The Early Stage Software Startup Development Model: A Framework for Operationalizing Lean Principles in Software Startups

Jan Bosch[1], Helena Holmström Olsson[2], Jens Björk[1], and Jens Ljungblad[1]

[1] Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden
jan.bosch@chalmers.se, {jensbj,jenslj}@student.chalmers.se
[2] Department of Computer Science
Malmö University, Sweden
helena.holmstrom.olsson@mah.se

Abstract. Software startups are more popular than ever and growing in numbers. They operate under conditions of extreme uncertainty and face many challenges. Often, agile development practices and lean principles are suggested as ways to increase the odds of succeeding as a startup, as they both advocate close customer collaboration and short feedback cycles focusing on delivering direct customer value. However, based on an interview study we see that despite guidance and support in terms of well-known and documented development methods, practitioners find it difficult to implement and apply these in practice. To explore this further, and to propose operational support for software startup companies, this study aims at investigating (1) what are the typical challenges when finding a product idea worth scaling, and (2) what solution would serve to address these challenges. To this end, we propose the 'Early Stage Software Startup Development Model' (ESSSDM). The model extends already existing lean principles, but offers novel support for practitioners for investigating multiple product ideas in parallel, for determining when to move forward with a product idea, and for deciding when to abandon a product idea. The model was evaluated in a software startup project, as well as with industry professionals within the software startup domain.

Keywords: Software startup companies, agile software development, lean principles, process support.

1 Introduction

New software companies are started every day, and emerging technologies such as smartphones, cloud infrastructure platforms and enhanced web development tools have made it even quicker and easier to get started. The many success stories surrounding software startups, such as for example Facebook, Twitter and Instagram, contribute to their popularity and allure. However, contrary to what media portraits, far from all software startup companies succeed [1], [2]. If looking at new product ideas, over 98% fail [1]. This has led researchers, e.g. [3], [4], [5] and [6], to try and identify what factors contribute to software startups succeeding. In recent years, several authors [1], [7], [8], [9], [10] have embraced lean thinking and customer focused development as the way forward.

In order to understand the many challenges that software startups face, we need to understand what constitutes a software startup. According to Ries [8] a startup is a human institution designed to deliver a new product or service under conditions of extreme uncertainty. Most often, startups have limited resources in terms of people and funding, and are run on very tight schedules. In addition to that, they are commonly exploratory in nature, lacking clear requirements, customers and even business models. With this in mind, being efficient and systematic is of high importance; efficient in terms of minimizing the develop effort while maximizing value gained, and systematic in terms of continuously validating if what you develop generate customer value. During the early 2000's, lean development principles [3], [4], [11], [12] gained popularity in the startup community. Especially, lean principles emphasizing continuous learning based on customer validation of functionality, experimentation with customers to test hypotheses and assumptions, and short feedback cycles to avoid efforts on activities that do not generate customer value, have attracted significant attention from practitioners. However, while lean principles have permeated the software development industry for a while now, research on how to apply these principles in practice, and especially in the context of startups, is scarce. In related research, we see that while well-established companies might succeed in implementing selected parts of agile and lean principles, they experience difficulties with maintaining these throughout the development cycle [13], and this becomes even more difficult in the context of a software startup. To explore this further, and to propose a decision-making model as support for software startup companies operating in highly uncertain contexts, this study aims at investigating the following research questions:

- What are the typical challenges when finding a product idea worth scaling?
- What solution would serve to address these challenges?

Based on the Design Science Research (DSR) framework [14] we iteratively explored our research questions based on qualitative interviews with practitioners in nine startup companies, in order to capture the challenges they experienced in relation to balancing multiple product ideas, deciding on when to move forward with an idea, and how to know when to abandon and exit a product idea. In parallel with this, and as a natural following on the insights we acquired in the interviews, we started formulating hypotheses as the basis for development of a solution to their problems. As a result of our study, we propose the Early Stage Software Startup Development Model (ESSSDM). The model extends lean principles by offering more operational process support and hence, better decision-making support, for startup companies. The model is evaluated in a startup setting and based on the results from this evaluation we see that it is found useful for addressing the challenges experienced by the practitioners involved.

The contribution of this paper is twofold. First, we present a validated process model that manages a portfolio of ideas, whereas existing approaches typically focus on only one idea. Second, the model we propose provides a detailed approach for managing ideas, i.e. it offers operational guidance in terms of stage gates and exit criteria that have so far been difficult to find in existing methods and supporting frameworks.

The paper is structured as follows. First, we outline agile and lean principles as the two most influential approaches in software startups. Second, we describe our research approach based on design science research involving a qualitative interview study at nine software startup companies in Sweden. We present our interview findings, followed by a section in which we introduce the Early Stage Software Startup Development Model (ESSSDM) as a solution to the challenges revealed in the interviews. Finally, we present the evaluation process of the model and the conclusions of the study.

2 Background

2.1 Agile Software Development

During the last decade agile methods have dramatically changed the way software development is performed. Unlike traditional development methods characterized by sequential phases and heavy upfront planning, agile methods deal with unpredictability and change by relying on people and close customer collaboration rather than formalized processes [15]. Agile methods are characterized by short development cycles, collaborative decision-making, rapid feedback loops, and continuous integration of code changes into the product baseline. Thus, agile methods operate on the principle of “just enough method” and seek to avoid cumbersome and time-consuming processes that add little value to the customer [16].

Today, many different agile methods are in use [11]. During the last decade, XP and Scrum have become well established in small-scale, as well as large-scale software development. While XP is basically a collection of well-known software engineering practices taken to the extreme, Scrum is a simple, low-overhead process for managing software development [17]. The two methods are highly compatible in that XP provides engineering techniques and Scrum essentially works as a wrapper for such techniques. Although agile methods differ in details and techniques, overall agile principles such as ‘flexibility’, ‘working code’ and ‘customer collaboration’ lie at the heart of all of them. As highlighted in this paper, agile methods have become increasingly common in software startups due to their flexible, lightweight and adaptive nature with a strong focus on close customer collaboration throughout the development process.

2.2 Lean Principles

In emphasizing customer collaboration, short feedback loops and flexibility, agile software development methods can be seen as one way to operationalize some of the values that permeate lean principles. These principles originate from lean manufacturing, i.e. a production practice that considers the expenditure of resources for any goal other than the creation of value for the end customer to be wasteful, and thus a target for elimination. Working from the perspective of the customer who consumes a product or service, "value" is defined as any action or process that a customer would be willing to pay for. Essentially, the lean concept is centered on preserving value with less work. The philosophy focuses on getting the right things to the right place at the right time in the right quantity, to achieve perfect work flow, while minimizing waste and being flexible and adaptable to change [18]. In aiming to make work simple to understand, do and manage, the lean concept is characterized by four rules: (1) all work shall be highly specified in terms of content, sequence, timing, and outcome, (2) every customer-supplier connection must be direct, (3) the pathway for every product and service must be simple, and (4) any improvement must be made in a systematic and scientific way at the lowest possible level in the organization.

2.3 The Lean Startup

While agile development processes are primarily solution focused and answers ‘how’ to build products fast, they do not answer ‘what’ products to build. That is, they are mainly applied in situations where the problem is fairly well understood but the solution is not. In a startup context, however, neither the problem nor the solution is well understood. Therefore, a software developer working in a startup context, needs to be focused not only on the technical solution itself, but also on overall business strategies and needs, e.g. an associated business model, targeted marketing efforts, and the establishment of customer relationship models. Recently, the solution focused thinking that characterizes agile practices has gained increasing attention due to Eric Ries [8] and the ‘Lean Startup’ movement. In his book, Ries notices that because of this solution focused thinking a lot of software startups are failing, including his own. Instead of actively evaluating what customers value, most startups spend time and money developing products that people are not interested in. While projects are delivered on time and on budget, nobody wants the product. Ries [8] underscores the importance of understanding the problem before developing the solution. In his work, Ries is heavily influenced by the ‘Customer Development Model’ that was outlined by Blank in 2005 [7]. In this model, customers are considered from the very start. It is a structured process for testing business model assumptions (or hypotheses) about markets, customers, channels and pricing etc. The model consists of four steps, i.e. customer discovery, customer validation, customer creation and company building. While the first two steps are about capturing the vision and break it down into testable business model assumptions, the last two steps concern building demand for the product, start scaling the business and transition from a startup to a fully fledged company executing the validated business model.

A central concept within the Lean Startup is ‘The Pivot’, a term used when a startup company changes direction based on what they have learned about customers. Ries [8] claims that having “pivoted” is the most frequently occurring commonality among successful startups, i.e. successful startups seldom end up doing what they initially set out to do, and they change direction based on efficient collection of customer data. By reducing the time between pivots, it is possible to increase the odds of success before running out of money. Another central concept is the ‘Build-Measure-Learn’ (BML) loop, which is described as the concept of validated learning [8]. In this loop, ideas are turned into products, data is gathered by measuring how the product is actually used by its customers, and ideas for product improvement and innovation are based on what is learned by analyzing the data collected from customers. In this way, focus is always on delivering customer value, and the risk of being too solution-oriented is minimized.

2.4 Lean in Practice

While the Lean Startup presents many interesting concepts and ideas, it can be difficult to understand how to apply them in practice. Hence, several authors have tried to clarify this, and there are a number of handbooks and guidelines to support the practical implementation of lean principles in software startups. One of the more influential ones is ‘Running Lean’ in which Maurya [10] outlines a process based on the principles in Blank’s work on customer centered development [7], and Ries work on lean startups [8]. As such, Running Lean provides a process for applying Lean Startup principles when developing a software business. The process is divided into three distinct steps:

- *Document the initial plan.* This is done by capturing and focusing on the entire business model, not only the product/solution. The “solution box” should be kept small to keep solution-oriented developers from spending too much time there. The goal is to capture the vision of the business.
- *Identify the most risky parts of the plan.* After having documented the initial plan, risks are assessed and prioritized. The risks that are considered highest should be dealt with first. Three types of risks are identified, i.e. product risks, customer risks, and market risks.
- *Systematically test the plan.* With an initial plan in place, and with risks prioritized, the rest of the process focuses on systematically testing and iterating the plan using methods such as Ries’ BML loop [8].

3 Research Method

This paper is based on a study conducted between the authors and Chalmers School of Entrepreneurship (CSE), in Gothenburg, Sweden. Two of the authors, together with master students from CSE, co-founded a startup that was run in an incubator setting, i.e. an advanced entrepreneurial education combined with real business incubation. This paper presents the results of the research conducted during an eight-months period (fall 2012 – spring 2013). In our study, the following research questions were investigated:

- What are the challenges in terms of finding a product idea worth scaling in early stage software startups?
- What solution would serve to mitigate the identified challenges?

The study was performed using Design Science Research (DSR) [14]. DSR differs from other research approaches in that it focuses on learning through design, i.e. the design and development of artifacts. DSR is conducted iteratively, with lessons learned from earlier phases feeding back into later ones. DSR consists of five phases: (1) problem or problem area awareness resulting in an initial research proposal; (2) first approach to solving the problem; (3) realization in the form of an artifact; (4) evaluation of the artifact according to defined criteria; (5) stop iterating and reach conclusions. DSR allows for moving back and forth between phases in an iterative fashion, iterating until the evaluation criteria are met. For the purpose of this study, DSR was deemed a good fit due to the context of our research. With the authors taking part in the forming of a startup, and with the design of an artifact aimed at mitigating typical challenges, DSR seemed as an interesting and relevant approach. Furthermore, the close proximity to a real-world startup meant that the artifact could be rapidly iterated and evaluated, an opportunity which is advocated in DSR.

3.1 Research Process

In applying DSR, the following activities were undertaken in each of the five phases:

(1) Problem awareness: In the first phase, research questions were formulated and a literature review focusing on agile and lean development practices was conducted. The research questions were defined as: (1) What are the typical challenges in terms of finding a product idea worth scaling in early stage software startups, and (2) What solution would serve to mitigate the identified challenges? After having our research questions formulated and our literature review completed, semi-structured interviews with practitioners in nine startup companies in Sweden were conducted. The purpose of these interviews was to get an in-depth understanding of how software startups typically work in the early stages, what challenges they face, and if any best practices could be observed. When selecting interviewees, we used criteria such as (1) they should work at a software startup company with at least one product on the market, (2) they should be the CTO or similar of the company, and (3) they should have worked at the company from its start. All interview sessions were about 60 minutes long, and two sessions per interviewee were conducted with two of the authors present. The first session was exploratory in nature and took a broad perspective. The second session aimed at exploring further what was said in the first session, and to move beyond the initial understanding. An interview guide with template questions served as a guideline. However, the structure of the interviews was flexible so that discussions could go in new and interesting directions. All interviews were recorded and transcribed.
(2) Problem solution: As a result of phase one in which the literature review and the qualitative interviews contributed to a deep understanding of the problem area, an initial research proposal was formulated. Our initial problem solution was to design a

process model that would address the challenges as pointed out in literature and in our interviews and hence, provide effective operational support for managing software startups and increase their odds of success.

(3) Development of artifact: In the third phase, and in response to the challenges that were identified in literature and in our interviews, we developed the Early Stage Startup Software Development Model (ESSSDM). The model extends existing lean startup principles by incorporating knowledge gained in previous research as well as recent challenges experienced by practitioners participating in our interview study.

(4) *Evaluation of artifact:* In the forth phase, the ESSSDM model was evaluated as part of a startup project at the School of Entrepreneurship at Chalmers University of Technology in Gothenburg, Sweden. In this evaluation, criteria such as perceived usefulness, multiple product portfolio support, support for pursuing or abandon a product idea, and techniques for validation were used. During this evaluation, revisits to the problem solution phase were frequent, and we were aware that revisions of our initial problem solution were most likely to happen. In addition, the model was evaluated by a subset of the interviewees in four of the nine companies involved in the study.

(5) Conclusion: As a result of our evaluation, we revised and improved the Early Stage Software Startup Development Model (ESSSDM) to include clear guidance on multiple product ideas, on when to move product ideas forward, on when to abandon an idea, and finally, what techniques to use for validating product ideas.

4 Interview Findings

Nine founders, CTOs and early employees of software startups in the Gothenburg region in Sweden were interviewed. The interviews focused on the following topics:

- How did the initial product idea emerge?
- Does the initial product idea differ from the current product?
- How are product ideas validated?
- What business and software development practices are used?
- What are the challenges faced in early software startup?

Based on our interviews, we see that all of the companies use agile practices when developing their software, and seem to have a good understanding of how to apply them in practice. However, lean development principles were unfamiliar to most of the interviewees, and those who did know them found them difficult to implement in practice. Overall, very few worked with continuously validating product concepts with customers to try to identify problems before building a full solution. One explanation for this might be that some companies started from existing solutions, thereby reducing the need for extensive problem validation. Companies that did not copy existing solutions, but instead tried to innovate, put more effort into understanding the problem. Their current products are the result of having pivoted multiple times. However, our interviews do not show on any systematic approach or method for knowing when to pivot.

Based on the interviews, no obvious best practices for building a successful startup could be derived. All companies understand how to build software efficiently using agile methods, but few work with lean startup practices such as the Customer Development Model [7], or validated learning [8]. Our interviewees confirmes that it is very difficult to know how to work in a straightforward manner in early stage startups, and that operational process support, i.e. decision-making support, is limited. Based on our interviews, we identify a number of key areas where operational process support is needed:

- Existing processes and theories do not adequately support working on, or investigating, *multiple product ideas in parallel.*
- Existing processes and theories provide insufficient validation criteria for *moving product ideas forward.*
- Existing processes and theories lack clear guidance on when to *abandon a product idea.*
- Existing processes and theories provide insufficient suggestions of what *techniques to use during validation of product ideas.*

5 The Early Stage Software Startup Development Model

In response to the challenges mentioned by our interviewees, and as summarized in the key areas above, we developed the Early Stage Software Startup Development Model (ESSSDM). The model extends existing Lean Startup principles [8], [9], [10], [19], incorporates the results from our interviewees with entrepreneurs within the software startup domain, and builds on the author's previous experience from working with software startups. Figure 1 presents an overview of ESSSDM.

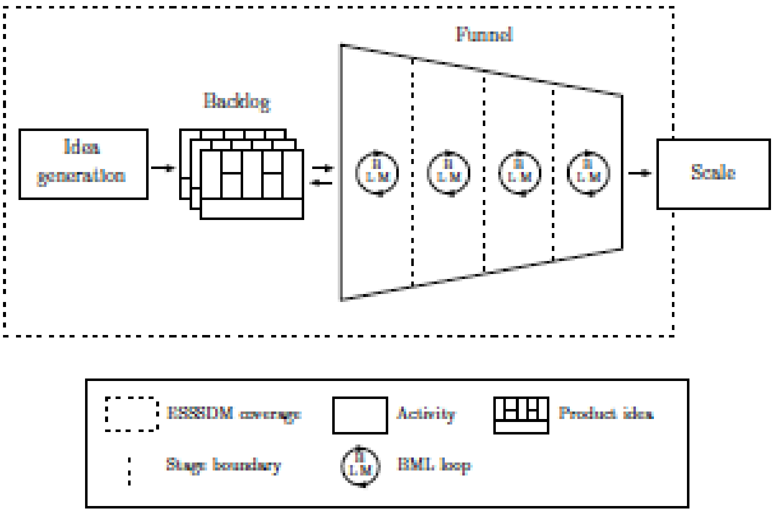

Fig. 1. The Early Stage Software Startup Development Model (ESSSDM)

The model supports multiple product ideas being investigated in parallel, it defines a step-by-step process with clear exit criteria, and it presents guidance concerning the techniques and practices to employ during the different stages. The purpose of the ESSSDM model is to find one product idea worth scaling. There are three parts to the process, i.e. (1) idea generation, (2) a prioritized ideas backlog, and (3) a funnel through which ideas are systematically validated using the Build-Measure-Learn (BML) loop [8].

5.1 Step 1: Idea Generation

We consider idea generation to be part of the startup process. Typically, it occurs prior to incorporation, but sometimes an existing company wants to expand their product portfolio, and thus needs to come up with new ideas. There are a number of techniques to generate and extract ideas:

- *Exploratory interviews:* One way to extract problems from potential customers is to go out and talk with them. It is recommended to investigate one customer segment at a time, so that the team stays focused and dig deep within each segment. The purpose is to understand how potential customers run their businesses, and what problems they experience.
- *Follow-me-homes:* One way to discover problems is to ask potential customers for permission to spend a day at their office in order to see their work habits in action. This is useful in order to extract tacit knowledge. However, the practice is very time consuming, and it can be hard to convince people to participate if there is no prior relationship. During follow-me-homes, monotonic work, complex workflows, communications paths, information load and time consuming tasks are useful items for observation.
- *SCAMPER:* This is a brainstorming technique used to systematically generate new ideas by modifying existing product concepts [20]. Each letter in the acronym represents a different way of thinking in terms of modification, (i.e. *S*ubstitute, *C*ombine, *A*dapt, *M*agnify/modify, *P*ut to other use, *E*liminate and *R*earrange/reverse).

5.2 Step 2: The Backlog

All ideas for potential products are put in a prioritized backlog. In similar with user stories within an agile product backlog, all product ideas in the backlog have to be written in a comparable format. If this is not done, the task of prioritization becomes increasingly complex. Being able to compare and prioritize among ideas is crucial when working on multiple ideas in parallel. After having documented ideas in a comparable format, they need to be prioritized. The following criteria are useful when prioritizing among product ideas:

- *How much do customers care about the problem?* The problem should be significant in order for a solution to generate interest and revenue.
- *How much does the team care about the problem?* A software startup will require an enormous investment in terms of effort and time. All involved need to be personally devoted to the task.

- *How large is the market potential?* If relevant, it is worth considering if the idea can be bootstrapped or if it will need investments.
- *How much domain knowledge exists within the team?* A skilled team with domain expertise reduces uncertainty regarding the problem and saves valuable time during the problem/solution validation stages.
- *Has the team experienced the problem themselves?* Known as ”scratching your own itch” [21], and reduces uncertainty regarding the problem and saves valuable time during the problem/solution validation stage.
- *Are customers easy to reach?* To get going, the team needs good access to potential customers. The easier access they have to people experiencing the problem, the easier it is to get rapid feedback.

5.3 Step 3: The Funnel

Ideas from the backlog are fed into a ‘funnel’ where they undergo systematic validation using the Build-Measure-Learn (BML) loop. Multiple ideas can exist in the funnel at the same time, and be investigated and validated in parallel. The funnel is divided into four stages, and each stage has its own set of risks and exit criteria. Ideas move through the funnel stages as a result of a validated learning process in which data needed to mitigate risks and fulfill exit criteria are provided. The four stages are (1) Validate problem, (2) Validate solution, (3) Validate Minimum Viable Product (MVP) small-scale, and (4) Validate Minimum Viable Product (MVP) large-scale (see sections 5.3.1 – 5.3.4 below for details on each stage).

There are several reasons why investigating multiple ideas in parallel is worthwhile during the early stages of a startup: (1) The increased ability to stay objective. Growing attached to one particular idea too early can be damaging [14] [30]. In the early stages, an open mind and a willingness to change direction are advantageous traits. (2) Having a pipeline of ideas means there is always something to work on when other ideas are on hold, e.g. waiting for experiments to run, or interview session dates to be set. It is also useful when neither pivoting nor persevering is an attractive option, i.e. when a risk becomes blocking. (3) Most startups investigate and prioritize multiple ideas prior to picking one around which the company is formed.

When working on multiple ideas in parallel, it is important to enforce a limit on how many ideas can be worked on simultaneously. This number becomes smaller during the later stages of the funnel. During stages one and two, we found three ideas in a team of five to be efficient. During stage three and onwards, it often becomes a matter of available resources, and the size of the MVP. A simple approach for dealing with ideas in different stages of the funnel is to assign points to each idea, and then limit the amount of points each team member can work on in parallel.

The process that each individual idea goes through while in the funnel can be described as a feedback loop comprising risk prioritization, followed by validated learning using the BML technique [8] [10]. At the end of each BML iteration, a decision is taken whether to move the idea forward, pivot, persevere, or put it on hold in favor of another idea. If an idea is not ready to move to the next stage, a decision need to be taken about whether to pivot, persevere or put the idea on hold. Pivoting is a significant strategic change, while still remembering what has been learned about customers so far [8]. Persevering means staying on the course, doing minor adjustments, and hoping for better results in time. The third option, to put on hold, is introduced as part of the concept of multiple ideas in parallel. If a risk becomes so severe that neither pivoting nor persevering is an attractive option, the risk becomes blocking and the product is put on hold until such time when the risk can be dealt with. In the meantime, a new idea is picked from the backlog and moved into the funnel to begin the process of validation.

5.3.1 Funnel Stage 1: Validate Problem

The purpose of the first stage of the funnel is to investigate and validate the underlying problem(s) that customers want to have solved. It specifically tries to answer (1) What is the problem? (2) Who has the problem? (3) Is the problem big enough to make a business out of?

Exit criteria for this stage are when a majority of customers (potential customers) indicate that they (a) want the problem solved, (b) are willing to pay for a solution, and (c) are willing to participate in solution testing.

5.3.2 Funnel Stage 2: Validate Solution

The purpose of the second stage of the funnel is to define a solution that solves the problem(s) that customers want to have solved. This stage specifically tries to answer (1) What features are needed for the Minimum Viable Product (MVP)? (2) Who is the early adopter? (3) How much is the solution worth to customers?

Exit criteria for this stage are when a majority of customers (potential customers) indicate that they (a) believe that the solution solves the identified problem, (b) are willing to test the MVP, and (c) are willing to pay for the MVP (verbal commitment).

5.3.3 Funnel Stage 3: Validate MVP Small-Scale

The purpose of the third stage of the funnel is to build an Minimum Viable Product and test it on a small set of early adopters. It specifically tries to answer (1) Does the MVP solve the problem(s) that customers want to have solved? (2) How to access early adopters? (3) Are customers willing to pay for the MVP?

Exit criteria for this stage are when a majority of customers (potential customers) indicate that they (a) customers understand the Unique Value Proposition (UVP), and (b) customers accept the pricing model.

5.3.4 Funnel Stage 4: Validate MVP Large-Scale

The purpose of the fourth stage of the funnel is to further validate the MVP on a larger group of early adopters. This stage specifically tries to answer (1) Has the MVP reached product/market fit? (2) Is there a viable path to early adopters? (3) Is the business model suitable for the product?

Exit criteria for this stage are when the MVP (a) has passed relevant tests such as the Sean Ellis Test [22], (b) develops inbound channels that repeatedly delivers early adopters into the conversion funnel, and (c) produces a Customer Lifetime Value (CLV) > User Acquisition Cost (UAC).

Once an idea has moved through all four stages of the funnel it is considered validated and ready for commercial scaling. At this point, the objective of the Early Stage Software Startup Development Model (ESSSDM) has been fulfilled.

6 Evaluation

The Early Stage Software Startup Development Model (ESSSDM) was evaluated as part of a startup project at the School of Entrepreneurship at Chalmers University of Technology in Sweden, and through interviews with industry professionals. Two of the authors, together with three master students from Chalmers School of Entrepreneurship, co-founded a startup that was run in an incubator setting for eight months. The team was provided with initial funding and office space, and experienced industry professionals, business advisors and legal experts were also made available. The purpose of the startup was to find a promising product in the small business segment. In accordance with the Design Science Research framework, the following design goals were defined for evaluating the model (see below). For each design goal the evaluation sought for consensus from both the project team and from the industry professionals that were interviewed.

1) The process must support working on, or investigating, multiple product ideas in parallel.
2) The process must provide clear guidance on when to abandon a product idea.
3) The process must provide clear guidance on when to move product ideas forward through process stages.
4) The process must provide clear guidance on what techniques to use and when, while validating product ideas.

In the evaluation, consensus from the project team was derived by talking to each individual team member, while consensus from industry was derived by talking to a subset of all interviewees (representing four of the nine companies that were initially interviewed). They were asked to rate whether the design goal had been fulfilled, by choosing a number between 1 (”strongly disagree”) and 5 (”strongly agree”).

1) The model must support working on, or investigating, multiple product ideas in parallel

Consensus of project team: The project group consisted of five students: three business developers and two software engineers, working in an incubator setting. At the beginning of the project, no ideas existed. Doing exploratory interviews with potential prospects made the team both knowledgeable with how small businesses operate, and provided ideas on promising product concepts.

Overall, the team felt investigating multiple ideas in parallel was worth doing from a project perspective. Having a prioritized backlog was a good way to keep work focused, although there was some struggling before the team aligned in how to interpret the prioritization criteria. Having to document ideas in a comparable format made the prioritization process easier and forced the team to consider all aspects of the business model, not only the solution. A potential drawback of this was that vague ideas were not entered to the backlog due to difficulties in documenting them.

The workload was distributed so that the three business developers were responsible for one idea each, while the software engineers worked on all three ideas at once. When it came to building the MVP for the most promising idea, all other ideas were put on hold. Working in this way allowed for good momentum; there was always something in the pipeline for the team to work on. Sharing of assets between ideas happened frequently. HTML-mockups could often be put together using code, libraries and frameworks used on previous mockups. The team felt that reusing assets mitigated switching cost that comes when working on multiple ideas in parallel.

Consensus of industry professionals: Mean value: 4.4. Lowest value: 3.5. Consensus reached.

2) The model must provide clear guidance on when to abandon a product idea

Consensus of project team: The process gave clear guidance on when to abandon a product idea. The team constantly evaluated whether exit criteria had been reached or not. When experiments began to reach diminishing returns, and there was no clear path towards fulfilling the criteria, the team took a decision: pivot, persevere or abandon. If there was no obvious way to pivot, the team usually opted to abandon the idea in favor of another idea from the backlog. This way-of-working was well supported by the ESSSDM.

Consensus of industry professionals: Lack of data.

3) The model must provide clear guidance on when to move product ideas forward through process stages

Consensus of project team: The team appreciated having exit criteria as guidance on when to move forward with an idea. Having such clear goals enabled the team to keep a good momentum and allowed each team member to work independently. Also, it made it easier for the team to not miss anything critical during the validation process, something that is otherwise common in a chaotic startup setting. The stages were felt to be appropriate, even though the clearest separation was perhaps between stage two and three; one and two could probably be rolled into a single stage. Stage four was never reached. The exit criteria themselves were generally clear and unambiguous. The biggest problem was deciding on how many people to talk to, and how to gauge their reactions and feedback.

Consensus of industry professionals: Mean value: 4.6. Lowest value: 4. Consensus reached, but with some reservations, e.g. exit criteria are not to be blindly trusted but used as guide together with common sense.

4) The model must provide clear guidance on what techniques to use when validating product ideas

Consensus of project team: The definition of a relevant technique in this context is that (1) the outcome is valuable learning: it mitigates important risks and supports stage exit criteria, (2) the time it takes to execute is kept to a minimum. The team felt that in general, the techniques provided by the process were relevant; there was a clear connection between techniques, risks and exit criteria. Although, future versions

of the process might benefit from more detailed instructions, taking additional consideration to context, i.e. what has been done, what is about to be done etc.
Consensus of industry professionals: Lack of data.

Concluding, the Early Stage Software Startup Development Model (ESSSDM) addresses the challenges identified in the problem awareness and problem statement phase of software startups. As a result of an evaluation made with team members in a startup setting (students), as well as with industry professionals in the startup domain, we see that the model provides operational support for implementing lean principles in both planning and in execution stages of a software startup.

7 Conclusions

Software startups are more popular than ever and growing in numbers. They operate under conditions of extreme uncertainty, and face a number of challenges. In this paper, we identify these challenges by conducting a literature study on agile and lean development, and by conducting an in-depth interview study with industry professionals within the software startup domain. The result shows that few practitioners apply Lean Startup methods because these are found too vague and imprecise to implement in practice, i.e. they provide limited operational support. In response to this, we propose the 'Early Stage Software Startup Development Model' (ESSSDM) addressing the challenges identified in literature, as well as by professionals. The evaluation of the model shows that:

- ESSSDM supports working on, or investigating, multiple product ideas in parallel, as part of an idea portfolio.
- ESSSDM provides clear guidance on when to move product ideas for- ward through process stages.
- ESSSDM provides clear guidance provides clear guidance on when to abandon a product idea
- ESSSDM provides clear guidance on what techniques to use when validating product ideas.

To conclude, ESSSDM provides operational support for early stage software startups. Novel parts include (1) having a backlog with product ideas written in a comparable format, (2) a list of backlog prioritization criteria, (3) the concept of validating ideas through a funnel, and (4) the introduction of abandoning ideas as an alternative to pivot or preserve.

References

[1] Mullins, J., Komisar, R.: Getting to Plan B: Breaking Through to a Better Business Model. Harvard Business Review Press (2009)

[2] Crowne, M.: Why Software Product Startups Fail and What to Do About It. In: Proceedings of the IEEE International Engineering Management Conference, Cambridge, UK, August 18-20 (2002)

[3] Baron, J., Hannan, M.: Organizational Blueprints for Success in High-Tech Start-Ups: Lessons from the Stanford Project on Emerging Companies. California Management Review 44(3), 8–36 (2002)

[4] Brinckmann, J., Grichnik, D., Kapsa, D.: Should Entrepreneurs Plan Or Just Storm the Castle? A meta-analysis on contextual factors impacting the business planning performance relationship in small firms. Journal of Business Venturing 25(1), 24–40 (2010)

[5] Kakati, M.: Success Criteria in High-Tech New Ventures. Technovation 23(5), 447–457 (2003)

[6] Watson, K., Scott, S., Wilson, N.: Small Business Start-Ups: Success Factors and Support Implications. International Journal of Entrepreneurial Behaviour & Research 4(3), 217–238 (1998)

[7] Blank, S.: The Four Steps to the Epiphany: Successful Strategies for Products that Win, 3rd edn., Cafepress.com. (2005)

[8] Ries, E.: The Lean Startup: How Constant Innovation Creates Radically Successful Businesses. Penguin Group, London (2011)

[9] Furr, N., Ahlstrom, P.: Nail It Then Scale It: The Entrepreneur's Guide to Creating and Managing Breakthrough Innovation. NISI Institute (2011)

[10] Maurya, A.: Running Lean: Iterate from Plan A to a Plan That Works, 2nd edn. O'Reilly Media (2012)

[11] Adlin, T., Pruitt, J.: The Essential Persona Lifecycle - Your Guide to Building and Using Personas. Morgan Kaufmann Publishers, Burlington (2010)

[12] Campbell, D.: Software as a Service: spend and payment solution. Summit: Canada's Magazine on Public Sector Purchasing (May 25, 2010), `http://www.summitconnects.com`

[13] Holmström Olsson, H., Bosch, J.: Post-Deployment Data Collection in Software-Intensive Embedded Products. In: Herzwurm, G., Margaria, T. (eds.) ICSOB 2013. LNBIP, vol. 150, pp. 79–89. Springer, Heidelberg (2013)

[14] Vaishnavi, V., Kuechler, W.: Design Science Research in Information Systems (May 25, 2004), `http://www.desrist.org/design-research-in-informationsystems`

[15] Cockburn, A.: Agile Software Development. Addison-Wesley, Boston (2002)

[16] Larman, C.: Agile and Iterative Development: A Manager's Guide. Addison-Wesley (2004)

[17] Schwaber, K., Beedle, M.: Agile Software Development with Scrum. Prentice-Hall, Upper Saddle River (2002)

[18] Poppendieck, M., Poppendieck, T.: Lean Software Development: An Agile Toolkit. Addison-Wesley Professional (2003)

[19] Blank, S.: The Startup Owner's Manual: The Step-by-Step Guide for Building a Great Company. K&S Ranch, Inc. (2012)

[20] Serrat, O.: The SCAMPER Technique. Knowledge Solutions (February 2009)

[21] Fried, J., Hansson, D.H., Linderman, M.: Getting Real: The smarter, faster, easier way to build a successful web application. 37signals, Chicago (2009)

[22] Ellis, S.: The Startup Pyramid. Startup Marketing, `http://www.startupmarketing.com/the-startup-pyramid/` (May 25, 2013)