Requirements Engineering in Software Startups: a Grounded Theory approach

Jorge Melegati and Alfredo Goldman
Department of Computer Science
University of São Paulo
São Paulo, Brazil
melegati@ime.usp.br, gold@ime.usp.br

***Abstract*—Software startups face a very demanding market: they must deliver high innovative solutions in the shortest possible time. Resources are limited and time to reach market is short then it is extremely important to provide the right requirements. Nevertheless, software requirements are usually not clear and startups struggle to know what they should develop. This context affects how requirements engineering processes in these organizations are executed. This work focus on how these processes are executed in software startups and which effects context has on them. Nine interviews were conducted with founders or managers of different Brazilian software startups that operate in several market sectors and have different maturity level. Data was analyzed using Grounded Theory techniques such open and axial coding. As a result, a conceptual model of requirements engineering state-of-practice for software startups was developed and influences from the context were identified and described.**

Keywords*—requirements engineering; software development; software startups; grounded theory

I. Introduction

Startups are in the midst of the technology revolution that took place in the world for the last fifty years. In this high changing environment, software requirements are difficult to extract and they change frequently [1]. Nevertheless, the capacity of innovative solutions delivery is very important for their success [2]. This effect got more apparent after the "dot.com" bubble burst in the beginning of the years 2000s. Since then, from used practices criticism, some new startup development methodologies emerged in the industry like Customer Developer [3] and Lean Startup [4]. These new techniques are presented through the description of practical experiences and examples of their use, like pointed by Patz [5] there is a gap between theory and practice. According to Paternoster et al. [1], there are few evidences on the use of these techniques in literature. Besides that, more specific software requirements engineering techniques would be very welcome for startups since they must handle several stakeholders (users, founders, investors, developers, etc.) with limited time and human resources.

A detailed description of software requirements engineering state-of-practice would make possible the evolution of these methodologies or even the creation of new practices that aim a better efficiency in startup creation and management.

Paternoster et al. [1] break their software startups systematic mapping in some sections and one of them is specific for requirements engineering. The authors acknowledge that requirements engineering practices:

- are limited to some key practices;
- present an increasing difficulty because of a growing number of users;
- demand, from several authors perspectives, more user involvement.

Besides that, the authors have not found any study focused on software requirements engineering practices. Given this context, the research question that will guide this study is:

How requirements engineering practices are used in software startups?

Then, this research main objective is to develop a requirements engineering model of state-of-practice in software startups.

As pointed out by Blank [15], the traditional way a new product was launched could only end up as a total success or a big fail. This formula consisted in developing a business plan, present it to potential investors, organize a team, build the product and, then, sell it. The "dot.com" bubble burst raised questions about this traditional formula and alternative methodologies to develop a product were proposed like Customer Development [3] and Lean Startup [4].

Given the methodologies used by startups, a secondary objective is derived: verify if these methodologies are used and check if they are well adapted. And finally to understand how this is done.

A. Startup definition

There are several definitions for what a startup is in literature [1]. In this context, it is important to make clear which definition is used for this research. Sutton [6] proposed one definition based on some characteristics: little or no operation, limited resources, multiple influences and dynamic technologies and markets. After several studies analysis, Paternoster et al. [1] expand this definition adding some aspects: innovation, fast growth, time pressure, third party dependency, focus on one product and flat organizational structures.

Although several other studies use definitions similar to this one, it can leave out some important organizations: small tech companies that received a huge investment or project teams inside a bigger corporation developing an innovative product. These organizations face similar problems and are in similar contexts. Then, we preferred to use Ries' definition: “a startup is a human institution designed to deliver a new product or service under conditions of extreme uncertainty”. This definition covers innovation, time pressure and focus on only a product that influences requirements engineering.

B. Requirements engineering

According to Nuseibeh and Easterbrook [7], “the primary measure of success of a software system is the degree to which it meets the purpose for which it was intended”. Then, the authors 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. Kotonya and Sommerville [8] enumerate requirements engineering activities: elicitation; analysis and negotiation; documenting and validation.

Requirements elicitation is the first stage of requirements engineering and its main objectives are to find out the problem to be solved and, then, define system boundaries. The first task is to identify system stakeholders. Generally, they are clients, system users and developers. This task can be extremely hard for startups that focus a large group of people like one that is building a website or an app. And through interaction with these stakeholders, it is possible to define which requirements will be developed.

During elicitation stage, several stakeholders provide different requirements. Unavoidably, several inconsistencies appear: different sources conflict, lack of necessary details to implementation or inadequacy to project scope. Then, it is necessary a stage to analyze and detail requirements and negotiation with stakeholders to what requirements will be implemented.

The requirements, once selected, are then documented to be monitored during development.

Requirements should also be validated before their implementation to check if they are complete and consistent. It is also important to define tests to verify if a requirement has accomplished its objectives after implementation. This stage is called validation.

Kotonya and Sommerville [8] recognize that different organizations execute requirements engineering in different ways but they provide an abstract model as described in Fig 1.

Given the context both on startups definition and on the basis of requirements follows the paper organization. In Section 2, the related works are presented. In Section 3, the research methodology is described including data collection and analysis techniques chosen and the reasons for that. In Section 4, the study results are presented including a discussion on threats to validity. Finally, in Section 5, a conclusion is presented.

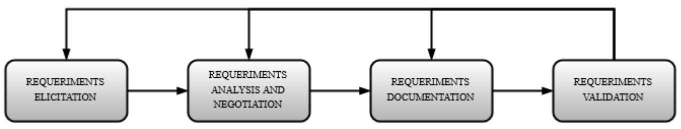
REQUIREMENTS ELICITATION
REQUIREMENTS ANALYSIS AND NEGOTIATION
REQUIREMENTS DOCUMENTATION
REQUIREMENTS VALIDATION

Fig. 1. Traditional requirements engineering process model

II. RELATED WORK

Coleman and O'Connor [10] employed Grounded Theory to study software development process formation in Irish software startups. The authors developed a theoretical framework with categories and their relations to explain software development process in startups. The background of software development manager determines the process model that will be the basis for the activities and this process will be tailored.

The software development manager is a founder or someone else hired in the beginning of company activities. Besides techniques, the manager brings also a management style that could be: “command and control” when there is no trust on developers and they should be constantly observed and analyzed or “embrace and empower” when developers perform their tasks with more independence because of a greater level of trust. The management style is also influenced by the background of founder(s). The software development process is then formed by the management style within a tailored process based on a methodology.

The authors also conclude that most of processes are based on industry standard models like RUP [22], waterfall [23] or XP [21]. The process is then tailored to suit organization reality, generally, removing practices but it is also possible to add new practices. These changes create a unique process adapted to the organization context including the market into it is.

Furthermore, the authors also conclude that agile methodologies have a lot to offer to startups since they are product-driven, work for small development teams and are led by developers. Taipale [11] also considers agile methodologies as the most viable process for startups since they embrace change instead of avoiding it.

Requirements engineering is closed related to software startups business activities. This is also important in agile methodologies where there are roles directly related to business like the product owner in Scrum [12] or the client always available on Extreme Programming [13]. In this sense, it is important to discuss software product management, which is the discipline and business process that guide a product since inception to its delivery [14]. Then, it is important to analyze which software management techniques are being used by startups.

Our intent is to reduce the gap between theory and practice providing a detailed study on what actually happens concerning software requirements on some current software startups.

III. Research methodology

This research aim is totally related to find out how a process or processes set (requirements engineering) is performed in a determined context (software startups). That is, it tries to understand a human behavior. According to Seaman [16], human behavior is one of few phenomena demanding qualitative methods to be studied. Hence, a qualitative research is a natural choice.

Given the research exploratory bias, it is important that the data collection techniques used in this study foster new facts not based on the researcher's expectation. Therefore, interviews are a well-suited option. Still under this perspective, semi-structured interviews are chosen because it is possible to let the interviewee tell more details and novel facts but, using an interview guide, restrict unnecessary digressions that could make interviews too long and miss important elements.

To perform the data analysis it was chosen as a support the techniques developed under Grounded Theory. Grounded theory was developed by Strauss and Glaser and, according to them, is the discovery of a theory based on data systematically obtained and analyzed in social research [17]. After a book published by Strauss, now in cooperation with Corbin, there was a clear divergence between both initial authors that created two streams. This research is based on the methodology and techniques described by Strauss and Corbin [18].

The authors describe the four stages of the codification process: open coding, axial coding, selective coding and coding for process. This research used only open and axial coding. The codification process iterative idea was also applied. Then, several phases consisting of open and axial coding will be done until theoretical saturation, when new interviews do not bring any novelty, is reached. In this paper, only the results from the first iteration will be described.

A. Interview guide

The interview guide was developed consisting of three parts. First, it was important to understand the interviewee background, level of knowledge about software development and startup development methodologies and role in the startup and the startup itself: a little about its history and its product. After that, the main part consisted of four open questions about each requirements engineering stage. And finally a feedback from the interviewee to add any point wanted and about the interview itself.

B. Data analysis

Open coding was made with AtlasTI tool [24] support. AtlasTI is a commercial tool developed to help qualitative research handling data and coding. The tool was used to add labels to interviews transcriptions and also group labels into categories. This was very useful because after labeling transcribed interviews there were more than two hundred labels. Then, an iterative process took place to refine the labels and categories: some of them were removed, new were created, others were merged. This process happened due to a growing understanding about the data by the analyst. This way a group of categories with their properties and categories emerged. To picture the categories and identify their relationships, a white board was used to draw categories. A white board was chosen because it was possible to erase and draw easily and this would be done quicker than in software. As a result of viewing the emerging model, new changes to categories and labels were made. A picture of the whiteboard is shown in Fig. 2.

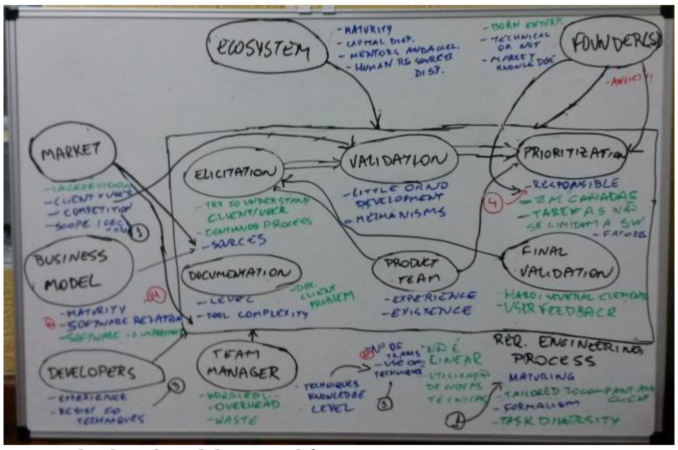

Fig 2. Whiteboard used during codification

IV. Results

A. Interviews

The interviewee's list was created opportunistically. It was created based on first author contacts and other university students who own startups. We also used a snowball approach to get other possible interviewees. They were then contacted and arrangements were made to perform the interview. In this first stage, nine interviews were made, recorded and then transcribed. They are shown on Table I. The number of interviews should not be too small because data is necessary and from different contexts and not too big because of the iterative idea. After nine interviews, we had enough data from different contexts and started coding. Interviews were made face-to-face or through a call depending on interviewee availability. In both cases, they were recorded using two recorders to avoid any data loss. They took between 25 and 45 minutes and, most of the time, followed the interview guide. The interviewee was set free to talk and, sometimes, the interviewee started to talk about another point but the interview followed then the guide was used to come back to a missing point. The interviews transcription summed 72 pages.

The startups were located in the cities of São Paulo, São José dos Campos and Campinas; the latter two are located in São Paulo state and in a radius of less than 100 km of the city of São Paulo. All cities contain great universities and a flourishing innovation culture.

B. Requirements engineering model

At the moment, our research results can be represented by a conceptual model on Fig. 3. The model is composed by two main groups: the context where the development team is inserted including elements that influence the whole startup (shaded boxes); and the requirements engineering process (bigger shaded box) itself within its properties and practices white boxes). Finally, a list of used practices from software or startup development methodologies will be given in subsection IV.C.

In this context, the following elements were identified: founders, software developer manager, market, business model, developers and entrepreneurship ecosystem. The first four elements are close related to categories identified in [10]. This was expected since the requirements engineering is part of the software development process.

Founders can be technical or non-technical and their knowledge can determine if some techniques will be used or not. Patience or anxiety they may feel will also play a role. One manager said that they used to have a Scrum based process but it was abandoned because of founders saw it as “waste of time”. After almost a year, the team will start again to use a Scrum based process because the result was worse than before. Generally, founders are “born entrepreneurs” having founded several companies, sometimes still running more than one, and many times they have never worked for a consolidated firm.

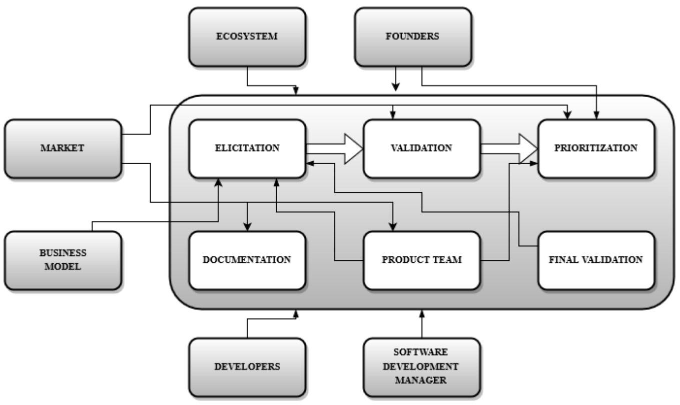

Fig 3. Requirements engineering process conceptual model

The *software development manager* can be a founder or someone hired in the startup history beginning [10]. His knowledge about requirements engineering techniques will be determinant to their use. For example, one manager said that his team was not using pair programming because he had a bad experience and “lost his faith” on it.

Market is a very powerful process determinant for startups. One interviewee was from a startup operating in markets considered critical like financial and defense. To be able to sell its products, the organization has to use strict processes because it is required and checked by clients. This is pointed out in [10] as market requirements.

TABLE I. INTERVIEWS

Startup	Market sector	Interviewee Position
1	Internet of Things	Founder and software director
2	Agricultural automation	Founder and software director
3	Health	Founder and CEO
4	Real state	Technical Leader
5	Finance and Defense	CEO
6	Retailer	Founder and technical leader
7	Sharing economy	Mobile leader
8	e-Learning	Product Director
9	e-Learning	Product Manager

It is also possible to classify startups depending on its product market. We will use the following classification: client-targeted market and user-targeted market. When a client is clearly identified,   (and is reachable, one might say that the market the startup operates is client-targeted. On the other side, if the software is targeted to a large number of users even when a company or organization will pay for the software, we will say that this market is user-targeted. This classification is different from the classical business-to-business (B2B) and business-to-consumer (B2C) classification since B2B and B2C are more related to organization strategy and sales and it was not fitting well in our study. For example, one platform was targeted to elementary students but the schools pay for the software: for sales, it has a small number of clients, the schools, then it should be classified as B2B but for software development, there are several users, the students, then it resembles more the development of common B2C products like a website.

The startup *business model* could already be stablished or still under construction when the organization tries to understand the market and how it can make profits. Generally, when the business model is not stablished, founders participate more in requirements engineering because it can change if the startup will success or fail.

The main reason why the startup develops software will also influence requirements engineering. Software can be a product as itself like when a machine to select vegetables is created and the pattern recognition software is core or just an instrument to other business model like an online real state agency. This will determine who will tell what the requirements are. If the software is the product itself, the requirements will be clearer but if it is not the case, the business rules will dictate what the software requirements are.

Developers themselves can refuse to use some techniques or they can slow down adoption like one manager said "*to implement concepts I just can't say guys here it is. I have to go slowly in such manner.*" But developers also bring their experience and suggest the use of some techniques.

The *ecosystem* where the startup is also plays an important role. When asked if he would use a different process if he had money, a founder answered categorically that yes. Acceleration programs and mentorship also provides knowledge to startup founders and favor techniques use. One founder said "we started at [an acceleration program] and they taught startup things, MVP and since our target was to build a MVP". A common mentioned problem is limited human resources. Sometimes, this problem is ecosystem related like when a manager said that his company had money to hire but it was not possible because they were not finding people with required knowledge. It is important to highlight that, according to Cukier et al. [19], São Paulo ecosystem, where all interviewed startups are, is considered as an Evolving ecosystem in a scale consisted of the following: Nascent, Evolving, Mature and Self-Sustainable.

After the context description, it is time to discuss the process itself. The *requirements engineering process* is composed of several techniques from different methodologies that were considered relevant for the process by the manager and/or founders. These practices are usually adapted to the startup context. It was observed that startups are at different stages of process maturity: from almost no process and without any common practices to clear phases and techniques to each one. But this is not static: the process matures and that is viewed by managers and sometimes they expect to such thing happen. The only case when the process was extremely formal occurred because of market requirements. It is also common to adapt the process to client needs.

Inside requirements engineering process is common the *product team* figure. This team is not made of developers and its role is to understand what the product goals are. It exists when the market is user-targeted and it acts like a proxy to the real users. These people execute most of requirements engineering disciplines and, specially, *elicitation* and *prioritization* phases.

The *elicitation* phase is clearly different depending if it is a client or user targeted startups. In the first case, the team just has to ask the client what he wants to be made. However, the latter is different: there is no such figure of client to ask for requirements. Generally, this role is played by the *product' team* but it is very common that the founder(s) do that. They usually have a market understanding and that is why they created the company in the first place. One common thing for all cases is continuity: requirements elicitation always occurs. In almost all cases, ideas come from everyone within the company.

After a requirement is created, a *validation* stage takes place to test if the idea is worth implementing. This is accomplished with little or even no software development and is similar to the Minimum Product Viable (MVP) concept from Lean Startup [4]. This concept calls for, when developing something new, build the least necessary to learn something from your market, creating and validating hypothesis. How the validation is made depends in the most on market: if it is client target, it is simply to ask to the client but when it is user targeted some kind of experiment is made.

The *prioritization* is generally made in layers. Because of the software importance to business survival (one interviewee said that what his team makes in a month can change a lot the financial result for that same month), there is a high level prioritization by founders and/or higher managers to determine what will be done for larger periods. But, the prioritization is also influenced by the market. If it is a client-targeted product, the client himself will tell which his priorities are. One founder who also played as a software developer manager said "We would say let's talk to the client. He decides". Then, in sprints, what is very common, the development team or even the product team determine what it will be done. Another interesting point is that, especially in small companies, it is common to mix technical to non-technical tasks like legal requirements, marketing, etc.

A *final validation* stage is not described in traditional requirements engineering process but for startups they are very important. At this point, after the software is implemented, it is the moment when the learning cycle about the market is closed and the process is done. The requirement correctness can be effectively tested, that is, if what was implemented is what the user or client has desired or expected. It is very common to new requirements emerge in this stage, re-feeding the whole process. The interviewees mentioned the user feedback importance on this stage. A difficulty mentioned was that, generally, several changes to the product can occur at the same time making hard to isolate effects from each change.

Finally, *documentation* is a set of practices that occur during all requirements engineering process. The documentation level varied substantially through interviewed startups. There are startups that do not have a clear process to keep track of requirements and just count on emails and contracts. On the other side, the startup that operates on defense market has a strictly documenting process as a result of market requirements. However, most of startups actually do some kind of requirements documentation generally using simple tools like physical or electronic boards. Bigger teams can also use medium complexity tools like issue trackers and agile project management tools. One startup tried traditional project management tools but it did not work out. This might have happened because of the agile mindset in this team that was not targeted by the tool.

Fig. 3 represents the conceptual model that was described. There is a clear boundary that separates the process from its context. The arrows that touch the boundary indicate elements that influence the process as a whole like *Ecosystem*, *Developers*, *Software Development Manager* and *Founders*.

There are arrows also to indicate stronger influences that context elements have on specific activities: Market determines *Validation, Prioritization, Documentation* and the presence or not of a *Product Team*. The *Business Model* maturity influences *Elicitation* and *Founders* influence the prioritization. The *Product Team*, when present, is responsible for *Elicitation* and *Prioritization*. The thicker arrows indicate the general flow: *Elicitation*, *Validation* and *Prioritization* that is different from the traditional flow from Fig. 1. That is an expression of how hard is to elicit the correct requirements and a validation stage has to be made before implementation to verify if that requirement is what the market wants. And an arrow was added from *Final Validation* to *Elicitation* to represent new requirements that arise in this stage.

C. Used practices

In a general way, no startup followed strictly a methodology. Instead of that, they used techniques from different methodologies for software or product development according to their needs as said before. But, there are more common practices that will be described now.

Software development is organized in sprints like Scrum and XP. From XP also it is usually used continuous integration, metaphors, tests and collective ownership. Continuous integration or even continuous delivery was mentioned in four different startups. It is usually seen when there are more experience developers in the team that generally setup the environment. These practices reduce the time needed to a new feature to reach the users what is very important for startups. Metaphors or the use of the client language was mentioned by a startup that is client-targeted and it shows the close relationship to the client. Automated tests use was cited by all companies but its depth depends on experience from simple acceptance tests to a mix of unit, integration and acceptance tests. Test-driven developed was not mentioned. A possible explanation is that it may be seen as a “slower” development process. Two startups fostered some kind of knowledge sharing of the code and architecture.

From Kanban, one startup uses an updated task list and limit work-in-progress. In this case, the prioritization stage is thinner and is only concerned about to order tasks depending on value. This is different from the other startups that also determine what will be done in the sprint. This startup also limits its work-in-progress as preconized by Kanban.

For product development, although not fully used, it is common to people have some level of knowledge of Lean Startup. Almost all startups run some kind of Build-Measure-Learn cycle. Some really implement the MVP (Minimum Viable Product) concept: they try to develop an experiment to test if an assumption about what the user wants is really true. One interviewee told about a new process that was made manually first to check if it will take users to signup to the product more often. Other startups just mentioned less complex tests like prototypes or mock-ups. Generally, they are used by client-targeted startups or for internal verification if something being built is what is expected. In this case, some tools to create wireframes are generally used.

From Design Thinking [20], which is a set of practices that describe designers’ activities during a product development, ideation process was mentioned by one startup. In the same sense, brainstorm sessions were also mentioned by another startup. In both cases, the intent was to foster new ideas to develop the product and they were performed by the product team.

D. Recommended attitudes

Given the requirements engineering process model, some recommended attitudes may favor a better process. For the context, *founders* should be aware that software development is complex and takes time. There is a knowledge body of better practices which enable better processes. The *founders* must control their anxiety and let their development team produce software. Of course, to be able to allow their team to be more independent, they should trust them. This is only possible if they hire the right people.

Developers and the *software developer manager* should be also aware of best practices and be open-minded. They should know that, generally, agile is a better approach for startups. Since the ecosystem is not yet mature, there is no clear indication on the use of methodologies Then everyone involved in the startup creation: developers, managers and founders should be aware of that and should take actions to increase familiarity with the better practices: participate in meetings, read books or other material, etc. From these attitudes, they will foster the ecosystem development what will benefit themselves and the society.

The startup as whole should be aware of the market that it is operating (client or user targeted) and how this aspect influences their practices. For example, if it is important to have a product team or a similar role.

The process itself should exist but people must be aware that it has to change to adapt to its context. *Elicitation* should allow anyone to give ideas since innovation is vital. *Validation* is very important to reduce development waste given the lack of resources and time. *Documentation* should be used to foster knowledge sharing and help process organization. Since *prioritization* generally involves several different tasks, not all software development related, it is important to have someone or a group of people that are aware of the whole organization.

E. Threats to validity

A clear threat to validity in this research was the researcher bias. Since the first author is a partner on a startup, his prior knowledge and conceptions about the subject could have an effect on data collection and analysis. The use of well-known techniques (grounded theory) was an attempt to mitigate this threat. Furthermore, the guide interview developed was another way to keep the interviewer focused.

The opportunistic approach used to reach interviewees could also be problematic. But as described earlier, it was possible to get startups and people who operate in different markets and have different size and maturity stages.

F. Summary

In the emerged model, it is clear that requirements engineering processes are influenced by external and internal factors. They are also closely related to business development, where the business model is a decisive factor in the choice of practices being used. The existence of a product team, its activities and its existence conditions are also important. Finally, the process mapping and suggested practices could improve startups processes and, therefore, their success rate.

V. CONCLUSION

The requirements engineering process problem in startups was described and a research based on Grounded Theory techniques was proposed and conducted. As a result, a conceptual model was presented containing elements from startups context that affect requirements engineering and how these processes are conducted. A list of used techniques was also provided.

This research is still a working in progress and conceptual model improvements are expected. Although, a general idea is clear. Sharing these findings with the community and absorbing its feedback could improve the final result. New interviews will also be conducted to help build the theory. Then it is possible to foresee some interesting developments. Startups founders and accelerators staff could use these findings to guide requirements engineering processes in new startups. For instance, a startup founder that is not satisfied with his startup requirements engineering process can detect if he can change something on the context like himself or the software developer manager or if it is something from the ecosystem that he has to live with. A manager can also take a reference when creating the process that his team will follow to develop software and the problems that his team will face because of the context that the company is inserted. Public policies managers can also take insights from this study to foster startup creation and survival.

Researches can use this model to propose new techniques and evaluate them against what is being done today. An attempt to generalize this model could also be made through a quantitative approach like a survey. It is important, tough, to keep in mind that there are always limitations in empirical research involving human beings. This study also provides more evidence to support part of Coleman and O'Connor software development process formation model in startups [10] by detecting some of the same influences that the authors did. The research methodology could also be used again to focus other points like software implementation that was beyond the scope.

This research focused on Brazilian startups and, as a result, the ecosystem influence the process, other replications of this research could be made in more mature ecosystems to verify if there is more capital and human resources available, the processes will really be different.

ACKNOWLEDGMENT

The authors would like to thank to all interviewees to give some of their time to this research. We would also like to thank Prof^a Dr^a Viviane Santos for her important help on Grounded Theory. This research was funded by a grant from CNPq, Proc. 485070/2013.

REFERENCES

[1] N. Paternoster, C. Giardino, M. Unterkalmsteiner, T. Gorschek, and P. Abrahamsson, "Software development in startup companies: A systematic mapping study," *Inf. Softw. Technol.*, vol. 56, no. 10, pp. 1200–1218, April 2014.

[2] E. Deakins and S. Dillon, "A helical model for managing innovative product and service initiatives in volatile commercial environments," *Int. J. Proj. Manag.*, vol. 23, pp. 65–74, 2005.

[3] S. Blank, "The four steps to the epiphany," *Cafepress. com*, 2005.

[4] E. Ries, *The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses*. Random House LLC, 2011.

[5] M. Patz, "Lean Startup – Adding an Experimental Learning Perspective to the Entrepreneurial Process," University of Twente, 2013.

[6] S. M. Sutton, "The role of process in software start-up," *IEEE Softw.*, vol. 17, pp. 33–39, 2000.

[7] B. Nuseibeh and S. Easterbrook, "Requirements engineering: a roadmap," *ICSE '00 Proc. Conf. Futur. Softw. Eng.*, vol. 1, pp. 35–46, 2000.

[8] G. Kotonya and I. Sommerville, *Requirements engineering: processes and techniques*. J. Wiley, 1998.

[9] "Manifesto for Agile Software Development", *Agilemanifesto.org*, 2001. [Online]. Available: http://agilemanifesto.org/. [Accessed: 28-Mar- 2016].

[10] G. Coleman and R. V. O'Connor, "An investigation into software development process formation in software start-ups," *J. Enterp. Inf. Manag.*, vol. 21, no. 6, pp. 633–648, 2008.

[11] M. Taipale, "Huitale - a story of a Finnish lean startup", in: Lean Enterprise Software and Systems, vol. 65, Lecture Notes in Business Information Processing, 2010, pp. 111–114.

[12] K. Schwaber, Agile project management with Scrum. Redmond, Wash.: Microsoft Press, 2004.

[13] K. Beck, "Embracing change with extreme programming," *Computer (Long. Beach. Calif).*, no. c, pp. 70–77, 1999.

[14] C. Ebert, "The impacts of software product management," *J. Syst. Softw.*, vol. 80, pp. 850–861, 2007.

[15] S. Blank, "Why the Lean Start Up Changes Everything," *Harv. Bus. Rev.*, vol. 91, no. May, p. 64, 2013.

[16] C. B. Seaman, "Qualitative methods in empirical studies of software engineering," *IEEE Trans. Softw. Eng.*, vol. 25, no. 4, pp. 557–572, 1999.

[17] B. Glaser and A. Strauss, *The discovery of grounded theory*. Chicago: Aldine Pub. Co., 1967.

[18] A. Strauss and J. Corbin, *Basics of qualitative research*. Newbury Park, Calif.: Sage Publications, 1990.

[19] D. Cukier, F. Kon, and N. Krueger, “Designing a Maturity Model for Software Startup Ecosystems,” in *Product-Focused Software Process Improvement*, Springer, 2015, pp. 600–606.

[20] T. Brown and B. Kātz, *Change by design*. New York: Harper Business, 2009.

[21] K. Beck, *Extreme Programming Explained: Embrace Change*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2000.

[22] K. Beck, *Extreme programming eXplained*. Reading, MA: Addison-Wesley, 2000.

[23] W. W. Royce, “Managing the development of large software systems,” in *proceedings of IEEE WESCON*, 1970, vol. 26, no. 8, pp. 1–9.

[24] "ATLAS.ti: The Qualitative Data Analysis & Research Software", *atlas.ti*, 2016. [Online]. Available: http://atlasti.com/. [Accessed: 05-Apr- 2016].