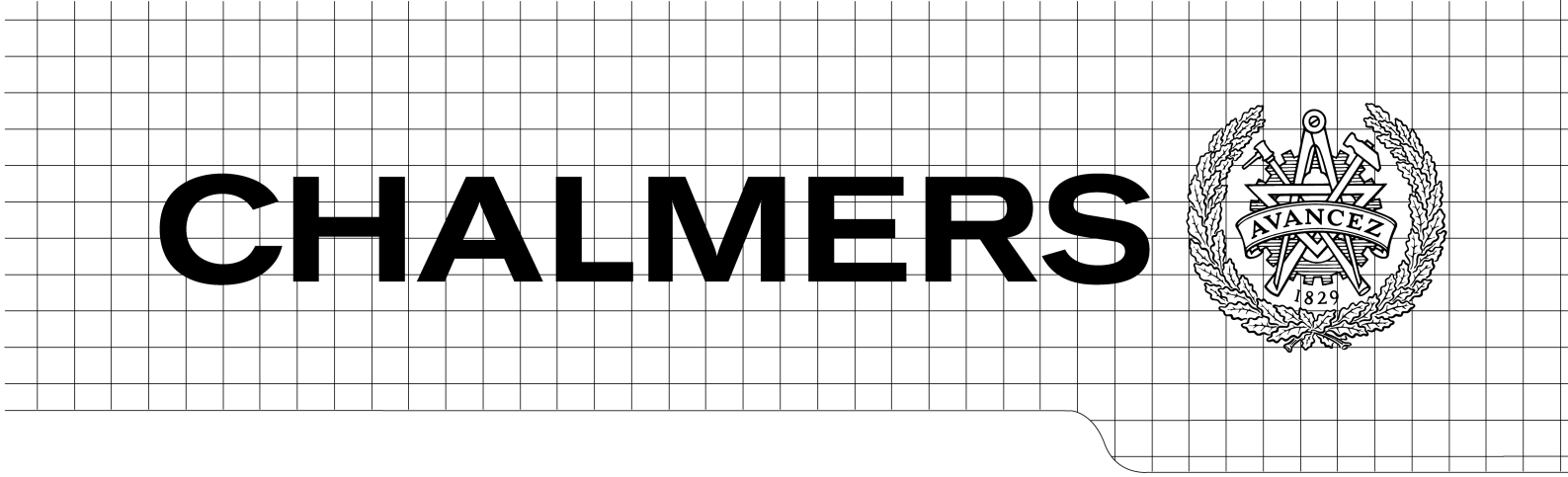

Managing Early Stage Software Startups

Applying Lean Startup Principles in Practice

Master of Science Thesis in Software Engineering

JENS LJUNGBLAD
JENS BJÖRK

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
Göteborg, Sweden, June 2013

The Author grants to Chalmers University of Technology and University of Gothenburg the non-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet.

The Author warrants that he/she is the author to the Work, and warrants that the Work does not contain text, pictures or other material that violates copyright law.

The Author shall, when transferring the rights of the Work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the Author has signed a copyright agreement with a third party regarding the Work, the Author warrants hereby that he/she has obtained any necessary permission from this third party to let Chalmers University of Technology and University of Gothenburg store the Work electronically and make it accessible on the Internet.

Managing Early Stage Software Startups
Applying Lean Startup Principles in Practice

JENS LJUNGBLAD
JENS BJÖRK

© JENS LJUNGBLAD, June 2013.
© JENS BJÖRK, June 2013.

Examiner: AGNETA NILSSON

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden June 2013

Abstract

Software startups are more popular than ever and growing in numbers. They operate under conditions of extreme uncertainty and face plenty of challenges, underlined by their high failure rate. Using Design Science Research, we set out to investigate (1) what the typical challenges are in terms of finding a product idea worth scaling, in early stage software startups and (2) what solution would serve to mitigate these challenges. A literature review showed that in recent years, several authors have suggested ways to increase the odds of succeeding as a startup, e.g. Customer Development and The Lean Startup. Interviews with industry professionals showed that working in a structured and organized manner is a big challenge, and that even though The Lean Startup offers guidance, many find the concepts difficult to implement in practice. The Early Stage Software Startup Development Model (ESSSDM) extends existing Lean Startup processes and is the suggested solution to the identified challenges. It is novel in that it supports investigating multiple product ideas in parallel, and provides clear criteria for when to move forward with product ideas. In addition, it gives advice on when to abandon product ideas and what techniques to use while validating them. Much of the model was used and successfully evaluated in a startup project co-founded by the authors. Further evaluation was done through interviews with industry professionals.

Glossary

A/B testing
: Two different versions (A and B) of the same design or feature is created. Users are then routed to one of the two versions, half of them seeing A, the other half seeing B. By measuring the performance of both versions, a decision on what version to use can be made.

AARRR!
: See Startup Metrics for Pirates.

Backlog
: A list of work waiting to be done, generally prioritized so that the most important tasks are located at the top of the list.

BML
: Build-Measure-Learn. A feedback loop for validated learning. Ideas are turned into products by building them, data is gathered by measuring how products are used by customers, and new ideas can then be formed from what is learned by analyzing the data [30].

Channels
: Channels are paths to customers, i.e. ways to reach them. Inbound channels are about getting found, through search engines, blogs, social networks etc. Individual customers find the product themselves. Outbound channels, on the other hand, are about reaching out to customers, such as through direct sales.

Conversion funnel
: A way to visualize conversion rates, e.g. sign-ups and purchases. Typically there is a decrease in numbers at each step, which is why it is thought of as a funnel. Example: there are 100 visitors to a landing page, 50 of them sign up, and of those only 10 make a purchase.

Customer pain
: See customer problem.

Customer problem
: A problem or problems that customers want solved. Some problems are more important than others; it is common to differentiate between ”nice to haves” and ”must haves”. A customer problem does not have to be a problem per se, it can just as well be a customer delight.

DSR
: Design Science Research. Research by way of iteratively designing/evaluating artefacts [35].

Encubation
: Advanced entrepreneurial education combined with real business incubation.

Inbound channels
: See channels.

Landing page
: Typically the first page reached when clicking a search result or an ad for a product. The purpose of a landing page is to get visitors to sign up for the product.

Lean Canvas
: A one-page business model format by Ash Maurya based on Alex Osterwalder's Business Model Canvas. An alternative to business plans [24].

MDD
: Metrics Driven Development. A methodology where data is favoured over opinion. Instead of relying on intuition for making decisions, metrics are gathered and analyzed [21].

MVP
: Minimum Viable Product. Typically the first version of a product released to customers. It contains only the absolute minimum in terms of features and design for it to become viable to customers [30].

Outbound channels
: See channels.

Pivot
: To pivot is to make a substantial change in direction, business model wise, while staying grounded in what has been learned (about customers) so far [30].

Problem/solution fit
: The point where a startup has found a problem and defined a solution that customers want [24].

Product
: A product or service is the solution a startup offers that solves the customer problem. More generally it refers to the entire business model surrounding the solution, including problems, UVP, what segments to target, revenue streams, channels etc.

Product/market fit
: The point where a startup has built a product for a market that wants it [24].

Product idea
: An idea that a startup has for a new product concept.

SaaS
: Software as a Service. A software delivery model where the application and its data is centrally hosted in the cloud.

Service
: See product.

Split testing
: See A/B testing.

Startup
: A human institution designed to deliver a new product or service under conditions of extreme uncertainty [30].

Startup Metrics for Pirates
: A metrics framework for software startups. The five metrics of interest are: acquisition, activation, retention, referral and revenue, or AARRR! for short [25].

UVP
: Unique Value Proposition. A single, clear, compelling message that states why a product is different and worth buying [24].

Validated learning
: A process for learning through experimentation [30]. See BML.

Contents

List of Figures

List of Tables

Chapter 1

Background

1.1 The startup phenomenon

New software companies are started each 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 Facebook, Twitter and Instagram contribute to their popularity and allure. However, contrary to what the media portraits, far from all startups succeed [11]. Among new product ideas, over 98% fail [29]. This has led researchers (e.g. Baron and Hannan [4], Brinckmann et al. [7], Kakati [20] and Watson et al. [36]) to try and identify what factors contribute to startups succeeding. In recent years, several authors [5], [30], [14], [29], [24] have embraced Lean thinking and customer focused development as the way forward. Whatever the reason for startups failing, statistics do confirm that being one comes with plenty of challenges.

To understand these challenges, we need to understand what constitutes a software startup. A popular definition, by Eric Ries [30], states that "a startup is a human institution designed to deliver a new product or service under conditions of extreme uncertainty." He puts no limit on the size of the company, arguing that startups might exist within big corporations. However, oftentimes startups have limited resources in terms of people and funding, and are run on tight schedules. In addition to that, they are commonly exploratory in nature, lacking clear requirements, customers and even business models from the get go.

With this in mind, being efficient and systematic is of high importance; efficient in terms of minimizing development effort spent while maximizing value gained, and systematic in terms of testing and monitoring if value is being generated. It is not enough to know how to *best develop* a product, it is just as important to know that you are developing the *right* product.

1.2 A new role for the software engineer

The software engineer working in a startup is often part of a small team. Lines between areas of responsibility tend to blur, and business and software concerns mix together. Being able to deal with business aspects is important, as is being comfortable with uncertainty. The ability to not only look at the solution and how to implement it, but to understand underlying problems and potential business models, will be key for software engineers working in future startups.

Chapter 2

Research questions and methodology

This thesis was written as part of a collaboration between the authors and Chalmers School of Entrepreneurship (CSE). The authors, together with master students from CSE, co-founded a startup that was run in an encubation setting (advanced entrepreneurial education combined with real business incubation) at Encubator AB for eight months. Described in this thesis are the results of the (applied) research conducted during this period.

2.1 Research questions

The following were the research questions investigated:

1. What are the typical challenges and problems in terms of finding a product idea worth scaling, in early stage software startups?
2. What solution would serve to mitigate the identified challenges and problems?

2.2 Research methodology

2.2.1 Research methodology overview

Design Science Research (DSR) [35] was chosen as the framework for the research. DSR differs from traditional research in that it focuses on learning through design, i.e. the construction of artefacts. The act of designing is, within DSR, used as a research method or technique.

Takeda, et al. [35] describes a model of the iterative design cycle, depicted in Figure 2.1. It comprises five phases. (1) Awareness of problem. Research proposal and research questions are formed. (2) Suggestion. Abductive reasoning,

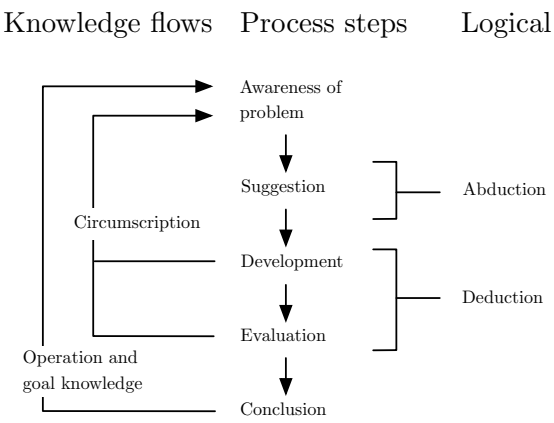

Figure 2.1: The iterative design cycle [35]

drawing from existing knowledge and theory within the field, leads to a suggestion of how to solve the problem. (3) Development. The suggested solution is realized in the form of an artefact. (4) Evaluation. The artefact is evaluated according to defined criteria. (5) Conclusion. When the artefact performs to satisfaction according to evaluation criteria, iteration stops and conclusions are drawn.

Being iterative, the model allows for moving back and forth between phases. If new information emerge during the development of the artefact, phase one and two can be revisited, and a new or modified suggestion formed. Similarly, during phase four, if evaluation criteria are not met, phase three is revisited and the artefact improved.

DSR was deemed a good fit due to the context of the research project. With the authors taking part in the forming of a startup, the design of an artefact aimed at mitigating typical challenges and problems seemed both interesting and relevant. Furthermore, the close proximity to a real-world startup meant the artefact could be rapidly iterated over/evaluated.

2.2.2 Research methodology instantiation

Awareness of problem. The research questions investigated were (1) What are the typical challenges and problems in terms of finding a product idea worth scaling, in early stage software startups? (2) What solution would serve to mitigate the identified challenges and problems?

Suggestion. A literature review was conducted, focusing initially on Agile practices and in later iterations on Lean Startup theory. In addition, semi-structured interviews with nine industry professionals in the Gothenburg region were carried out. The purpose of these interviews was to get a good understanding of how software startups typically work in the early stages, and if any patterns,

processes or best practices could be observed. The following criteria were used to select interviewees: (1) they should work at a software startup company with at least one market product (2) they should be the CTO of the company or have a similar position (3) they should have worked at the company from an early stage.

Interview sessions were approximately 60 minutes long, and two sessions per interviewee were conducted. The first session was more exploratory and took a broader perspective. The second session aimed to investigate, in further detail, what had been said during the first session and to better understand previous reasoning. Although an interview guide with template questions was written, structure was kept loose so that discussions were free to go in new and interesting directions [19]. All interviews were recorded, which has been shown to increase the detail richness and prevent data loss [19]. Additionally, the first sessions were fully transcribed.

Abductive reasoning [35] based on the literature and the interviews led to a set of problems (see chapter 5) and a suggested solution in the form of a process (see chapter 6). Interviews were analyzed by comparing the transcripts and marking commonalities. Identified common problems, together with gaps in literature, led to the problem statements while identified common solutions, together with best practices from literature, led to the first version of the suggested solution.

Development and evaluation. During the deductive stages, data was gathered mainly through participatory observation and reflective journals. The process was built for and evaluated on the aforementioned startup project. Additional evaluation was done through interviews with industry professionals. Revisits to the suggestion phase were frequent. In total, the process saw three major revisions, and multiple minor ones.

Chapter 3

Literature review

Throughout the DSR iterations, the focus of our literature review shifted. We concentrated initially on agile software development before emerging ourselves in the Lean Startup movement, its origins and, finally, practical implementations thereof.

What follows is a summary of topics, works and concepts relevant to the startup domain.

3.1 Agile software development

So called lightweight software development processes began to appear during the mid 90s, as a reaction against the heavier, waterfall driven processes that were the norm. Scrum and Kanban are examples of such lightweight processes. In 2001, a group of prominent software developers met and discussed many of these ideas, after which they published the Agile Manifesto. It is summed up in four brief statements, and the writers proclaim that while there is value in the items to the right, agile proponents value the items to the left more:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

What is key about agile methods is that they try to address the problem of things changing. Traditional methods, such as waterfall processes, do not respond well to change [34]. The reality is, of course, that things change all the time. Customers change their minds, the market changes, what we believed to be true a month ago is no longer so. Not only does things change, but the problems we as software engineers try to solve are often complex in nature, with many parts needing to fit together to realize a solution. Thus we are dealing with complex problems that are frequently changing, a recipe for failure if not handled properly [9].

3.1.1 Scrum

Scrum is one of the most popular agile development processes and is founded on empirical process control theory. Empiricism states that knowledge comes from experience and that decisions should be made based on what is known, not on what is believed. Empirical process control theory is a way to deal with "imperfect processes that generate unpredictable and unrepeatable outputs" [32] by prescribing frequent inspection and adaptation. In Scrum, inspection and adaptation is applied not only to the software product in development, but to the process as well [32].

Scrum, as many other agile methods, implements an iterative approach in order to maximize opportunities for inspection and adaptation. Instead of developing software by first gathering requirements, then creating an architecture and finally spend a year implementing it all according to a specification, the product is developed iteratively. By doing this, and by continuously delivering increments of working software every two weeks or so, a Scrum team inspects and adapts their product on a regular basis, making it possible to better respond to change, and to catch changing requirements as fast as possible. Working with requirements, architecture, implementation and deployment becomes a fluid, overlapping and parallel process. The iterations, or Sprints as they are referred to in Scrum, act as feedback loops. For every passing Sprint, valuable information on how the product is being received and used is gathered, which enables the team to better plan future Sprints, and to build the right product [32].

3.1.2 Kanban

Kanban was originally created at Toyota to reduce waste within their production system, using the "just in time"-approach [3]. The success of Kanban has led to adoption in other areas, such as in agile software development, where it has proven to be efficient [26].

Kanban contains six core practices: (1) visualize workflows, e.g. with a Kanban board (2) limit work-in-progress (3) actively manage workflows, e.g. monitor, measure, report (4) make policies explicit, so that everyone understands the process (5) implement feedback loops (6) improve the process collaboratively, with small incremental changes [3].

Kanban boards are used to visualize the workflow. On a Kanban board, columns represent task states such as "todo", "in progress", "completed" etc. Tasks are written on cards and put on the board. Work-in-progress is limited by only allowing a certain number of cards for each column [3].

3.2 Metrics driven development

Another methodology used by developers to reduce uncertainty and promote fact-based decision making is metrics driven development (MDD) [21]. Instead of relying on intuition, one should strive to base decisions on real data, gathered by testing and measuring the product. A/B testing (or split testing), for

instance, is a common MDD technique. When doing an A/B test, two different versions of the same design or feature is created. Users of the software are then routed to one of the two versions, half of them seeing version one, the other half seeing version two. By measuring the performance of the two versions, a decision on what version to use can be made [22]. Google, perhaps somewhat excessively, once tested which shade of blue out of 41 gradations was most likely to be clicked by users [18].

There is often some difficulty in understanding exactly what to measure. Measuring in itself is relatively easy, and there is the common notion of more data is always better, which has led to an increase in the amount of data being gathered. This, however, means the data is sometimes hard to analyze and make sense of. That is the reason why *actionable metrics* are being increasingly talked about [30] [24]. An actionable metric is one that is helpful when making decisions, offering clear guidance on what action to take next. Many common metrics, such as the number of visitors to a website, are often not actionable because it is unclear what to do with the information. An A/B test, on the other hand, offers clear action: deciding between the two (or more) tested versions [22].

While metrics driven development is increasing in popularity, many insist that intuition still plays a crucial role, and that it is important to find a balance between the two [30] [29]. Additionally, quantitative measurements require a large enough sample size in order to give statistically significant results [22], which might not be available in early stage startups.

3.3 The Lean Startup movement

Agile development processes are solution focused. That is, they are mainly applied in situations where the problem is well known/understood but the solution is not. In a startup context, however, uncertainty is even greater: both problem and solution are typically unknown/not well understood [30]. Agile answers how to build good products and fast, not so much what products to build. Engineers are solution minded, and this is reflected in existing research and writing. For the software engineer working in a startup, however, being focused on the solution is often not enough. A product is more than a solution, it is a business model, and in a startup the software engineer is often involved in both business and technical development efforts.

This customer and problem focused thinking has been advocated in the past by people such as Steve Blank [5], John Mullins and Randy Komisar [29], but has in recent years gained traction because of Eric Ries and the Lean Startup [30] movement. Ries noticed that, because of solution focused thinking, a lot of software startups were failing, including his own. It turns out, many were spending time and money developing products that people were not interested in. He calls this ”achieving failure”: successfully executing a bad plan. While projects may have been delivered on time and on budget, and with good design to boost, nobody wanted the product. This underscores the importance of understanding the problem before defining a solution. Being solution minded, we have a tendency to go ahead and start building, even before we know there

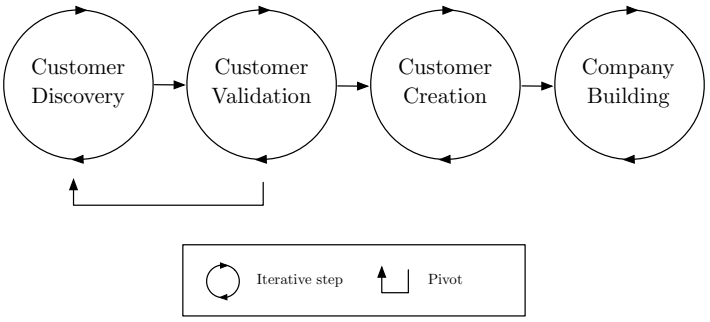

Figure 3.1: The Customer Development Model [5]

is a real problem to solve. Ries draws parallels with waste management in the Lean manufacturing process formulated by Toyota in the 90s. Lean specifies that everything that does not add value to the customer, is a form of waste and thus target for elimination. Ries captured his learnings first on his blog and later in his book The Lean Startup, in 2011 [30]. It has since turned into a worldwide movement that continues to grow.

While Ries can be credited with coining the term Lean Startup and bringing the word to the masses, his work is heavily influenced by, in particular, that of Steve Blank, who outlined the Customer Development Model in 2005 [5]. Others, such as John Mullins and Randy Komisar, contributed greatly to the field before Ries with Getting to Plan B in 2009 [29]. Likewise, Jason Fried and David Heinemeier Hansson touched upon many similar concepts with their book Getting Real, in 2006 [13].

3.3.1 Customer Development

In his book The Four Steps to the Epiphany [5], Steve Blank presents the Customer Development Model, which is further developed in his 2012 follow-up The Startup Owner's Manual [6]. Blank argues that the highest risk in building a business is not building the product, but finding people to pay for it. Startups generally do not lack products, they lack customers. Therefore, the traditional product centric development model, where a product is thought of, developed, beta tested then launched is flawed because it ignores customers up until product launch, which is also mentioned by Crisan and Nahirny to be a key factor in why startups fail [33]. The Customer Development model, on the other hand, considers customers from the start. It is a structured process for testing business model assumptions (or hypotheses) about markets, customers, channels and pricing. The model consists of four steps, where the first two mark the search for the business model, and the last two its execution. The steps are depicted in Figure 3.1.

All steps are iterative, and only when enough measurable progress has been made is it suitable to move to the next step. It is important to remember that

failure can and will happen and should not be viewed as failure per se, but as opportunities for learning.

The first step, customer discovery, is about capturing the vision and break it down into testable business model assumptions. A plan is formulated to test these assumptions with actual potential customers and turn assumptions into facts. To validate assumptions (or hypotheses) it is critical, Blank states, to "get out of the building" and face customers. Throughout his books, this is the one point he keeps on making over and over, stating it to be the single most critical lesson for startup founders to learn.

Customer discovery contains two distinct phases. First, customer perception of the problem is tested. That is, is the problem important enough for a large enough audience? Once that has been confirmed, a solution is built and shown to customers. Customer discovery is complete when the solution is good enough to persuade lots of customers to buy it.

During the second step, customer validation, the plan is validated with customers to make sure the business model is repeatable and scalable, enough so that it is possible to build a profitable company. The idea is to field-test marketing and pricing strategies before hiring sales and marketing people. Furthermore, customer validation is where customers actually start paying for the product, and, as Blank [6] puts it, "there's no surrogate for people paying for a product" in terms of validating it.

If validation fails, one goes back to customer discovery and tweaks the plan based on what has been learned so far. This is known as a pivot. It is assumed that a typical startup will go through the discovery and validation steps multiple times.

The third and fourth steps, customer creation and company building, is about building demand for the product, start scaling the business and transition from a startup to a full fledged company executing the validated business model.

In addition to the four steps, Blank outlines 14 rules as part of the Customer Development Manifesto. To demonstrate what they look like, here follows the first five rules:

1. *There are no facts inside your building, so get outside.* As previously stated, it is crucial to talk with actual customers in order to understand them. They live, usually, outside the building.
2. *Pair Customer Development with Agile Development.* "Customer Development is useless unless the product development organization can iterate the product with speed and agility." [6]
3. *Failure is an integral part of the search.* Failure is in most contexts something inherently negative and needs to be avoided. In a startup context, however, failure is simply part of the process. When searching for the right path forward, it is necessary to experiment and try many different things, which will inevitably lead to failures. It is the only way to make progress.
4. *Make continuous iterations and pivots.* Pivots are substantial changes in direction in the business model, such as going from freemium to subscrip-

tion, targeting men instead of women and so on. Blank phrases this as embracing failure. When it seems like there is no way forward, take a step to the side instead.

5. *No business plan survives first contact with customers, so use a business model canvas.* Except for financing, a business plan contains nothing but unproven hypotheses and is therefore useless. Many entrepreneurs believe it to be a "cookbook for execution" [6] which is simply not the case. A business model, on the other hand, is dynamic, flexible, and much better suited for startups. Osterwalder's Business Model Canvas is an excellent way to document such business models.

3.3.2 Getting to Plan B

According to John Mullins and Randy Komisar, new ventures operate under conditions of high uncertainty. In their book Getting to Plan B [29] they state that only one out of fifty-eight new product ideas turn into successful products. Furthermore, studies show that those who insist on sticking with their initial plan (Plan A) fail more often than those willing to adapt. As it turns out, many successful business end up doing something quite different from what they initially set out to do. The authors believe that by following a systematic process, there is a way to move from the initial Plan A to a better Plan B that actually works. There are four major building blocks in the process:

1. Analogs: "successful predecessor companies that are worth mimicking in some way." [29]
2. Antilogs: "predecessor companies compared to which you explicitly choose to do things differently, perhaps because some of what they did has been unsuccessful." [29]
3. Leaps of faith: "beliefs you hold about the answers to your questions despite having no real evidence that these beliefs are actually true." [29]
4. Dashboards: "a tool that drives an evidence based process to plan, guide, and track the results of what you learn from your hypothesis testing." [29]

A case study about Apple illustrates the first three building blocks. In 2000, Apple was searching for a new product. The iPod came to be after witnessing the success of Sony's Walkman, the famous portable music player — an analog. When Apple wanted to get into the business of selling music, Napster could be seen as an analog, proving that there was a huge demand for downloadable music. Although music on Napster was free, Apple took a leap of faith in that they believed people would pay for the music. As it turned out, people did, and the iPod and the iTunes store became big successes for Apple.

The fourth building block, dashboards, are used to test leaps of faith and other hypotheses in a structured and clear manner. They track these hypotheses and their key metrics over time. Quantitative measurements are highly favoured over qualitative ones. The key takeaway is that decisions should be made based on facts rather than assumptions. This includes leaps of faith: until tested and validated they are nothing but assumptions.

The book also mentions the danger of business plans in startups, and how such plans typically assume that everything is already known at the outset, and not learned as time progresses which is actually the case. They also contain very little except untested assumptions, and often turn a blind eye to the fact that new ventures operate under significant uncertainty. The future, as always, remains hard to predict.

Finally, the authors emphasize the importance of a good problem. The best ideas solve problems that constitute a major pain for customers. The role of the entrepreneur is to identify these problems, resolve the pain and make a profit in so doing.

3.3.3 The Lean Startup

Eric Ries published The Lean Startup in 2011 [30], wherein he states that entrepreneurship is a form of management. It is fundamentally different from traditional management in that the unit of progress is learning. That is, learning about customers and what they want. And because agile methodologies are not enough for this purpose, Ries brings in Steve Blank's Customer Development Model to fill the gap.

In a typical scenario, a company builds a product going in with many untested assumptions about their customers. Lots of code gets written that has to be thrown out six months later because the assumptions were wrong from the get go, and lessons were only learned once the product was in customer hands. Lots of time is spent arguing about what bugs absolutely must be fixed and what features are needed in version one, but the more pressing matter is: will customers even use the product? Ries argues that instead of producing code, there are other activities that might yield the same amount of learning, but significantly faster. By doing what he refers to as validated learning, experiments can be performed to test whatever hypotheses or assumptions we have. Building things that does not support learning is a form of waste.

The Pivot. A central concept within The Lean Startup is The Pivot, which is the term Ries uses for when a startup changes direction, but stay grounded in what they have learned (about customers) so far. He claims that having pivoted is the most frequently occurring commonality among successful startups. By reducing the time between pivots, it is possible to increase the odds of success, before running out of money.

Build-Measure-Learn. The Build-Measure-Learn (BML) loop (Figure 3.2) is a good way to visualize validated learning. Ideas are turned into products by building them, data is gathered by measuring how products are used by customers using various techniques, and new ideas can then be formed from what is learned by analyzing the data. One major iteration through the feedback loop constitutes a potential pivot. By reducing the time it takes to get through the BML loop, time between pivots can be reduced, and the odds of success increases.

The scientific method. Ries suggests treating this process as if one were a scientist: by applying the scientific method. Think in terms of learning experiments.

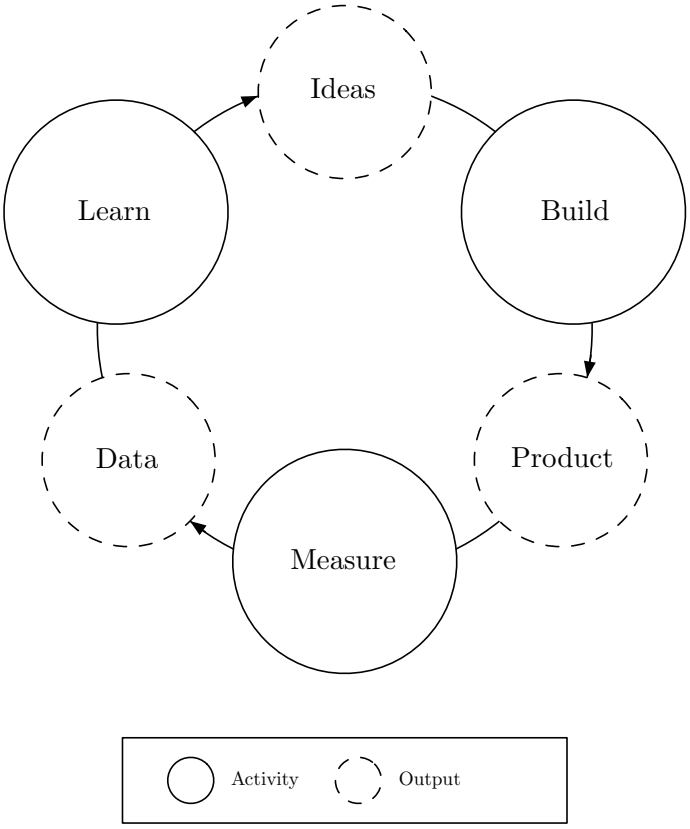

Figure 3.2: The Build-Measure-Learn loop [30]

By formulating falsifiable hypotheses (statements that can be proven wrong by empirical data) learning objectives can be defined up front. By running experiments, hypotheses are validated (proved valid/invalid), by analyzing the data typically leading to the formulation of new hypotheses.

Minimum Viable Product. The Lean Startup suggests many techniques for speeding up the BML loop time. One of them is building Minimum Viable Products, or MVPs. An MVP is typically the first version of a product released to customers, and should contain only the absolute minimum in terms of features and design for it to become viable to the customer, i.e. it solves the customer's problem. This is done so that the customer feedback process can begin as early as possible, shown by Crisan and Nahirny to be a key factor in why startups succeed [33]. Stripping a product down to its bare essentials to find what lies at its core is often a difficult, but worthwhile process.

Speeding up learning. More drastic approaches to speeding up learning would be to drive traffic to "fake" landing pages or product pages, thus testing customer interest in the product before even beginning to develop the MVP. Other techniques mentioned in Eric Ries's book are typical agile methods such as unit testing, continuous integration, utilizing open-source and so on. For measuring, A/B-testing is brought up as a particularly useful technique.

Innovation accounting. The final concept discussed is innovation accounting. It tries to answer the question of how can we, in a startup, be held accountable for what we do. How can we know what progress, if any, is being made, when traditional metrics do not apply. Ries terms these vanity metrics, and suggests focusing on actionable metrics instead. A vanity metric could be, for instance, the total number of messages sent over a technical platform. While the number might sound impressive, mapping it to something of actual value, such as revenue, is often not straightforward.

The Lean Startup proposes three learning milestones to focus on from an innovation accounting perspective:

1. *Establish the baseline.* Build an MVP. Measure how customers currently behave.
2. *Tune the engine.* Experiment to see if metrics can be improved from the baseline towards the ideal.
3. *Pivot or persevere.* When experiments reach diminishing returns, decide whether to pivot (change direction) or persevere (stay the course and hope for continued metrics improvements).

3.3.4 Running Lean

While The Lean Startup presents many interesting concepts and ideas, it can be difficult to understand how to turn them into practice. It is a philosophical book, presenting its lessons in the form of real world cases. This makes it difficult for beginners to get going. Ries himself has been quoted as saying: "I have heard an overwhelming demand for practical guidance for how to put Lean Startup principles into practice" [24]. Ash Maurya wrote the first edition of

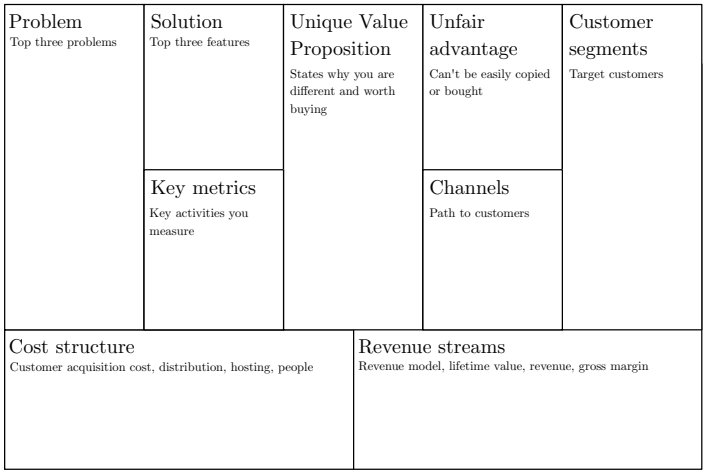

Figure 3.3: The Lean Canvas captures problem, customer segments, unique value proposition, solution, channels, revenue streams, cost structure, key metrics and unfair advantage. [24].

Running Lean in 2010 [24]. It is a rigorous process and handbook for creating Lean Startups, based on principles by both Steve Blank [5] and Eric Ries [30]. The process is divided into three steps:

1. Document Plan A
2. Identify the riskiest parts of the plan
3. Systematically test the plan

Documenting the initial plan is done in the form of a Lean Canvas (Figure 3.3), which is Maurya's version of the Business Model Canvas [24] [6]. The Lean Canvas captures and focuses on the entire business model, not only the product/solution. The solution box is kept intentionally small, so as to keep solution focused entrepreneurs from spending too much time there. The canvas is a living document, and is continuously updated as the plan iterates from Plan A to a plan that works. The canvas captures the vision of the business.

After having documented the initial plan, risks are assessed and prioritized. Highest prioritized risks should be dealt with first. Maurya lists three risk categories: product risks as in getting the product right, customer risks as in building a path to customers, and market risks as in building a viable business.

With an initial plan drafted up and risks prioritized, the rest of the process focuses on systematically testing and iterating over the plan using the scientific method and Ries's BML loop. Maurya defines four stages that the product moves through, where each stage has different risks associated with it, as well as defined exit criteria that needs to be fulfilled before moving into the next stage.

First stage: understanding the problem. The purpose of the first stage is to learn if the problem is worth solving, who has the problem, and what competition there is. The technique suggested for validating hypotheses concerning these risks is structured customer interviews. Running Lean provides ready made templates for such problem interviews. The exit criteria for the problem stage are, to name a few, being able to identify an early adopter, having identified a must have problem, and being able to describe how the problem is currently solved.

Second stage: defining the solution. During the second stage, learning experiments are focused around defining the MVP and figuring out pricing models. Techniques suggested are structured customer interviews (solution interviews) and creating demos, mockups, videos, or prototypes: anything that can communicate to potential customers how the MVP will look and function. Important exit criteria include being able to define the feature set for the MVP, and having a price the customer is willing to pay.

Third stage: validate qualitatively. In the third stage the MVP is built and tested on a small scale with early adopters. Learning experiments focus on finding out if the MVP demonstrates the Unique Value Proposition (UVP), and having a price the customer actually pays. Again, structured customer interviews is the preferred technique for validating hypotheses. The defined exit criteria is that 80% of early adopters make it through the conversion funnel.

Fourth stage: validate quantitatively. During the fourth stage the product is launched to a wider audience. Learning switches to understanding how to scale the product, how to build inbound customer channels and optimizing cost structures. Techniques also switch from relying on customer interviews to relying on measured data. Maurya deems the product scalable when 40% of users are retained.

Running Lean thus provides a process for applying Lean Startup thinking when developing software businesses. There are clear steps to follow and ready made templates for the structured customer interviews. The tools provided, such as Lean Canvas, are heavily focused on clarity and brevity, which Maurya believes to be key when it comes to raising the odds of success for new startups.

3.3.5 Nail It then Scale It

In 2011, Nathan Furr and Paul Ahlstrom published Nail It then Scale It [14], a handbook for creating successful, innovative new businesses. Innovation is defined by the authors as a combination of an invention (new or existing) with insight about a market need. Similar to Running Lean, the book presents a more practical approach when compared to The Lean Startup.

The authors begin by presenting what they call the entrepreneur's paradox: "if you act like an entrepreneur is (traditionally) supposed to, you actually increase the chance that you will fail." It is based on the following three observations:

1. Having passion, determination and vision are, contrary to popular belief, in fact dangerous traits for an entrepreneur. It frequently leads to

entrepreneurs falling in love with their products, turning a deaf ear to negative feedback and spend years developing something nobody wants.

2. Executing a business plan is not an appropriate process for startups, despite being taught by many business schools. Startups are about searching, not executing, and therefore the product development model is better replaced by the Customer Development Model [5].

3. Early stage funding, traditionally seen as something positive, can actually be damaging. Instead of focusing on validating assumptions with customers, early funding can be seen by entrepreneurs as a form of validation, and turn them towards the product development model.

With regards to innovation, there are two big risks. The technology risk (can we build it) and the market risk (will customers buy it). Startups tend to put too much effort into the technology risk, even though 90% of businesses fail because of a lack of customers, not because they failed to build their product. This is why the authors strongly favour a Customer Development Model, and a "get out of the building" mentality. As part of Customer Development, a fact-based decision making approach is promoted. Fast, cheap experiments are to test any hypotheses in a scientific manner. Instead of building fully functioning software, experiment using prototypes such as drawings, mock-ups etc which are shown and tested during customer meetings.

The NISI process is divided into five phases (although stage two and three run in parallel):

1. Nail the customer pain
2. Nail the solution
3. Nail the go-to-market strategy
4. Nail the business model
5. Scale it (not within scope of this article)

Phase 1: Nail the customer pain. The goal of phase one is to define and understand the customer pain, and to determine if that pain is a market opportunity. The phase is divided into four steps. Step one is to formulate a monetizable pain hypothesis about a believed customer pain. In order for a startup (typically without track record, brand and reputation) to attract attention, the customer pain needs to be big. Unless the pain is big enough, the NSIS process will not work. Step two is to formulate a big idea hypothesis, which is the plan for solving the problem. The authors suggest following a bullet point format from Crossing the Chasm [27] which is similar to Maurya's Lean Canvas [24]. In step three, the hypotheses are tested by making cold calls (B2B) or email sendouts (B2C) to prospective customers. The goal is to have >50% success rate after pitching the customer pain. If unsuccessful, tweak pitch or customer segment until successful. Step four is a quick exploration of market dynamics and competition.

Phase 2: Nail the solution. The goal of phase two is to discover the minimum feature set (MFS) that drives customer purchase. In order to find the core of

what drives a purchase, it is necessary to keep the MFS to an absolute minimum. The phase is focused around three sequential tests. During test one, a customer profile is developed and a rapid virtual prototype built. Customer interviews are performed and the MFS further developed. During the second test, a functioning prototype is built and tested on many customers during a prototype road show together with the team. MFS further refined and price points discussed. A market segment for initial launch (early adopters) is defined. During the third test the solution is developed together with selected pilot customers. The product is built to suit the needs of the early adopters. When demonstrating, customers must be excited about the solution and willing to pay. Price point at this stage needs to support a viable business.

Phase 3: Nail the go-to-market strategy. The third phase run in parallel with phase two, and the goal is to learn how customers would learn about and purchase the product. It is organized around the same three tests as in phase two, and the following questions are explored during those same customer meetings. Test one: customer buying process discovery. How do customers learn about new products and what are the channels they go though? How do customers evaluate solutions and what drives them to purchase? Test two: market infrastructure discovery. What does the infrastructure look like between the product and the customer, i.e. partners, social media, ads. Who are the key players? Test three: build customer relationships. Turn interviewees into pilot and reference customers.

Phase 4: Nail the business model. The goal of the fourth phase is to develop a repeatable business model. The authors strongly suggest avoiding business plans, which are very product development minded, and instead focus on creating a business model. Success in this regard depends on two factors: continuous data flow and measuring the right things (i.e. actionable metrics), with continuous data flow meaning continued customer interaction. As distribution channels, revenue streams, customer acquisition costs, customer lifetime value, break-even points, etc are measured and analyzed, the entrepreneur can estimate whether the business is viable.

3.4 Conclusions

The works presented in this chapter all build upon each other with many ideas being shared by the different authors. First and foremost, there is a heavy preference towards customer focused development, moving away from traditional product or solution focused development. It is seen as absolutely critical to bring on potential customers as early as possible, in order to validate the product concept and business model, and deal with the high amount of uncertainty that comes with being a startup. Moreover, focusing on really understanding the problem, by going out and talking to people and potential customers, is a point that is made over and over again.

Another frequently occurring idea is that of the pivot, and that the one shared commonality between successful startups seem to be that they have pivoted: few successful companies are doing today what they once set out to do. They

have gone from an initial plan to a plan that works. Related to this, failure is not seen as failure, put as part of the startup learning process. Failure is an opportunity for learning and for pivoting.

Decisions should be taken based on facts, not assumptions or guesses or faith. This lies at the core of agile methodologies such as Scrum, but also features heavily in Lean Startup literature. Iterative development supports such decision making, as does metrics driven development, as does validated learning. Actionable metrics is a concept that is gaining significant attention, as is tackling the learning process as if one were a scientist, using the scientific method.

A general leanness is advocated, both in terms of developing the software (agile) and in developing the business: minimize waste, i.e. anything that does not provide customer value, employ just-in-time decision making, avoid hiring people before they are really needed, construct MVPs, do not spend time on things that do not contribute to learning etc.

Finally, traditional business plans are criticized for being little more than untested assumptions, often mistaken by entrepreneurs for being "cookbooks of execution" [6]. Such thinking is one of the reasons why many cling to their initial plan, instead of pivoting to a more successful one. The problem with business plans is confirmed by Alvarez and Barney [2] who mentions that too much planning early on is a waste of resources. In addition, a study of 116 startups by Lange et al [23] showed that writing a business plan in an early stage had no positive effect on the performance of the startup. Many software engineers have embraced agile thinking, and understand that the future is near impossible to predict, and that we instead have to get better at responding to change. The same goes for business developing: the future cannot be predicted, and thus traditional business plans have limited purpose. Business models in the form of business model canvases are touted as a much more appropriate tool for lean startups.

Chapter 4

Industry practices

This chapter presents the results gathered through the conducted interviews with industry professionals from nine software startups in the Gothenburg region, as described in chapter 2.

For each startup, the following will be discussed: (1) *Context.* Area of business, technology platform, type of product, size of company, year founded. (2) *Development practices.* Business and software development practices. How the company conducts its operations. (3) *Problems/challenges.* Things that are viewed either as problematic or challenging when running a startup.

4.1 Appello

Context. While Appello might no longer be considered a startup, having existed since 2004, they still operate much in the same way. The core team were originally part of a company doing services for the automobile industry that was in turn bought by Framfab. There, they experimented with different concepts, including map based navigation solutions, which is what Appello eventually ended up doing. They are currently 25 employees, eight of which are software developers. While they have two products in the market, they are always looking for new opportunities where they can apply their domain expertise; they actively try to avoid becoming a one-product company.

Development practices. The company is trying to build a culture of innovation, with new ideas and product concepts constantly being developed. Being innovative is considered a competitive advantage, due to the fact that competition within the domain is increasing. The overall process of evaluating new ideas can be seen as a funnel, with gates along the way that filter out the ideas not worth pursuing. The first step is doing light research. Many ideas are filtered out already in this stage. Next, the product team selects the most promising ideas and performs *product assessments*, a more in depth research stage where they look at: purpose of the product, which partners to work with, competitor analysis, similar products, revenue potential, time to market etc. At the next gate, management looks at the product assessments and decides which ideas can

move into development/prototyping. The further down into the funnel, the less ideas can be worked on simultaneously. If a product idea fails in a late stage, other ideas can be picked up from where they were left off. When documenting these ideas, the company tries to avoid extensive business plans, instead opting for short slide-decks.

The company works with agile practices, such as Scrum and Kanban. Scrum is used by the development team, with three week sprints, while Kanban is used by the maintenance team. Kanban is considered too light/small when working in bigger teams, i.e. more than a couple of people. The company also has scheduled "creative time" where people are free to switch from their current tasks in order to do exploratory work, e.g. on new ideas.

Problems/challenges. The company states that it is hard to compare and evaluate ideas in the early stages because the products differ so much, particularly when it comes to goals and business models. They believe that in order to compare ideas, they must be built and then used by customers for a while. However, the company is interested in finding better ways to compare and evaluate ideas as early on in the process as possible.

In the early stages of an idea, it is difficult to focus on the underlying problems, without thinking too much about a solution. There is always a risk of doing too much, too early. Programmers, for instance, tend to start coding as soon as possible. Moreover, the company considers it hard to do rapid prototyping and continuous deployment on mobile platforms and marketplaces. Expectations on quality is also higher today than ever before.

The company want to work and think more like a startup in terms of flexibility and mindset, but think it hard to achieve due to having a lot of existing customers, as well as existing products to maintain. It is believed that having listened too much to customers may have limited to amount of product innovation over the years. In addition, convincing investors to spend resources on exploratory work that was not part of the initial investment is seen as a challenge.

Finally, it is hard to know when and how much to scale. This became evident when the company realized they had grown too much, and had to scale down, something that is also difficult.

4.2 Burt

Context. Burt creates web based analytics tools for the publishing industry. The company was founded in 2009 by people well acquainted with the domain, having worked within advertising. Since the start, they have grown from five to 28 employees. Such speedy growth can be attributed to the fact that the market is a niche one, and competition has been scarce.

Development practices. The company believes that their products must be founded in a solid problem with high customer pain level. They stress the need of concentrating on the problem and not the solution in an early stage. As

a rule of thumb, a problem that customers are not willing to pay for in advance to have solved, is deemed not critical enough.

The company made two major pivots early on, switching customer segment and product focus, while remaining within the domain. Initially the product was meant for copywriters, enabling more dynamic ads. The first pivot was doing the backend for serving such ads, and the second to build an analytics tool for such backends. For a while, all three products ran simultaneously, before the decision was made to focus on the analytics tool.

After coming up with ideas for new products/features, they are communicated to customers through sales material or HTML-mockups. The company then finds one or two customers that they work closely together with while developing the idea. They argue that it is important to build for specific cases initially, then evaluate the result to see if the idea is worth pursuing. Developing such HTML-mockups takes 1-2 days and they are kept relatively rough. The need for detailed mockups is lower in the B2B segment, the company believes, due to the high amount of shared domain knowledge. Not all mockups are implemented, about one in three is the typical ratio.

The company has a product development team and a software development team. The product team is responsible for product direction, features and design, while the software team is responsible for implementation and metrics measuring. In addition, there is a labs team, responsible for experimental work on new ideas and concepts. Each experiment is assigned three people, and two experiments can be conducted in parallel. The company believes having a labs team generates rapid learnings about new ideas.

Problems/challenges. Getting the amount of process right can be difficult in a startup, especially in the early stages. It is important to be structured, but too much process can harm productivity. While it is relatively easy to implement process changes in the early stages, this becomes much harder as more people are added to the team. Related to this, knowing when to scale and how much to scale is hard.

Communication within and between teams is also a challenge, especially within startups, which are typically quite chaotic. In addition, cross-functional teams, a technique from agile software development, may sound good in theory but are difficult to implement in practice. Oftentimes, experts are required, and if the team is constantly rotating responsibilities, no one becomes really good at one particular task.

4.3 Destly

Context. Destly is a web based travel service offering deals on trips and hotels. Similar services exist, but not in Sweden. The purpose of the venture was to find an existing, successful business model that could be applied in a new segment. The company consists of two business developers (the founders) and one software developer, and was started in 2010.

Development practices. As the company wanted to copy an existing concept,

they looked at areas where they had passion for the domain, and where there was a scalable business model to be found. In general, they practice Lean Startup and Customer Development principles: hypotheses, experiments, early customer interviews, MVPs etc.

After having done the initial market research, the company set out to talk to a lot of people in order to get feedback on their idea and to find the MVP feature set. For this purpose, a low-fidelity prototype in the form of a slide-deck and a pitch was all that was needed. The goal was to secure 3-4 customers who would advertise on the service, before they started building the MVP. Being able to sell the product before building it validated the potential of the business model.

The MVP was built rapidly in order to get early feedback from customers. At the same time, the company feels that it is important not to be sloppy in terms of quality; customers buy products they *feel* are well executed. The two guiding principles when building the MVP were, therefore, usability and simplicity.

In general, short decision paths within the company are touted as the main competitive advantage for small startups. By being small and nimble, it is possible to compete with much larger, and much slower, organizations. By applying lean principles and working smarter, it is possible to speed up learning while minimizing waste. For instance, the company frequently uses analogs — looking at existing solutions and how they are received in the marketplace — as a way to (partially) validate hypotheses.

Problems/challenges. Communication within the team is crucial, and worth putting extra effort into. It is important for everybody to feel involved in what goes on at the company, and to believe in the company vision. Communication is often challenging, and even more so in the chaotic world of startups.

The company tried outsourcing development, but found it difficult when trying to iterate rapidly over the MVP. Outsourcing may work in some cases, but when building a product in close collaboration with customers, it is useful to have the software developers nearby.

It is easy to spend too much time and resources on the wrong things. Design and brand are, contrary to what some believe, not the most important things in an early stage. Instead, copy design and copy brand until at least the product concept has been validated.

4.4 Duego

Context. Duego is a social networking service for emerging markets, founded in 2010. Initially it was geared towards dating on the web (for a younger, trendier audience than your usual dating site) but has since transitioned into a mobile-first meet new people service. The founders sought to find and copy an existing business model that could be tweaked and applied in new, emerging markets. The company employs around ten people.

Development practices. Several potential product ideas in the emerging markets segment were researched. The company looked into what existing concepts could be copied, current competition, potential for growth, attractiveness from their own perspective, and whether the idea could be implemented remotely, i.e. from Sweden. After having picked the most promising idea, investments were secured and a team assembled.

No proof of concept was created or tested on users prior to building. Intuition was deemed sufficient, mainly because the product copied existing (successful and therefore validated) concepts. Also, since the user base existed in remote countries, face to face meetings were unpractical.

The first version of the product was built during a period of six months. Seeing how it was to be heavily marketed once released, it was crucial for it to be as polished as possible; beautiful design and simplicity were the guiding principles. A year after releasing the first version, the company pivoted. From having targeted five countries and desktop-first, to targeting a single market and mobile-first. As part of the pivot, the scope of the product was reduced and a much smaller MVP created.

Strategic decisions regarding the product are based on a combination of intuition and quantitative data analysis. The company employs a variety of data collection techniques for measuring user activity, including Google Analytics, A/B-testing and an in-house event tracking framework. The company believes the amount of intuition versus data that plays into decisions is dependent on what phase the startup is in. While metrics are always important, experimentation and intuition plays a bigger role in a user acquisition phase than in a growth phase.

The company uses many agile practices and follows Scrum, with two week iterations, in order to keep the development team organized and focused. External pressure stops at management level, so as to keep the development team undisturbed.

Problems/challenges. The concept of an MVP is difficult to put into practice when going up against larger, established players. The company feels that it is very hard to build something that would only solve a fraction of what the competition does. Also, releasing a rough or unpolished MVP is not an option when it is to be heavily marketed.

Along similar lines, the company feels that some of the Lean Startup thinking is not applicable when building products that are dependent on a network effect. When the value of the product can only be demonstrated after reaching a critical mass of users, perhaps a startup needs to scale *before* reaching product/market fit, or "scale it then nail it".

Finding the correct amount of process is challenging when starting up a company, especially when assembling quite a large team. It takes time and experimentation until things move fluently and with good pace.

Some things that sound nice in theory are difficult to prioritize when you are in the thick of it. Making time for innovation in the early stages can be hard, because time is necessarily spent trying to make ends meet. Another challenge

is pivoting, which is seen as something very difficult when you have investors and operational costs to consider.

4.5 Evisto

Context. Evisto is a web based personal finance management solution, founded in 2012. The product concept is copied from a successful service in the US called Mint. The company currently employs three people: two business developers and one software developer. The product is invite-only for the time being, and has ∼900 users.

Development practices. No extensive investigation validating the business model was conducted; it was assumed that the model would work just as well in Sweden as in the US. The company spent three months developing their MVP. They focused their efforts on copying Mint, and prioritized technical risks first, such as synchronization with banks and categorization of transactions. They also prioritized scalability from the start. Since the launch of the MVP, the company has continued to work on nailing the core functionality.

When prioritizing features, the company keep their tight budget in mind, and focuses on creating as much value as possible given a defined period of time. Intuition is what most decisions are based on, but when it comes to more complex problems they go out and talk with customers. They follow the approach where they test, book a meeting, evaluate and then iterate. They also encourage feedback from their users, making it easy to get in touch with the team through the product, e.g. using services like User Voice.

In terms of metrics, the company look at how often users log in and how often they use the core functionality within the product. As an acquisition strategy, they drive traffic to their landing page using Google Adwords. Once the customer lifetime value (CLV) is higher than the user acquisition cost (UAC), the time is ready to scale the product.

Problems/challenges. Trying to work in a Lean Startup way is hard. Listening to customers is crucial, but there is a risk in listening too much. In order to succeed, the company believes that startups need to be founded in strong opinions.

Being data-driven is another difficulty, as it requires a lot of time and resources; sometimes it is better to choose a direction based on intuition in order to get things done. To confirm that a product concept works, the company believes that between half a year and a year is not uncommon. It is not only building the software and gathering the feedback that takes time, but also building customer relationships.

It is important to quickly recognize and correct bad decisions, even though it is sometimes emotionally hard to abandon or change something that time and effort has been spent on.

Knowing when to seek investors is hard. It is easier to negotiate when the concept has been proved, so the longer it can wait, the better. It is also harder

to pivot once investors are involved.

4.6 Lean Machine

Context. Lean Machine, started in 2012, builds a customer feedback analytics tool for businesses. The two founders have recently hired a third member to their team. The product is web based, but with a hardware component (an iPad) acting as the interface towards consumers. The vision is to create multiple products within the B2B segment, thus avoiding becoming a one-product company.

Development practices. The company generally tries to follow Lean Startup practices. Initially, they went out and performed exploratory interviews with potentially interesting customer segments, trying to find big enough problems to solve. The company believes that any product must be founded in a problem with a high customer pain level. Interviews were structured so that half and hour was spent exploring, while the second half was spent honing in on the most interesting problem.

After having performed interviews and analyzed the data, the five most interesting ideas were investigated, largely in parallel. Market research was conducted, and solutions thought of. HTML-mockups were created in some cases and shown to customers in order to gather feedback. From past experience, the founders felt that feedback is more earnest before the product has been built. Working with ideas in parallel was considered efficient and useful, although concerns were raised that ideas might be abandoned prematurely when working in this way.

At this stage, the ideas were prioritized in order to pick one that could be implemented. It was felt that not more than one MVP could be built and maintained at the same time. Prioritization criteria included: market potential, cost per unit, geographical location of customers, speed to market, the product being something used by people in their everyday lives, and, most importantly, the product and domain being something the founders felt passionate about.

The MVP was built and testing began on early customers. The company believes that qualitative feedback is preferred in the early stages, and that it is important to work closely together with customers. Sometimes, startups implement quantitative metrics measurements just because they can.

Problems/challenges. Working in a structured manner in a Lean Startup is seen as a challenge. Startups are inherently chaotic, and when doing Customer Development, a lot of information needs to be processed. It was felt that as more and more customer interviews were conducted, the results became difficult to keep track of. Documenting in general, including documenting decisions and experiment results, was a problem and sometimes forgotten.

It was difficult to keep a good separation between problem and solution. Even though much of the literature claims that the problem should be investigated on its own, it was hard to do so without looking at potential solutions.

Only one potential solution was mocked and tested on customers for each product idea. The company felt that perhaps because they worked on multiple ideas in parallel, too little effort might have been spent investigating alternative solutions within each product idea. On a similar note, the company felt that working on more than one idea at the same might lead to ideas being abandoned prematurely.

Finally, some critique against the Lean Startup was voiced. The company felt that it is most suited for people who value creating a profitable business over feeling passionate about the product itself. The founders felt that while creating a profitable business is obviously a requirement for the company to succeed, an additional requirement on their behalf is to have fun while doing so.

4.7 Let's Deal

Context. Let's Deal is a web/mobile deal-of-the-day service, founded in 2009. It is a copy of Groupon, a similar service from the US. The company employs 65 people.

Development practices. The two founders started off by brainstorming ideas, with the goal of creating a business they thought fun. They had multiple ideas in the early stages. Some were investigated, but most were abandoned for various reasons. Ideas were prioritized based on: how interested the founders were in the domain, time to market, development costs, scalability, competition, technical feasibility and intuition. Two promising ideas emerged: a copy of Mint (see Evisto) called Mynto and a copy of Groupon, which later became Let's Deal. Copying existing services from the US is a good way to kick-start a business, according to the founders.

Mynto and Let's Deal ran in parallel for half a year, with both ideas being pitched and tested in the market. Mynto was slow and complex to work with, both from a technical and from a business perspective. Let's Deal, on the other hand, revealed its potential early on and started to gain traction. The product was was much easier to build, with the MVP only taking three weeks to complete. When competitors to Let's Deal started to show up (e.g. CityDeal who was later acquired by Groupon), a choice had to be made and the company concentrated all their efforts on Let's Deal, even though they were interested in both products. Mynto was put on hold, not to be continued. Had CityDeal or similar competition been around from the start, maybe the founders would have gone for that product instead.

The company works with a modified Scrum-process, tailored to their needs. They work quite short-term oriented and occasionally schedule slack time to do creative sessions, i.e. coming up with and researching new ideas. They believe that it is important to document decisions and to prioritize tasks. Furthermore, the company tends to work mainly on urgent matters, trying to cut as much waste as possible and not be "too smart too early". They measure all their users' behavior and the feedback drives what deals to focus on. A/B testing is considered resource heavy and currently used only for GUI optimizations. A/B testing features is something the company is looking to do at a later stage.

Problems/challenges. While Let's Deal proved to be a successful concept in an early stage, when big competitors came to the Swedish market it created a challenging situation for the company. They had to raise funding in order to keep up with the aggressive marketing from CityDeal (and later Groupon).

Another challenge was to see if an existing, successful concept from the US would have the same impact in Sweden. The company did nothing to validate their assumptions prior to building the MVP. However, since both founders had been living in the US previously, they felt they had a good understanding of how the societies differed.

The company states that is is difficult to work in a structured and organized manner. Looking back, however, patterns of structure can almost always be gleaned in retrospective, even though they are hard to see in day-to-day work.

4.8 Saltside

Context. Saltside builds a classifieds product for emerging markets such as Pakistan, Bangladesh and Sri Lanka. The concept is copied from successful services such as eBay and Blocket. The company employs 27 people, of which 17 are based in Gothenburg, of which ten are software developers.

Development practices. The idea originated during the market research for Duego, described in section 4.4. After having secured investments and assembled a team, three months were spent building the MVP. During its construction, it was optimized for the *key success factors* that the company believes in, such as speed and uptime.

In deciding which countries to launch in, different metrics were looked at, such as GDP per capita, total GDP, number of Internet users, population size, competition etc. It is believed that in order to succeed in the market, the company must outspend the competition when it comes to marketing because classifieds products require a network effect in order to become valuable. In such scenarios, it can be harmful to do too much validation of the concept prior to scaling. When the concept is well known and has been proved to work many times over, it is a matter of taking a position on the market more than validating the concept.

The company looks at metrics in order to measure their success. They focus on different KPIs each quarter and the KPIs are derived from the visions and goals for the year. The product team, for instance, looks at KPIs such as pages per visit, time on site, conversion, page speed etc. In addition to looking at KPIs, the company conducts usability tests in terms of seeing how people interact with their products, does A/B testing for the GUI, and they believe that reading support tickets is important for everyone in the company.

Problems/challenges. One of the most difficult things is prioritizing the backlog, and deciding what to focus on in upcoming sprints. Such decisions are based on a combination of data, intuition and what the strategic goals for the company are.

It is dangerous to rely too much on KPIs, or to be too data-driven. Being data-driven is about optimization, whereas having a strategic perspective allows you to see new opportunities, otherwise easily missed. This can be likened with the hill climbing technique: while being data-driven takes you towards a maxima, there is no saying whether it is local or global. Leaps of faith are required in order to reach potentially higher hills.

Another challenge for the company is that they are not located geographically close to their customers, although they do have people working in these countries, and who help out with marketing and communication.

4.9 Shpare

Context. Shpare develops a web/mobile product that makes it easier for people to network at conferences and trade shows. The vision is to create the highest quality face to face meetings between people. The company was started in 2011 and currently has three employees, of which two are the founders.

Development practices. The company follows Lean Startup practices, and stresses the importance of getting out of the building and talking to customers. Instead of going "stealth mode" in fear of the idea being stolen, pitch it to as many as possible. Ideas are a dime a dozen, execution is what matters.

Initially, the idea was to offer a way for people to schedule their free time together with their friends. An MVP was constructed (300 hours) and tested on potential customers. It turned out that although people had responded well to the idea when pitched, they were unwilling to schedule their free time. The company pivoted, and applied their product in another context: that of business networking during conferences and trade shows. This demonstrates the importance of getting the MVP out to customers as soon as possible.

The quality of the MVP was kept intentionally low in order to speed up the process. The company believes that quality is not the most important thing in the early stages, getting feedback on the concept is. If a startup is afraid of losing customers over a low-quality MVP, they have too small a market in the first place.

The company uses the Lean Canvas business modeling technique from Running Lean [24], believing that writing business plans is wasteful. The iterative nature of a business model canvas is much more suited to the fast paced startup world. Furthermore, they are hypothesis driven, typically validating their assumptions with experiments. Experiments must be easy to execute and, after having conducted one, it is very important to sit down and analyze the results and what they mean.

An example of such an experiment is the product's matchmaking algorithm. The assumption was that receiving a (short) list of suggested people to contact would increase networking. Instead of coming up with an algorithm, the company populated the lists with randomly selected people. The feedback from this experiment dictated what was prioritized when implementing the actual feature.

Continuous deployment is touted as an important technique in order to speed up learning. Being able to work on and introduce changes during conferences, when the product is being actively used, is and has been instrumental to the company. Also, instead of debating whether to include something in the product or not, release it and test it with customers. Working with small batches makes it easier to throw away things that turn out bad.

In order to know what metrics to focus on, the company uses a concept known as Startup Metrics for Pirates [25]. The five metrics of interest are: acquisition, activation, retention, referral and revenue, or AARRR! for short. These metrics must be dealt with in that order; the product is optimized for acquisition before moving on to activation, then retention and so on. This keeps the company focused on the most important metric.

Problems/challenges. It is difficult to work in a structured and organized manner in a startup. Even though the company is hypothesis driven and works with experiments, data can be fuzzy and a lot of decisions are based on intuition.

It is easy for startups, even those following Lean, to build too big of an MVP. This has been the experience of the founders when working on previous ventures, which is why they stress the importance of keeping the MVP as small as possible.

Having a product that is used infrequently, in this case only during conferences and trade shows, makes the feedback and testing process challenging. Instead of having a continuous stream of data, with changes to the product being rolled out at a steady pace, the company must be present during conferences and trade shows in order to iterate and gather as much feedback as possible.

4.10 Conclusions

From a software development perspective, all companies used agile practices, especially Scrum and Kanban. From a business development perspective, a few companies were aware of Lean Startup methodologies and worked in that manner, but most were either unaware or found it difficult to apply in their situation, e.g. it is viewed as too abstract, and hard to implement in practice. Some did, however, follow principles similar to Lean Startup, without necessarily labeling it so. That includes working closely with customers and pivoting towards product/market fit.

Of those not following Lean Startup practices, few worked actively with validating product concepts early and often with customers (trying to pinpoint underlying problems) before building and scaling a solution. In some cases this was due to products and business models having been copied from existing ones, to be applied in different contexts/countries, thereby reducing uncertainty and the need for extensive validation. Also, the opinion was voiced that Lean Startup is difficult to apply in situations where the product is depending on a network effect. In such cases, scaling before reaching product/market fit might be necessary.

Startups that did put a lot of effort into understanding underlying problems either followed Lean Startup or created new products, i.e. not copying existing ones. Those same startups had also pivoted the most.

Many proclaimed to be data-driven to some extent, keeping close track of various metrics. Even so, most strategic decisions were based on intuition and gut feeling. Many dabbled in A/B testing of their user interfaces, but this was mostly viewed as an optimization technique. No one A/B tested features. Those following Lean Startup did perform experiments using validated learning but admitted it was difficult to base strategic decisions on data alone. Thus, there is still ways to go before startups are truly data-driven, or apply fact-based decision making.

It became apparent that there is an early-stage process not heavily discussed in the literature, where different product ideas are weighed against each other before a decision is made on what product to develop. This often happens prior to the forming of the company. A structured approach to tackle this task seemed to be lacking. Some startups brought this early-stage idea selection process further, by actively investigating multiple ideas in parallel even after forming the company.

Startups are run in many different ways and there are many different types of startups. On the software development side, all interviewed companies adhere to agile methods, but on the business development side, few agreed upon best practices could be observed. What all *can* agree upon, however, is that it is difficult to work in an organized and structured manner in an early stage software startup.

Chapter 5

Problem statement

In chapter 2 we presented the research questions:

1. What are the typical challenges and problems in terms of finding a product idea worth scaling, in early stage software startups?
2. What solution would serve to mitigate the identified challenges and problems?

After having conducted a literature review and interviewed industry professionals, we came to the following conclusions. (1) It is difficult to work in an organized and structured manner in an early stage software startup. The Lean Startup offers some guidance, but many find the concepts hard to implement in practice. (2) A process that supports implementing Lean Startup principles in practice would serve to mitigate this difficulty. Although some authors [24] [14] claim to provide such processes, we have identified some key areas where improvements are needed.

Several companies (see sections 4.1, 4.6 and 4.7 worked with or expressed an interest in investigating multiple product ideas in parallel, in order to improve decision making during the idea selection process, as opposed to picking an idea based on intuition. The purpose of the idea selection process is to find *one* product idea worth scaling. The literature advocates fact-based decision making over intuition, but assumes that the idea selection process has already taken place and that only one idea is being worked on.

In addition, working with multiple product ideas in parallel with the purpose of finding *one* idea worth scaling, gave rise to the need to know when to abandon product ideas. This is also not covered in the literature.

These observations led to us formulating the first and second problem statements:

1. Existing processes and theories do not adequately support working on, or investigating, multiple product ideas in parallel with the purpose of finding *one* idea worth scaling.

2. Existing processes and theories give no clear guidance on when to abandon a product idea.

All interviewed companies agreed that it is difficult to work in an organized and structured manner and those aware of or working with Lean Startup principles found them difficult to implement in practice. This notion is supported by literature [24]. The startup project team expressed a wish early on for more concrete goals to work towards, as well as knowing what techniques to apply and when, something that is currently lacking in literature. This led to us formulating the third and fourth problem statements:

3. Existing processes and theories provide insufficient criteria for when to move product ideas forward through process stages.
4. Existing processes and theories provide insufficient suggestions of what techniques to use and when, while validating product ideas.

Chapter 6

ESSSDM

In response to the identified challenges, we have developed the Early Stage Software Startup Development Model (ESSSDM). The model extends existing Lean Startup approaches [30] [24] [6] [14], incorporates the results from interviews with entrepreneurs and is based on earlier experiences with startups by the authors. The process supports multiple product ideas, constituting a product idea portfolio, being investigated in parallel by a team of entrepreneurs. It is defined in a clear step-by-step fashion with exit criteria for each stage. In addition, the model presents guidance concerning the techniques and practices to employ during the different stages.

The model is applicable to startups as defined by Eric Ries: human institutions designed to deliver new products or services under conditions of extreme uncertainty [30]. This includes startups within larger organizations. Copying existing product concepts and bringing them to new markets reduces uncertainty to such a degree that these scenarios are disregarded. The process was designed for B2B SaaS products; implications of other setups are discussed in chapter 8.

6.1 Overview

The purpose of ESSSDM is to find *one* product idea worth scaling. There are three parts to the process: idea generation, a prioritized ideas backlog and a funnel through which ideas are validated systematically, in parallel, using the Build-Measure-Learn (BML) loop [30]. The funnel is divided into four distinct stages, each with its own set of risks, suggested techniques and exit criteria. Multiple ideas exist in the funnel simultaneously, as they are being investigated in parallel. The number of ideas that can be worked on at the same time, however, decreases the further along the funnel one travels. Figure 6.1 shows an overview of ESSSDM.

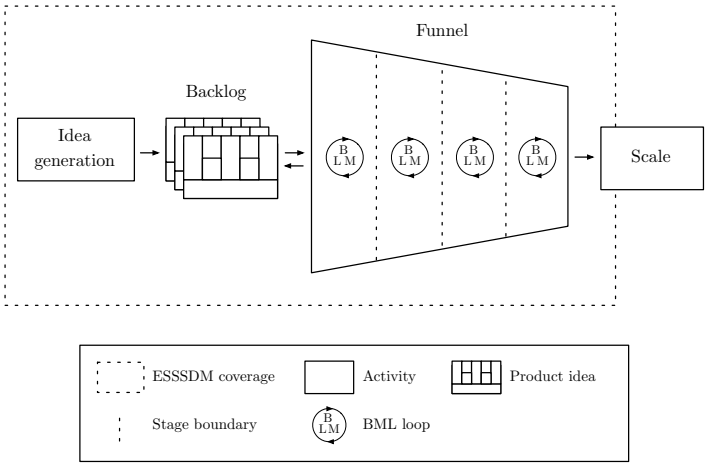

Figure 6.1: Overview of ESSSDM

6.2 Idea generation

6.2.1 Overview and starting points

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, see section 4.1.

A product idea contains, at a minimum, a problem or collection of problems that needs solving. In order to extract such problems from the market (referred to as market-pull), different techniques can be used. The following questions serve as good starting points:

- Does the team have specific domain expertise?
- Does the team have specific technical expertise?
- Does the team experience any problems of their own (”scratch your own itch”) [13]?

6.2.2 Techniques

Exploratory interviews

One way to extract problems from potential customers is to go out and talk with them, i.e. ”get out of the building” [6]. When doing so, it is important to segment the market and work in a structured way.

When segmenting, striking the right balance between broad and narrow is hard.

Too broad a segment will make it difficult to spot patterns and common problems; too narrow and the team might miss a great opportunity.

It is recommended to investigate one segment at a time, so that the team can stay focused and dig deep within each segment. There are few guidelines on this topic, but the most important thing is for the team to be aligned with what to investigate and what the learning objectives are.

Prior to booking an interview with a company, it is advisable to do some lightweight research to ensure that the appropriate person (i.e. someone with decision making powers) is contacted and booked.

Interviews should be relatively short, around 30-40 minutes. The purpose is to understand how potential customers run their businesses and what problems they experience. Even if these types of interviews can be hard to plan for, it is important to define learning objectives, otherwise it can be hard to draw conclusions afterwards [24]. Because the interviews are exploratory in nature, it is easier to do them in person than over the phone or through e-mail. Being on-site also allows for the interviewer to observe first-hand how the company operates.

An example of how an exploratory interview can be conducted is presented in appendix A.

"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, a practice popularized by Intuit [30]. This is useful in order to extract tacit knowledge. Unfortunately, the practice is very time consuming, and it can be hard to convince people to participate if there is no prior relationship.

These are some suggestions on what to observe:

- *Monotonic work.* When tasks are constantly being repeated, there is an opportunity to automate them.
- *Overly complex workflows.* When a workflow is perceived as complex or cumbersome, there is an opportunity to simplify it by removing steps through automation or by coming up with a different solution.
- *Complex communication paths.* When communication is handled in an inefficient way, i.e. more communication paths than necessary, there is an opportunity to improve/remove/automate these communication paths.
- *Heavy load of information.* When there is a lot of information not trivial to structure and understand, there is an opportunity to organize it and present it in a better way.
- *Time consuming tasks.* When one task takes significantly more time than others, there is an opportunity to speed it up.
- *Avoidance of work.* When work is tedious or perceived as boring and therefore delegated to another person, there is an opportunity to provide

a simpler, faster or more fun solution in order to get the work done.

SCAMPER

SCAMPER is a brainstorming technique used to systematically generate new ideas by modifying existing product concepts [31]. Each letter in the acronym represents a different way to think in terms of modification.

- *Substitute.* A part of the idea or product (e.g. rules, design) is substituted for something else, in order to come up with new ideas.
- *Combine.* Previously unrelated ideas or products are combined in order to come up with new ideas.
- *Adapt.* Concepts from other existing ideas or products (e.g. from other contexts) are borrowed and applied to the idea or product in order to come up with new ideas.
- *Magnify/modify.* Specific parts of the idea or product is magnified or exaggerated in order to come up with new ideas.
- *Put to other uses.* The idea or product is put to other uses, e.g. a completely new use case, a new market segment etc, in order to come up with new ideas.
- *Eliminate.* The idea or product is made smaller by eliminating parts or features. This can be done in order to find the core of an idea or product.
- *Rearrange/reverse.* The purpose of an idea or product can be reversed, meaning doing the complete opposite, in order to come up with new ideas.

In order to find existing product concepts that SCAMPER can be applied to, consider the following methods:

- Talking to industry professionals.
- Attending conferences and trade shows.
- Reading technology blogs and magazines, e.g. TechCrunch.

In addition to products, the SCAMPER technique can be applied to processes/workflows. This makes it useful when analyzing workflows during ”follow-me-homes”, as described in the previous section.

6.3 The backlog

All ideas for potential products are put in a prioritized backlog. Much as user stories within an Agile product backlog must be written in a comparable format, so must ideas within the ideas backlog. If this is not done, the task of prioritization becomes a difficult one. Being able to compare and prioritize among ideas is crucial when working on multiple ideas in parallel.

6.3.1 Comparable format

Blank [6], Furr and Ahlstrom [14] all advocate using Osterwalder's Business Model Canvas as a way to document product ideas. It captures all the important parts of a business model (synonymous with product idea for the remainder of this paper), while remaining crisp and to the point. Nine areas are covered: (1) customer segments, (2) value propositions, (3) customer relationships, (4) channels, (5) key activities, (6) key resources, (7) key partners, (8) cost structure, (9) revenue streams. Because of its brevity, the Business Model Canvas is a good format for sharing with advisors and investors.

Maurya uses a customized version of the Business Model Canvas called Lean Canvas [24], see figure 3.3, which is made specifically for software startups following Lean Startup practices. It requires the team to consider the following nine areas: (1) problem, (2) customer segments, (3) unique value proposition (UVP), (4) solution, (5) channels, (6) cost structure, (7) revenue streams, (8) key metrics, (9) unfair advantage.

The Lean Canvas is the recommended way to document product ideas in ES-SSDM.

6.3.2 Prioritization

After having documented ideas in a comparable format, they need to be prioritized. The following are some useful criteria for doing so. Prioritize by:

- How much customers care about the problem. The problem the team is trying to solve needs to be big in order to generate interest. Preferably, the customer should lie awake at night with stomach pains thinking about the problem [14]. Also see section 4.2.
- How much the team cares about the problem. "You need to be personally invested in some way. If you're going to live with something for two years, three years, the rest of your life, you need to care about it." [13]. Also see sections 4.7, 4.9, 4.4 and 4.6.
- How large the market potential is. 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. Reduces uncertainty regarding the problem and saves valuable time during the problem/solution validation stages.
- How much the team experiences the problem themselves. Known as "scratching your own itch" [13]. Reduces uncertainty regarding the problem and saves valuable time during the problem/solution validation stages.
- How easy customers are to reach. To get going, the team needs good channels to potential customers that they can talk to. The easier access they have to people experiencing the problem, the better it is in order to get rapid feedback.

- How clear analogs and antilogs there are. Analogs and antilogs are indicators that a similar business model has been successfully executed in the past.
- How clear the UVP is. Ideas with clear value propositions are easier for the team to communicate and easier for customers to understand.
- How frequently occurring the problem is. Preferably the problem occurs ∼1/week. When this is not the case, the testing and feedback process run the risk of becoming too slow.
- How technically feasible the problem is to solve within a realistic time horizon.

6.4 The funnel

Ideas from the backlog are fed into a funnel where they undergo systematic validation. Multiple ideas can exist in the funnel at once, as they are investigated in parallel. The validation process for each idea can be described as a feedback loop comprising risk prioritization followed by validated learning in the form of BML looping.

The funnel is divided into four stages, each with its own set of risks and exit criteria. Ideas move through the funnel stages as the validated learning process provides the data needed to mitigate risks and fulfill exit criteria. For each stage, techniques that accelerate validated learning are provided. The four stages are:

1. Validate problem
2. Validate solution
3. Validate MVP small-scale
4. Validate MVP large-scale

These stages can be mapped to familiar startup milestones. Problem/solution fit [30] [24] should be reached at the end of stage two, and product/market fit [30] [24] at the end of stage four. These are roughly similar to Customer Development's Customer Discovery and Customer Validation stages [6]. Beyond the fourth stage awaits the transition between early adopters and early majority, sometimes referred to as "crossing the chasm" [27] [14]. An overview of the funnel is depicted in figure 6.2

6.4.1 Working with ideas in parallel

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 can be damaging if it happens too soon [14] [30]. Being overly attached may lead to data skewing and an inability to see things as they truly are. In the early stages, an open mind and a willingness to change direction are advantageous traits. (2) Having a pipeline

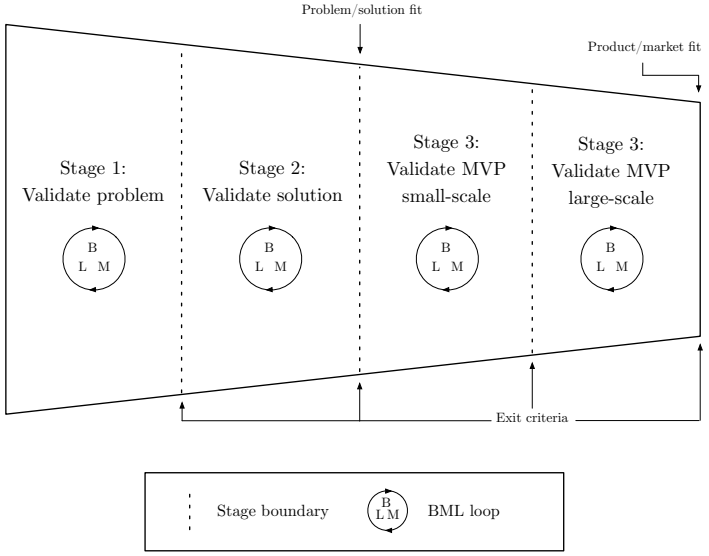

Figure 6.2: Overview of the funnel

of ideas means there is always something to work on when other ideas are on hold: waiting for experiments to run their course or interview session dates to arrive. It is also useful when neither pivoting nor persevering is an attractive option, i.e. when a risk becomes blocking. (3) Many do investigate and prioritize multiple ideas prior to picking one around which the company is formed, see chapter 4. Typically, this only entails doing standard market research. Extending this investigation so that ideas are actually validated against customers makes it easier for the company to pick the right ideas.

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 becomes a matter of available resources and the size of the MVP. During stage four, the number of ideas should ideally be distilled down to one. A simple approach for dealing with ideas in different stages is to assign points to each idea, depending on stage, and then limit the amount of points that can be worked on in parallel. For instance, a team of four could assign one point for an idea in stage one, two points in stage two, etc., and allow themselves to work on four points in parallel. That way they could, for instance, work on: four stage one ideas, two stage one ideas and one stage two idea, or one stage four idea.

Depending on team size, it can be worth thinking in terms of problem teams and solution teams [30]. Problem teams focus on doing Customer Development [6] and solution teams focus on doing product development. There may, for instance, be several problem teams, all investigating their own ideas, and one solution team, catering to all their needs. These internal teams can be small,

and may not even require more than one person.

Efficiency can of course be an issue; there is a switching cost for team members when working on multiple ideas in parallel. The negative impact is mitigated to a degree due to great opportunities for reuse in terms of concepts and assets. Analogs/antilogs, business models, customer channels, software frameworks, design guidelines etc. can all be tweaked and modified and shared between ideas. Also, "constraints are often advantages in disguise" [13]. Working on multiple ideas enforces a lean-by-necessity approach. Building two MVPs, for instance, ensures they are kept as small as possible.

6.4.2 Life in the funnel

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 [30] [24]. At the end of each BML iteration, a decision can be made whether to move the idea to the next stage, pivot, persevere, or put it on hold in favour of a different idea.

Risk prioritization

There are many types of risks associated with an idea. Initially the idea, or business model, is based mostly on assumptions that need to be tested. Prioritization of risks is key, as it is easy to start working on things that are not crucial at the moment [30] [24]. There are several ways to identify risks but an easy way to get started is to list everything about the business model that is uncertain to some degree. These risks must be thought of in terms of what stage the idea is currently in. Upcoming sections describing the four stages of ESSSDM offer some guidance as to what risks are the most important to focus on at any given time.

One approach to prioritizing risks is to use a traditional likelihood/impact matrix, where likelihood is the likelihood of the risk occurring (the more uncertainty, the harder this is to judge) and the impact is how damaging it would be to the business model if the risk would occur. Figure 6.1 provides an example of such a matrix, outlining risks that are important to consider during stage one.

Validated learning

With risks identified and prioritized, the validated learning process commences. The purpose of the BML technique is to learn from carefully crafted experiments, and use the gained knowledge to mitigate crucial risks and tweak the business model until it passes validation criteria. Using the scientific method [30] as a point of reference, the BML process can be thought of as follows. (1) For each risk that needs to be mitigated, the team formulates one or many falsifiable hypotheses and defines and prepares experiments to test them. (2) The team performs the experiments and collects/measures data. (3) The team

Table 6.1: A likelihood/impact matrix used to prioritize risks during stage one, with ”the team does not know if the problem is big enough to warrant solving” being the most important one to investigate.

High likelihood	The team does not know the competitive landscape (medium)		The team does not know if the problem is big enough to warrant solving (high)
Medium likelihood			The team is not familiar with the problem domain (high)
Low likelihood		The team does not know who experiences the problem (low)	
	Low impact	*Medium impact*	*High impact*

analyzes collected data and documents what was learned. Newfound knowledge is fed back into the business model and the validated learning process, typically leading to new hypotheses. See next section for more on what happens at the end of a BML iteration.

A good way to keep track of experiments is to put up an experiment board: a Kanban board (see section refsec:kanban) with cards listing all experiments as they move through the BML process and through the funnel stages. Such a board is depicted in Figure 6.3.

Defining experiments. It is important to minimize the time it takes to perform experiments, while maximizing the amount of learning gained. However, it is crucial that the learning is useful learning, otherwise the experiment is a form of waste and should not be conducted. Thus we have two requirements on any experiment:

- The outcome of the experiment must be valuable learning
- The time it takes to perform the experiment should be kept to a minimum

The worst case scenario is an experiment that takes a long time to perform and with low learning value. An example of such an experiment might be building a product for six months without testing it on customers, which once released, turns out to be something the customers are not interested in [30].

Defining experiments that fulfill both requirements is no easy task, and is considered by some [24] to be ”more art than science”. Upcoming sections describing the four stages of ESSSDM offer some guidance as to what techniques can be utilized in order to define and conduct experiments.

Documenting results. After having performed an experiment, it is crucial to document any learning gained. Failure to do so can have damaging consequences,

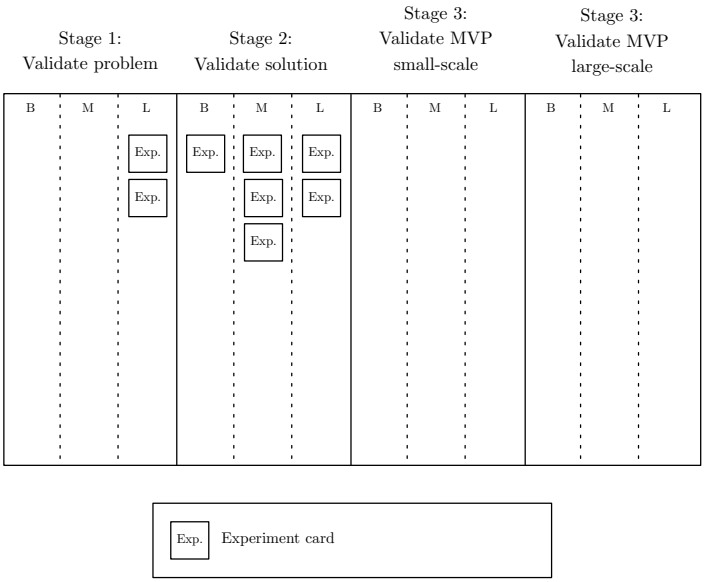

Figure 6.3: An experiment board for keeping track of experiments [24].

including running in circles: spending valuable time performing similar experiments [24]. Another consequence would be taking important decisions based on inaccurate/believed data.

Keep the following in mind when documenting. (1) Learning tied to a particular experiment should be documented (briefly) in a way so that it is clear what experiment it pertains to. Traceability in this regard is required. (2) Any learning gained must be transferred back into the business model and documents holding the identified, prioritized risks (the purpose of running experiments is to validate hypotheses so that risks can be mitigated).

There is software available, e.g. Nvivo, designed to deal specifically with the qualitative research process and the types of mixed data associated with it: interview transcripts, surveys, audio, video, web pages etc. Being able to collect and organize this kind of data in a structured, searchable fashion is very useful for documenting experiment results.

Pivot, persevere or put on hold

At the end of a BML iteration, there is an opportunity for the team to reflect upon all that has been learned, and to act upon it. The first decision point is whether the idea is ready to move on to the next funnel stage or not. This is done by consulting the stage exit criteria (see sections 6.4.3, 6.4.4, 6.4.5 and 6.4.6) during a team meeting. A good setup for such a meeting is for the person/team responsible for the idea to defend it, while the rest of the team tries to invalidate it. The idea moves through to the next stage if the team feels

that the exit criteria have been fulfilled.

If an idea is not ready to move to the next stage, a second decision must be taken: whether to pivot, persevere or put the idea on hold. When experiments reach diminishing returns, hard decisions must be made. This is yet another example of where something is "more art than science" [24]. Persevering means staying the course, doing slight tweaks and hoping to see better results in time. Pivoting, on the other hand, is a significant strategic change, while still remembering what has been learned about customers so far [30]. A third option, 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.

Software development practices

From stage three and onwards, the MVP will be actively developed and placed in the hands of real customers. On the software development side, these are some important general practices to implement.

Agile software development. While there is no requirement to use any specific agile process (such as Scrum or XP) working in an agile fashion is highly recommended. In particular: short iterations, frequent delivery of working software, and close customer collaboration. In addition, building adaptive software that responds well to change is critical in a Lean Startup.

Continuous integration and continuous delivery. Working with small batches is a technique from the Lean Manufacturing process, and discussed by Ries in Lean Startup [30] as a way to speed up the validated learning process. The more frequently code can be deployed to customers, the better. Some companies are known to deploy several times a day [30] [24], a practice that greatly reduces the time it takes to get feedback on a particular feature. The trade off in software quality is outweighed by the benefits of such speedy feedback. In addition, smaller batches are easier to throw away if it turns out the feature provided little value, see section 4.9.

6.4.3 Stage 1: Validate problem

The purpose of the first stage (see figure 6.2) is to investigate and validate the underlying problem(s) that customers want 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?

Risks

1.1 The team is not familiar with the problem domain

1.2 The team does not know if the problem is big enough to warrant solving

1.3 The team does not know who experiences the problem

1.4 The team does not know the competitive landscape

Exit criteria

1.1 After having talked to 30[1] potential customers (strangers, early adopters with similar characteristics), 50% (majority) must give strong positive indications when pitched the problem:

(a) Wants the problem solved

(b) Willing to pay for a solution

(c) Willing to participate in solution testing

1.2 Being able to describe a promising customer segment (+ rationale)

1.3 Being able to describe how potential customers currently solve the problem

Techniques

- **Domain research**

  Description. If the domain has been researched to some extent before talking with potential customers, it is more likely that (1) the right people will be contacted (2) a trustworthy impression will be communicated and (3) fewer misunderstandings will occur. Domain knowledge will of course be built up over time by talking to people, but this is a slow process, and it is recommended to do more structured research, initially, to get underway.

  Examples of areas to investigate when doing a domain research:

  – How do companies within the domain conduct business? What are their business models?

  – What does the infrastructure look like within the domain, e.g. which companies rely on each other?

  – What does a typical company look like within the domain? What kind of roles are there in the company?

  – What rules and regulations are companies within the domain obliged to follow?

  – What jargon and terminology is used within the domain?

  Rationale. Use in order to eliminate potential wasteful work due to not understanding the core domain, to find new customers and to learn rules and restrictions.

  Risks. Mitigates risks 1.1 and 1.3.

  Exit criteria. Supports learning for exit criteria 1.2.

[1] See appendix B for details on sample size calculations

- **Problem cold calls**

  Description. These interviews are held over the phone and the purpose is to recognize if, and to what extent, people have the given problem. When doing these cold calls it is recommended to have a script (even if it is not followed to the letter) in order to, as brief as possible, state the reason why calling. During these sessions there is a risk of being perceived as someone who wants to sell a product; it is important to convince people otherwise. The caller should give an impression of wanting to learn from, help and sympathize with people and their problems [24]. Before finishing, it is important to ask for permission to call back and for potential referrals. If people are willing to go out of their way and spend valuable time helping out, that is a good indicator that the problem is important.

  A problem cold call script is available in appendix A.

  Rationale. Use in order to get a quick indicator whether the problem is common or big enough, to learn how to pitch the problem, and to gather potential customer leads.

  Risks. Mitigate risks 1.1, 1.2, 1.3 and 1.4.

  Exit criteria. Support learning for exit criteria 1.1, 1.1a, 1.1c, 1.2 and 1.3.

- **Problem interviews**

  Description. Conducting interviews and "getting out of the building" is an essential part of Customer Development [6], and problem interviews are described by Maurya in Running Lean [24]. These interviews should focus on understanding the underlying problems that potential customers have, and not be about solutions. Problem interviews have the following setup:

  1. *Welcome and introduction.* The stage is set and the interviewer briefly describes how the interview works.
  2. *Collect demographics.* This is done in order to refine the early adopter definition.
  3. *Tell a story.* The interviewer explains a typical problem scenario and sees if it resonates with the customer.
  4. *Problem ranking.* The interviewer asks the customer to rank a predefined set of subproblems based on importance.
  5. *Explore customer's worldview.* This part of the interview requires no script. The customer talks about their general opinions on the topic.
  6. *Wrapping up.* The interviewer sums up what has been said, asks for permission to follow up and asks if the customer has any referrals who might have the same problem.
  7. *Document results.* Directly after the interview, the interviewer takes some minutes to document the learnings, in a structured format, so that interviews can be compared.

Rationale. Use in order learn if the problem is big enough, to better understand the problem, to refine the early adopter definition and to gather potential customer leads.

Risks. Mitigate risks 1.1, 1.2, 1.3 and 1.4.

Exit criteria. Support learning for exit criteria 1.1, 1.1a, 1.2 and 1.3.

- **"Follow-me-homes"**

  Description. This technique is described in section 6.2 as a way to generate ideas, but it can also be used to investigate a given problem. When so doing, a less exploratory approach can be taken, and less time spent observing.

  Rationale. Use in order to understand the daily workflow of customers and to extract tacit knowledge.

  Risks. Mitigates risks 1.1 and 1.3.

  Exit criteria. Supports learning for exit criteria 1.2 and 1.3.

- **Personas**

  Description. A persona is a fictional, tangible representation of the typical user who has the given problem. The technique makes defining the early adopter more tangible, as it creates a picture of who to communicate with, develop for and eventually sell to. A persona is something that will be refined over time; in the beginning it will most likely be based on assumptions. Eventually, as more knowledge is gained, the persona will become more accurate. There are several frameworks for creating personas. One of them is described in the book *The Essential Persona Lifecycle: Your Guide to Building and Using Personas* [1].

  Rationale. Use in order to continuously refine the early adopter definition, and to improve the communication and decision making process.

  Risks. Mitigates risk 1.3.

  Exit criteria. Supports learning for exit criteria 1.2.

6.4.4 Stage 2: Validate solution

The purpose of the second stage (see figure 6.2) is to define a solution that solves the problem(s) that customers want solved. It specifically tries to answer (1) what features are needed for the MVP? (2) who is the early adopter? (3) how much is the solution worth to customers?

Risks

2.1 The team does not know the minimum feature set for the MVP

2.2 The team does not know what constitutes an early adopter

2.3 The team does not know what customers would pay for the MVP

2.4 The team cannot find enough potential early adopters to support learning

2.5 The team and potential customers might interpret the suggested solution differently

Exit criteria

2.1 After having talked to 30[2] potential customers (early adopters with similar characteristics), 50% (majority) must give strong positive indications when shown the solution prototype:

(a) Believes the solution solves the problem

(b) Willing to test the MVP

(c) Willing to pay for the MVP (verbal commitment)

2.2 Being able to describe the characteristics of an early adopter (+ rationale)

2.3 Being able to define the minimum feature set needed to solve the problem

2.4 Being able to define a sufficiently small feature set so that the MVP can be built within a realistic time horizon

2.5 Having secured 2-3 frequently available pilot customers that will participate actively in the MVP development process

Techniques

- **Prototypes**

  Description. Low fidelity prototypes (LFPs) e.g. sketches, paper- or slideshow prototypes, are generally rough and lacking in detail. They are best used in an early stage, in order to get feedback on the solution *concept*. As a rule of thumb, three different solutions should be prototyped as a way to stay creative and objective.

  High fidelity prototypes (HFPs) e.g. wireframes, images or HTML-mockups, closely resemble the finished solution. HFPs are best used later in the stage, when the feature set has been roughly defined. A problem with doing HFPs too early is for the team and customers to get stuck on details, such as layout and button placement, instead of focusing on the concept.

  HTML-mockups are particularly useful. They look real, and the more real a prototype looks, the better it will be able to validate the solution. Another advantage is that the mockup can be reused when building the MVP, leading to less waste.

  Rationale. Use in order to quickly and cheaply create prototypes that can be used to communicate the solution to customers.

[2]See appendix B for details on sample size calculations

Risks. Mitigates risks 2.1, 2.2 and 2.5.

Exit criteria. Supports learning for exit criteria 2.1, 2.1a, 2.3 and 2.4.

- **Solution interviews**

  Description. Solution interviews are semi-structured interviews where the solution is shown to potential customers. Solution interviews are described by Maurya in Running Lean [24] and consist of the following steps:

  - *Welcome and introduction.* The stage is set and the interviewer briefly describes how the interview works.
  - *Collect demographics.* This is done in order to refine the early adopter definition.
  - *Tell a story.* The interviewer explains a typical problem scenario and sees if it resonates with the customer.
  - *Demo.* The interviewer shows a prototype of the product, e.g. and HFP in the form of an HTML-mockup. Each problem that the product solves is gone through to see if it resonates with the customer.
  - *Test pricing.* The interviewer gives the customer a price after having shown the demo and immediately tries to read the response.
  - *Wrapping up.* The interviewer sums up what has been said, asks for permission to follow up and asks if the customer has any referrals who might have the same problem.
  - *Document results.* Directly after the interview, the interviewer takes some minutes to document the learnings, in a structured format, so that interviews can be compared.

  Rationale. Use in order to learn if the solution will satisfy customer needs, to get feedback on the solution, to discuss and prioritize features and pricing, and to further refine the early adopter definition.

  Risks. Mitigates risks 2.1, 2.2, 2.3, 2.4 and 2.5.

  Exit criteria. Supports learning for exit criteria 2.1, 2.1a, 2.1b, 2.1c, 2.2, 2.3, 2.5.

- **Pre-sale cold calls**

  Description. These interviews are held over the phone and the purpose is to learn if people respond to the UVP and are interested in purchasing the solution. The technique is straightforward but many find it frustrating to deal with the many rejections. It is key, however, to find out why people do not want to use the product, so that the pitch can be tuned. It is also advisable to ask for referrals.

  Rationale. Use in order to quickly test the UVP and pricing, and gather potential customer leads. Faster than meeting people in person.

  Risks. Mitigates risks 2.1, 2.2, 2.3 and 2.4.

  Exit criteria. Supports learning for exit criteria 2.2, 2.3, 2.5.

- **E-mail sendouts**

  Description. E-mail sendouts are similar to pre-sale cold calls, but they allow for reaching a bigger group of people at a relatively low cost. When doing e-mail sendouts, it is crucial to (1) know what the goal of the e-mail is, e.g. test the UVP, and (2) measure how well the e-mail achieves the goal [28]. Measuring usually means tracking if the recipients opened the e-mail and if they clicked links included in it. There are many services available that support this, e.g. MailChimp or Google Analytics.

  The goal of an e-mail sendout can be, for instance, to:

  - Test the pitch
  - Test the UVP
  - Test the pricing
  - Test different customer segments

  When doing e-mail sendouts, A/B testing on different batches of recipients is a good way to test and optimize alternative pitches, UVPs, price points etc.

  It is important not to send too many e-mails too frequently to the same recipients, otherwise there is a high risk of being perceived as spam.

  Rationale. Use in order to quickly test the UVP and pricing. Faster than meeting people in person or making pre-sale cold calls.

  Risks. Mitigates risks 2.2 and 2.4.

  Exit criteria. Supports learning for exit criteria 2.2, 2.3, 2.5.

- **Fake landing pages**

  Description. A fake landing page looks like an ordinary landing page, e.g. it contains a description of the UVP, the features, pricing and other relevant information. The only difference is, the product has not yet been built. This is not explicitly stated on the page, the purpose is for people to believe that there is a product. Instead of a signup procedure, users will be prompted to subscribe to a newsletter or similar in order to get notified when the product launches.

  Creating a landing page does not have to be very time consuming. Even if the landing page is a key touchpoint for customers, at this stage it only has to clearly state the UVP together with features, pricing etc. At this stage, design, brand and other types of user experience optimization can be copied from existing concepts and then emerge over time, e.g. see section 4.3.

  Rationale. Use in order to get feedback on a solution, specifically the UVP and the feature set, before it is implemented. The landing page can be reused when building the MVP.

  Risks. Mitigates risks 2.1, 2.2, 2.3, 2.4 and 2.5.

  Exit criteria. Supports learning for exit criteria 2.2, 2.3, 2.4 and 2.5.

- **Fake product videos**

  Description. A fake product video is a relatively short video presentation of the product. The video can be conceptual, not showing the solution at all, or it can be a screencast of someone using the solution. This can be done even if the solution has not yet been built, e.g. by using HTML-mockups. The video should clearly demonstrate the UVP.

  Rationale. Use in order to quickly and cheaply create videos that can be used to communicate the solution to customers. Videos can clearly demonstrate functionality and interactions, more so than static prototypes such as LFPs and HFPs. Videos are also easy to distribute, and can be used in combination with other techniques, e.g. fake landing page.

  Risks. Mitigates risks 2.1 and 2.5.

  Exit criteria. Supports learning for exit criteria 2.3, 2.4.

6.4.5 Stage 3: Validate MVP small-scale

The purpose of the third stage (see figure 6.2) is to build an MVP and test it on a small portion of early adopters. It specifically tries to answer (1) does the MVP deliver in terms of solving the problem(s) that customers want solved? (2) how to reach early adopters? (3) are customers paying for the MVP?

Before experimentation can commence, the MVP (landing page included) must be built. This is best done together with the 2-3 pilot customers secured during stage two. Working together with actual customers is a good way to ensure speedy feedback, and avoid making faulty assumptions. Even so, it is important to keep in mind that the product is meant for a broader segment, and not for pilot customers only.

In addition, a conversion funnel should be prepared. The important metrics at this stage are activation, retention and revenue. For further details, see section 6.4.6.

Risks

3.1 The team builds an MVP that does not demonstrate the UVP

3.2 The team builds an MVP that lacks in software quality according to customer needs (e.g. availability, performance, usability, security)

3.3 The team cannot sell the MVP to customers due to flawed pricing

3.4 The team cannot find enough early adopters to support learning

3.5 The MVP is not technically feasible

Exit criteria

3.1 50% of early adopters (>30[3]) make it through the conversion funnel

  (a) Customers understand the UVP

  (b) Customers accept the pricing model

3.2 50% of customers willing to give positive testimonials

3.3 Having developed outbound channels that repeatedly deliver early adopters into the conversion funnel

Techniques

- **MVP interviews**

  Description. In an MVP interview, as described by Maurya [24], the purpose is to introduce the potential customer to the built MVP and landing page. The interview consists of the following steps:

  – *Welcome.* Briefly set the stage for how the interview works.

  – *Show landing page.* Run a five-second test to test the site navigation/call to action.

  – *Show pricing page.* The interviewee should eventually end up on the pricing page, where they are asked if they accept the pricing model.

  – *Sign up & activation.* Ask the interviewee to sign up and watch how he/she navigates through the activation flow.

  – *Wrapping up.* Make sure the user knows what to do next, and keep the conversation channel open with the interviewee.

  – *Document results.* Directly after the interview, take some minutes to document the learnings, in a structured format, so that interviews can be compared.

  Rationale. Use in order to see if customers respond to the UVP and accept the pricing model.

  Risks. Mitigates risks 3.1, 3.2 and 3.3.

  Exit criteria. Supports learning for exit criteria 3.1, 3.1a, and 3.1b.

- **Follow up on activity**

  Description. When the MVP has an initial user base, it is interesting to see if and how actively people use the product. Measuring basic activity, e.g. number of signins, is important for two reasons:

  – Being able to identify which customers use the product the most is a good way to further refine the early adopter definition. It is also important to follow up and find out why certain users are more active than others.

[3]See appendix B for details on sample size calculations

- Being able to identify which customers are not using the product but previously indicated that they wanted to is a good way to test the UVP and to refine the early adopter definition. It is *very* important to follow up and find out why these users are not using the product. Here, the Five Whys [30] is appropriate.

When doing follow ups, it is recommended to do them in person, or over the phone.

Rationale. Use in order to refine the early adopter definition, by learning why certain customers use the product more than others. Also to find out why some customers do not use the product at all.

Risks. Mitigates risks 3.1, 3.2 and 3.3.

Exit criteria. Supports learning for exit criteria 3.1.

- **Streamlined trial period**

  Description. In many situations, customers expect to receive a trial period before making a purchase, especially with SaaS products. This can be problematic when trying to quickly validate the MVP, as it introduces a delay (typically 30 days) before a payment is made. There are a couple of things that can be done in order to speed up validation:

  - If possible, customers should be charged up front and offered to have their money back at the end of the trial. This strengthens the validation of the MVP.
  - Shorten the trial period. Do customers really need 30 days in order to test the product? As long as they have time to evaluate all of the features, the length of the trial period should be kept as short as possible.
  - Introduce feedback meetings halfway through the trial period. These are similar to the ones in the "follow up on activity" technique in that they should answer: (1) if some users are more active than others, why? (2) if some users are not using the product at all, why? Already at this point, it can often be determined whether or not the customer will pay at the end of the trial.

    When doing halfway meetings, it is recommended to do them in person, or over the phone.

  Rationale. Use in order to minimize time spent validating the MVP and to refine the early adopter definition.

  Risks. Mitigates risks 3.1, 3.2 and 3.3.

  Exit criteria. Supports learning for exit criteria 3.1.

6.4.6 Stage 4: Validate MVP large-scale

The purpose of the fourth stage (see figure 6.2) is to further validate the MVP on a larger portion (not possible to meet them all in person) of early adopters.

It specifically tries to answer (1) has the MVP reached product/market fit? (2) is there a sustainable path to early adopters/customers? (3) is the business model working?

The conversion funnel should at this stage be expanded to include acquisition and referral.

Risks

4.1 The team does not have a sustainable path to customers

4.2 The team does not have a profitable business model

4.3 The team does not have a scalable business model

Exit criteria

4.1 Having passed The Sean Ellis Test (40%). See appendix C.

4.2 Having developed inbound channels that repeatedly delivers early adopters into the conversion funnel

4.3 Customer Lifetime Value (CLV) >User Acquisition Cost (UAC)

Techniques

- **Startup Metrics for Pirates**

  Description. It is important to focus on actionable metrics as opposed to vanity metrics, see section 3.2 for more on the concepts. Startup Metrics for Pirates [25] is a metrics framework for startups by Dave McClure. The five metrics of interest are:

  1. *Acquisition.* Users arrive to the product from various channels, e.g. they find the landing page.
  2. *Activation.* Users have their first "happy" user experience. This is often includes signing up plus additional activities.
  3. *Retention.* Users return to the product and uses it again.
  4. *Referral.* Users like the product enough to recommend it to others.
  5. *Revenue.* Users pay in some way for the product.

  Optimizations are done one metric at a time. The order, however, is not set in stone. Generally though, acquisition, activation and retention need to be optimized for before referral and revenue.

  Rationale. Use in order to optimize the conversion funnel by focusing on actionable metrics.

  Risks. Mitigate risks 4.1, 4.2 and 4.3.

  Exit criteria. Supports learning for exit criteria 4.2 and 4.3.

6.4.7 The light at the end of the funnel

Once an idea has moved through all four stages of the funnel it is considered validated and ready for scaling. At this point, the objective of ESSSDM has been fulfilled.

Chapter 7

Evaluation

7.1 Design goals and evaluation criteria

ESSSDM was evaluated in a startup project and interviews with industry professionals. In accordance with DSR, the following design goals and evaluation criteria were defined for evaluating ESSSDM:

1. The process must support working on, or investigating, multiple product ideas in parallel

   Evaluation criteria:

   (a) Consensus of project team

   (b) Consensus of industry professionals

2. The process must provide clear guidance on when to abandon a product idea

   Evaluation criteria:

   (a) Consensus of project team

   (b) Consensus of industry professionals

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

   Evaluation criteria:

   (a) Consensus of project team

   (b) Consensus of industry professionals

4. The process must provide clear guidance on what techniques to use and when, while validating product ideas

   Evaluation criteria:

   (a) Consensus of project team

(b) Consensus of industry professionals

7.2 Context

The two authors, together with three master students from Chalmers School of Entrepreneurship, co-founded a startup that was run in an incubator setting at Encubator AB for eight months. Encubator provided the team with initial funding and office space. Experienced industry professionals, business advisors and legal experts were also made available. All of the students were entitled to shares in the company, in the event of an incorporation at the end of the incubation period. The purpose of the startup was to find a promising product in the small business segment.

7.3 Instantiation

This chapter describes how ESSSDM was instantiated and applied to the startup project. The project was active during eight months, and worked on product ideas that passed through two complete stages and partly through stage three (see figure 6.2). No further evaluation was possible due to the project's cancellation at the end of the incubation period.

7.3.1 Idea generation and the backlog

This section describes what ideas were eventually worked on and where they originated from. Table 7.1 lists the product ideas that were at some point picked from the backlog, and what techniques were used to generate them.

Table 7.1: Lists the various product ideas with a brief description, and also what methods were used to generate each idea.

Product idea	*Description/problems*	*Idea generated from*
Tradesmen	Tradesmen need a tool for creating invoices, distributing work to employees and logging time	Exploratory interviews
Quotes	Companies face problems comparing and choosing quotes from multiple suppliers	Exploratory interviews + domain knowledge (scratch own itch)
Liquidity	Small business owners find it difficult to manage and understand the implications of liquidity	Exploratory interviews + "follow-me-homes"

Product idea	*Description/problems*	*Idea generated from*
Automated accounting material (AAM)	Always having to prepare accounting material is a hassle for small business owners	Exploratory interviews
Regulatory compliance (RC)	Small companies struggle to be regulatory compliant and maintain quality assurance	Exploratory interviews
Customer Relationship Management (CRM)	Small businesses would like a CRM-system that is automated, flexible and which visualizes a sales funnel	Exploratory interviews
On-site construction communication (OSCC)	Construction workers have no effective way of communicating, sharing documents etc. when working on construction sites	Exploratory interviews
Retail	Small businesses in the retail industry find it hard to bridge the gap between connecting with their customers on a personal level and maintaining operational hours with staffing costs being so high	Copied existing solution (SCAMPER: Put to another use, looked at existing foreign solutions within retail in North Korea) + exploratory interviews
Stock and inventory management (SaIM)	Small business retailers waste a lot of time in inventory management and ordering new stock	Exploratory interviews
Invoice management	Small business have no good way of archiving and managing invoices when receiving them in different format	Exploratory interviews + domain knowledge (scratch own itch)

During the course of the project, many more ideas than those presented in table 7.1 were generated, but were down-prioritized and thus never entered the funnel.

Example: When the AAM product idea was abandoned, a new one had to be picked from the backlog. Liquidity and Quotes were compared, with Liquidity

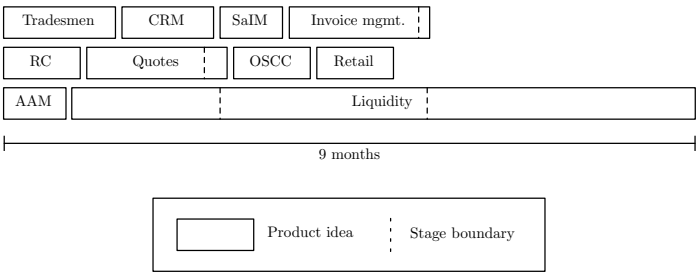

Figure 7.1: A timeline of when the ideas were generated and brought into the funnel, which were worked on simultaneously, and when they were discarded. The stage boundary shows where an idea moved into the next stage.

being the idea that was eventually favoured, even though Quotes got picked soon after. The following prioritization criteria, as defined by the process, led to Liquidity being ranked higher in the backlog:

- Ease of reach and customer availability. At this stage, the team felt that it had higher chances of reaching out to the customer segments defined in the Liquidity canvas. Small business, especially startups, were close at hand. Construction companies, on the other hand, were more difficult to initiate contact with, even though the team had a few lined up.
- Problem importance. While the team had good indications that the quotes process was a major hassle, dealing with liquidity and cash flow problems was felt to be an even more important problem for small business owners.

7.3.2 The funnel

This section describes how the team worked with ideas through the funnel stages (see figure 6.2). First, details on how multiple ideas were investigated in parallel is presented. Then, a subsection for each individual idea lists prioritized risks, techniques used and exit criteria not fulfilled for relevant stages.

Figure 7.1 shows a timeline of when the ideas were generated and brought into the funnel, which were worked on simultaneously, and when they were eventually discarded. The stage boundary shows where an idea moved into the next stage. A maximum of three ideas was worked on in parallel. Work was normally distributed so that the three business developers were responsible for one idea each, while the software engineers worked on all three ideas at once. Eventually, more and more effort was put into the most promising idea, with others being put on hold.

Tradesmen

Tradesmen need a tool for creating invoices, distributing work to employees and logging time.

Stage 1: Risks. (1) The team is not familiar with the problem domain. (2) The team does not know if the problem is big enough to warrant solving. (3) Tradesmen want to use pen and paper over computer software. (4) Tradesmen do not experience this problem.

Stage 1: Techniques. Problem cold calls: 20 tradesmen. Problem interviews: 10 tradesmen.

Reasons for abandoning product idea. Exit criteria 1.1 not fulfilled. Interviews proved that tradesmen were resistant in using computer software over pen and paper.

Quotes

Companies face problems comparing and choosing quotes from multiple suppliers.

Stage 1: Risks. (1) The team does not know if the problem is big enough to warrant solving. (2) The team does not know who experiences the problem. (3) The team is not familiar with the problem domain. (4) The team does not know the competitive landscape.

Stage 1: Techniques. Problem cold calls with 20 construction companies. Problem interviews: with 10 construction companies.

Stage 2: Risks. (1) The problem does not occur frequently enough, hindering the testing process. (2) The quotes cannot be written in a standardized format. (3) The team does not know the minimum feature set for the MVP. (4) The team does not know what constitutes an early adopter. (5) The team does not know what customers would pay for the MVP.

Stage 2: Techniques. High fidelity prototypes: HTML-mockups. Solution interviews with 10 construction companies. Email sendouts to 20 construction companies.

Reasons for abandoning product idea. (1) Exit criteria 2.5 not fulfilled. It turned out the problem did not occur frequently enough (it occurred $\sim$1/month) and during the winter, construction companies worked on fewer projects. (2) The problem of defining standardized formats for quotes was deemed too complex and time consuming from the startup project's perspective.

Automated accounting material (AAM)

Always having to prepare accounting material is a hassle for small business owners.

Stage 1: Risks. (1) The team does not know if the problem is big enough to warrant solving. (2) The team does not know the competitive landscape. (3) Automating standard accounting procedures is not technically feasible.

Stage 1: Techniques. Problem interviews with 15 small businesses. Problem interviews with two bookkeepers (same problem domain).

Reasons for abandoning product idea. Exit criteria 1.1 not fulfilled. Turned out few people experienced the problem at all.

Liquidity

Small business owners find it difficult to manage and understand the implications of liquidity.

Stage 1: Risks. (1) The team does not know if the problem is big enough to warrant solving. (2) The team does not know who experiences the problem. (3) The team does not know the competitive landscape.

Stage 1: Techniques. Problem interviews with 20 small businesses. Problem interviews with five people within the same problem domain: invoicing, sales management. Two "follow-me-homes".

Stage 2: Risks. (1) The team does not know the minimum feature set for the MVP. (2) The team does not know what constitutes an early adopter. (3) The team does not know what customers would pay for the MVP.

Stage 2: Techniques. Solution demos in the form of wireframes and HTML-mockups. Solution demos in the form of videos. Interviews with 15 startups. E-mail sendouts of solution demos to an additional 20 startups.

Stage 3: Risks. (1) The team builds an MVP that does not demonstrate the UVP. (2) The team cannot sell the MVP to customers due to flawed pricing. (3) The team cannot find enough early adopters to support learning.

Stage 3: Techniques. MVP interviews. Follow up on activity. Streamlined trial period.

Reasons for abandoning product idea. As of this writing, the idea is still in stage three.

Regulatory compliance (RC)

Small companies struggle to be regulatory compliant and maintain quality assurance.

Stage 1: Risks. (1) The team does not know if the problem is big enough to warrant solving. (2) The team does not know who experiences the problem.

Stage 1: Techniques. Problem interviews with four small business in Ireland.

Reasons for abandoning product idea. Exit criteria 1.1 It was hard to get companies to talk with since the problem came from an abroad market.

Customer Relationship Management (CRM)

Small businesses would like a CRM-system that is automated, flexible and which visualizes a sales funnel.

Stage 1: Risks. (1) The team does not know if the problem is big enough to warrant solving. (2) The team does not know who experiences the problem. (3) The team does not know the competitive landscape.

Stage 1: Techniques. Problem interviews with 12 small businesses.

Reasons for abandoning product idea. Exit criteria 1.1 not fulfilled. There was difficulty in isolating and understanding what the underlying problems were. Area deemed too complex and the idea was put on hold.

On-site construction communication (OSCC)

Construction workers have no effective way of communicating, sharing documents etc. when working on construction sites.

Stage 1: Risks. (1) The team does not know if the problem is big enough to warrant solving. (2) The team does not know who experiences the problem. (3) The team does not know the competitive landscape.

Stage 1: Techniques. Problem interviews with 17 construction companies. Some companies that were interviewed came from the quotes idea.

Stage 2: Risks. (1) The team does not know the minimum feature set for the MVP. (2) The team does not know what constitutes an early adopter. (3) The team does not know what customers would pay for the MVP.

Stage 2: Techniques. High fidelity prototypes: HTML-mockups. Solution interviews with 17 construction companies.

Reasons for abandoning product idea. (1) Exit criteria 2.1 not fulfilled. General unwillingness to use software systems. (2) Exit criteria 2.5 not fulfilled. (3) The team could not find enough potential early adopters to support learning.

Retail

Small businesses in the retail industry find it hard to bridge the gap between connecting with their customers on a personal level and maintaining operational hours with staffing costs being so high.

Stage 1: Risks. (1) The team does not know if the problem is big enough to warrant solving. (2) The team does not know who experiences the problem.

Stage 1: Techniques. Problem interviews with 12 small retail businesses.

Reasons for abandoning product idea. Exit criteria 1.1 not fulfilled. There was difficulty in isolating and understanding what the underlying problems were. Area deemed too complex and the idea was put on hold.

Stock and inventory management (SaIM)

Small business retailers waste a lot of time in inventory management and ordering new stock.

Stage 1: Risks. (1) The team does not know if the problem is big enough to warrant solving. (2) The team does not know who experiences the problem. (3) The team does not know the competitive landscape.

Stage 1: Techniques. Problem interviews with 10 small businesses.

Reasons for abandoning product idea. (1) Exit criteria 1.1 not fulfilled. Few people experienced the problem. The problem was not viewed as big enough. (2) Exit criteria 1.2 not fulfilled. There was difficulty in finding who exactly had the problem.

Invoice management

Small business have no good way of archiving and managing invoices when receiving them in different format.

Stage 1: Risks. (1) The team does not know if the problem is big enough to warrant solving. (2) The team does not know who experiences the problem. (3) The team does not know the competitive landscape.

Stage 1: Techniques. Problem interviews with 15 small companies.

Stage 2: Risks. (1) The team does not know the minimum feature set for the MVP. (2) The team does not know what constitutes an early adopter. (3) The team does not know what customers would pay for the MVP.

Stage 2: Techniques. High fidelity prototypes: wireframes. Solution interviews with 15 construction companies. E-mail sendouts: wireframes to 15 additional companies.

Reasons for abandoning product idea. The idea was deemed similar in many ways to Liquidity: same customer segments, same channels and some of the same underlying problems. Put on hold while investigating it as a potential add-on to Liquidity.

7.4 Conclusions

This section goes through each of the design goals and their evaluation criteria. Consensus of project team was derived by talking to the individual team members. Consensus of industry professionals was derived by talking to the subset of the companies interviewed in chapter 4 best matching the requirements for ESSSDM (four companies). 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 process must support working on, or investigating, multiple product ideas in parallel**

   Consensus of project team Overall, the team felt investigating multiple ideas in parallel was worth doing from a project perspective, and well supported by the process.

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 (i.e. Lean Canvas) made the prioritization process easier (an early iteration of the process was missing this) and forced the team to consider all aspects of the business model, not only the solution. A side effect of this was that not all ideas entered the backlog due to them being too vague to be documented as a Lean Canvas.

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 while waiting for data, interview dates etc to arrive. It also allowed for increased objectivity in that the business developers could often offer each other unbiased advice.

Sharing of assets between ideas happened frequently. Domain knowledge and sometimes even customers could be shared due to similar problem areas or customer segments. HTML-mockups could often be put together using code, libraries and frameworks used on previous mockups. The team felt that reusing assets in this way mitigated the inherent switching cost that comes from working on multiple ideas in parallel.

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

2. **The process must provide clear guidance on when to abandon a product idea**

   Consensus of project team The team frequently 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 favour of another one from the backlog. This workflow was well described by the process.

   While the process gives clear guidance on when, timewise, to consider abandoning an idea, it might not provide good enough criteria for making the decision. Concerns were raised that perhaps some ideas were abandoned prematurely, something that others working in a similar fashion have acknowledged to be a problem , see section 4.6.

   Consensus of industry professionals Lack of data.

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

   Consensus of project team The team thought the idea of having exit criteria was a good way to give guidance on when to move forward with ideas. Having such clear goals enabled the team to keep a good momentum and allowed each business developer 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 typically 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 process must provide clear guidance on what techniques to use and when, while 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. They also proved speedy to implement, no technique took more than two working days to execute.

   The team felt having techniques divided by stage made it relatively clear when to use them. Although, future versions of the process might benefit from more detailed instructions, taking into consideration context, what has already been done, what is about to be done etc.

   Consensus of industry professionals Lack of data.

Chapter 8

Discussion

This chapter discusses the results of the study, why they are important and how they relate to existing research and literature. It also touches upon relevant limitations and areas for improvements. Finally, the validity of the research is examined.

8.1 The results

8.1.1 Summary and contributions

The resulting artefact of the research — the Early Stage Software Development Model (ESSSDM) — provides practical guidelines for managing early stage software startups. It supports investigating multiple ideas in parallel, and provides stages with clear exit criteria. In addition, it gives advice on when to abandon ideas and what techniques to use while validating them.

With software startups being more popular than ever and growing in numbers, there is an increasing demand for processes like ESSSDM, that try to bring structure to where there is mostly chaos. The field is relatively young (the Lean Startup movement has been active only since 2011) and there is so far very little academic writing on the topic. We feel that this needs to change. These issues, even if somewhat fuzzy and unclear, need to be brought forward.

In terms of existing research and literature, ESSSDM is inspired in particular by the works of Eric Ries [30], Steve Blank [6] and Ash Maurya [24]. The primary contribution is that the process supports investigation of multiple product ideas in parallel, with the purpose of finding a single, scalable idea. Novel parts include (1) having a backlog with ideas written in a comparable format (the format itself previously exists) (2) a compiled list of backlog prioritization criteria (criteria 8 and 9 are own contributions, while the others were derived from literature and interviews) (3) the concept of validating ideas through a funnel (validated learning and the four stages are existing concepts) (4) the introduction of abandoning ideas as an alternative to pivot or persevere and (5) guidelines for how to work in a parallelized way.

Secondary contributions are (1) considering idea generation to be part of the startup process (exploratory interviews as a technique is novel, while "follow-me-homes" and SCAMPER are previously existing) (2) the composition of exit criteria for each stage (1.1, 2.1, 2.5, 3.2 and 4.3 are own contributions, while the others were derived from literature and interviews) and (3) the mapping of existing lean startup techniques to stages, risks and exit criteria.

An additional contribution is the interview data from nine software startups in the Gothenburg region.

8.1.2 Limitations

These are some important limitations with the current version of ESSSDM.

Big enough customer pain. Processes such as Running Lean [24], Nail It then Scale It [14] and ESSSDM all come with one big reservation: the customer pain (or the problem) must be big enough. The solution must be a "must have" and not a "nice to have". The reason for this is simple. Without significant customer pain, it is very difficult for a startup, typically without track record, brand and reputation, to attract enough attention to build a large and successful business. For instance, activities such as cold calling and booking interviews become incredibly time consuming if customers do not believe the problem to be important enough.

This, however, creates a dilemma. Finding ideas that meet this criteria can obviously be very difficult. But not everyone is looking to create the next multi-billion dollar success story. There are plenty of companies out there supplying "nice to have" solutions while still making a profit. There is a discussion to be had whether ESSSDM should be followed by everyone, or if some of the validation criteria could be relaxed depending on company ambition.

B2B vs B2C startups. There is a difference between building products for consumers and building products for companies. ESSSDM was designed for B2B startups, but could be modified to suit B2C situations as well. The following are some key areas to pay attention to.

Selling the product prior to building it is probably more difficult to pull off in B2C than in B2B. Pre-selling is not common in the consumer segment, where free trials and demos are the norm. Thus, alternative exit criteria, that provide similar levels of validation, might be necessary in the early stages of the process.

It is possible that the perceived quality (e.g. user experience) of the MVP needs to be higher in B2C. While companies mostly care about having their problems solved (and better understand the concept of MVPs) consumers tend to demand easy to use products with sensible user interfaces. Slightly counterintuitive, seeing how B2C products are often cheaper than B2B products, sometimes even free.

Network effect startups. At the core of the Lean Startup and Customer Development lies the notion that by testing and validating assumptions prior to scaling, the product is more likely to be used by people. However, startups

with products that depend on a strong network effect might not fit this model perfectly, see section 4.4. Only after reaching a critical mass of users can the true value of such products be demonstrated.

This begs the question: do network effect startups need to scale *before* reaching product/market fit? Maybe. There might be cases where the effects of the network can be simulated, using carefully crafted experiments, on a small scale. Also, the number of users required to reach critical mass varies between products, and could be lower than believed. Despite this, it may very well be that other rules apply to network effect startups, and that the best course of action for them is to "scale it then nail it".

8.2 The research

As described in chapter 2, the study was performed in an encubation setting. The setup provided the authors with the opportunity and freedom to build a business from scratch, and perform research while doing so. It allowed time to be spent building and documenting theory, and for testing different approaches when designing ESSSDM. This would have been hard in a traditional startup setting.

Although very close to real industry conditions, some might argue that the project did not reflect an authentic startup scenario. In a non-encubation setup, personal risks can be seen as higher, e.g. personal investments. However, many startups have emerged from Encubator, whereof many still exist and have grown into self-sustaining companies. Throughout the year, the project team was strongly motivated to build a sustainable business. All involved parties were entitled to shares in the company, and the project was fully funded by Encubator during the incubation period. The term *student* might confuse readers when evaluating the validity of this paper, and it is important to remember that the team were not treated as students, not by the project's board nor by customers. The term *student* was never communicated to customers since that most likely would have affected the credibility of the project. Therefore, while not identical, we do claim there to be many similarities between encubation and non-encubation startups, and that learnings discovered in this paper can be applied by startups in general, assuming the requirements for ESSSDM are met, see chapter 6.

As described in chapter 2, ESSSDM was mainly validated through project instantiation in order to understand and conclude as much as possible from a practitioner's perspective. The decision to do so was based in consultation with the authors' educational institution, who agreed that this approach would benefit both the startup project and provide the research with good practical insights into running a startup. One rationale was that building a startup requires a lot of customer interaction; taking on a more observatory role, such as doing a case study, would not provide the information necessary to fully understand the environment. However, all parties were aware of the complexity of, and criticism against validating through project instantiation and understood the risk of the authors becoming biased by their opinions. This concern was discussed from

the start and the project team, especially the authors, have continuously scrutinized and criticized the work and tried to have an objective attitude towards the research results.

The detailed feedback gathered throughout the startup project was more than sufficient in order to make appropriate decisions when designing ESSSDM. Of course, it would have been good to evaluate the process on more startups, but since the process was designed iteratively, we thought it would be hard to run more trials until a ”first” version had been developed. Therefore, we considered it appropriate to first design the process and then, as future work, run more trials on more startups. In addition, more feedback from more industry professionals would further strengthen the validation.

Chapter 9

Conclusions

9.1 Summary and conclusions

Software startups are more popular than ever and growing in numbers. They operate under conditions of extreme uncertainty and face plenty of challenges, underlined by their high failure rate. Using DSR, we set out to investigate: (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 these challenges. A literature review and interviews with industry professionals led to a set of problems (see chapter 5) and a suggested solution in the form of a process (see chapter 6).

Evaluation of the process on a startup project and through interviews with industry professionals showed that:

- The process supports working on, or investigating, multiple product ideas in parallel. Supported through consensus of project team and industry professionals.
- The process provides clear guidance on when to abandon a product idea. Supported through consensus of project team, although further evaluation is needed to make sure ideas are not prematurely abandoned.
- The process provides clear guidance on when to move product ideas forward through process stages. Supported through consensus of project team and industry professionals, although further evaluation is needed; the startup project never entered the final stage of the process.
- The process provides clear guidance on what techniques to use and when, while validating product ideas. Supported through consensus of project team, although elaborating on the *when to use* would be good for future revisions.

To conclude, ESSSDM provides practical guidelines for managing early stage software startups. It supports investigating multiple ideas in parallel, and provides stages with clear exit criteria. In addition, it gives advice on when to

abandon ideas and what techniques to use while validating them. Even though further evaluation of the process is necessary (preferably on additional real world startups) initial results are promising.

9.2 Future work

The third and fourth stages of the process, qualitative and quantitative validation, have yet to be fully evaluated. These stages are integral to providing a comprehensive process for how to manage early stage startups. Also, exit criteria and guidelines for how to know when to start scaling (when has the business and product been validated?) are important additions that could be made.

Another difficulty that has been observed is that of designing experiments. Thinking in such terms, trying to find the least expensive way of testing hypotheses, is a creative effort. A list of common techniques and even patterns for designing and conducting such experiments would reduce the time it takes to complete a BML iteration.

Finally, storing the results from experiments and making sure lessons stay learned can be difficult in high pace startups. A more strict process that outlines activities similar to Scrum (e.g. daily scrum, sprint reviews, sprint retrospectives) could be worth investigating.

Bibliography

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

[2] Alvarez, S. A., Barney, J. B. (2007) Discovery and creation, alternative theories of entrepreneurial action, Strategic Entrepreneurship Journal, Vol. 1, pp. 11-26.

[3] Anderson, D. J., Reinertsen, D. G. (2010) Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press.

[4] Baron, J., Hannan, M. (2002) Organizational Blueprints for Success in High-Tech Start- Ups: Lessons from the Stanford Project on emerging companies. California Management Review, Vol. 44, No. 3, pp. 8-36.

[5] Blank, S., Dorf, B. (2006) The Four Steps to the Epiphany: Successful Strategies for Products that Win (3rd edition). Cafepress.com.

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

[7] Brinckmann, J., Grichnik, D., Kapsa, D. (2010) 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, Vol. 25, No. 1, pp. 24-40.

[8] Campbell, D. (2010) Software as a Service: spend and payment solution. Summit: Canada's magazine on public sector purchasing. http://www.summitconnects.com (25 May 2013).

[9] Cockburn, A. (2006) Agile Software Development: The Cooperative Game (2nd edition). Addison-Wesley Professional.

[10] Creative Research Systems. (2013) Sample Size Formulas. http://www.surveysystem.com/sscalc.htm (25 May 2013).

[11] Crowne, M. (2002) Why software product startups fail and what to do about it. In 2002 IEEE International Engineering Management Conference; 18-20 August 2002, Cambridge. pp. 338–343.

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

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

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

[15] Gartner, Inc. (2011) Forecast: Software as a Service, All Regions, 2010-2015. http://www.gartner.com/it/page.jsp?id=1791514 (25 May 2013).

[16] Giardino, C., Paternoster, N. (2012) Software development in startup companies. Karlskrona: Blekinge Institute of Technology. (Master's thesis).

[17] Gustafsson, A., Qvillberg, J. (2012) Implementing Lean Startup Methodology: An Evaluation. Gothenburg: Chalmers University of Technology. (Master's thesis).

[18] Holson, L.M. (2009) Putting a Bolder Face on Google. The New York Times, February 28. http://www.nytimes.com/2009/03/01/business/01marissa.html (25 May 2013).

[19] Hove, S. E., Anda, B. (2005) Experiences from conducting semi-structured interviews in empirical software engineering research. In 11th IEEE International Software Metrics Symposium; 19-22 September, 2005, Como. pp. 23:1-10.

[20] Kakati, M. (2003) Success criteria in high-tech new ventures. Technovation, Vol. 23, No. 5, pp. 447–457.

[21] Klasavičius, M. (2012) Metrics-Driven Development. InfoQ. http://www.infoq.com/articles/metrics-driven-development (November 30 2012)

[22] Kohavi, R., Henne, R. M., Sommerfield, D. (2007) Practical guide to controlled experiments on the web: listen to your customers not to the hippo. In Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining; 12-15 August, 2007, San Jose. pp. 959–967.

[23] Lange, J. E. et al. (2007) Pre-startup formal business plans and post-startup performance: A study of 116 new ventures. Venture Capital Journal, Vol. 9, No 4, pp. 1-20.

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

[25] McClure, D. (2007) Startup Metrics for Pirates: AARRR! 500 Hats. http://500hats.typepad.com/500blogs/2007/09/startup-metrics.html (25 May 2013).

[26] Middleton, P., Joyce, D. (2012) Lean Software Management: BBC Worldwide Case Study. IEEE Transactions on engineering management, Vol. 59, No. 1, pp. 20-32.

[27] Moore, G.A. (2006) Crossing the Chasm: Marketing and Selling Disruptive Products to Mainstream Customers. HarperBusiness.

[28] Mullen, J., Daniels, D. (2009) Email Marketing: An Hour a Day. Indianapolis: Wiley Publishing, Inc.

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

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

[31] Serrat, O. (2009) The SCAMPER Technique. Knowledge Solutions. 31 February.

[32] Schwaber, K., Sutherland, J. (2011) The Scrum Guide. Scrum.org

[33] Smagalla, D. (2004) The truth about software startups. MIT Sloan Management Review, Vol. 45, No. 2, pp. 7.

[34] Sommerville, I. (1996) Software Process Models. ACM Computing Surveys, Vol. 28, No. 1, pp. 269-271.

[35] Vaishnavi, V. and Kuechler, W. (2004). Design Science Research in Information Systems. http://www.desrist.org/design-research-in-information-systems/ (25 May 2013).

[36] Watson, K., Scott, S. and Wilson, N. (1998) Small business start-ups: success factors and support implications. International Journal of Entrepreneurial Behaviour & Research, Vol. 4, No. 3, pp. 217-238.

Appendix A

Scripts

In this chapter, examples of scripts that can be used in ESSSDM are presented and described.

A.1 Exploratory interview script

Example of an exploratory interview, 30-40 minutes:

1. The purpose of the interview is introduced.
2. The interviewer asks some general questions about the company in order to collect demographics, e.g. number of employees, organization infrastructure, suppliers, what type of customers they have, what are their core services etc.
3. The interviewer asks what a typical working day looks like.
4. The interviewer asks what takes time away from working on the core services.
5. The interviewer asks if the company has any specific problems that they know of and want solved.
6. If any problems have revealed themselves during the session, the rest of the interview is spent honing in on the most interesting ones:
    - The interviewer asks how often and why the problem occurs.
    - The interviewer asks how much time is spent dealing with the problem.
    - The interviewer asks who are affected by the problem, e.g. employees, suppliers etc.
    - The interviewer asks if the company has looked for or are using any existing solutions, and if so, which one and why.

- The interviewer asks if the company believes they are unique in having the problem.

7. The interviewer asks permission to follow up if a problem was found.
8. The interviewer asks if the company has referrals to others with the same problem.
9. The interview is summed up and ended.

A.2 Problem cold call script

Example of a problem cold call, 5-10 minutes:

1. The caller introduces him/herself.
2. The caller explains the reason for calling and that the purpose is not to sell anything.
3. The caller asks for some minutes of the customer's time.
4. The caller explains a typical problem scenario.
5. The caller asks if the customer can identify with the scenario.
6. The caller asks if the customer uses any existing solutions to deal with the problem.
7. The caller asks for the customer's general opinions on the topic.
8. The caller asks if the customer would be interested in being a test user for a future solution.
9. The caller asks for referrals to others that might have the same problem.
10. The cold call is ended.

Appendix B

Calculating statistical sample size

In order to roughly estimate how many potential customers to interview, consider using a formula for calculating the statistical sample size n. The following variables are needed: (1) the confidence level and the equivalent Z-score (2) the confidence interval c, which specifies the margin of error (3) the variance p in results expected from the population. Population size can be disregarded if large or unknown. The formula [10] reads:

$$n = \frac{Z^2 * p * (1 - p)}{c^2}$$

Using a confidence level of 90% equals a Z-score of 1.645. A confidence interval of 15% and a variance of 50% (the most forgiving variance) leads to a suggested sample size of 30 people:

$$n = \frac{1.645^2 * 0.5 * (1 - 0.5)}{0.15^2} = 30$$

This assumes the sample to be representative of the population, which is why random sampling (within the customer segment) is preferable.

The same formula can be used to calculate the confidence interval or variance as the customer segment or early adopter definition is refined.

If the population is known and relatively small, such as when selling to larger companies, the finite population correction formula can be applied. With the finite population size denoted N, the formula [10] reads:

$$n = \frac{n_0 * N}{n_0 + (N - 1)}$$

Appendix C

The Sean Ellis Test

On his blog, *Startup Marketing*, Sean Ellis published a test for measuring product/market fit [12]. Users of a product are asked to answer how they would feel if they could no longer use the product. If at least 40% say they would be "very disappointed", there is a good chance product/market fit has been reached.

The 40% threshold was chosen after comparing results from over 100 startups; those above it generally gained traction whereas those below it generally struggled.

Maurya's version of the test [24] involves a survey with the following answer options: (1) very disappointed, (2) somewhat disappointed, (3) not disappointed, it is not really that useful, (4) no longer use the product.

Depending on the situation, the exact wording of the question might need tweaking. If users have invested heavily in the product, hearing that it might be taken away could harm relations.