Cite as:
J Melegati, A Goldman, F Kon, X Wang. A model of requirements engineering in software startups in Information and Software Technology, Volume 109, pp. 92-107, 2019.

A model of Requirements Engineering in Software Startups

Jorge Melegati[a], Alfredo Goldman[b], Fabio Kon[b], Xiaofeng Wang[a]

[a] *Faculty of Computer Science - Free University of Bozen-Bolzano*
[b] *Department of Computer Science - University of São Paulo*

Abstract

Context: Over the past 20 years, software startups have created many products that have changed human life. Since these companies are creating brand-new products or services, requirements are difficult to gather and highly volatile. Although scientific interest in software development in this context has increased, the studies on requirements engineering in software startups are still scarce and mostly focused on elicitation activities. **Objective:** This study overcomes this gap by answering how requirements engineering practices are performed in this context. **Method:** We conducted a grounded theory study based on 17 interviews with software startups practitioners. **Results:** We constructed a model to show that software startups do not follow a single set of practices but, instead, build a custom process, changed throughout the development of the company, combining different practices according to a set of influences (Founders, Software Development Manager, Developers, Market, Business Model and Startup Ecosystem). **Conclusion:** Our findings show that requirements engineering activities in software startups are similar to those in agile teams, but some steps vary as a consequence of the lack of an accessible customer.

Keywords:
software startups, requirements engineering, empirical software engineering, customer development, product validation

1. Introduction

Since the beginning of this century, humanity has seen an enormous number of innovations in several areas. Information access is ubiquitous, buying and consuming products and services can be done with a few taps on a smartphone. Many of these innovations come from nascent companies (startups), formed specifically to address that innovation. These organizations are present in several markets. For instance, a company that develops a new biotechnological product could be a startup as well as another that creates an innovative device using nanotechnology. Software startups are startups in which software development represents a core part of their product or services [1, 2, 3]. The product or service can be a piece of software like a software-as-a-service platform or can be the means that makes the product or service possible like a new commuting app.

According to Nuseibeh and Easterbrook [4],"the primary measure of success of a software system is the degree to which it meets the purpose for which it was intended". Then, they define Requirements Engineering (RE) as the process of discovering that purpose by identifying the stakeholders and their needs, documenting the discoveries for future analysis, communication and implementation. The authors also stressed that the use of the term engineering in RE is "an important part of an engineering process" remembering that typical definitions of engineering bring the idea of creating cost-effective solutions to practical problems. Kotonya and Sommerville, in their seminal RE textbook [5], described RE as a "term which has been invented to cover all of the activities involved in discovering, documenting, and maintaining a set of requirements for a computer-based system."

It has been argued that RE is a misnomer, and the term "engineering" serves as a reminder that RE is an important part of software development that is concerned with anchoring development to solve a real-world problem [4]. RE is even less an engineering process in the context of software startups since they are creating brand-new prod-

ucts or services in highly uncertain and turbulent business environments. The teams do not know the requirements in the first place, and they are highly volatile [6]. Besides that, in startups, success is represented by business results and not necessarily technical accomplishments [7, 8]. Such a context may influence how a development team performs RE activities, which are crucial to the success and even to the survival of these startups [9].

There is evident proximity between requirements and business aspects of software. While other software artifacts are related to the solution, requirements are mainly concerned with the problem that software is supposed to solve [10]. In this sense, startup development methodologies like Customer Development [11] and Lean Startup [12] may influence how software startups perform RE activities because they focus on the business aspects of software startups. These methodologies appeared as a response to the "dot-com bust", a phenomenon occurred at the beginning of the 2000s when several IT-based companies were not able to deliver real results after raising a significant amount of money from investors. Their valuation went to almost nothing, and they were no longer capable of getting more capital to continue running resulting in several of them closing [13].

According to Blank [11], the reason for most of the startup failures was that they were following a traditional product development path. In this path, a company should develop a business plan, get funded by presenting it to potential investors, build a team, develop the product and sell it. For Blank, this path does not work for startups because customers may not accept the product. That is, acceptance or adoption-related problems could happen, and he claims that they are commonplace in software and web markets. Lean Startup and Customer Development tackle this problem by proposing practices to validate if there will be enough customers willing to pay for the product. For instance, in Lean Startup, teams should follow Build-Measure-Learn cycles in which they create a Minimum Viable Product (Build) like a prototype with only the necessary features to test a business hypothesis through metrics (Measure). The validation or not of the hypothesis will represent validated learning (Learn) for the team. Through validated learning, the team can move to another cycle to test other hypotheses or change the product idea which is called a *pivot*. Nevertheless, the existence of such methodologies does not ensure the adoption and correct use of them.

Although very important for startups, in a systematic mapping study of software engineering practices in software startups, Paternoster et al. [6] had not found any study focused on RE practices. This gap motivated our study guided by the following research question:

RQ: How do software startups carry out requirements engineering activities?

To achieve this goal, we conducted 17 interviews with the founders or employees of software startups located in the region of São Paulo, Brazil, following a Grounded Theory (GT) [14] approach. We published our preliminary results [15] after conducting nine interviews and presented a preliminary conceptual model of RE processes in software startups. In this paper, we present the results based on all 17 interviews and extended our previous work by adding more empirical evidence and exploring in-depth various influences and their effects on RE processes in software startups. We also add a comparison of RE in software startups with that in non-startup contexts, especially in the agile software development context.

The ecosystem where the research was conducted, São Paulo, Brazil is one of the largest tech startup hubs in the world. It hosts nearly a thousand startups, contains a large research university, ranked first in Latin America, and is located in a large metropolitan area. The local government and the state research agency has special programs to fund innovation projects in small companies that have funded over 1,350 different companies in the past 20 years[1]. Although the amount of private investment in startups is still much lower than in leading ecosystems such as Silicon Valley, New York, and Tel-Aviv [16], there is a growing level of investment, both from local angels and local and international VCs, which has led to the creation of three unicorns in the past two years. Thus, it is a good environment for collecting research data due to its size, diversity, and complexity.

The remainder of this paper is organized as follows: Section 2 presents software startup and RE concepts, Section 3 provides an overview of related works. Section 4 describes the research design including data collection and analysis and Section 5

[1] https://bv.fapesp.br/en/266/small-business-research/

presents the results. Finally, Section 6 contains a discussion on results and Section 7 gives the conclusions.

2. Background

This section presents the elements that support our study. First, Section 2.1 contains a discussion about the startup definition and Section 2.2 describes RE stages.

2.1. Software startups

In scientific literature, scholars have used several definitions for the term startup. Sutton [17] considered as a startup an organization that has little or no history of operation; has limited resources (human and financial); is influenced by multiple elements; and handles changing technologies and markets. In a systematic mapping about the subject, Paternoster et al. [6] analyzed other definitions used in several studies and expanded this definition taking the following aspects into account: innovation, fast growth, time pressure, third-party dependency, focus on one product and flat organizational structures. In a more recent systematic mapping study replicating that of Paternoster et al., Berg et al. [18] still found an inconsistency of characterizing startups. They analyzed 27 papers published between 2013 and 2017 and performed a thematic analysis of how the authors characterized software startups. According to their analysis, the most used themes were innovation/innovative and uncertainty.

Other well-known definitions come from gray literature (e.g., blogs and magazines). Blank defines a startup as "a temporary organization used to search for a repeatable and scalable business model" [11]. In the book where the Lean Startup methodology is described, Ries proposes that "a startup is a human institution designed to deliver a new product or service under conditions of extreme uncertainty" [12]. Although very simple, these definitions focus on probably the most crucial factor that distinguishes startups from other companies: innovation. Startups develop an innovation: from a new concept to a real usable product or service.

These definitions have another interesting fact: they do not require the company to be small or new. Then, teams from consolidated companies creating innovative products or services could also be called startups. Both authors give examples of such teams and how similar their problems are from other startups. Researchers including Alpkan et al. [19] and Reuther and Schumman [20] discussed the relationship between innovation and the intrapreneurship. In summary, we considered as a startup a human institution searching for a repeatable and scalable business model under conditions of extreme uncertainty.

2.2. Requirements engineering

Nuseibeh and Easterbrook [4] define Requirements Engineering (RE) as the process of discovering the purpose that a piece of software is intended by identifying the stakeholders and their needs, documenting the discoveries for future analysis, communication and implementation. In their Requirements Engineering textbook, Kotonya and Sommerville [5] enumerate the activities performed under RE: elicitation; analysis and negotiation; documentation and validation.

Elicitation comprehends the listing of features and behaviors that the system should implement and also what it does not have to, that is, during this stage analysts draw the system boundaries [4]. These elements can be functional or non-functional requirements [21]. Functional requirements are the features themselves representing which activities the system will be able to perform such as to list users or to open a file. Non-functional requirements describe how the system should behave like, for example, how fast or reliable it should be. The first task to list requirements is to find stakeholders, people who have some interest in the system. Evident stakeholders are users, but also can be whoever orders the system (who may not be the user), or developers, or other developing companies [5].

From a set of requirements gathered from different stakeholders, there can be several problems: duplicates, not well-described features, or requirements beyond the scope of the project or not technologically feasible [5]. In the analysis and negotiation phase, the team should analyze, detail and, in the case they are not feasible or beyond project scope, to discard requirements [5]. Then, the team describes the selected requirements in detailed documents [5]. This documentation stage is essential to guide the implementation and, after that, to check if a feature certainly fulfills the demand [4]. Nevertheless, before implementing, there is a validation stage when analysts check requirements [5]. They should be complete, that is, they should have all the details to allow the correct implementation of the functionality and be consistent in the sense that they meet the users' expectations [10].

Since the introduction of agile methodologies, a prolific topic has been RE in agile contexts [22]. For instance, Inayat et al. [23] performed a systematic literature review on the subject and found 21 studies on practices and challenges in this context. Among these studies, Ramesh et al. [24] studied 16 organizations that employ agile practices to understand how they perform RE practices. The authors identified the following practices: face-to-face communication, iterative requirements engineering, requirements prioritization goes extreme, managing requirements through constant planning, prototyping, test-driven development, user review meetings, and acceptance tests.

3. Related work

The scientific interest in software startups is growing. In a systematic mapping study on software development in startups, Paternoster et al. [6] examine 43 primary studies and dedicate a subsection to RE. In this subsection, they analyzed studies focused on requirement elicitation and practices. The authors drew several conclusions. First, RE in software startups is limited to some key practices [25]. Second, because of market innovativeness, requirements change very rapidly representing an increasing difficulty [26]. Meanwhile, although startups create products for growing markets, several authors [27, 28, 29] suggests more user involvement in eliciting and prioritizing requirements.

A relevant study mentioned in the systematic mapping study was Coleman and O'Connor's [30] work on software development process formation in Irish software startups. The authors developed a model composed of the following categories: software development manager experience, founder experience, management style, process tailoring, and market requirements.

More recently, Berg et al. [18] replicated the mapping study of Paternoster et al. [6]. They selected 27 new papers from the period 2013-2017 and concluded that the publishing frequency is higher than in any other period and these papers have greater pertinence and rigor than those in the 1994-2013 period. They also classified selected papers according to SWEBOK areas. In software requirements, they only mentioned a paper by Rafiq et al. [31]. In this paper, the authors investigated how three software startups from different parts of the world conducted requirements elicitation and what are the techniques used. The authors concluded that the process is mainly informal and occurs alongside product evolution.

Some studies also focused on the efficacy of methodologies applied in the startup context. Chanin et al. [8] analyzed three startups that took part in a startup development program based on the Customer Development methodology. Although having limited data, the authors believe that there is an indication that the methodology improved the startup understanding of customer needs, that is, the software requirement process. Frederiksen and Brem [32] compared Lean Startup with "leading theories and empirical evidence found in the scientific literature" and concluded that the key methods and main approach find substantial evidence to its efficacy.

After the mapping study of Berg et al. [18], more recent studies have focused on RE in software startups. Klotins et al. [9] performed a multi-vocal exploratory study based on experience reports of software startups and concluded that these teams "apply market-driven requirements engineering practices to discover and validate ideas" but "practices are often rudimentary and lack alignment with other knowledge areas." Tripathi et al. [33] focused on how RE is practiced in software startups performing a multi-vocal literature review and a case survey with 80 cases. Their results consisted of lists of techniques used by software startups to perform RE activities like elicitation. The authors covered several startups, but it does not address the reasons for choosing different techniques. Gralha et al. [34] investigated the evolution of RE practices in software startups. In their study, six dimensions emerged: Requirements artifacts, Knowledge management, Requirements related roles, Planning, Technical debt, and Product quality. They concluded that software startups evolve in all dimensions in 3 different phases; the evolution is characterized by turning points like the number of clients, the number of features and negative feedback. This work does not allow a comparison between software startups and conventional companies since it does not rely on traditional RE terminology and practices (e.g., analysis and validation).

In our previous work based on nine interviews with people working in software startups [15], the construction of a conceptual model of the RE process in software startups was initiated. Based on the emerging model, we discovered that software startups did not follow a specific set of activities; instead, they were dictated by a set of context influ-

ences (Founders, Software Development Manager, Developers, Business Model, Market, and Ecosystem). RE in software startups is closely related to business development, where the business model is a decisive factor in the choice of RE practices being used. However, this preliminary paper did not address how and which activity each influence dictates.

In summary, the reviewed related studies do not cover all RE activities in the specific context of software startups and do not fully explore the problem and compare it to other environments like teams employing traditional or agile methods. The importance of such RE in software startups is even higher: "inadequacies in applying" software engineering practices in startups could be a significant factor to their failure [1, 9]. In the present paper, we intended to address the above-mentioned knowledge gap. Besides exploring the consequences of the influences, we added more empirical pieces of evidence to our previous work to strengthen the arguments, and also presented a comparison between our results and those found in the literature regarding RE in traditional and agile software development teams.

4. Research methodology

With the research question as a starting point, this study focuses on how a group of people (software development team) develops a set of processes (requirements engineering) in a specific context (software startups). In other words, this research focus on human behavior. According to Seaman [35], "human behavior is one of the few phenomena that is complex enough to require qualitative methods to study it". Given the lack of a theory to explain how software startups carry out RE, Grounded Theory is a reasonable method choice since its goal is to develop theory from data [14, 36]. Grounded Theory is a method originally proposed by Glaser and Strauss [37] which purpose is to inductively generate theory from data instead of validating an existing one [38]. It has been used in several disciplines like medical sociology, nursing, education, and management [38]. Lately, several studies employed it to draw theories in the software engineering field, including studies in the software startups ([30]) and RE ([39]) contexts.

As pointed out by Stol et al. [38], there are at least three main streams of Grounded Theory: classic or Glaserian GT; Strauss and Corbin's (Straussian GT); and constructivist GT. Then, while reporting a GT study, it is essential to state explicitly which stream is being used. This study is based on Strauss and Corbin's version, using their book [14] as the primary guideline during research execution. This choice is reasonable since our research question was defined at the beginning of our study. Figure 1 depicts the overall research process used in our study. The following subsections elaborate on data collection and analysis in detail, then highlights the iterative nature of the research process. The GT's core principles, according to Stol et al. [38], are in bold.

4.1. Data collection

Our study is exploratory in nature. According to the classification of Easterbrook et al. [40], our research question is of description and classification type, and it is a base-rate question, the answering to which will result in a clearer understanding of RE process in software startups. The selected research methods should "offer rich, qualitative data, which help us to build tentative theories". Surveys or structured interviews would focus on the elements present in the questionnaire or questions made. On the other hand, unstructured interviews could lead to digressions that would not be valuable to this study. Then, semi-structured interviews were more suitable. In this method, the interviewer has a question set, but he or she is allowed to add questions, modify their order or suppress them depending on the interview evolution. This method enables the interviewee to tell new details that the interviewer has not known about, but restricts the time spent on unnecessary digressions. It prevents missing essential elements meantime generates better data than sticking to the guide [41].

4.1.1. Interview design

To support the interviewer, we developed an interview guide [41] represented in Table 1. As Weiss [41] proposes, we listed topics with "just enough detail to make evident what is wanted", and added some answers examples to show the interviewee(s) what we wanted. As mentioned by Stol et al. [38], we used the literature to formulate questions for data collection what is a possibility under Straussian GT. The interview guide consisted of three parts.

With the first part, we wanted to get information about the interviewee(s) and the startup, and

Table 1
Interview guide

		Question	Options/Suggestions
Background	1	How many startups have you worked for and/or founded?	a) 1 b) 2 c) 3 or more
	2	How long at total have you worked for startups?	a) Less than 1 year b) From 1 to 2 years c) From 2 to 3 years d) More than 3 years
	3	Which title best describes the position you had in your last startup?	a) Software developer b) Tech manager or director c) Marketing d) Operations e) Strategy f) Other, what?
	4	For how long does your current startup exist?	a) Less than 1 year b) From 1 to less than 2 years c) 2 years or more
	5	Briefly describe your last startup product. How software development is related to it?	(open question)
	6	Which of the following techniques do you know? Do you try to use them?	a) Lean Startup b) Design Thinking c) Scrum d) Extreme Programming e) Kanban f) AB tests g) MVP
Main questions	7	Which of the following phrases best describes how the features that your company develop will have:	a) Developers' ideas b) Ideas from other company teams (marketing, operations, etc.) c) Suggestions form in an website/app? d) Surveys or user interviews e) Other?
	8	Once a idea is created, is it discussed somehow before getting implemented? Who takes into this discussion?	(open question)
	9	Given the possible features list, how, in your startup, is defined which will be implemented in the next release?	a) Development team meeting (with or without other company members) b) Managers' decision (manager, director, etc) c) Developer will (what is most interested in) d) Survey with users e) Other?
	10	Given the features that will be implemented in your startup, is there any kind of verification if it will attend an user expectation before getting implemented? If yes, how?	(open question)
	11	In your startup, is it verified if a implemented features is being used by users? If yes, how?	(open question)
	12	How does your startup document the features that will be implemented?	a) E-mails between team members b) Paper writing c) Stick notes in a board d) Wiki e) Issue tracker f) Project tracking tools (e.g. Jira) g) Other?
	13	In your opinion, what determines how the requirements engineering process will be made in your startup?	(open question)
	14	Is anyone responsible to guide the product development?	(open question)
Closing	15	Any other details about features to be implementation choice that was not covered in this interview?	
	16	Any comments?	

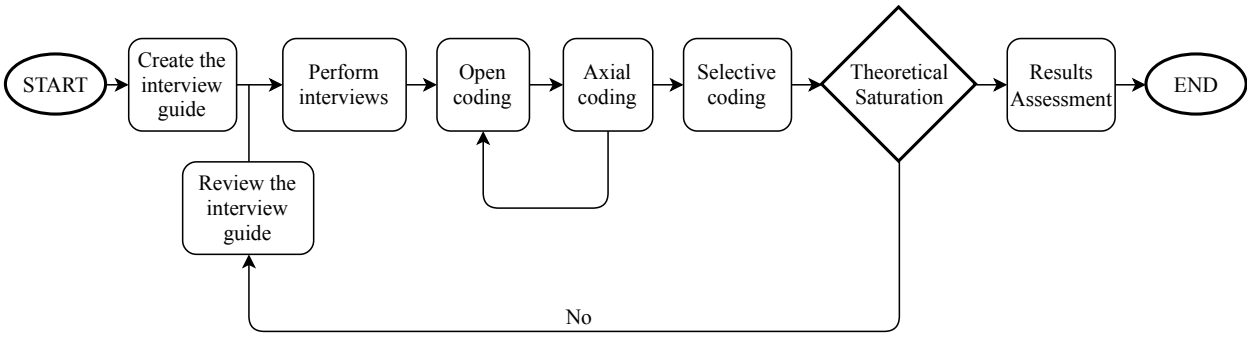

Fig. 1. Research process.

also make them feel comfortable throughout the interview. The first three questions concerned the interviewees' past and current experience in startups. The following two concerned the startup that the interviewee is currently working on its age and a description of the startup product and its relationship to software. Through these two questions, we aimed to select startups in different stages and also check if they are actually software startups. Finally, the sixth question got information about the knowledge and use of: agile methodologies, like Scrum, Extreme Programming and Kanban; startup development methodologies, like Lean Startup, and some techniques like MVP (Minimum Viable Product) and AB tests. Given that these methodologies and techniques discuss *what* a startup or a software development should do, it is reasonable that they may influence RE.

The second part consisted of open questions about how the startup performs each RE activity since those represent our lines of inquiry [41]. Question 7 focused on elicitation giving, if necessary, some suggestions about requirements sources. Question 8 concern requirement analysis, Question 9, prioritization, Questions 10 and 11, requirements validation and the real use of features, Question 12 versed about documentation. Finally, Questions 13 and 14 concerned about the product team and were added to get more information about this actor that showed up during initial interviews. We deliberately did not use the "product manager" and "product owner" not to bias the interviewees to answer based on our expectations.

Finally, in the third part, interview feedback was gathered aside from any other information that he or she thought that would be interesting to the research. At the end of the interview, as proposed by Weiss [41], the interviewer checked the guide to see if all areas were covered.

4.1.2. Interviews

The set of interviewees consisted of our contacts, and a snowball approach was used to reach more software startups. We considered a software startup, an organized team (legally a firm or not) that met the following criteria: i) developed an innovative product or service (a brand-new or a copy of a similar product already existent in another market) and ii) software development is essential to the team to reach their business goals. The potential interviewees' list reached thirty prospects. The first author contacted them through emails trying to arrange a meeting in the startup working environment. In this way, the interviewer could also write field notes (**treat everything as data**). After some contacts, this requirement became more flexible because of the interviewees' busy schedules.

In the end, we conducted 17 interviews in total, some with more than one person, totalizing 23 people that have founded or worked in over 30 startups. Table 2 presents the list of interviews. Among the 17 interviews, eight were conducted in person and 9 through video conference calls. All covered startups were in São Paulo startup ecosystem, one of the biggest startups hubs in the world, in the 12th position according to the Global Startup Ecosystem Ranking 2015 [42]. The interviews took place between December 2015 and September 2016, and their duration ranged between 21 and 48 minutes. In the interviews conducted in person, the interviewer also observed the working environment and, when possible, even the tools used by the teams like spreadsheets and boards. We recorded all interviews using two devices to prevent data loss. Then,

the first author transcribed the first four interviews, and a professional service did the remaining. The interview time was summed up to 9 hours and 37 minutes, resulting in 181 pages of transcription.

4.2. Data Analysis

Data analysis consisted of three stages: open, axial, and selective **coding**. According to Strauss and Corbin [14], open coding is "the analytical process in which concepts are identified, and its properties and dimensions are discovered in data". Axial coding "is the process of connecting categories to its subcategories, and is called axial because it occurs on a category axis, associating categories on properties and dimension level". In selective coding, the analyst builds and refines the theory, identifying a central category and relating all others to it.

The open coding started with a careful reading of the transcripts followed by labeling and creation of memos (**memoing**). We used some labeled excerpts to illustrate this text. In order to differentiate them from literature excerpts, we italicized them. After reading the first interviews, the first author decided labeling phrase by phrase. After that, he compared labels and grouped them into categories starting the axial coding. These steps were conducted using the AtlasTI[2] tool. Throughout the whole data analysis process, the second author was consulted, providing constant feedback. Then, relations between categories are drawn, for instance, subcategories, cause and consequence, and also properties and dimensions are organized (**theoretical sensitivity**). Finally, in the selective coding, a core concept emerged, and categories were associated with it (**cohesive theory**). These steps were mainly performed on a whiteboard because it facilitated visualizing relationships between categories(**theoretical sorting**). Besides that, drawing and erasing was more straightforward than doing it in software and kept the analyst focused. The process was not as linear as this description; instead, it followed an iterative approach (**constant comparison**) as presented in the next subsection.

To illustrate our data analysis process, we will show how we coded pieces of data to reach the Market category. The CEO in I2 said *"so the main company products precisely are machines, using these techniques, to agriculture automation"* and we labeled it as "B2B". I9 describe their product: "And the students access these materials, and the platform generates... gets metrics on that, and create reports on students' learning," and we labeled it as "B2B with several users." I8 said: "Anyone can watch a video in the free mode and then if he likes our course, its quality, he subscribe [the company name] to have access to all courses in the catalog," and we labeled it as "B2C." In the axial coding phase, all three labels were put together under the category Market turning dimensions of it. During this stage, we also related them to labels in other categories like I8 which title was "Product Director", labeled as "Product team", and I2 that said "In the case of the project [name], we defined the prioritization quite the client's", labeled as "Prioritization by client" categorized under "Prioritization." In the selective coding phase, we used the RE process as the core category and defined Market as an influence on this process.

4.3. Iterative research process

Strauss and Corbin [14] encourage an iterative process to the Grounded Theory research designs (**immediate and continuous data analysis**). Our study design had cycles consisted of interview guide creation (on the first cycle) and enhancement (on following cycles, **theoretical sampling**) followed by data collection, through interviews and field notes, and data analysis. The processes of data collection and analysis were also intertwined. Data analysis started even before data collection for that cycle finished as suggested by Eisenhardt [43] and axial coding started before labeling all interviews. This tactic improved labeling results. This process was followed until the **theoretical saturation** was reached, as depicted in Figure 1.

Besides that, given the exploratory nature of the study, it is advisable not to fix the interview guide in advance and update it when necessary. In our study, the interview guide changed as information gets available through interviews. In question 6, we also asked if startups were using Customer Development, once it was clear that they were not aware of it, we stopped asking about it. Question 8 was added to get more information about requirements analysis and questions 13 and 14 were added to get more information about the product development team without mentioning it to prevent biasing interviewees. In this sense, the research design should provide a means to update it. Besides that, knowledge absorption also influences data analysis. After the axial coding stage, data labeling was improved

[2] http://atlasti.com/

Table 2
Interviews performed in startups. The letter F indicates those interviewees that were also the founders of the startups. CEO is an acronym for Chief Executive Officer, that is, a person responsible for strategy decisions, the startup general leader. CTO is an acronym for Chief Technology Officer, that is, a person responsible for technical decisions in the startup.

Study cycle	Interview	Interviewee(s) position(s) in the current startup	Current startup market sector	Current startup age
1	I1	Software director (F)	Internet of Things	2 years and a half
	I2	CEO (F) Software development director (F)	Automation	3 years and a half
	I3	CEO (F) CTO (F)	Health	1 year
	I4	Technical Leader	Real state	3 years
	I5	CEO (F)	Finance and Defense	13 years
	I6	Technical leader (F)	E-commerce	2 years and a half
	I7	Mobile development team leader	Sharing economy	1 year
	I8	Product Director	e-Learning	2 years and a half
	I9	Product Manager	e-Learning	3 years and a half
2	I10	CEO (F) Product manager Software development manager A Software development manager B	Advertising	5 years
	I11	CEO (F)	Virtual reality	3 years and a half
	I12	CTO (F)	Social network and Data Mining (2 startups)	4 years, 2 years and a half
	I13	CTO	E-commerce	3 years
	I14	CTO (F)	Specific office software	3 years and a half
3	I15	CTO	Advertising	5 years
	I16	CTO CEO (F)	Web	1 year and a half
	I17	Product director	E-commerce	4 years

since similarities and differences between categories were more visible.

Following the process described in Figure 1, the data analysis reached theoretical saturation[3] after three cycles. The first cycle consisted of nine interviews, and the criteria to pause interviewing was practical: the original plan was to stop after the fifth interview and focus on analysis but, given the difficulty to schedule interviews, the first author preferred to continue the scheduled ones. This decision may not have influenced results, at most, postponed some modifications on the interview guide. The second cycle consisted of five interviews and the third, three. The number of interviews in the third cycle was fewer because theoretical saturation seemed close.

[3]Theoretical saturation happens when additional data provides no new knowledge [30].

4.4. Results assessment

During this text preparation, we presented the results back to the interviewees. In this way, we performed a *member checking* (or respondent validation), which is considered a valuable technique to assess ground theory studies [44]. We sent an

email to an interviewee from each interview with a summary of our results and a link to an online survey. The survey consisted of four questions with a five-point Likert scale about overall aspects of the results and the usefulness of the model, as well as an additional open question. We sent 17 emails and waited for two weeks. In between, we sent follow-up emails to those that had not yet answered. In the end, we collected 11 answers and incorporated them into the results.

5. Results

From the data analysis, a model of the RE process in software startups emerged and is presented in Figure 2. One core category of the model is the RE process which includes common RE activities performed by startups. Section 5.1 describes the overall characteristics of the whole process. Instead of following a common set of practices, each startup follows different ones determined by a series of influences (Founders, Software Development Manager, Developers, Market, Business Model, and Startup Ecosystem). Section 5.2 describes these influences in detail, especially Market and Business Model. These influences determine the presence or not of a specific actor in the process, the product management team, as described in Section 5.3. Section 5.4 describes observed practices and how influences acted on the selection of them.

5.1. RE Process

During data analysis, some dimensions of the whole process emerged. First, rather than strictly following a methodology, software startups create a custom-made process arranging techniques from different methodologies. For instance, I13 was the software development manager in several startups and mentioned that for each one he created a specific process combining practices from different methodologies. I17 mentioned: *"[we tried] always to consume these better practices but never getting stuck too much... to follow a methodology by the book, but always trying [...] to extract the principles behind it, what is good, to incorporate [...] in our culture"* or like I8 that mentioned that "we use a 'thinner' Scrum." I8 preferred to follow a successful startup, Spotify, that made public their process[4]. Coleman and O'Connor [30] already mentioned this process tailoring mechanism.

Second, teams do not set the process upfront and follow it indefinitely. Instead, they change it according to the product or team needs. This change occurs through adding new practices as the team grows (for instance, I12 mentioned: *"in the beginning we were not using [Scrum], but after we had two developers, a more stable thing, we started doing sprints, software deliveries"*) or when a team member proposes a new practice. For instance, I2 mentioned that a new employee brought Design Thinking practices to the team. Design Thinking is a set of techniques proposed to systematize activities performed by designers during product development. The methodology tackles "wicked problems" [45]. According to Rittel and Webber [46], these problems can have "an exhaustive inventory of [...] conceivable solutions", in contrast to "tame" problems that have clear missions and the solutions can be verified. These are the problems scientists and engineers face. There are several versions of Design Thinking, but one (IDEO's version [47]) got very famous in the startup community. Nevertheless, the opposite can also happen: a team can abandon a practice like when a startup (I4) stopped using Scrum including RE related practices like sprint planning and product backlog. A team can also postpone an improvement movement as mentioned by I3 that feels they are not using the best possible process, but he prefers to focus on creating the product first and then improve the process.

5.2. Influences

Six different influences on RE process emerged from data. They are: *Founders*, *Software Development Manager*, *Market*, *Business Model*, *Developers* and *Startup Ecosystem*. In the following subsections, a description of each influence is presented.

5.2.1. Founders

Founders are the people who created the company. Therefore, they are very influential in how the startup operates (represented in Fig. 2 by arrows from Founders to Elicitation, and Analysis, Validation, and Prioritization). For instance, I4

[4]The Spotify development process is available in https://labs.spotify.com/2014/03/27/spotify-engineering-culture-part-1/ and https://labs.spotify.com/2014/09/20/spotify-engineering-culture-part-2/. Accessed in 2018-06-04.

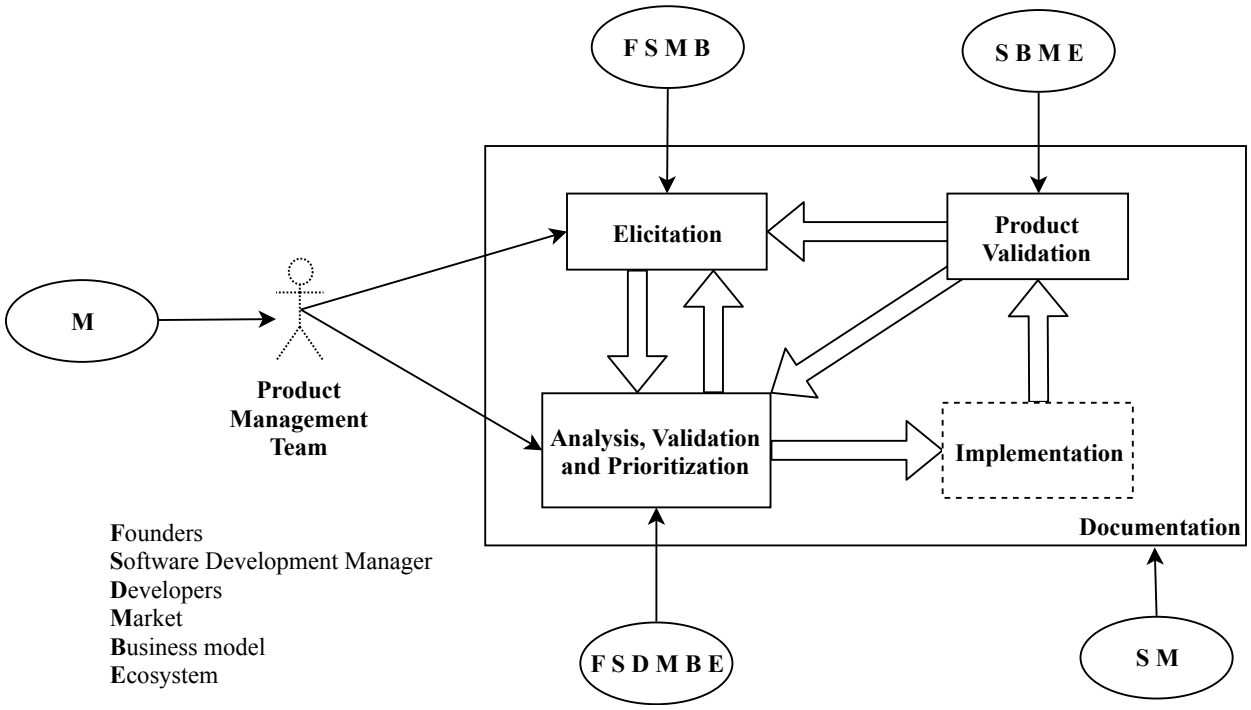

Fig. 2. The Requirements Engineering in Software Startups Model. The influences on each activity are represented by the initials inside the ovals. Rectangles represent activities. Documentation permeates the entire process. Thin arrows represent influences and thick arrows represent most common requirement flows.

mentioned that his team had used Scrum before as a base process. However, since founders viewed it as a *"waste of time"*, the team abandoned it. After almost a year, the team would come back to use a Scrum-based process since the results were not good. I2 found a similar difficulty: while trying to use some Scrum and Kanban elements like *"a blackboard with tasks"*, *"tasks meeting and sprints"*. He blamed his partner: *"[the co-founder] did not help too much"*. When the interviewer asked what determined the process they are following for RE, I17 mentioned *"the big challenge is that there is a culture inside the company, obviously, and this come even from founders"*. I11, the CEO and founder of the startup, mentioned: *"we use a basic Scrum because it has a lot of overhead."*

The interviewees who were founders, generally, had a long experience on startups. Several had founded several companies like the CEO in I3, I6, I8, I12, in some cases running more than one at the same time (the CEO of I3 and I12). Many have never worked for big or consolidated companies (I2, I3, I11). Sometimes, they had a technical background like I1 but in other cases like the CEO of I16 that admitted *"[another co-founder] and I do not understand anything about technology"*.

5.2.2. Software developer manager

Despite the founders influence, the software development team leader or manager still sets which practices the software development will use (represented in Fig. 2 by arrows from SW manager to all activities in the process). I1, I5, I12, I13 and the CTO of I16 corroborated this assertion. I7 mentioned *"our CTO [...] has a good background in Lean, then he helps us a lot in several processes [making the process well-defined]"*.

I13 performed the role of software development manager in several startups and told: *"In each context, I made a composite of practices that will better fit there, and the process will be based on that"*. Then, it is natural to expect that they will also define RE practices. For instance, I14 said *"[..] I have always intended to try simple and lean solutions to check if that feature would indeed add value."*

5.2.3. Market

The external market influence on internal RE practices is also present. I5 mentioned that for his startup to be able to sell products in a critical market like defense, the organization had to follow a strict process required by clients. Beyond obligations, according to the interviewees, the market represented a strong influence on the RE process. It was clear that the process was different depending on the size of its user pool.

A natural classification for companies towards its user pool would be between business-to-consumer (B2C) and business-to-business (B2B). This classification is used in marketing [48, 49] and operation studies [50]. In this classification, one would expect that B2B companies have a smaller user pool than a B2C company. However, that was not necessarily true based on our interviews. For instance, I9's product was a platform for elementary and high schools where users are students, parents, and teachers. Since the school pays for the platform, the startup should use B2B practices for marketing and operations purposes; nevertheless, its user pool is as big as a traditional B2C as the interviewee mentioned that they used "metrics" to see how students are using the platform.

Therefore, we propose a new market, or even product, classification to software startups concerning RE. A startup could be a user-targeted or a client-targeted one. The first concerns startups with a large user-pool like most of the B2C companies or B2B companies with many users under the same client. On the other hand, client-targeted startups have a few user-pool like most B2B companies. These differences affect, especially, how these different teams perform elicitation and analysis (as represented in Fig. 2). A large pool of users demands techniques to assess the needs for most of the users, and generally the presence of a product team (represent in Fig. 2 by an arrow from Market to Product Team). Table 3 summarizes the differences between user-target and client-targeted startups regarding RE activities.

5.2.4. Business model

Data collected showed the business model influence on RE through two different aspects: the maturity stage and reason why the company develops software.

Business model maturity. The startup development stage changes how the team processes requirements. There were two different stages: first, the team is still building its product, understanding its market, sometimes without even delivering a first MVP or prototype, that is, the business model is still not mature. In this case, founders participation in processes was stronger because of their concern about startup survival: the result can change if the company will succeed or fail. This phenomenon was identified in the current startups of I1, I3, I4, I6, I9, I12 and I13. I8 highlighted that founders participate in prioritization meetings, for instance, because development team achievements *"sometimes changes the company's month [revenue]"*. Meanwhile, if the startup already set its business model, most of the requirements are fine-tuning or bug fixes, then founders involvement is less frequent as observed in the current startups of I5, I7 and I10. Figure 3 represents these results.

Reason for software. The role of software in the startup business model also shapes the RE process. The software can be either the product itself, like an agricultural automation tool to select vegetables (I2) or office software for a specific scenario(I14). Either a business model piece to achieve a different goal, like an online real state agency that seeks to make the renting process simpler through the web(I4 or a platform for a sharing economy app (I7) In the first case, requirements from users are closer to development while, in the second case, the business analysis will dictate RE.

5.2.5. Developers

The influence of developers in software development practices selection is also relevant regarding Analysis and Validation as represented in Fig. 2. For instance, they can refuse to use a technique because of their disbelief on it or just not willing to perform it. For instance, I1 said *"to implement concepts I just can't say: 'guys here it is,' I have to go slowly in such manner"*. I9 mentioned that *"it is always tough to convince that we have to iterate in small cycles"* and that *"[user interface designers] want everything perfect."* On the other hand, they can also improve the process through their past experiences. I7 mentioned that this had happened to her actual and previous teams. I2 said that they used design thinking for a while because of a former employee who had had experience with it. Sometimes, the existence of more developers in the team influenced the decision of using specific techniques. I12 mentioned: *"for a while, we had two other developers, a more stable thing, we started using [some practices]"*. Table 4 summarizes human

Table 3
Differences on requirements engineering based on the market they operate.

	User-targeted	**Client-targeted**
Number of potential customers	Large pool of users	Small number of customers
Traditional classification	B2C or B2B (a customer represents several users)	B2B
Process	Product team as a proxy to real users influencing, specially, requirements elicitation, prioritization and validation	Use of few or only one customer to guide the product development
Validation techniques	Prototypes and metrics use	Interviews

influences on the process.

5.2.6. Startup Ecosystem

Four influential aspects grounded on data are related to the entrepreneurship environment where startups find themselves, known as its ecosystem [16]. These influences are knowledge spread, human resource availability, capital availability, and investors. These elements are some characteristics of the startup ecosystem of a region or country. A startup ecosystem is formed by startups, entrepreneurs, investors (angels and venture capital firms), universities and government [51].

Knowledge spread. Startups could acquire knowledge about different techniques and practices from other players like accelerators programs. For instance, I3 mentioned: *"we started at [an acceleration program], and they taught us startup things, MVP, and since then our target is to build an MVP"*. Alternatively, university courses like one mentioned by the CTO in I16: *"I had a college course about entrepreneurship and [...] I indeed liked it"*. Also in the same interview, the CEO regretted: *"if I have seen [that it would be better to check user interest before building a product], I would not have spent so much time, so much dedication building the product"*.

Human resource availability. I8 and I14 mentioned the difficulty to find people able to work for them. For instance, I8 said that his company had financial resources to hire more people, but they were not able to find people with the required [technical] knowledge. An undersized team will demand more work hours from members or deadlines will be longer. The time pressure by other people in the company, founders or even investors may lead to the removal of "unnecessary" practices like requirements analysis and validation.

Venture capital availability. Another issue faced by several interviewees' current startups was the lack of capital. Several of them (I1, I2, I3, I6, I11) develop more than one product or, besides their main product, offer consulting services on the same field. In such a way, they obtain money to sustain their main idea. Dealing with different products and services will increase the development team load leading to similar effects of those described in the previous item.

Investors. Represented by angel investors and Venture Capital funds (VCs), these actors have their expectations and expertise on how to build the product and may influence RE specially through acting on founders. This issue was raised both during the review process of this paper and during the results assessment step. I4 missed the influence of venture capitals and angel investors. About that, he conjectured *"maybe VCs and investors have not shown up in your model because their influence is indirect? That is, there is an influence that is made directly over founders, or through training of software developers or managers."*

5.3. Product management team

Many interviewed startups, especially those with user-targeted products, had a product management team. I12 mentioned that his startups have this

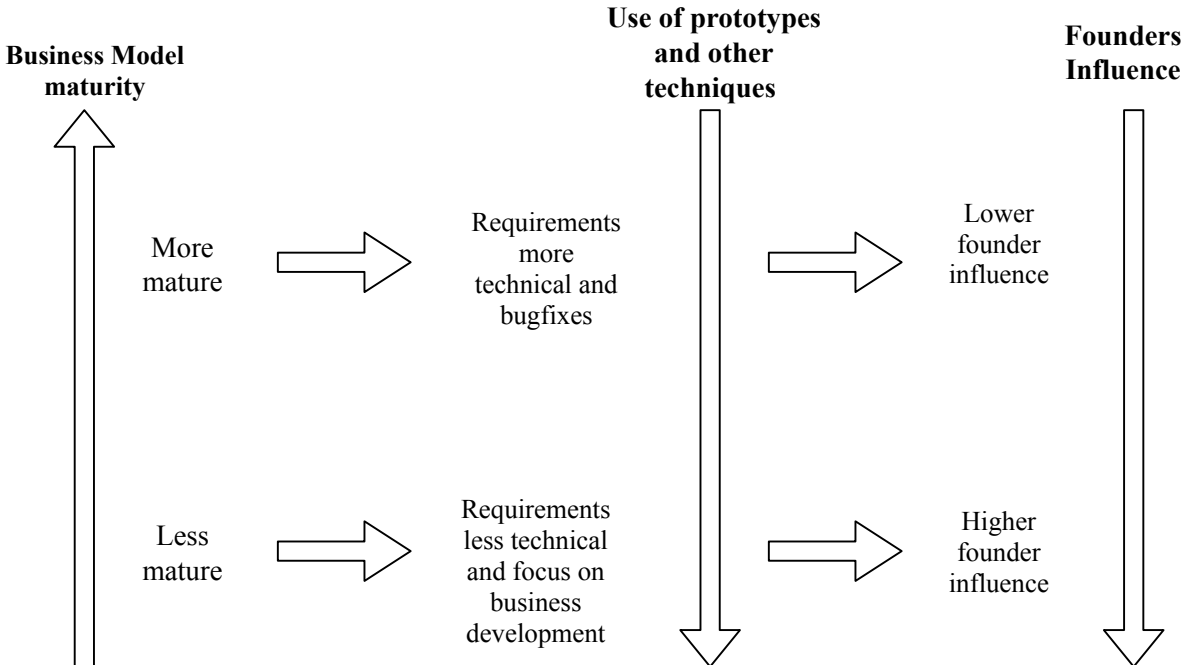

Fig. 3. The influence on business model maturity on RE process.

Table 4
Summary of human influences on RE process.

Software development Manager (sometimes a founder)	Based on his or her experience defines the overall team process
Founders	Even without technical background, they can hinder the use of practices depending on how they view them
Developers	They can promote the use of practices based on their previous experiences but can prevent the use if they do not have a positive opinion of them.

team because *"in the beginning everyone was lost and the product was bad [...] it is important to have someone here [to think about the product]."* This team was not composed of developers although its components can have a technical background, but also business or marketing profiles. They worked together with the software development. Usually, they are responsible for most of the RE stages, particularly, elicitation and prioritization. A product manager in I10 described his role: *"Part of my job is to be a filter... Even inside the company, I see my role as a filter because we cannot do everything. [...] I have to understand what the real problem is and tackle from only one side and all".*

I12 mentioned the product team importance: *"we have two guys concerned... trying to develop the product. [...] there must be someone responsible because it is very important. In the beginning, there was no one, and everyone was lost. Someone should be responsible for the product to have a vision".* I7 compared two different product managers: when a more experienced person took care of the product: *"he used AB tests and was very good with it"* but when a not so experienced person performed the role, according to her, they should have gathered more data to avoid waste of work.

5.4. Activities

The following subsections describe how interviewed software startups performed each of the RE

activities. Validation activities will be discussed together with analysis and prioritization since startups generally perform them before implementation. In Fig. 2, thick arrows represent the flow of a requirement. Besides the traditional flow elicitation, analysis and implementation, it is also represented additional flows that we observed in software startups. During analysis and validation, new requirements emerge. There is also a discussion on a post-implementation stage called product validation that is important for startups to understand if a requirement satisfies user needs. New requirements can emerge at this moment as well. In the section concerning requirements documentation, there will also be a discussion on requirements communication within the startup.

5.4.1. Elicitation

Elicitation of requirements is probably the most laborious task for software startups since they are building something innovative and users are not aware of what the software will in reality do. One solution was to map problems instead of solutions. In interview I9, the product manager said: *"several times, people come with the solution, but the solution always changes, the idea is always to map the problem never to lose the point"*. Then, the CEO added: *"the client has a problem. Instead of saying that he has a problem to solve, he already comes with a solution. However, this is wrong. How can he imagine a solution better than someone who makes a living of it?"*. In other words, the primary goal of this activity is to understand the user's problem.

Still, understanding the users' problem can be very difficult if the targeted market is vast. It is impossible to reach all the users and understand their needs. The general solution used by startups was to have a product team. This team is responsible for understanding users and market and guiding the development. Its tasks comprehend features selection and product market positioning.

An important fact confirmed by the interviewees is that new requirements are created continuously. Ideas come from anyone in the company and are constantly created. I8 mentioned *"we are always hearing everyone [...] obviously all the company brings ideas to put in discussion"*. In this sense, I15 mentioned *"everyone has a role"*. This phenomenon contributes to a continuous requirements creation and flow throughout the whole process. To create new requirements, the interviewees mentioned several sources like business objectives analysis (I4, I8, I10), use of competitors' products (I9) or from similar products (like an e-commerce platform for I13), ideas from developers or product team (I10), specially for features not visible to the final user (I4 and I11), sales team (I3 and I5). Some techniques mentioned were user interviews (I4, I7, I8, I9, I11, I12 and I13); brainstorm sessions (I7); and ideation (I8).

5.4.2. Analysis, Validation and Prioritization

Requirements analysis comprehend tasks "to discover problems with the system requirements and reach agreement on changes to satisfy all system stakeholders"[5]. There was not a general way the interviewed startups adopt. Some promote a discussion between developers and product managers to understand the best option to implement the requirement (I3, I15, I16) including the feature scope (I17). I13, I14, I15 and I16 mentioned an alignment between strategy and technology. Although commonplace in RE textbooks, consistency check was only mentioned by I5 and I15. In general, teams do not spend so much time in these practices.

Validation is probably the most important RE stage for startups. Generally, in this moment, the team will check if the users want the features represented by the requirements. Most of interviewed startups perform some kind of requirements validation with little or no development. I1 said: *"generally we perform a validation before implementation through user interface. We develop a wireframe using what we understood from the requirement and check with the clients"*. I4 mentioned a wireframe before implementing the solution. I8 alluded that once a feature was performed by humans before being implemented to validate if the user really wanted it. Other techniques mentioned by interviewees were: MVP (I3, I4, I10, I11 and I14); mock-ups (I4) and prototypes (I1, I2, I3, I5 and I11); surveys (I16 and I17) and focus groups (I17).

I9 said that his team only performed validation for more complex tasks. According to him, the time spent to perform validation for a more straightforward task could be higher than the feature development. I3 and I4 mentioned a concern while using validation techniques like MVP was the company image for their customers. In I4 words: *"[we are trying to adopt] an MVP that brings value and the user sees that has quality and not just a mock, that is, a lousy version that in the future will be great"*.

Requirements prioritization is an important task for startups given their small team sizes. They may use sprints like in Scrum (I1, I4, I10 and I14) or maintain a prioritized list like in Kanban (I9 and I12). Sometimes, the list also contains tasks not related to software development (I1, I2 and I6). Given the importance of software development, some times (I1, I4 and I13) a higher management layer (founders and/or other managers) defines milestones then the development or the product team are responsible to determine the task implementation order. For client-targeted startups, prioritization is generally restricted by asking to the first clients what are the most important features to them (I2, I11, I12 and I14).

Startup teams mentioned the following criteria for prioritization: firm strategy (I7, I8, I9, I12 and I13), need to demonstrate the product (I2, I3 and I4), value to the user (I12 and I15), prevent other teams blocking (I1), essential features (I16), high priority situation like critical bugs (I14), cost-effective analysis (I14).

5.4.3. Product validation

In RE textbooks like [5], implementation is the ultimate step in a requirement lifespan. For startups, though, it is actually more important to understand if a requirement satisfies a user wish. Then, teams revisit this problem after implementing a requirement. In this stage, metrics are used to evaluate user interaction with the product. A very common tool was Google Analytics (I1, I3, I4, I6, I9, I11, I12, I13 and I14). I3 described their goal: *"[a final product validation and metrics analysis] are very important for us because change how we develop our final product. Our current goal is precisely to iterate over the product with user feedback"*.

A difficulty mentioned (I9) in this stage was that, sometimes, several new features reach production together then it is hard to isolate the effect each of them cause. Another curious aspect is that practitioners see failure as natural. For instance, I6 told that several times had thrown away several hours of programming even performing validation before implementation. When asked if he thought that practitioners could avoid failures, he said: *"I do not think so, it is going to happen things like that"*.

5.4.4. Documentation and communication

Concerning requirements documentation, startups' major concern was to communicate what should be done. I3 mentioned: *"we are documenting everything even as a communication means among us"*. The main reason to create artifacts was to spread knowledge between developers and also to other people or teams in the organization. Nevertheless, in some startups, tacit knowledge is still vigorous. For instance, I13 mentioned: *"we did not spend much time documenting because information is in people"*.

The degree in which software startups document their requirements varied a lot. Some startups do not keep track of their requirements and just use email communications or contracts (I2), others operate in a strict market that demands complete requirements documentation (I5). Though, generally, startups perform a shallow documentation using simple tools like physical or electronic boards. A common tool used by teams was Trello[5] (I2, I4, I8, I11, I16, I17). Bigger teams also mentioned using issue trackers or agile project management tools (I4, I10, I15). I4 mentioned its experience on a traditional project management (Asana) that did not work out.

5.5. Results assessment

To assess our results, we performed member checking through a survey. Since each startup had its context, it would not be possible to ask all interviewees if all influences were correct. Hence, we created questions to evaluate if the interviewee's startup RE process was represented in the model. Appendix A displays the survey questions and results.

The last survey question was open and asked for comments and suggestions. Based on the analysis of these responses, we improved the figure that represents the model.

I7 raised another point: *"The research is interesting and seems to bring good inputs to the RE process in startups, but in the last question my answer was neutral because I would not know how to start applying this model to improve the process in my startup [...] some examples or material would be interesting explaining how startups could benefit from this result."* The message we sent for the interviewee's validation contained only the explanation of the model without guidelines of how to apply it to a real scenario. The potential improvements based on the model are numerous. For instance, a software startup struggling with its RE

[5] https://www.trello.com.

process could check if a product management team is needed based on their market. Similarly, in case of problems in any activity, the team could try to figure out which influence is harming it and act on the problem.

The basic conclusions one can draw from the interviewee's final feedback, summarized in Table A.5, is that a large majority (91%) of the interviewees believe that the model is good in capturing the reality of RE both in general and in their startups. However, only 45% of them thought that the awareness of the model would be useful in practical terms for their startup's daily life, with 27% of them saying explicitly that it would not be useful. This shows that the model seems to be very successful in advancing scientific knowledge about software startups requirements engineering and partially successful in providing a useful tool for practitioners. Additional future work could focus on resolving this last issue.

6. Discussion

From the results presented, software startups do not execute RE practices in a single way. Instead, they change them according to a set of influences (Founders, Software Development Manager, Developers, Market, Business Model, and Startup Ecosystem). Coleman and O'Connor [30] found a similar result for software development process formation in software startups. The authors recognized as influences the background of the software development manager, the background of founder and market requirements. Zettel et al. [25] already diagnosed the startup process immaturity and Giardino et al. [3] viewed it as an evolutionary approach. Our model confirmed the influences previously discussed in the literature and revealed others such as Developers and Startup Ecosystem.

In comparison to the model proposed by Gralha et al.[34], our results indicate that the maturity of a startup is just one of several conditions that influence how startup teams perform RE activities. Besides that, practices do not follow a linear evolution. Instead, they may begin already in a complex stage because of market requirements, but they can also sometimes regress depending on founders' or developers' opinions. One reason for such difference could be that, in their study, "in most of the studied companies, the business followed a subscription plan", while we interviewed companies operating in different markets with several business models. Finally, our study investigated each of the activities already described in RE literature allowing the comparison between software startups with other contexts.

The following subsection describes a detailed discussion on the influences grounded in our study and Section 6.2 details the practices in software startups and compares them to what the literature for traditional and agile software development teams describes.

6.1. Influences on Requirements engineering process

Founders regulate practices that their startups use especially in elicitation, analysis, validation and prioritization stages. They do that based on their previous experience in the market or other startups and because they are anxious about having a product ready. Seppänen et al. [52] already observed the impact of founders in software startups. They performed interviews with European software startups studying the initial team competencies and found that even when the founder did not do any software development "s/he participated at a higher level: target setting, management, and evaluation". This involvement is a reflection of founders importance in the team and their attitude more as owners than managers [53]. Coleman and O'Connor [30] describe an indirect influence on practices selection rather than an explicit choice made by a founder without a technological background. Our results indicate that this can also happen in a "lower" level, that is, if specific requirements are valid and how they should be prioritized. Besides that, since some software development activities do not create a code output, founders could take them as unnecessary. This problem refers to the newness problem on founding team experience. The founders should, generally, learn new roles [54, 55] including managing software development.

Software development managers, in the end, determine which activities the team will perform in all process steps. Coleman and O'Connor [30] found a similar result: the founder and the software development manager backgrounds will determine the *management style* used in the organization. This style could be: "command and control" or "embrace and empower." Regarding the software development manager, they state: "it was clear that where the software development manager had worked before, what their responsibilities were, what process and process improvement model was

used[...] shaped the process that the software development manager used in their current company". Our results give more pieces of evidence to support these claims.

The market is also a significant influence on all aspects of the process including the activities and even the presence of a product team. Generally, if a market is user-targeted, there will probably be a product team, and all activities will focus to diminish the uncertainty about what the user wants. Meanwhile, in a client-targeted market, the product team was not usually present, and the communication was made directly with some customers, that is, all the activities will somehow be related to these customers. There are also some specific requirements that could determine the practices the team uses. These aspects give more evidence but also expands what Coleman and O'Connor [30] observed. According to the authors, market requirements influence in software startups process formation what several authors [56, 57, 58] discussed it in market-driven requirements literature.

The business model is vital for new ventures success [59]. Given its importance, if it is not mature, all activities will be performed with the goal of reaching a sustainable one. In this case, founders are more present and their influence on practices selection will be substantial. In the other side, if the business model is mature, generally, the requirements are more technical, and founders do not intervene much in practices. Some studies focus on software startup maturity. These two stages are close to the differences observed in a study by Wang et al. [60]. The authors distinguish two main stages for a startup: problem/solution fit when a problem is observed, and a proposed solution is evaluated to check if it solves the problem, and product/market fit when a product development process takes place to deliver a mature product. In another study, Nguyen-Duc et al. [61] proposed a model based on the Cynefin framework. It consists of two different types of activities: hunting (in the chaotic domain) comprising of "generating ideas, elicitation requirements and market and customer development" and gathering related to "requirements description, prototype implementation, automated testing, system integration, and deployment." The authors conclude that in the early phases most of the startup activities are hunting and, as the company evolves, more gathering activities take place. These two models have in common two different stages (although Nguyen-Duc et al. stress that they are simultaneous): one focusing more on elementary questions and doubts about the business model and the other more about focused on a determined task. These two models could be essential tools to explore deeper this problem in future work. Another possibility would be the model proposed by Croll et al. in the Lean Analytics book [62]. The authors suggested another startup maturity model composed of five stages: Empathy, Stickiness, Virality, Revenue, and Scale. When it comes to the reason for developing software, if the software is a product itself, requirements will be close to business perspectives and founders may also influence. Meanwhile, if it is not the case, again, the technical actors determined the practices the team uses.

Developers also have an impact on practices chosen, especially in analysis, validation, and prioritization. They can propose the practices to be used or refuse to use them preventing their adoption. Concerning this aspect, some explanations are possible. First, they might have earned an equity share as an advantage to joining a risky enterprise. Second, it is also hard for startups to attract talents then they are well treated not to leave the company. For instance, I8 and I14 mentioned that talents are rare and really hard to find. The developers' importance in software development practices selection is already present in the literature. Kajko-Mattsson and Niktina [63] tried to improve startups processes and recognized that developers bring their experiences and suggest techniques. Several authors [64, 65] studied people-related factors while migrating to or adopting agile. Hardgrave et al. [66] investigated the determinants of software methodologies adoption and concluded that "an organizational mandate is not sufficient to guarantee use of the methodology in a sustained manner". These results analyzed jointly with ours make reasonable to think about specific capabilities to evolve in developers and technical students that want to be, or already are, part of a startup. This claim is also supported by studies in information systems literature concerning how software developers or managers characteristics influence on organization innovation like individual innovation orientation [67] or understanding of business [68]. Chow and Cao [69] also found a similar result that team environment and team capability are some of key project success factors for agile software development.

The Startup Ecosystem has several aspects, but only those concerning capital, human resources and knowledge availability had an influence detected on

activities, especially, validation, prioritization, and product validation stages. As presented in Section 1, there are several methodologies for startup development in non-scientific literature. Nevertheless, this is not enough for them to be widely adopted. Taking these practices as innovations, we can use the innovation diffusion theory from Rogers [70] to explain it: "most people depend mainly upon a subjective evaluation of an innovation that is conveyed to them from other individuals like themselves who have previously adopted the innovation". Through acceleration programs or other startup community events, startups have contact with other startups and mentors incorporating new techniques and learn from mistakes other startups made. Since all interviewed startups were in the São Paulo startup ecosystem, it is important to assess its characteristics. On a scale to assess software startup ecosystem maturity proposed by Cukier et al. [71], São Paulo is in an Evolving stage. This scale consists of the following steps: Nascent, Evolving, Mature and Self-sustainable. An Evolving stage startup ecosystem has "few successful companies, some regional impact, job generation, and small local economic impact". As a comparison, a Self-Sustainable startup ecosystem like Silicon Valley has "thousands of startups and financial deals, at least a 2nd generation of entrepreneurs mentors, specially angel investors, a strong network of successful entrepreneurs[...] and presence of high-quality technical talent." A strong network and entrepreneurs mentors would facilitate the diffusion of practices towards startups-focused practices and better availability of capital and human resources tackling the other two factors discussed.

6.2. Requirements engineering process

The RE process in software startups presented all activities described in textbooks, but practices used depended on influences above. Besides that, there is an essential actor responsible for some activities present in software startups: the product team.

The product team is responsible for acting as a proxy between a market and the startup, especially in user-targeted startups. They are active usually in elicitation, analysis, validation and prioritization activities. The role performed by this team is similar to the Product Owner in Scrum [72] or product manager role in XP [73]. Initially a one-person job, some authors, like De-Ste-Croix and Easton [74] and Bass [75], already discussed a team performing this role. In the startups' context, Blank [11] recommends two separated teams existing throughout the whole startup history: product development and customer development. The customer development for Blank is what is called the product team by our interviewees. The role of the product team, viewed by the interviewees, is to assist in the customer development process. Founders can also perform this role since they, generally, already have experience on the targeted market as Seppänen et al. [52] mention that the "founder tends to be the sole owner of the innovation and its related competency domains".

Ebert and Brinkkemper [76] evaluated the product manager role in fifteen different organizations worldwide and concluded that an empowered product management role improved project success rates regarding schedule predictability, quality and project duration. They also identified that product managers generally come from a technical background without a formal education in this field as we observed in our interviews. According to the authors, this role ensures "that products have a clear business focus and are not mere feature collections" that was a concern for several of our interviewees. Software product managers cover technical and business activities [77]. In an agile context, if a real customer is not available, they act as surrogates [24]. In this sense, Racheva et al. [78] make a distinction like ours: "while in a custom context, the agile approaches rely on the on-site customer, in a vendor model, it is the product manager". The product owner is not always present in an agile context as Diebold et al. [79] mentioned. In their study, only half of the companies had a product owner in their projects. Although, some have a dedicated department to handle that. In this sense, our results indicate when such department or role generally occurs.

Requirements elicitation is, indeed, a hard task for software startups as shown in the interviews. They continuously use different sources to create a flow of requirements. The product team acts as a surrogate to a big market. Use of prototypes and MVP was also widespread.

In a systematic review of empirical studies on requirements elicitation techniques, Davis et al. [80] had interesting conclusions. They state that "experience has shown that for simple and well-defined problems, elicitation techniques are more or less equivalent" and the most used technique in practice are interviews being the most effective in a wide range of domains and situations. It is interest-

ing, though, that the authors did not find the use of prototypes, very differently from our study. In a very large organization, Werner and Berner [81] found that this phase is performed by the product marketing team that has "substantial market knowledge." In an agile context, Ramesh et al. [24] identify that practitioners prefer face-to-face communication over written specifications as our results also indicate in the software startup context.

First, concerning elicitation techniques, software startups resemble agile teams since they gather requirements throughout the development process instead of a long phase of requirements gathering found in traditional approaches. Specifically, it is also clear the difference between a single or a small group of clients and a market-driven product similar to what Ramesh et al.[24] mention. The difference is that the process is more complicated because even the possible customer does not know what she surely wants. In this sense, startups should use several other techniques besides interviews, which are considered the most-well suited technique for a wide range of problems in traditional or even agile projects. Another difference was prototypes being mentioned several times in opposition to what Ramesh et al. [24] observed in agile contexts. Nevertheless, requirements demanded much more validation even when compared to the ones in agile contexts with market-driven products. The reason is that innovation in the product makes requirements gathering harder, transferring the effort to requirements validation.

On the other hand, requirements analysis and negotiation are shallower as in the agile context. In software startups, though, at this moment, there is a focus on alignment between business and technology aspects. The reason is that the features implemented are critical from a business perspective, frequently influencing the company survival.

As mentioned earlier, the requirements validation is much more crucial and represents more work in software startups than in traditional or even agile projects. In the latter, this is achieved through constant communication with clients or a surrogate, especially, in iteration planning meeting [24]. This fact leads to a stronger validation when compared to traditional RE. Instead, in software startups, even when a product manager or a product team is present, they are not sure, or should not be, about the users' needs or desires. Several tools such as interviews, mock-ups or prototypes are used at this moment to decrease the uncertainty level.

Requirements prioritization is, usually, layered: founders or other higher level manager dictate the overall goals or milestones, and at an operational level, teams decide what to do in smaller cycles like sprints. The activities are also informal as already found in agile and even traditional teams as mentioned in a systematic mapping study [82] and a systematic literature review [83] on the topic. Kukreja et al. [84] identified 17 prioritization techniques most used in the industry, including Analytical Hierarchy Process (AHP), Cost-value approach, MoSCoW, Planning game. Although our interviewees (I4 and I10) only mentioned planning game, several criteria used by the interviewees resemble other techniques: prevent other teams blocking is similar to Cost of Delay, value to the user similar to Cost-Value approach. Nevertheless, these techniques are not performed strictly instead they emerge according to the practitioners' needs without even being mentioned by interviewees. A smaller set of techniques were found mainly the ones also used in agile contexts like Planning game and a MoSCoW like approach. The difference observed in software startups was the input from strategic and business people in prioritization. Again this is motivated by the importance of software development to company survival.

Other works already found the lack of a formal process in other companies. In a multiple case study in two companies focused on requirements prioritization practice challenges, Lehtola et al. [85] concluded that "requirements prioritization is an ambiguous concept and current practices in the companies are informal". Svensson et al. [86] found a similar result. Although the study focuses on the prioritization of quality requirements, nine out of eleven companies analyzed indicated that they perform prioritization in an ad-hoc fashion and seven of them indicated that functional requirements prioritization used the same method as quality ones. The authors also elicit the criteria used by their interviewees: no criterion, customer input, value, and cost. These results are very similar to the findings in this study. Our results can be a particular case of these criteria specific to software startups.

Bakalova et al. [87] performed a similar study focused on agile methods. After interviewing 11 practitioners from ten projects, they concluded that the "prioritization process itself varies significantly in terms of participants involved, prioritization criteria applied, purpose and frequency of the prioritization" and that "the project context has a signifi-

cant impact on the prioritization criteria." Racheva et al. [78] identified techniques used in agile teams which are very similar to the already mentioned studies for traditional teams. They argue that in contrast to traditional development, agile projects have to decide what to implement in each iteration and which requirements will deliver the maximum value to the customers. These constraints are the same for software startups.

Product validation is an after-implementation stage for requirements in software startups related to the validation stage. It is not a part of RE in traditional software development processes. However, it plays an important role in understanding requirements in the software startup context. Especially the use of MVPs in many cases is to understand user needs/requirements better rather than to deliver a product. Startup teams use the features already implemented to learn about user needs especially through metrics. Actually, in Lean Startup methodology [12], the product validation phase is critical: the learning can take place, and the build-measure-learn cycle is closed. Sometimes, users might not use implemented features at all even after the team had spent an extended period (I10 and I14). Through user feedback, new requirements can emerge, e. g., new features, bug fixes or even code refactoring, feed-backing the whole process, that is, giving a cycle-like form to the RE process in software startups.

Nevertheless, startups very often develop something and then realize that users did not want it. Even though, validating assumptions as soon as possible is present in well-known startup development methodologies like Lean Startup [12] and Customer Development [11] and agile methodologies recommend listening to clients. Unawareness of such methodologies would explain that, but even interviewees aware of them still made these mistakes. Startups massively adopt metrics tools in this phase what is essential to consolidate learning.

Documentation is shallow, based on simple tools and more concerned about task descriptions. Such simplicity can become an issue if a person leaves the company or when the team grows and newcomers have to get information, and it is not structured. Valtanen and Ahonen [88] already notice this problem in small software companies. Again this is similar to agile contexts since generally the depth is lower than in traditional approaches as pointed out by studies in the field [24, 89]. In this context, Diebold et al. [79] mentioned JIRA to handle product backlog, but the reliance on oral communication still makes it challenging to trace the evolution of the application [24]. The exceptions, again as in agile contexts, are when the market requires more complex documents like financial or defense markets. Practitioners prefer face-to-face communication and use simple tools to write a list of features rather than describing requirements as our interviews also showed. In general, software developers still use requirements documentation. Power and Moynihan [90] interviewed thirty practitioners and identified differences in requirement documents across different contexts. There are also proposals to improve the documentation process through a lighter process [91] or wikis [92].

6.3. Study validity

The study validity was a concern since the beginning, especially given how hard it was to reach interviewees and the amount of time spent transcribing and analyzing data. Wohlin et al. [93] discuss that threats to validity are of the following types to qualitative studies: construct validity, internal validity, external validity, and reliability. Runeson and Host [94] give good definitions to them. Construct validity reflects if what is investigated represents what the investigator have in mind, for instance, a threat could happen if the interviewee interprets the concepts differently from the researcher. Internal validity concerns causal relations and regards if the investigator is not aware or does not know to what extent a third factor influences the relationship between two factors. External validity concerns to what extent it is possible to generalize the results. Reliability concerns to what extent the results are dependent on the researcher, that is, if another person performs the study, the results would be the same.

The use of semi-structured interviews improved the **construct validity** since it allowed the interviewer to check if the interviewee understood what was expected, interpret what she wanted to say and clarify any doubts. The construct validity was also increased through embracing and comparing with other studies in the literature. We used this technique throughout the discussion especially versus Coleman and O'Connor's study [30] who also identifies influences on the software development process in software startups and Ramesh et al.'s [24] who identified the RE state-of-practice in the agile context. Besides that, the first author, who was

the main responsible for data collection and analysis, is a software startup partner with good experience in software development. This allowed him to use or understand the terms used by interviewees. Nevertheless, his prior knowledge and preconceptions about the topic could bias the results. The use of well-known techniques and a constant discussion with the second author, an academic researcher without any startup experience, mitigated this threat.

In terms of **internal validity**, the amount of the data, the choice of analyzing all data and constant comparison until theoretical saturation, inherent to GT, reduced potential threats to it. Constant updates on the interview guide allowed us to ask added questions to the following interviewees. Besides that, the results assessment phase allowed us to ask interviewees if they agreed with our results.

The use of well-known techniques together with a detailed description of data collection analysis improved the **reliability**. The opportunistic approach used to reach the interviewees could be a threat to **external validity**. However, in the end, the studied startups operate in distinct markets, have different sizes and maturity stages. However, given that the interviewees worked for or founded software startups in the São Paulo region, it is not possible to generalize the results to all software startups. Further studies in other ecosystems should be performed to allow such generalization.

Jantunen and Gause [44] describe some techniques to assess grounded theory studies: prolonged engagement, triangulation, peer debriefing, referential adequacy, and member checking. Since data collection and analysis took almost a year considering the three stages, it achieved a prolonged engagement. Although not possible in all cases, the preference for interviews in the startup workplace provided data triangulation through field notes. Peer debriefing consisted of a constant discussion among the authors and publishing a paper after the first stage end in a software startups research workshop [15]. For referential adequacy, we recorded all interviews, transcribed them and saved the analysis in AtlasTI. Finally, we performed member checking in the results assessment phase. Since this stage was performed two years after the data collection, a potential threat is interviewee's memory loss of what had been discussed. Nevertheless, we asked broad questions that would apply in different scenarios.

7. Conclusions and Future work

This study improved the understanding of RE practices in software startups. Using a three-stage iterative research process, through 17 interviews with 23 people covering more than 30 software startups from the region of São Paulo, Brazil, operating in different markets, the study unveiled the employed practices and a list of factors that influence how startup teams choose these practices. Our results indicate that software startups do not follow a single set of practices; instead, they are dictated by a set of influences (Founders, Software Development Manager, Developers, Market, Business Model, and Startup Ecosystem). We presented how each of these context characteristics influences RE activities. In particular, we proposed a classification for software startups product in user and client-targeted based that improves the well-known B2B/B2C that showed to be insufficient to understand RE in software startups. Moreover, we discussed how the business model maturity influences the use of validation techniques and mediate the control of founders over RE practices. The paper also presents a comparison between startups and the state-of-practice described in RE literature to teams using traditional or agile approaches. In summary, software startups employ practices similar to the ones used by agile contexts, mainly, in analysis and documentation that are also faster and smaller. Elicitation, prioritization, and validation are also similar, but they are influenced by the general lack of an accessible customer found in agile contexts and the increasing importance of software development activities to the company survival.

Several future studies could be performed based on these results. First of all, the presented model needs to be validated in a systematic manner. The model was built following the Grounded Theory approach which ensures that the model is grounded in empirical evidence. Future validation could follow a more deductive approach, using a survey or focused group with experts to systematically examine every element in the model. Another valuable future study would be to replicate this study in different startup ecosystems to analyze the differences in practices besides the possibility of other startup ecosystem aspects also influence the practices performed. Studies could further explore the factor of founders effect, and a possible correlation between this influence and startup success. The developers' experience level could also be further

investigated to check if more senior employees have a more significant impact on practices selection. A similar study could focus on product management teams existence. This study did not focus on non-functional requirements, which have been neglected by agile practitioners but some practices have been proposed to tackle this problem [23]. Future work could focus on this topic in software startups. Another interesting topic would be the use of big data and machine learning to guide software development. A recent trend that definitely could influence how RE is performed in software startups.

Appendix A. Assessment survey

Table A.5 presents assessment survey questions.

References

[1] M. Unterkalmsteiner, P. Abrahamsson, A. Nguyen-duc, G. H. Baltes, K. Conboy, D. Dennehy, R. Sweetman, H. Edison, S. Shahid, X. Wang, J. Garbajosa, T. Gorschek, L. Hokkanen, I. Lunesu, M. Marchesi, L. Morgan, C. Selig, M. Oivo, S. Shah, F. Kon, Software Startups - A Research Agenda, e-Informatica Software Engineering Journal 10 (1) (2016) 1–28. doi:10.5277/e-Inf160105. 1, 5

[2] M. Gutbrod, J. Münch, M. Tichy, How Do Software Startups Approach Experimentation? Empirical Results from a Qualitative Interview Study, Vol. 10611 of Lecture Notes in Computer Science, Springer International Publishing, Cham, 2017, pp. 297–304. doi:10.1007/978-3-319-69926-4_21. 1

[3] C. Giardino, N. Paternoster, M. Unterkalmsteiner, T. Gorschek, P. Abrahamsson, Software Development in Startup Companies: The Greenfield Startup Model, IEEE Transactions on Software Engineering 42 (6) (2016) 585–604. doi:10.1109/TSE.2015.2509970. 1, 17

[4] B. Nuseibeh, S. Easterbrook, Requirements engineering: a roadmap, ICSE 2000 Proceedings of the Conference on The Future of Software Engineering 1 (2000) 35–46. doi:10.1145/336512.336523. 1, 3

[5] G. Kotonya, I. Sommerville, Requirements Engineering: Processes and Techniques, John Wiley & Sons, Inc., New York, NY, USA, 1998. 1, 3, 15, 16

[6] N. Paternoster, C. Giardino, M. Unterkalmsteiner, T. Gorschek, P. Abrahamsson, Software development in startup companies: A systematic mapping study, Information and Software Technology 56 (10) (2014) 1200–1218. doi:10.1016/j.infsof.2014.04.014. 2, 3, 4

[7] C. Giardino, X. Wang, P. Abrahamsson, Why early-stage software startups fail: A behavioral framework, in: C. Lassenius, K. Smolander (Eds.), Lecture Notes in Business Information Processing, Vol. 182 LNBIP of Lecture Notes in Business Information Processing, Springer International Publishing, Cham, 2014, pp. 27–41. arXiv:1709.04749, doi:10.1007/978-3-319-08738-2. 2

[8] R. Chanin, L. Pompermaier, K. Fraga, A. Sales, R. Prikladnicki, Applying Customer Development for Software Requirements in a Startup Development Program, Proceedings - 2017 IEEE/ACM 1st International Workshop on Software Engineering for Startups, SoftStart 2017 (2017) 2–5doi:10.1109/SoftStart.2017.3. 2, 4

[9] E. Klotins, M. Unterkalmsteiner, T. Gorschek, Software engineering in start-up companies: An analysis of 88 experience reports, Empirical Software Engineering (2018) 1–19doi:10.1007/s10664-018-9620-y. 2, 4, 5

[10] B. H. C. Cheng, J. M. Atlee, M. Joanne, Research Directions in Requirements Engineering, Proceeding FOSE '07 2007 Future of Software Engineering (2007) 285–303doi:10.1109/FOSE.2007.17. 2, 3

[11] S. Blank, The four steps to the epiphany: successful strategies for products that win, BookBaby, 2013. 2, 3, 19, 21

[12] E. Ries, The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses, The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses, Crown Business, 2011. 2, 3, 21

[13] S. Blank, Newtv is the antithesis of a lean startup. can it work?, https://hbr.org/2018/08/newtv-is-the-antithesis-of-a-lean-startup-can-it-work (2018). 2

[14] A. L. Strauss, J. M. Corbin, Basics of qualitative research: grounded theory procedures and techniques, Sage Publications, 1990. 2, 5, 8

[15] J. Melegati, A. Goldman, Requirements Engineering in software startups : a grounded theory approach, in: 2nd International Workshop on Software Startups, 2016. 2, 4, 22

[16] D. Cukier, F. Kon, A maturity model for software startup ecosystems, Journal of Innovation and Entrepreneurship 7 (1) (2018) 14.
URL https://doi.org/10.1186/s13731-018-0091-6 2, 13

[17] S. M. Sutton, The Role of Process in a Software Startup, IEEE Software (2000) 33–39. 3

[18] V. Berg, J. Birkeland, A. Nguyen-Duc, I. Pappas, L. Jaccheri, Software Startup Engineering: A Systematic Mapping Study, Journal of Systems and Softwaredoi:10.1016/j.jss.2018.06.043. 3, 4

[19] L. Alpkan, C. Bulut, G. Gunday, G. Ulusoy, K. Kilic, Organizational support for intrapreneurship and its interaction with human capital to enhance innovative performance, Vol. 48, 2010. doi:10.1108/02517471080000697. 3

[20] K. Reuther, C.-A. Schumman, Intrapreneurship : Increasing Employees' Responsibility for an Enhancement of Innovation Performance, in: Engineering, Technology and Innovation (ICE) \& IEEE International Technology Management Conference, 2016 International Conference on, 2016, pp. 147–149. 3

[21] J. Siddiqi, Requirement engineering: The emerging wisdom [Guest Editor's introduction], IEEE Software 13 (2) (1996) 15. doi:10.1109/MS.1996.506458. 3

[22] A. Eberlein, J. Leite, Agile Requirements Definition: A View from Requirements Engineering, International Workshop on Time-Constrained Requirements Engineering, TCRE 2002 (2002) 1–5. 4

Table A.5
Questions and results of the member checking survey.

Question	Totally agree	Agree	Neither agree nor disagree	Disagree	Totally disagree
The set of influences that determine RE practices in the model is representative of the reality.	7	3	0	1	0
The RE activities executed in my startup are represented in the model.	7	3	0	1	0
In general, requirements in software startups do not follow a cyclical flow, that is, new requirements could be created or come back to previous steps (like analysis or validation) even after they are implemented.	8	3	0	0	0
The presence (or absence) of a person or team responsible by product management in my startup can be explained by the model.	1	5	4	0	1
The model will be useful to understand and change the way a software startup execute RE activities.	2	3	3	3	0

[23] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, S. Shamshirband, A systematic literature review on agile requirements engineering practices and challenges, Computers in Human Behavior 51 (0) (2015) –. doi:10.1016/j.chb.2014.10.046. 4, 23

[24] B. Ramesh, L. Cao, R. Baskerville, Agile requirements engineering practices and challenges: an empirical study, Information Systems Journal 20 (5) (2010) 449–480. doi:10.1111/j.1365-2575.2007.00259.x. 4, 19, 20, 21

[25] J. Zettel, F. Maurer, J. Münch, L. Wong, LIPE: a lightweight process for e-business startup companies based on extreme programming, Product Focused Software Process Improvement (2001) 255–270doi:10.1007/354044813623. 4, 17

[26] J. L. Mater, B. Subramanian, Solving the software quality management problem in Internet startups, in: Eighteenth Annual Pacific Northwest Software Quality Conference, Proceedings, 2000, pp. 297–306. 4

[27] M. Crowne, Why software product startups fail and what to do about it. Evolution of software product development in startup companies, IEEE International Engineering Management Conference 1 (2002) 338–343. doi:10.1109/IEMC.2002.1038454. 4

[28] B. May, Applying lean startup: An experience report - Lean & lean UX by a UX veteran: Lessons learned in creating & launching a complex consumer app, Proceedings - 2012 Agile Conference, Agile 2012 (2012) 141–147doi:10.1109/Agile.2012.18. 4

[29] C. Midler, P. Silberzahn, Managing robust development process for high-tech startups through multi-project learning: The case of two European start-ups, International Journal of Project Management 26 (5) (2008) 479–486. doi:10.1016/j.ijproman.2008.05.003. 4

[30] G. Coleman, R. V. O'Connor, An investigation into software development process formation in software start-ups, Journal of Enterprise Information Management 21 (6) (2008) 633–648. doi:10.1108/17410390810911221. 4, 5, 9, 10, 17, 18, 21

[31] U. Rafiq, S. S. Bajwa, X. Wang, I. Lunesu, Requirements elicitation techniques applied in software startups, Proceedings - 43rd Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2017 (2017) 141–144doi:10.1109/SEAA.2017.73. 4

[32] D. L. Frederiksen, A. Brem, How do entrepreneurs think they create value? A scientific reflection of Eric Ries' Lean Startup approach, International Entrepreneurship and Management Journal 13 (1) (2017) 169–189. doi:10.1007/s11365-016-0411-x. 4

[33] N. Tripathi, E. Klotins, R. Prikladnicki, M. Oivo, L. Pompermaier, M. Arun Sojan Kudakacheril, A., Unterkalmsteiner, K. Liukkunen, T. Gorschek, An anatomy of requirement engineering in software startups using multi-vocal literature and case survey, Journal of Systems and Software 146 (2018) 130–151. doi:10.1016/J.JSS.2018.08.059. 4

[34] C. Gralha, D. Damian, A. I. T. Wasserman, M. Goulão, J. Araújo, The evolution of requirements practices in software startups, in: Proceedings of the 40th International Conference on Software Engineering - ICSE '18, ACM Press, New York, New York, USA, 2018, pp. 823–833. doi:10.1145/3180155.3180158. 4, 17

[35] C. B. Seaman, Qualitative methods in empirical studies of software engineering, IEEE Transactions on Software Engineering 25 (4) (1999) 557–572. doi:10.1109/32.799955. 5

[36] C. Urquhart, H. Lehmann, M. D. Myers, Putting the 'theory' back into grounded theory: Guidelines for

grounded theory studies in information systems, Information Systems Journal 20 (2010) 357–381. doi:10.1111/j.1365-2575.2009.00328.x. 5

[37] B. G. Glaser, A. L. Strauss, The discovery of grounded theory: strategies for qualitative theory, New Brunswick: Aldine Transaction. 5

[38] K.-j. Stol, P. Ralph, B. Fitzgerald, Grounded theory in software engineering research, Proceedings of the 38th International Conference on Software Engineering - ICSE '16 (Aug) (2016) 120–131. doi:10.1145/2884781.2884833. 5

[39] M. Daneva, D. Damian, A. Marchetto, O. Pastor, Empirical research methodologies and studies in Requirements Engineering: How far did we come?, Journal of Systems and Software 95 (2014) 1–9. doi:10.1016/j.jss.2014.06.035. 5

[40] S. Easterbrook, J. Singer, M.-a. Storey, D. Damian, Selecting Empirical Methods for Software Engineering Research, in: Guide to Advanced Empirical Software Engineering, Springer London, London, 2008, pp. 285–311. doi:10.1007/978-1-84800-044-5_11. 5

[41] R. S. Weiss, Learning from strangers: The art and method of qualitative interview studies, Simon and Schuster, 1995. 5, 7

[42] The global startup ecosystem ranking 2015, Tech. rep., Startup Compass Inc. (2015). 7

[43] K. M. Eisenhardt, Building theories from Case Study Research, The Academy of Management Review 14 (4) (1989) 532–550. arXiv:z0037, doi:10.2307/258557. 8

[44] S. Jantunen, D. C. Gause, Using a grounded theory approach for exploring software product management challenges, Journal of Systems and Software 95 (2014) 32–51. doi:10.1016/j.jss.2014.03.050. 9, 22

[45] R. Buchanan, Wicked Problems in Design Thinking, Design Issues 8 (2) (1992) 5. doi:10.2307/1511637. 10

[46] H. W. J. Rittel, M. M. Webber, Dilemmas in a general theory of planning, Policy Sciences 4 (2) (1973) 155–169. arXiv:arXiv:1011.1669v3, doi:10.1007/BF01405730. 10

[47] T. Brown, Change by design: How design thinking creates new alternatives for business and society, Collins Business, 2009. 10

[48] K. Swani, B. P. Brown, G. R. Milne, Should tweets differ for b2b and b2c? an analysis of fortune 500 companies' twitter communications, Industrial Marketing Management 43 (5) (2014) 873 – 881. doi:10.1016/j.indmarman.2014.04.012. 12

[49] M. Subramani, E. Walden, The impact of e-commerce announcements on the market value of firms, Information Systems Research 12 (2) (2001) 135–154. 12

[50] S. Hoejmose, S. Brammer, A. Millington, "green" supply chain management: The role of trust and top management in b2b and b2c markets, Industrial Marketing Management 41 (4) (2012) 609–620. 12

[51] Y. Motoyama, K. Knowlton, Examining the Connections within the Startup Ecosystem: A Case Study of St. Louis, Entrepreneurship Research Journal 7 (1). arXiv:35, doi:10.1515/erj-2016-0011. 13

[52] P. Seppänen, M. Oivo, K. Liukkunen, The initial team of a software startup, in: Engineering, Technology and Innovation (ICE) / IEEE International Technology Management Conference, 2016 International Conference on, 2016, pp. 57–65. 17, 19

[53] M. J. Fern, L. B. Cardinal, H. M. O'Neill, The genesis of strategy in new ventures: escaping the constraints of founder and team knowledge, Strategic Management Journal 33 (4) (2012) 427–447. arXiv:1, doi:10.1002/smj.1944. 17

[54] F. Delmar, S. Shane, Does experience matter? The effect of founding team experience on the survival and sales of newly founded ventures, Strategic Organization 4 (3) (2006) 215–247. doi:10.1177/1476127006066596. 17

[55] F. Muñoz-Bullon, M. J. Sanchez-Bueno, A. Vos-Saz, Startup team contributions and new firm creation: the role of founding team experience, Entrepreneurship and Regional Development 27 (2015) 80–105. doi:10.1080/08985626.2014.999719. 17

[56] B. Regnell, S. Brinkkemper, Market-driven requirements engineering for software products, Engineering and Managing Software Requirements (2005) 287–308doi:10.1007/354028244013. 18

[57] C. Potts, Invented requirements and imagined customers: requirements engineering for off-the-shelf software, Proceedings of 1995 IEEE International Symposium on Requirements Engineering (RE'95) (1995) 128–130doi:10.1109/ISRE.1995.512553. 18

[58] L. Karlsson, Å. G. Dahlstedt, B. Regnell, J. Natt och Dag, A. Persson, Requirements engineering challenges in market-driven software development - An interview study with practitioners, Information and Software Technology 49 (2007) 588–604. doi:10.1016/j.infsof.2007.02.008. 18

[59] C. Zott, R. Amit, Business model design: An activity system perspective, Long Range Planning 43 (2-3) (2010) 216–226. doi:10.1016/j.lrp.2009.07.004. 18

[60] X. Wang, H. Edison, S. S. Bajwa, C. Giardino, P. Abrahamsson, Key Challenges in Software Startups Across Life Cycle Stages, in: Lecture Notes in Business Information Processing, Vol. 179 LNBIP, 2016, pp. 169–182. arXiv:9780201398298, doi:10.1007/978331933515514. 18

[61] A. Nguyen-Duc, P. Seppänen, P. Abrahamsson, Hunter-gatherer cycle: a conceptual model of the evolution of software startups, Proceedings of the 2015 International Conference on Software and System Process - ICSSP 2015 (Idi) (2015) 199–203. doi:10.1145/2785592.2795368. 18

[62] A. Croll, B. Yoskovitz, Lean Analytics: Use Data to Build a Better Startup Faster, Lean (O'Reilly), O'Reilly Media, Incorporated, 2013. 18

[63] M. Kajko-Mattsson, N. Nikitina, From Knowing Nothing to Knowing a Little: Experiences Gained from Process Improvement in a Start-Up Company, 2008 International Conference on Computer Science and Software Engineering (October 2007) (2008) 617–621. doi:10.1109/CSSE.2008.1370. 18

[64] S. Nerur, R. Mahapatra, G. Mangalaraj, Challenges of Migrating to Agile Methodologies, Communications of the ACM 48 (2) (2005) 72–78. arXiv:0704.1294, doi:10.1145/1060710.1060712. 18

[65] L. R. Vijayasarathy, D. Turk, Agile software development: A survey of early adopters, Journal of Information Technology Management XIX (2) (2008) 1–8. 18

[66] B. Hardgrave, F. D. Davis, C. K. Riemenschneider, Investigating Determinants of Software Developers' Intentions to Follow Methodologies, Journal of Management Information Systems 20 (1) (2003) 123–151. doi:10.1080/07421222.2003.11045751. 18

[67] S. Nambisan, Software firm evolution and innovation-

orientation, Journal of Engineering and Technology Management - JET-M 19 (2) (2002) 141–165. doi:10.1016/S0923-4748(02)00007-3. 18

[68] S. R. Gordon, M. Tarafdar, How do a company's information technology competences influence its ability to innovate?, Journal of Enterprise Information Management 20 (3) (2007) 271–290. doi:10.1108/17410390710740736. 18

[69] T. Chow, D.-B. Cao, A survey study of critical success factors in agile software projects, Journal of Systems and Software 81 (6) (2008) 961–971. doi:10.1016/j.jss.2007.08.020. 18

[70] E. M. Rogers, Diffusion of innovations, Simon and Schuster, 2010. 19

[71] D. Cukier, F. Kon, N. Krueger, Designing a Maturity Model for Software Startup Ecosystems, in: Product-Focused Software Process Improvement, Springer, 2015, pp. 600–606. doi:10.1007/978331926844645. 19

[72] K. Schwaber, Agile Project Management with Scrum, Best practices, Microsoft Press, 2004. 19

[73] K. Beck, C. Andres, Extreme Programming Explained: Embrace Change, The XP series, Addison-Wesley, 2005. 19

[74] A. De-Ste-Croix, A. Easton, The product owner team, Proceedings - Agile 2008 Conference (2008) 274–279doi:10.1109/Agile.2008.94. 19

[75] J. M. Bass, Agile method tailoring in distributed enterprises: Product owner teams, Proceedings - IEEE 8th International Conference on Global Software Engineering, ICGSE 2013 (2013) 154–163doi:10.1109/ICGSE.2013.27. 19

[76] C. Ebert, S. Brinkkemper, Software product management - An industry evaluation, Journal of Systems and Software 95 (2014) 10–18. doi:10.1016/j.jss.2013.12.042. 19

[77] A. Maglyas, U. Nikula, K. Smolander, What are the roles of software product managers? An empirical investigation, Journal of Systems and Software 86 (12) (2013) 3071–3090. doi:10.1016/j.jss.2013.07.045. 19

[78] Z. Racheva, M. Daneva, L. Buglione, Supporting the Dynamic Reprioritization of Requirements in Agile Development of Software Products, IEEE International Workshop on Software Product Management, 2008. IWSPM '08. (i) (2008) 49–58. doi:10.1109/IWSPM.2008.7. 19, 21

[79] P. Diebold, J.-P. Ostberg, S. Wagner, U. Zendler, What Do Practitioners Vary in Using Scrum?, in: C. Lassenius, T. Dingsøyr, M. Paasivaara (Eds.), Lecture Notes in Business Information Processing, Vol. 212 of Lecture Notes in Business Information Processing, Springer International Publishing, Cham, 2015, pp. 40–51. doi:10.1007/978-3-319-18612-24. 19, 21

[80] A. Davis, O. Dieste, A. Hickey, N. Juristo, A. Moreno, Effectiveness of Requirements Elicitation Techniques: Empirical Results Derived from a Systematic Review, in: 14th IEEE International Requirements Engineering Conference (RE'06), IEEE, 2006, pp. 179–188. doi:10.1109/RE.2006.17. 19

[81] C. M. Werner, D. M. Berry, An Empirical Study of the Software Development Process, Including Its Requirements Engineering, at Very Large Organization: How to Use Data Mining in Such a Study, Vol. 809 of Communications in Computer and Information Science, Springer Singapore, Singapore, 2018, pp. 15–25. doi:10.1007/978-981-10-7796-82. 20

[82] M. Pergher, B. Rossi, Requirements prioritization in software engineering: A systematic mapping study, 2013 3rd International Workshop on Empirical Requirements Engineering, EmpiRE 2013 - Proceedings (2013) 40–44doi:10.1109/EmpiRE.2013.6615215. 20

[83] P. Achimugu, A. Selamat, R. Ibrahim, M. N. R. Mahrin, A systematic literature review of software requirements prioritization research, Information and Software Technology 56 (6) (2014) 568–585. doi:10.1016/j.infsof.2014.02.001. 20

[84] N. Kukreja, S. S. Payyavula, B. Boehm, S. Padmanabhuni, Selecting an appropriate framework for value-based requirements prioritization, in: 2012 20th IEEE International Requirements Engineering Conference (RE), IEEE, 2012, pp. 303–308. doi:10.1109/RE.2012.6345819. 20

[85] L. Lehtola, M. Kauppinen, S. Kujala, Requirements Prioritization Challenges in Practice, 2004, pp. 497–508. doi:10.1007/978354024659636. 20

[86] R. B. Svensson, T. Gorschek, B. Regnell, R. Torkar, A. Shahrokni, R. Feldt, A. Aurum, Prioritization of quality requirements: State of practice in eleven companies, Proceedings of the 2011 IEEE 19th International Requirements Engineering Conference, RE 2011 (2011) 69–78doi:10.1109/RE.2011.6051652. 20

[87] Z. Bakalova, M. Daneva, A. Herrmann, R. Wieringa, Agile requirements prioritization: What happens in practice and what is described in literature, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 6606 LNCS (2011) 181–195. arXiv:9780201398298, doi:10.1007/978364219858818. 20

[88] A. Valtanen, J. J. Ahonen, Big improvements with small changes: Improving the processes of a small software company, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 5089 LNCS (2006) (2008) 258–272. doi:10.1007/978354069566022. 21

[89] F. Paetsch, A. Eberlein, F. Maurer, Requirements engineering and agile software development, WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. (2003) 308–313doi:10.1109/ENABL.2003.1231428. 21

[90] N. Power, T. Moynihan, A theory of requirements documentation situated in practice, Proceedings of the 21st annual international conference on Documentation - SIGDOC '03 (2003) 86doi:10.1145/944885.944887. 21

[91] Z. Zhang, Towards Lightweight Requirements Documentation, Journal of Software Engineering and Applications 03 (09) (2010) 882–889. doi:10.4236/jsea.2010.39103. 21

[92] B. Decker, E. Ras, J. Rech, P. Jaubert, M. Rieth, Wiki-Based stakeholder participation in requirements engineering, IEEE Software 24 (2) (2007) 28–35. doi:10.1109/MS.2007.60. 21

[93] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson, B. Regnell, A. Wessln, Experimentation in Software Engineering, Springer Publishing Company, Incorporated, 2012. 21

[94] P. Runeson, M. Höst, Guidelines for conducting and reporting case study research in software engineering, Empirical Software Engineering 14 (2) (2009) 131–164. arXiv:9809069v1, doi:10.1007/

s10664-008-9102-8. 21