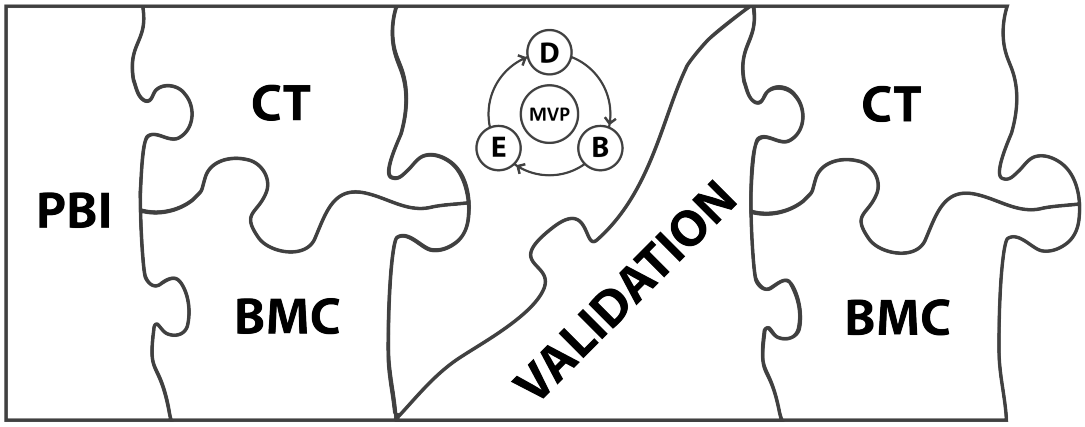
Aalborg University

SW10 Project

Specialization

Entrepreneurial Software Innovation



Group:

Anders Christian Kaysen

Frederik Bruhn Mikkelsen

Stefan Mailund Thilemann

Supervisor:

Ivan Aaen

AALBORG UNIVERSITY
STUDENT REPORT

Department of Computer Science

Software 10th semester

Address: Selma Lagerlöfs Vej 300

9220 Aalborg Øst

Phone no.: 99 40 99 40

Fax no.: 99 40 97 98

Homepage: `http://www.cs.aau.dk`

Project title:
Entrepreneurial Software Innovation

Subject:
Specialization

Project period:
Spring 2016

Group name:
IS1012F16

Supervisor:
Ivan Aaen

Group members:
Anders Christian Kaysen

Frederik Bruhn Mikkelsen

Stefan Mailund Thilemann

Copies: 5

Pages: 68

Appendices: 0

Finished: June 8, 2016

Abstract:

A case study for developing a software business using Lean Startup has shown how it prioritizes short development time over software quality. Focusing on best software design principles to improve the software quality, allows practitioners to rapidly embrace change in regards to business and product development by combining concepts from Software Innovation and Software Entrepreneurship. The relationship established between Software Innovation and Software Entrepreneurship identifies a gap when developing a business. Software Innovation focus on development of the product, while Software Entrepreneurship focus on the surrounding software business.
The Entrepreneurial Software Innovation model – a suggestion towards bridging the gap between the two methodologies, by emphasizing good software design while maintaining the relationship between product and business.
The model presented is based on experience gained while developing a software solution for a Danish sports organization, in which more than 1700 users registered during a four month test period.
By developing software using best software design principles, Entrepreneurial Software Innovation enables entrepreneurs to extend their business by exploiting contingencies. This is achieved by *branching*, which allows practitioners to expand their customer segments by making only minor changes to the product.

Preface

With the prospect of becoming engineers empowered with the skills to create the software technologies of the future, and with a desire to innovate, it is not difficult to see why we aspire for the entrepreneurial lifestyle. Entrepreneurs are free spirits that are not bounded by any limits but their own imagination and their ability to carry their ideas to life. However, being an entrepreneur also implies that decisions have to be made; often ones with consequences that can make or break their visions. An old saying states that "*a bird in the hand is worth two in the bush*" which can be interpreted as to be happy with what you have, instead of risking what you have by chasing the unknowns. Another characterization of this old saying in the context of entrepreneurship, is Sarasvathy's definition of effectual logic; the *bird in hand* principle here refers to entrepreneurs beginning their journey by looking at who they are, what they know and who they know. By using the *means* at their disposal, entrepreneurs are able to find a path that suits them well, in order to achieve the desired *effects*. Sarasvathy also describes causal logic; another approach to start a business, where means are chosen in order to achieve the desired effect. As such, entrepreneurs with a causal mindset looks for opportunities not immediately at their disposal, to be able to predict aspects of an otherwise uncertain future [15].

This Master Thesis addresses how to establish a software-intensive business based on two very distinct approaches:

- Turning an idea into a business by exploiting contingencies, and
- Creating a business by collaborating with a customer to steer initial development.

During our project we strive to learn from our practical experiences by evaluating on our previous actions, combined with our new knowledge obtained from utilizing theories such as Lean Startup (Ries [14]) and the Early Stage Software Startup Development Model (Bosch et al. [5]), as described in Kaysen et al. [11].

References and citations are mostly provided in number notation without listing the authors, for example as: [14], or with the authors listed as: Ries [14].

We would like to give special thanks our supervisor Ivan Aaen for his work on software innovation and especially Essence [1], a pragmatic way to add creativity to software development projects. His guidance and involvement throughout the project has been invaluable, and have greatly helped us see the connections between software innovation, software development and entrepreneurship.

Summary

This thesis is based on work from a previous report written by the same authors. The report describes a software product called *Gymnastbillet*, developed for a Danish sports organization, DGI Nordjylland. The product was developed using Lean Startup and the Early Stage Software Startup Development Model (ESSSDM) with the aim of establishing a business. During our work with Lean Startup and ESSSDM we discovered that they do not emphasize any processes or practices for developing software. In turn, Lean Startup prioritizes minimizing the development time, partly by sacrificing software quality.

We believe that there should be more focus on the actual software product in Software Entrepreneurship. Thus we have dived into the area of Software Innovation where we have found the methodology Essence. Essence allows developers to maintain the relationship between product, customers, and the context in which the product is to be used. However, Essence lets developers decide which development process is followed when developing the product. Thus, by combining Software Innovation and Software Entrepreneurship we believe that we are able to both minimize the development time and at the same time focusing on developing high-quality software. Although combining these two areas there is still a gap that must be covered.

Thus, this thesis suggests a new model, Entrepreneurial Software Innovation (ESI), for entrepreneurial practitioners, that permits low development time without sacrificing software quality. ESI is based on concepts from both the area of Software Innovation and Software Entrepreneurship. By including concepts from Essence we are able to keep practitioners' focus on the product and customer, such that it is never overshadowed by the business aspect. The entrepreneurial aspects of ESI are based on concepts from Lean Startup and ESSSDM. We use the term MVP, however, extending the notion such that it supports development of business ideas and low fidelity prototypes, to validate the potential of an idea using an MVP.

ESI introduces branching as a way to expand the customer segment from one customer to several similar customers with a minimal effort. Branching is facilitated by the software practices suggested in ESI. By maintaining good software design with a focus on near-decomposability, it is possible to branch out only by making minor changes to the existing product. We demonstrate branching by presenting how Gymnastbillet can serve customers similar to DGI, without making significant changes to the product.

Contents

CHAPTER 1

Introduction

This thesis follows the path that were decided during the pre-specialization (9th) semester, where results were obtained by establishing the grounds for how a software-intensive business can be formed (Kaysen et al. [11]). Leaning on well-established entrepreneurship theories such as The Lean Startup (Ries [14]) and Effectuation (Sarasvathy [15]), two products were developed following very different approaches to establishing such business.

The first product was initiated with the desire to find a mass-market fit. Such potential was discovered by a member of the project group; he found that it was difficult to find other cycling enthusiasts to ride with, without being member of a cycling club. This vision was disclosed to friends and family members who all vouched for its potential and need. As a result, a social platform called *Juvono* (*"juvo"*, Latin word meaning: "help", "assist", "aid", "support") was developed, enabling cycling- and running enthusiasts to find others to exercise with. When referring to Juvono throughout this report, we are referring to this social platform.

However, as later described (in Past Experiences (Chapter 2)), the initial vision proved not to hold, as we were unable to successfully evaluate its foundation by attracting crucial early adopters. This led to a change in direction where a new business model was developed to support a potential customer. The interesting part of this collaboration is that the customer provided us with an initial use case to base the developments on, as they were having trouble managing free tickets handed out to members. Without the ability to effectively manage free tickets they are at risk loosing important revenue, as they cannot guarantee that only eligible members enter an event. Considering the number of events held by this customer each year, this can amount to a significant loss of revenue. The developed solution facilitates this functionality with an administration system used by managers to grant or deny members access to certain events, and a mobile application used by members, in order to generate a personalized ticket to be shown to ticket checkers. Thus, the customer had their problem solved, and we were provided with initial users from their organization – taking away the hazzle of attracting early adopters, enabling us to validate the solution in collaboration with the customer. Throughout the report, we refer to this solution as *Gymnastbillet*.

We find the working relationship with this customer interesting, as they are part of a larger organization where their members come to act as early adopters. Basing developments on this customer requires us to make substantial changes to the way we

work; working under the assumption that the product eventually will cater to mass-market adoption (Juvono) is significantly different from working in collaboration with a potential customer in order to develop a product that eventually will fit the need of others. With Gymnastbillet, this is our exact goal – using DGI as a catalyst to establish the grounds for a *potentially* viable business that can be extended by adhering to established software development practices to support morphological customers, with the least amount of effort required.

As described above, we have been keen on investigating entrepreneurial processes while also focusing on our core competences, namely software development. As such, the research is grounded in the field of Software Engineering in the area of Information Systems, focusing on how we can combine existing knowledge about entrepreneurship and software development, to the extent that practitioners are able to synchronize both business- and software developments. Currently, it is not clear in present research how a decision involving either product or business affects the other. Therefore, our aim is to develop a model that takes part in this decision process by defining a framework describing their interdependencies, such that practitioners at all times are able to identify their progress and act accordingly upon their results.

This thesis is divided into 8 chapters: In Chapter 2 we describe past developments and how well-established entrepreneurial processes have been put to use during our pre-specialization semester. Section 2.5 outlines our findings and provides the reader with a discussion and reflections on their use, to form the basis of Chapter 3 that sets to clarify the exact questions to be answered with this thesis. Chapter 4 introduces parts of the Software Innovation methodology, Essence [1], as the theoretical foundation for combining entrepreneurial processes with software development, while ensuring the best grounding for creativity and innovation. Chapter 5 introduces the result of this thesis: the *Entrepreneurial Software Innovation* model. How we recommend the model to be used, as well as our actual usage, is described in Chapter 6. Chapters 7 and 8 concludes and reflects on the results.

CHAPTER 2

Past Experiences

In our previous work (Kaysen et al. [11]) we started the development of a business and product which also serves as the case for this Master's thesis. During this work, we gained experience using various models such as The Lean Startup [14] and Early Stage Software Startup Development (ESSSDM) [5] while working on a product in close collaboration with a customer. These models, as well as the theories of Sarasavthy (Sarasvathy [15]) and other business-related development tools such as Business Model Canvas (Osterwalder et al. [13]), were utilized to develop the business from idea to a fully implemented product used by more than 1700 users. A description of the theories relevant to this work is presented in Sections 2.1 to 2.3.

Kaysen et al. [11] describes the development of the product in three iterations where each iteration represents a unique stage of the development and thus the state of the product. This chapter summarizes these iterations and uses them to present the experiences gained during this process, thus forming the basis of the rest of the report. Figure F2-1 presents a timeline showing the start of the work and the three iterations made during our previous work, until the beginning of this Master's thesis.

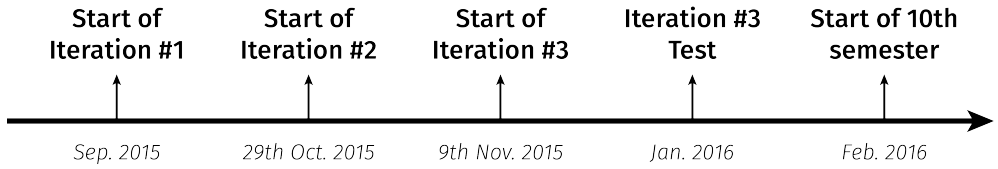


Figure F2-1: Timeline providing an overview of the iterations described in Kaysen et al. [11].

2.1 Lean Startup

This section is taken from Kaysen et al. [11].
Eric Ries, a student of Steve Blank[1], has developed a methodology called the *Lean Startup* which enables founders to deliver a product to customers faster than using traditional methods, and minimize potential wasted resources by focusing on what customers want. This section introduces the most important aspects of Lean Startup.

[1] An american entrepreneur and now teaching entrepreneurship at U.C. Berkeley, Stanford University, Columbia University, NYU and UCSF. Author of the book *Four Steps to the Epiphany* which has been called the book that launched the "Lean Startup Movement" [4].

2.1.1 The Methodology

Ries [14] establishes with the Lean Startup methodology a new approach for building businesses and products that breaks with the traditional thinking of how a business should be constructed. In short, it takes out the tedious phase of planning and focuses instead on the core business hypotheses and how to attract early adopters that are willing to use the product. This is achieved by adopting *lean thinking* (as known from Lean Manufacturing, a process focusing on minimizing resource-waste in production) and *agile development* (as known from software development). Of course, planning can not be completely eliminated as *making a choice* often involves some kind of planning. As agile development implies that work is done iteratively and incrementally, founders are able to adapt "as they go" and make choices accordingly – something that is very difficult when working sequentially – especially if no early adopters are willing to use the product. In such case, a lot of development efforts (resources) have been wasted, and the business might not be able to recover.

Build-Measure-Learn

Working iteratively is supported by the Build-Measure-Learn (BML) feedback loop as illustrated in Figure F2-2. It is a three-phased loop where the minimal viable product is established during the *Build* phase, analytics generated based on customers in the *Measure* phase and new ideas for improvements and innovation are based upon the measured data obtained from the *Learn* phase. What should follow from the *learn* phase is what Ries characterizes as *Validated Learning*, which is the parts of a (business or product) hypothesis that have been validated by the output of the *Measure* phase. The *Learn* phase describes the decisions to be made during the next iteration of the BML loop, based on what have just been learned. BML thus becomes an essential activity of a startup as it helps turning ideas into products. Ries further recommends that all phases of a startup are optimized in order to accelerate the feedback cycle of the BML loop.

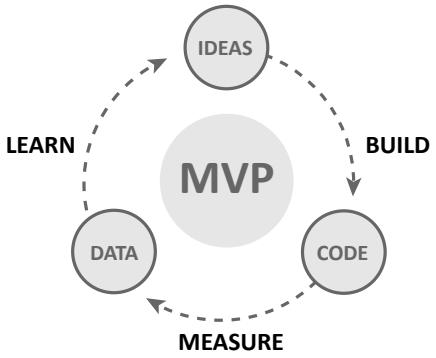


Figure F2-2: The Build-Measure-Learn feedback loop.

When completing a BML loop cycle, the startup must validate the result and choose to either *Persevere* or *Pivot*. When one choses to *Persevere* it means that the startup

continues in the direction established the last time going through the stages of the loop, which happens when good results are obtained from customers. However, in some cases it is necessary to steer the ship out of trouble to avoid unnecessary resource waste; such action involves aligning with customer needs and thus a change in product development. This is the *Pivot* and can be characterized as a drastic change in the business strategy, for example if you learn that your initial strategy has no market fit, you are required to make a major change. Ries [14] gives an example of a pivot based on his own startup IMVU, which initially strategy was to provide an virtual 3D environment that could use existing Instant Messaging (IM) services as add-ons for their customers to chat with existing friends. What they learned was that customers was not interested in this particular functionality, they would rather meet new people and thus not use this service with their existing friends. Thus IMVU made a pivot and changed their strategy, dropping the support for IM add-ons by developing their own IM client customers could use to meet new people. If they did not perform this pivot and continued doing "business as usual", resources might be exhausted before awareness of the problem could be raised.

Minimum Viable Product

The Minimum Viable Product (MVP) is a version of the product that allows the startup to take one full cycle of the BML loop as fast as possible, and with minimal effort ensuring resources are not wasted. This means that the MVP typically is lacking features which later in the process may end up being essential for the resulting product. It might also be necessary to "ignore" well-known product errors (bugs in software terminology), in order to reduce time to market for the product. Leaving out features and ignoring errors can affect how customers sees the product, but according to Ries, passionate early adopters does not take much notice of this; rather, they are happy just being able to access the product (assuming the Value Propositions cater their needs).

2.2 Early Stage Software Startup Development Model

This section is taken from Kaysen et al. [11], although we have elaborated on the description of the first step, *Idea Generation*.
Using Lean Startup in a software startup is proven by Ries [14] to work, but Bosch et al. [5] identifies a number key areas that need to be improved. The result of their attempt to improve these key areas is the Early Stage Software Startup Development Model (ESSSDM). An overview of ESSSDM is shown in Figure F2-3. It consists of three steps (idea generation, backlog and a four-stage funnel), as described further below.

The first step, *idea generation*, is where the ideas for new products come to life. It is thus an essential part of understanding the needs of potential customers. Typically, it involves talking to customers to understand the potential of the idea. Idea generation can help expand an existing product portfolio or generate ideas for entirely new products. The techniques proposed, by Bosch et al., for generation of extraction of ideas are: 1) *Exploratory Interviews* which is talking to the customers to get an

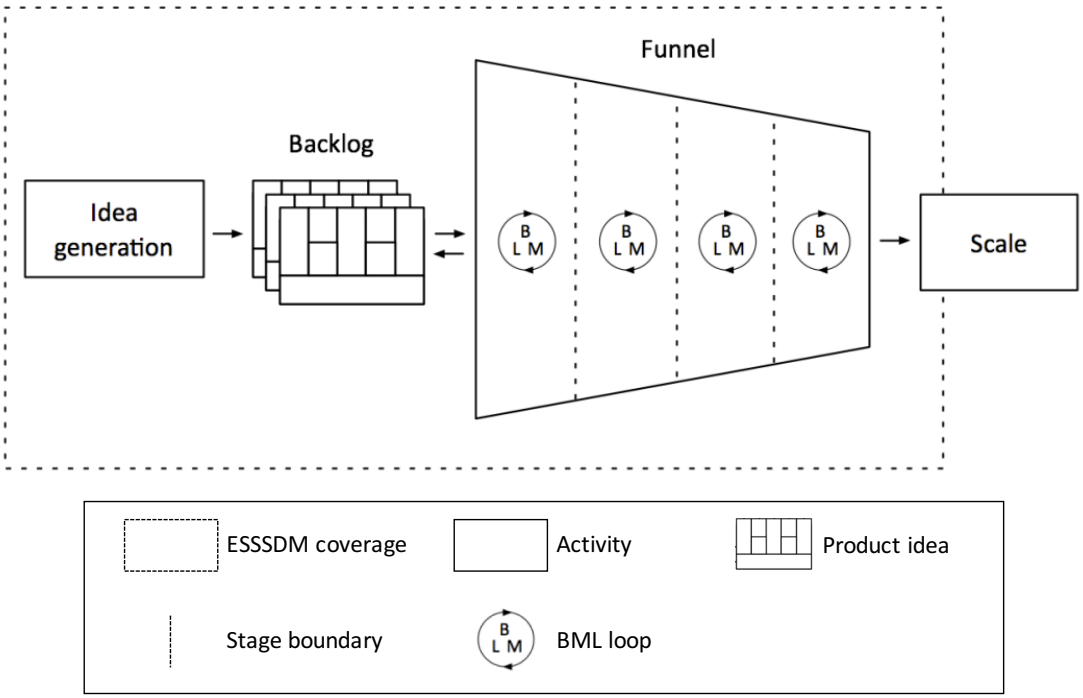


Figure F2-3: Overview of the Early Stage Software Startup Development Model [5].

understanding of how these potential customers run their business, such that you are able to identify potential market potentials. 2) *Follow-me-homes* which is a technique where you spend one or more days at the office of potential customers to observe their regular work day, in order to identify problems that can serve as a potential business case. 3) *Brainstorming* for generating ideas based on some topic. Specifically Bosch et al. proposes *SCAMPER* which is a brainstorming method for creating new ideas by modifying existing products [18].

The second step, the *backlog*, is a prioritized collection of business models that represents one or more ideas. Bosch et al. [5] notes that being able to compare ideas is crucial as it allows to work on multiple products at the same time while being able to easily decide between them if need be.

The third and last step, the *funnel*, consists of four stages that helps with developing the ideas that are put into the funnel using the BML loop [14]. Each stage in the funnel has clear exit criteria which defines when to move on and when to repeat the current funnel stage. The stages are characterized as follows:

1) *Funnel Stage 1: Validate Problem*,

2) *Funnel Stage 2: Validate Solution*,

3) *Funnel Stage 3: Validate MVP Small-Scale*, and

4) *Funnel Stage 4: Validate MVP Large-Scale*.

1) Focuses on validating the problem, which can be compared to the "get out of the building" approach coined by Steve Blank in his Customer Development methodology (as adopted by Ries [14]). 2) Validates if a possible solution is appropriate for solving the problem and that customers are willing to pay for it. 3) Solves the problem by focusing on building an MVP and testing it on a small set of early adopters while answering questions such as if the MVP solves the problems that the customers want solved. 4) Tests the solution on a large set of early adopters to see if a product/market fit exists.

Following the funnel stage is what Bosch et al. [5] characterizes as an *idea worth scaling*. The step is reached when it has been possible to successfully answer the exit criteria defined for the *Validate MVP Large-Scale* funnel stage. If answered successfully, the idea is validated and thus ready for commercial scaling. This fulfills the purpose of following ESSSDM, and it is safe to presume that further development can be initiated by following Lean Startup or another methodology, as Bosch et al. [5] only propose a solution for the *early stage* software startups.

2.3 Business Model Canvas

This section is taken from Kaysen et al. [11].
Managing a company requires a deep understanding of the context it is operating within. Without a clear definition and understanding of how it creates, delivers and captures value to and from customers, it is difficult to operate in a competitive market. In order to develop the company's products and services, a business model can be formulated to clarify how it expects to create revenue, what funds it requires to operate its services or maintain and develop products, and what activities that should be focused on to maximize the value delivered to its customers.

Osterwalder et al. [13] presents the Business Model Canvas (BMC), a visual tool that structures how such business model can be built. It establishes a shared language that eases the process of developing and maintaining it, while empowering its designers to see patterns in the model that can attribute to new innovations in the way the business is operated.

The structure imposed by the BMC is obtained by connecting its nine building blocks as seen in Figure F2-4. They help business model designers to cover the most important aspects of a business and to some extend in what order they should be defined. They include customers, offers (products and services), infrastructure (how to deliver products and services) and its financial viability. Each building block is logically connected to at least one other block in order to illustrate how value flows from one block to another. Below, each of these building blocks are described.

Customer Segments (CS)

The Customer Segments block groups types of customers into distinct segments. If for example a business caters to multiple individuals within an organization, a possible

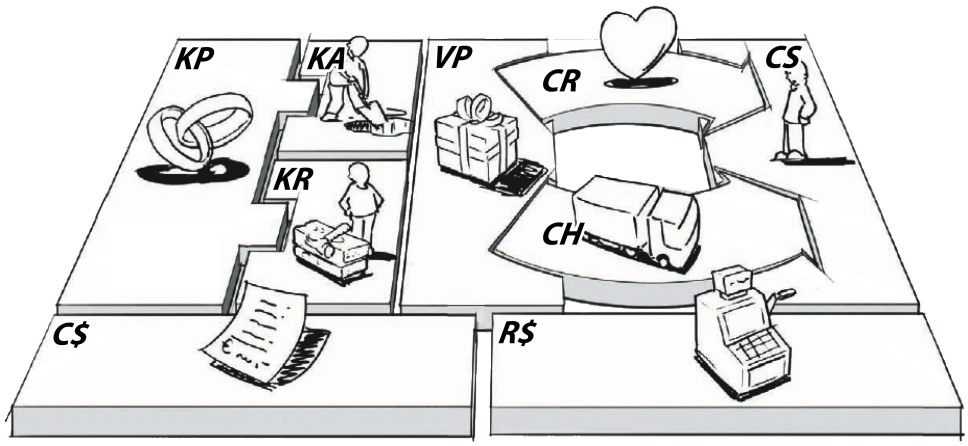


Figure F2-4: The nine building blocks of Business Model Canvas [13].

customer segment can be the grouping of individuals based on their role in the organization. This distinction enables business model designers to for example find value according to customer needs, their behaviours and how they can be reached.

Segmenting customers also enables designers to identify the most important customers that the business ought to focus on in order to maximize its profits. Gaining insight to who customers are and how they behave, help businesses to understand how to steer in the markets they are operating.

Value Propositions (VP)

Knowing who your customers are obtained from the insights of the CS building block can enable a business to better design the value that it should provide, as each segment often caters to different needs. Value Propositions are the main offers a company has to its customers. It can be characterized as the selling-point that accomondates customer retention and satisfaction, as it is what sets competing businesses apart.

Channels (CH)

Having established the VP building block, Channels helps a company deliver value to a given Customer Segment. They comprises every way of communicating with a customer – for example through advertising, sales and support. Choosing the right channels to deliver a Value Proposition can impact how customers will adopt a product; would a customer prefer to be contacted directly by an in-house sales department, or, does channeling VPs through an distributor make your company more credible? If the customer is interested in minimizing costs, he might prefer to establish a direct channel – on the other hand, a global distributor might be able to cut delivery time, improving the competiveness of the customer.

Customer Relationships (CR)

Customer Relationships is related to how a company interacts with a specific CS. It is not related specifically to the products or services offered by a company (VPs), but rather how we can aquire new customers and maintain or improve the relationship to existing ones. Thus, the CR building block optimizes the value a company provides to and recieves from a customer in order to improve aquisition and retention rates and maximizing sales.

Revenue Streams (R$)

In order to keep a business profitable it is necessary to investigate how it can generate revenue from the Value Propositions offered. Revenue Streams establishes what values a customer is willing to pay for and if alternatives should be investigated. For a electronics store, it can choose to include a service fee in the purchase price or offer it as a subscription. Choosing one over the other can greatly impact the amount of sales a company is able to generate, and thus the amount of profit generated from sales.

Key Resources (KR)

The Key Resources building block focuses on the most important aspects of a business that are required in order to create and deliver Value Propositions. Key Resources can be either physical, intellectual, human or financial. They can also be characterized as the requirements needed to fulfill the goals of the previous mentioned building blocks. For example, if a website is relying on pay-per-click advertisement to generate revenue, a KR related to the Revenue Stream (R$) is the number of visitors coming to the website, as this have the potential to increase the click-rate (assuming the cohesion between visitors and click-rate is linear), which in turn would increase the revenue.

Key Activities (KA)

Key Activities are closely related to Key Resources. But, instead of being dependent on the assets belonging to a company, KA describes what a company should do to enact the business model. For a software development company, it is of course to develop software – but depending on the business model, it can also be activities such as enforcing developer guidelines or best practices to help the company deliver their projects on time and of the best possible quality.

Key Partnerships (KP)

Most companies rely on external resources to deliver its Value Propositions. The Key Partnerships building block describes who the key partners are and provides an overview of where Key Resources come from.

Cost Structure (C$)

In order to be profitable, one need to know the cost of operating a business. Having defined Key Resources, Key Activities and Key Partnerships, it is possible to identify

the primary costs incurred by delivering value to customers. The Cost Structure building block is dominated by two extremes; cost-driven and value-driven. Obviously, the costs of operating a business should be lower than the revenue generated for it to be a profitable business.

2.4 Past Developments

As mentioned earlier, Kaysen et al. [11] describes the product developments in three distinct iterations. In order to gain an understanding of the product and its current state, each iteration is briefly described in the following sections (Sections 2.4.2 and 2.4.3). In the section that follows this description (Section 2.5), we present the findings as well as a discussion of the past developments, in order to uncover a potentially improved process for aligning business and software developments, which is presented in Chapter 5.

2.4.1 Motivation

The background for establishing a business during 9th semester does not stem from nothing; being software developers and thus being able to create and innovate, we wanted to put our skills to good use and develop a product that lasts more than a semester. Before starting 9th semester we were introduced to the New Venture Creation semester at Aalborg University, which sparked the desire to mix software development and entrepreneurship. Even though it was not possible to attend this semester, their description embodies our motivation for focusing on entrepreneurship:

> *New Venture Creation is an international cross-disciplinary semester with focus on business creation and business management. It is a real-life experience in entrepreneurship. And it is an opportunity for students from all disciplines and from all over the world to learn how to start your own company, design sustainable business models, and importantly – interact with a lot of interesting business people (Danish and international practitioners) through professional discussions* [6].

Using this motivation to drive the developments described in Kaysen et al. [11], two products were developed. Each of the following iterations reflect major stages in the development of these products.

2.4.2 Iteration #1

This section describes the idea that formed the basis of Juvono. The idea originated from a member of this project group following his passion for road cycling and the fact that he was unable to easily find others to share his passion with, if not having to join his local cycling club. On this basis, an idea for a web-based social platform to support both running and cycling *events* was developed. The platform lets users post their events to make them publicly available, and thus make it possible for other interested users to sign up for events that matches their interests, such as a bike ride in

their living area. Figure F2-5 shows the product developed (initial MVP) during this iteration.

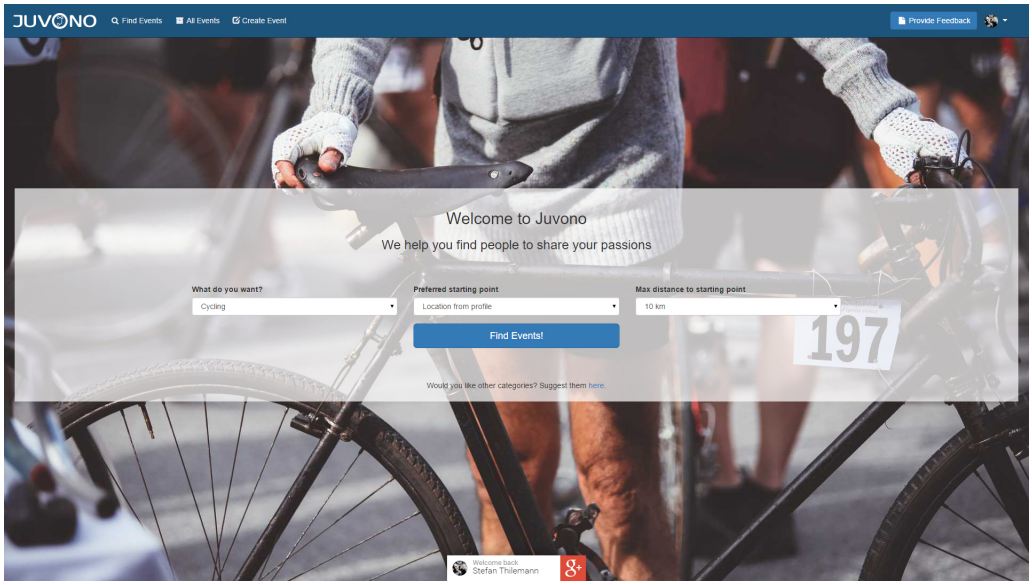


Figure F2-5: The MVP following the first *Build* phase while developing Juvono.

As the target audience for this product includes everyone searching for a cycling- or running partner to exercise with, its adoption caters to what Ries characterizes as *mass-market adoption*. Hence, the Lean Startup methodology takes center stage in this iteration. The methodology also encapsulates many concepts we find critical to a startup; the concepts of an MVP and thus the idea of minimizing waste by continuously learning (as what Ries characterizes as *validated learning*) by looping through the BML feedback-loop. Our problem was validated by friends and family based on their feedback.

Unfortunately, after building the initial MVP, it was not possible to attract the required early adopters in order to test the product, and thus validate the MVP during the measure phase. As a result, we were unable to begin the learn phase to obtain validated learning. According to Ries, validated learning is the concept that helps steer the project in the direction of the early adopters, and therefore decide whether to persevere or pivot. This posed a high risk for the future of this MVP; without validated learning it would not be possible to ensure rightful use of resources and that a market for the product exist.

While we struggled to attract early adopters, an opportunity to start a new project in the same area as before (event management) was discovered. This opportunity resulted in the decision to pivot, in order to follow this new and undiscovered potential, as the nature of the project guaranteed enough early adopters in order to obtain the essential validated learning. This process is summarized in the next section.

2.4.3 Iteration #2 and #3

The gymnastics department of DGI Nordjylland (a Danish sports organization) were discussing the idea of a way to overcome abuse of free tickets handed out to their members. On a board meeting they brought up the idea of a mobile application that could facilitate access to their events by generating digital personal free tickets for only eligible members. The most common case of abuse happens when a gymnast attending multiple teams receives one ticket per team, and a gymnast with two (or more) tickets, hands out tickets to another ineligible person, such as a friend or family member, who normally would pay the required spectators entrance fee. This is a consequence of DGI being unaware of who their members are; they are only aware of the associations that are under the organization of DGI, and not the members of each association. Associations sign each of their members up for a team, but since this process is not governed by a centralized system, members will receive multiple tickets when taking part on multiple teams, or in multiple associations. The hierarchy of DGI is illustrated in Figure F2-6.

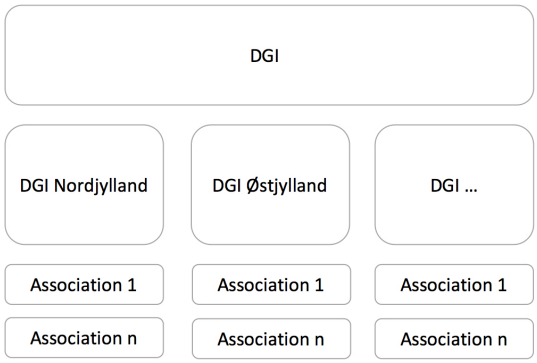
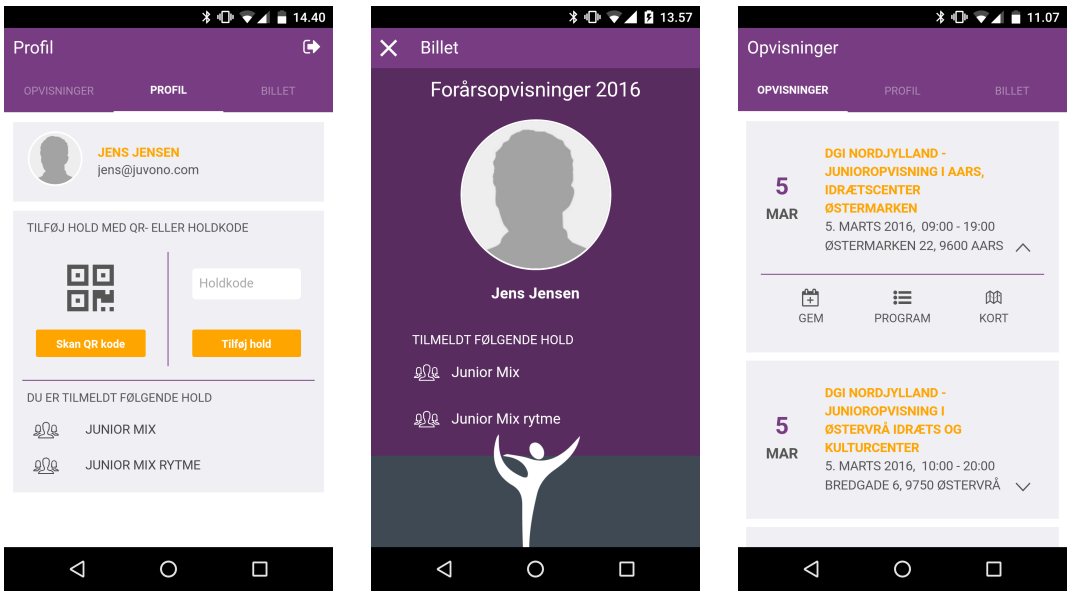


Figure F2-6: The hierarchical structure of DGI. Please note that each association depicted can consist of 1 to n teams.

Being informed of the idea, we reached out to DGI and convinced them that the system should be developed as part of this thesis, and as a step in the process of establishing a new business. The output of the developments is a mobile application that shows the ticket, as well as a web-based administration system that makes it possible to add events and assign tickets to eligible members. These applications are supported by a cloud solution that when combined, minimizes the risk of abuse by generating a digital personal ticket verified for each individual member, and thus ensuring that tickets are only generated for members that are eligible to receive a free ticket. The solution is aptly named Gymnastbillet, or in English, *Gymnast-ticket*. Screenshots of the most important parts of the mobile application is shown in Figure F2-7. A small excerpt of the administration system is provided in Figure F2-8.

What makes DGI interesting is the fact that they relieve us from attracting early adopters (or end-users), as their members are required to use the system in order to gain a free-pass to DGI's events. This establishes DGI as an early adopter of the system and also as a mediator between us and our end-users. This is fundamentally different from the approach taken during the developments of Juvono in the previous iteration



(a) The profile screen of the application.

(b) The ticket screen of the application.

(c) The list of all events as presented in the application.

Figure F2-7: Screenshots of the three most important screens within the mobile application [11].

(Section 2.4.2). Instead of building a business based on our own ideas, the system is modeled according to the needs of DGI, with the idea of being able to generalize it for other customers in mind. A potential disadvantage of this approach is that we establish a relationship where we are relying on the willingness of DGI to pay for their use; if they do not commit to it, resources are wasted and we are required to find another early adopter who fits the requirement specification outlined by DGI.

In hindsight, we attribute the deficiencies uncovered after finishing the first iteration to how Lean Startup handles progress and validation. To overcome these deficiencies we formally introduced ESSSDM (introduced in Section 2.2) as a layer on top of Lean Startup to our process. As such, the purpose of ESSSDM is to help identify the state of the idea by defining exit criteria for each of the four funnel stages, ensuring progress and whether or not an idea is worth pursuing.

The following section further elaborates what we have learned from these iterations. Thus, Section 2.5 will form the basis for the research topic throughout this thesis, and help developing the problem statement Chapter 3 in order to concretize the problem further.

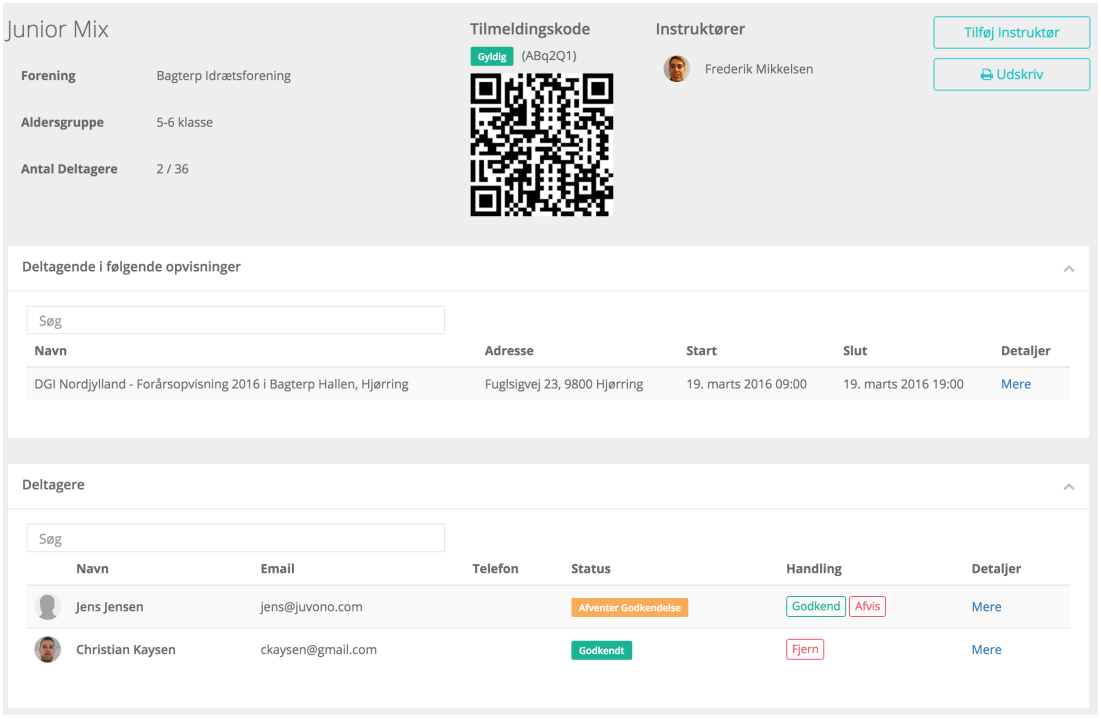


Figure F2-8: Example of a team page as seen by an instructor [11].

2.5 Findings

Based on the experiences just described in relation to the work in Kaysen et al. [11], we present our findings in the following sections.

When developing a product using Lean Startup, practitioners are advised to spend as little time as possible on development. However, in this section we discuss how software processes and practices can help practitioners build a high-quality software-product following best software design, without increasing the development time significantly.

Customers play a very important role for entrepreneurs, since without customers there would be no business. Thus, this section also presents reflections on how customers should be approached when developing a software product, as well as included in the development of a software product.

2.5.1 Reflections on Lean Startup

Ries defines the Lean Startup as a methodology that helps to get a desired product into the hands of customers as fast as possible [14, p. 228]. Furthermore, he emphasizes the importance of gaining insight into the minds of customers via *validated learning* in order to reach a broad uptake; if customers do not like your idea, your startup is at great risk of failing. As such, the Lean Startup methodology advocates developers to steer away from the "just do it" approach of development, by turning to what he describes as *innovation accounting*. The key in innovation accounting is to be able to distinguish

between *vanity* and *actionable* metrics; discerning what is important to your startup, and what is not. For example, Ries characterizes the total number of visitors of a website as a vanity metric, as it can be almost impossible to infer the decisions that preceded an increase compared to previous measurement. Actionable metrics on the other hand are easier to measure and distinguish from each other, as they are carefully planned. For example, if one need to confirm or refute that a new product feature for a website is worth its while to implement, split testing can be utilized such that some visitors are served the current version of the website, and others a version with the new feature. By analysing and comparing their movements and actions, it becomes easy to choose between them; if the new feature yields a lower bounce rate[2], measured over the entire test period, it is clear that it is superior to the current version and should thus be served to all visitors.

Interpretation of concepts

We believe that Lean Startup encompasses many elements that are helpful for entrepreneurs in order for them to quickly establish a viable business. The ability to navigate under high levels of uncertainty, while ensuring that the minimum amount of resources are wasted, is not an easy feat to accomplish. Our experience with Juvono shows that the *just do it* approach prepares the ground for resources being wasted, as the business hypothesis was refuted before finishing the initial development cycle of the BML loop. With a stronger focus on innovation accounting, it should have been a priority to attract early adopters from the inception of the product. However, having practiced good software design principles such as near-decomposeability and seperation of concerns while developing Juvono, we were able to rather easily support the requirements outlined by DGI.

Even though we are avid believers in the notion of a minimum viable product, we find its definition by Ries to be contradictory in relation to good software design principles. Being able to quickly, and without much effort, roll out a product can be essential in order to establish a market lead; however, if it happens on the expense of good software design, the joy can end up being short-lived. In Lean Startup, Ries describes [14, pp. 228]:

> *[...] the Build-Measure-Learn feedback loop is a continuous process. We don't stop after one minimum viable product but use what we have learned to get to work immediately on the next iteration. Therefore, shortcuts taken in product quality, design, or infrastructure today may wind up slowing a company down tomorrow.*

Even though it is clear that Ries agrees on our point on product quality, his argumentation becomes vague when mentioning an example from his startup, IMVU, where he recalls how customers were unwilling to try his product because it was full of bugs and badly designed. He further states: *"It's a good thing we didn't waste a lot of time fixing those bugs and cleaning up that early version"*. As such, according

[2]Bounce rate is the percentage of visitors that leaves a website immediately after entering it, without viewing other pages.

to Ries, quality is only an issue if it is directly visible to the customers. Although our system architecture is invisible to customers, we do not believe it is viable to slack on good design. Had this been the case with Juvono, we reckon that a significantly larger development effort would have been required in order to support the case presented by DGI.

2.5.2 Reflections on ESSSDM

During our previous work [11, ch. 6-8] we have worked with ESSSDM in relation to ideas defined by DGI. The model is defined as a wrapper for Lean Startup that enables practitioners to find an appropriate idea to scale, beginning with the *idea generation* stage.

We first started using ESSSDM during the development of Gymnastbillet. We believe that it would probably have been better suited during the development of Juvono where work was done according to an untested business hypothesis: That amateur athletes would use a social platform to find others to exercise with. This hypothesis was approved by friends and family, as they could see a need for such system. As such, the validation of Juvono bares some similarities with idea generation in ESSSDM, where ideas are validated according to exploratory interviews with potential customers and brainstorming techniques. Gymnastbillet is mainly based on requirements provided by DGI, and can therefore not be characterized as an idea we have generated in order to establish a business. Instead, it can be seen as a business opportunity that leans towards more traditional software development, where developments typically follow a requirements specification outlined by a customer. The difference lies in the fact that the system is not developed for a specific customer; we are merely using DGI and their members as early adopters that can help form the base functionality of the system, such that it can be generalized to other customers at a later point in time.

The second step of ESSSDM is the *backlog* where ideas for potential products are prioritized. It is clear that the idea of a backlog originates from agile software development, where ideas are prioritized according to their importance to the product, or by the estimated time (cost) of development. However, the authors do not explicitly define how the backlog should be prioritized other than it should be written in a "comparable format". Another part of the model that is not quite clear is how to build the backlog, but, by analyzing the iconography described by the model (as seen in Figure F2-3), it can be inferred that the Business Model Canvas should be used to capture business ideas for the backlog.

We followed the backlog step as intended by the authors, even though Kaysen et al. [11] bares no mention of it. It becomes evident when looking at the change in direction between Juvono and the introduction of DGI; Juvono was put on hold in order to focus on developing Gymnastbillet in collaboration with DGI. From this point onwards Juvono was never given any attention, thus having no reason for keeping Juvono in the backlog. Thus, in hindsight we see no good use of a backlog, especially in small project teams. Being able to work on multiple ideas in parallel as the underlying intention of the backlog, as specified by the authors, removes unnecessary focus from the idea

at hand, and might even hinder developers in fulfilling its potential. As such, we see the agile definition of a backlog being more suited, as a backlog of features enables practitioners to prioritize among ideas for current developments.

Entering the *funnel* step of ESSSDM is where we have found its true contribution with reference to Lean Startup. As previously mentioned, we failed to establish *validated learning* for Juvono following the development of its initial MVP. Working closely with DGI and validating progress through each stage of the funnel by fulfilling the exit criteria, proved useful in order to move the development forward and preventing inconsistencies between product- and business developments. If the solution could not be validated according to the problem specified by DGI, it would make no sense to continue to the next stage, as it would involve a high risk of the business failing, or more moderately, increase development costs. In case such inconsistency is found, one has to reiterate through the BML feedback loop. When the solution is found to be consistent in a certain stage, in relation to the problem it solves, the exit criteria enable developers to easily move the project forward. The funnel stages thus become a form of milestones that help practitioners validate when a product is ready for the market.

The last step, the *funnel*, clearly defines operational processes that wraps around Lean Startup to help validate when an idea is ready for further expansion, before leaving the funnel as *an idea worth scaling*. What happens after the last stage of the funnel is left unknown to the reader, but it seems reasonable that ESSSDM can be abandoned and thus only use Lean Startup, since many uncertainties should be eliminated when moving ideas forward through the funnel. The essence of the funnel stages is the exit criteria, ensuring that each stage is validated according to customer feedback. It is similar to *funnel metrics* in Lean Startup, where a sequence of actionable metrics are defined in order to reduce the amount of "learning" for each metric, into a single actionable metric, decisions can be based on. As such, the funnel helps evaluating an idea before staking everything on developing that idea into a product. If the exit criteria are not honored, it is required to reorder priorities; for example by picking another idea from the backlog or making a pivot. Similarly in Lean Startup, a business hypothesis cannot be validated by vanity metrics, as they do not generate validated learning.

Bosch et al. have created a model for early stage software startup development, but as with Ries they fail to dicuss software practices and processes. This means ESSSDM piggybacks on the definitions in Lean Startup. We consider this a poor choice since Ries defines Lean Startup in a way that allows it to be used in industries beyond software. Thus, the only connection Bosch et al. has to software development are the software companies they interviewed, their own software startup (which they do not talk much about), and the name of their model.

The following section dicuss our advantage in following software processes and practices. It explains our beliefs and elaborates on the specific processes and practices that we have followed during the development of Juvono and Gymnastbillet.

2.5.3 Focus on Software Development

Having read Ries [14] and worked with Lean Startup, it has left the impression that Ries has created the Lean Startup in order to support software development – without discussing any software processes or "best practices" for building the actual software product (Minimum Viable Product). Continuous Deployment is only briefly mentioned in relation to support rapid development and iteration through the BML feedback loop. By using Continious Deployment, a feature is deployed to production immediately after it is ready to be tested with customers, ensuring that users always have access to the latest development build. If a problem is detected, a rollback is issued, and the development team is notified. Everyone on the team is blocked from deploying any new features until the current problem is solved.

We believe the reason Ries does not go further into software processes, is that the Lean Startup is intended to be usable "*Beyond Software*". In Ries [14, pp. 191], just after the definition of Continuous Deployment, he explains how it can be facilitated in "*slower moving*" industries. For example in the automotive industry, where more parts of a car are being controlled by software, from the entertainment system to tuning the engine. Another example is that lean manufacturing is being picked up in production industries, allowing products produced on assembly lines to be customized immediately when designers get feedback about the current version of a product [14].

However, we believe there is a need for suggestions of software processes that are proven to work for developing software following Lean Startup. We found that including practices from Extreme Programming (XP) supported our development well, including only those that made sense to us to support our specific needs for developing our MVPs. We thus included the following:

Pair programming allowed us to write higher quality code.

Collective code ownership allowed everyone on the team to be able to extend and add features to a given component. This way we avoided having one team member being a bottleneck.

Coding standards increased the readability of our code and made it easier to understand already written code.

Unit tests ensured that new features did not break existing ones.

Extreme programming actually emphasizes unit tests through Test Driven Development (TTD), but we found TDD to clash with the notion of MVP, since TDD increases the development time by up to 15-16% [3, 9]. Ries [14, pp. 77] defines MVP as follows, "*The MVP is that version of the product that enables a full turn of the Build-Measure-Learn loop with a minimum amount of effort and the least amount of development time*". When using TDD the unit tests are written before the actual code, this means that time is spent thinking up test cases, which is a time consuming task that does not comply with "[...] a minimum amount of effort and the least amount of development time".

In addition to the concepts included from XP, we used the design principles of *near-decomposability*, as proposed by Sarasvathy [16] in relation to her work on Effectuation. Near-decomposability allows us to embrace fast changes, since components are decomposed in a way that make it possible to swap a component with a new, providing extended or totally different functionality. An example of this is an authentication module written for our web client for the first product, Juvono (Section 2.4.2). When we pivoted to work on Gymnastbillet, we were able to use the same component for our new application both in the mobile and web client without further changes, thus saving a lot of time not having to develop the same functionality once again.

Although we used different software techniques and processes to ease our product development, customers played an important role ensuring fast iteration through the BML loop. Our insights of the importance of working closely with customers when developing a software product, is presented in the following section.

2.5.4 Working with Customers

As described in Past Developments (Section 2.4), two different approaches were followed to establish the foundation for a business, namely Juvono and Gymnastbillet. Juvono aimed at finding a mass-market fit for an untested business hypothesis without a specific customer at hand, following our own ideas and vision. Next, a potential customer (DGI) took center stage in order to drive Gymnastbillet to drive the developments of Gymnastbillet with their requirements in mind and their members at our disposal.

Even though working with a customer to drive developments according to a requirement specification somewhat locks in the product according to that customer's needs, it did not influence our decision whether or not to focus entirely on this customer, as we would retain product ownership throughout the process.

In spite of Lean Startup aiming at finding a mass-market fit, it does not specify a specific number of required customers to do so. As such, working with even a single customer can be very useful; according to Ries [14, pp. 107]:

> *If we do not know who the customer is, we do not know what quality is.*

This underlines the importance of having users readily available to test and evaluate the product, as the quality of the product is decided by its actual users, and not the developers. Obtaining user feedback and insight thus become essential to drive developments further.

Reflecting on the customer relationships observed throughout the development of Juvono, it becomes clear that we did not succeed with obtaining any insights into our possible customers. Below we provide a characteristic of why we believe this is the case:

Market Research. Having established the idea to develop into a business, a social platform enabling users to share sports events with each other, this idea should have been thoroughly validated by conducting appropriate market research, and not only disclosed it to friends and family. As we were unable to acquire early adopters in order to test Juvono, the different types of research could have been

conducted. We could have establishing a list of potential early adopters from any possible user segment would have allowed us to establish *focus groups* categorized by interesting criteria such as amateur vs. professional, young vs. old, member/non-member of a club, etc. Such sampling would allow us to find the early adopters most likely customer to use the product in furtherance of finding the best market-fit for the initial MVP, and Other types of research for confirming an idea and getting more knowledge about users are described by the *Idea Generation* step in ESSSDM, see Section 2.2.

Customer vs. Early adopter. Establishing a collaboration with DGI has proven to work as they have provided their members as early adopters; however, our distinction between *customer* and *early adopter* has not been appropriate. Up until this point, we have seen DGI as both our customer and early adopter. This is problematic since the board members, who are the stakeholders at DGI, are not equivalent to the group of people that are actually using the product. Therefore, discussing features and use scenarios with DGI cannot yield the optimal results; focusing on their members using the product would have provided a more accurate sentiment and feedback we could have used to further improve the solution.

Product vs. Business. Both during the development of Juvono and Gymnastbillet (as described in [11]), priorities were arranged such that the product was given most attention, effectively giving business development second priority. However, during the development of Juvono this was more evident as Lean Startup only establishes the grounds for how to develop and test the product. In this case, the business hypothesis is supposed to be validated through the product itself (the MVP). Improving this by introducing the Early Stage Software Startup Development Model during Gymnastbillet, did however not entirely provide the desirable effects according to prove the business hypothesis – that DGI would in fact pay for the system.

CHAPTER 3

Problem Statement

In the previous chapters we found that the models and processes we have used and described in Chapter 2 do not facilitate the kind of project we have been doing, which is starting a business based on one customer's problem, with the goal of reaching other morphological customers.

We have previously worked with Software Innovation using the methodology *Essence* [1]. Essence facilitates innovation while solving a concrete problem by developing a software product via concepts and terms that helps one come around every corner of the development process, while keeping focus on the problem that is being solved.

As stated in Chapter 2 we have also worked with methodologies for Software Entrepreneurship, namely Lean Startup [14], and ESSSDM [5] which extends Lean Startup with operational processes to enable practitioners to gain a better overview of the state of their vision. While these methodologies are about developing a software product, they are merely focusing on the actions and processes around the actual product, without no substantial focus on software development. Essence on the other hand is continuously keeping focus on the actual software product and thus more anchored in software development than the Lean Startup and ESSSDM. Essence does not specify how the software product should be built, since the development model is generalized in a way that lets the developer team choose the software processes and practices they find suitable. Instead Essence encourages Software Innovation by keeping focus on the solution and problem at hand.

During our work we believe to have identified a gap between the fields of Software Entrepreneurship and Software Innovation. Software Entrepreneurship focuses mainly on the business and to get customers to use the product. It lacks focus on the actual software development as also outlined in Reflections on Lean Startup (Section 2.5.1). Software Innovation, however, focuses on development of a software product, without having a specific business opportunity or model in mind. We have not discovered any work that attempts to close this gap.

We believe that with a larger focus on Software Innovation, we are able to build a model that facilitates our approach of developing a business. While doing so, we also believe that it will bridge the gap between software innovation and software entrepreneurship by proposing and solving the following problem statement.

> *How is it possible to bridge the gap between Software Innovation and Software Entrepreneurship, allowing entrepreneurs to focus on developing a*

software-intensive product while being able to accommodate changes to the business model and thus support these with minimal effort?

In the remainder of this report we present a model that combines Software Innovation and Entrepreneurship to enables a developer team to solve a concrete problem for a concrete customer. While the model focus on building a business based on the problem specified by a customer, it also facilitates innovation in addition to the requirements specified by the customer. Whether the business focuses on solving a specific problem for one or more similar customers, or the business focuses on serving multiple customers with different problems, our model, the *Entrepreneurial Software Innovation* model, supports either. The reason is that the two different approaches begins as a problem that one customer wants solved; if you then identify similar customers you will be able to reach these by making minor changes to your product.

CHAPTER 4

Foundation of the Entrepreneurial Software Innovation Model

In the previous chapter we defined our problem statement in which we propose a new model for bridging the gap between Software Innovation and Software Entrepreneurship. This chapter introduces the foundations for our suggestion toward bridging this gap; the Entrepreneurial Software Innovation (ESI) model. ESI uses concepts of the Software Innovation methodology *Essence* by Aaen [1], and concepts from Lean Startup [14] as well as its extension, ESSSDM [5]. We do, however, think that Essence is too large for our need, so we piggyback on some of the most essential concepts, namely ETVX, the Configuration Table, and the Problem, Vision & Warrant Review, Sections 4.1 to 4.3 respectively. This chapter also includes our definition of a business concept that we include in our model, namely *branching* presented in Section 4.5. In addition to concepts from Essence we also use some of the concepts we used in Kaysen et al. [11], the Business Model Canvas, and an extended notion of the MVP (Section 4.4).

4.1 Configuration Table

Aaen [1] presents a Configuration Table (CT) as the entire reasoning behind a project. A *configuration* is thus representing the current state of the project at the present moment. It describes a *Prospect*, which is one understanding of the problem, and one idea of how to solve the problem. A simple problem may be easy to understand, but complex problems can be difficult to cope with. Thus it may result in different Prospects, one at a time, each representing one understanding of the problem. The Prospect is one part of a configuration, and it is represented along with a challenge, problem, and vision for the solution. This is accompanied with some criteria specifying what is expected of the solution.

The configuration is presented in the CT using four views that each represent a part of the configuration. The views are presented as columns in the table, as seen in Figure F4-1. The *Paradigm* view describes the challenge and problem that we are working to solve, and represents the problem domain and its users. The *Product* view focus on how the product should be built and what it should consist of. This helps

the developers identify relevant features and technologies for the solution. The *Project* view states the idea and purpose of the project, while also presenting the status of the project, as well as the vision covering the entire project. The *Process* view is about how we handle alternative ideas and solutions, and how to assess and evaluate the solution. The assessment and evaluation is used for decision making in future Prospects.

In addition to the columns, the CT contains three rows that each specify a part of the four views. These are shown and described in Figure F4-1.

View	**Paradigm**	**Product**	**Project**	**Process**
Value	*Reflection*	*Transaction*	*Reasoning*	*Appreciation*
Rationale: Why?	The overall *challenge* and the *problem* for the current configuration.	*Key technologies* for solving the problem.	Here the *Vision* and *Warrant* are described. The vision suggests a solution, which is supported by a warrant that argues for solving the problem. The Vision represents the the *Prospect*.	*Rational Review* How the solution solves the problem (defined before and during development). When reviewing the development, we might identify unused potential, problems etc.
Strategy: What?	A list of *key elements* from the problem domain.	*Key components* to be built to solve the current problem, as well as useful, but not necessary, features.	The *justification* for the warrant, why the problem should be solved. • *Backing*: What supports your warrant. • *Qualifier*: What contexts is the Prospect unable to solve the problem. • *Rebuttal*: What makes this Prospect still worth creating.	*Strategic Review* What *expectations* are there to the key components? Are they sufficient to solve the problem?
Tactics: How?	Important *key scenarios* representing the problem.	Required *key features* to solve the problem.	*Key mappings* ensure that key features are mapped to key scenarios, such that all key scenarios are enabled by features.	*Tactics Review* The *expectations* to the features that solve the problem. E.g. they are effective, they solve the problem completely etc.

Figure F4-1: Explanation of the cells within the Configuration Table.

In ESI the CT works as a "tool" for describing the problem being solved (*Paradigm View*) and the product being built (*Product View*). Defining the product is an essential part of establishing a viable business, and to decide how the product should look and which features it should provide, knowledge about the problem domain must be acquired. In the *Paradigm View* we have used the problem to describe the concrete problem our customer experienced during our work with Gymnastbillet, which is the main, but not only problem they experience; the challenge presents what is gained by solving the main problem. Thus we say that solving the customers problem using our solution induces what is described in the challenge. The *Product View* describes the main technologies and features that must be implemented in the components to solve the problem.

As stated in the introduction of this chapter we do not use all of Essence. Thus we exclude the *Value* row in the CT, since the parts it is referring to in Essence are not relevant in the ESI model. In addition to the Configuration Table we also use the

ETVX model as a basis for the ESI model, to ensure our model is usable within both the traditional and the agile paradigms.

4.2 ETVX

Essence presents a general software process model called ETVX that consists of four parts: **E**ntry Criteria, **T**ask, **V**alidation and e**X**it Criteria. This is depicted in Figure F4-2. ETVX can represent any development process since all processes, when broken into small steps, are all sequential steps performed again and again. For example, an iterative development process consists of the same sequence of steps performed multiple times. Representing this using the ETVX model is simply a sequence of ETVX models. The four parts of the ETVX model is related to the four views in Essence as follows.

Entry Criteria is criteria that must be fulfilled before entering the *Task* step. This is related to the *Paradigm* view in Essence, that focus on understanding the problem that must be solved before actually trying to solve it.

Task is the description of what must be done. It is related to the *Product* view, where the focus is entirely on one problem at a time, and on finding the technology best suited to solve it.

Validation is used to assess the quality of the work done in *Task*, and how it matches the *Problem*, *Vision*, and *Warrant*.

Exit Criteria must be fulfilled before finishing, and they are used to make sure the *Vision* matches the *Problem* and the solution.

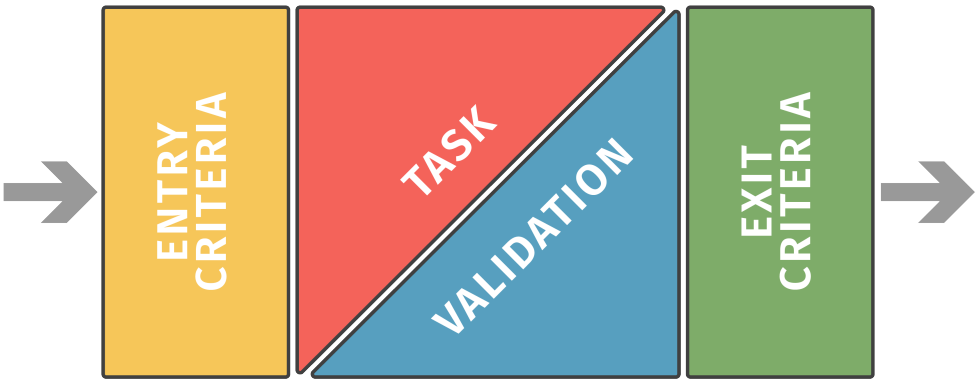


Figure F4-2: The ETVX model.

The *Entry Criteria* can be thought of as a *precondition* that must be fulfilled to enter a specific activity, and the *Exit Criteria* a *postcondition* that must be fulfilled to exit that activity. In between lies the *Task* and *Validation* steps, and as can be seen in Figure F4-3 they are characterized by being *invariant*. The *Invariant* represent the parts of the Configuration Table, see Section 4.1, that does not change during the

activity. Thus the development in the activity can be measured against the invariant to see how it solves the problem. It is essential that this invariant exists, since it is the only thing that is guaranteed to stay unchanged from the beginning to the end of the activity, and thus the only thing that can be measured against to see the progress in relation to solving the problem.

The following section elaborates on the PVW review performed in the validation step of the ETVX model.

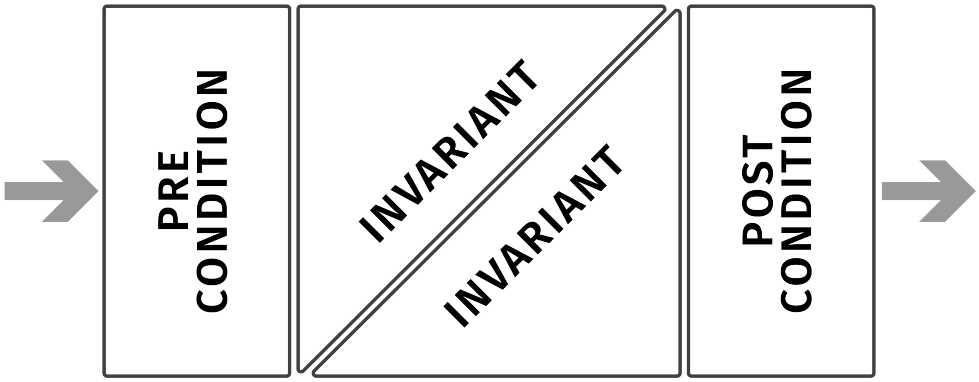


Figure F4-3: The stages describing the principles of the ETVX model.

4.3 Problem, Vision & Warrant Review

In Essence, an extension to the agile development model's Sprint Review called the Problem, Vision & Warrant (PVW) Review is proposed [1]. This extension incorporates the development and findings when evaluating what has been achieved, such that it can be used in future sprints.

The aim of the PVW Review is to ensure that the three elements of the CT, Problem, Vision, and Warrant, still represent the "world" that they represented before the sprint. Meaning that the way the problem was interpreted in previous Sprint Planning is still the same at the Sprint Review. If they are not meaningfully aligned, corrections must be made in order to align the three elements. These corrections may influence the rest of the CT – thus the PVW Review is actually a review of the entire configuration for the Prospect in question. The findings from the iteration may change the understanding of the problem, which can then be changed during the PVW review.

Figure F4-4 presents one iteration of development. The red circle represents developing software using Scrum while illustrating a Scrum sprint and the daily standup meeting. The green circle illustrates the problem that is subject to be solved, the overall vision for the project, and warrant for what is offered and what values are created by the Prospect. The colors matches the colors of the CT (presented in Section 4.1) with the exception of the problem which is presented in the yellow *Paradigm* column.

Even though PVW Review is defined in Essence as an extension to the agile development model, it allows the PVW Review to be used with any development model.

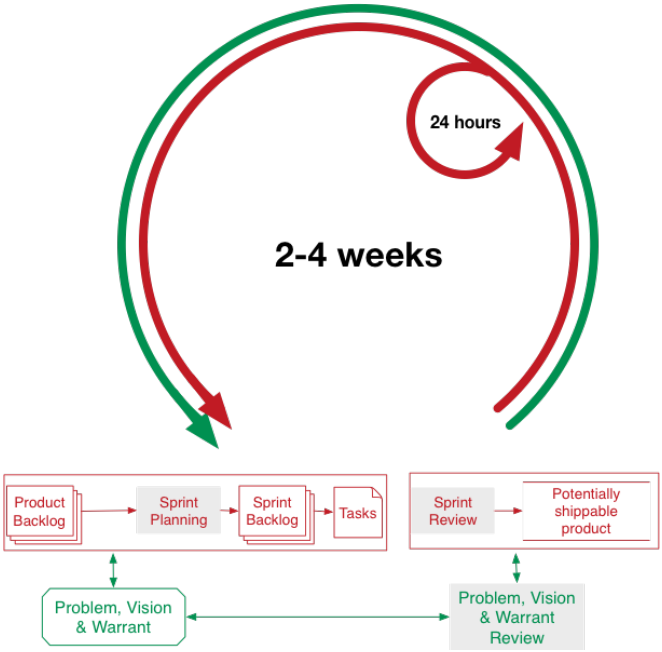


Figure F4-4: Problem, Vision & Warrant Review as illustrated in Aaen [1].

4.4 Expanding Minimum Viable Product

In this section we expand the MVP term defined by Ries (as described in Section 2.1), such that it encompasses terms that we find useful in a development iteration. Ries [14] defines the MVP term in Lean Startup to be the following:

> *The MVP is that version of the product that enables a full turn of the Build-Measure-Learn loop with a minimum amount of effort and the least amount of development time. The minimum viable product lacks many features that may prove essential later on.*

In Section 2.5.1 we outline how we appreciate the idea of a MVP according to the definition by Ries, as outlined above. We believe that being able to define a minimum viable product in conjunction with establishing validated learning through the BML loop is a strong concept, and very advantageous to entrepreneurs in order to reduce wasted resources and quickly set a business into motion. Our expansion thus includes the following items:

1. Adhering to best software design practices should be prioritized from the beginning, such that the product with the least amount of effort required can be adapted according to new business models. This is especially the case when working under high levels of uncertainty, as good design will allow entrepreneurs to quickly perform a pivot to get the business back on track, if its hypothesis proves not to hold.

2. We expand the definition of MVP to support more facets of product development by including low fidelity prototypes and business ideas. This develops a new

notion of MVP to underline the importance of being able to prove a business hypothesis during all stages of product development. This allows entrepreneurs to further ensure that resources are not wasted during the early stages of product development.

One reason to use the MVP for developing a business idea is that at some point you might want the MVP to reach more customers, which means expanding the customer segment. In order to do this, we suggest iterating over a business idea just as we are able to iterate over a product idea when developing a regular MVP as defined by Ries. The section below presents some approaches to expanding the customer segment.

4.5 Branching

The theory in this section is loosely based on a combination of information found in multiple sources [7, 10, 19].

When running a business, there might be several reasons why you might want to expand. It can be due to business plateaus (no increase in revenue, profit, number of new users/customers, net worth etc.), increasing demand, current revenue is not enough to keep the business afloat, etc. It might also be because the company sees potential in another customer segment. The solution to this problem is in many cases about finding new ways or means to reach new customers, thus increasing the revenue and hopefully the profit of doing business. This can be done in many ways, and below are some of the most obvious ways to reach new customers.

Pricing is one way to branch out a business, and it is commonly used by car manufacturers. Makers such as Maserati, Ferrari and Porsche have made "cheap" cars that are interesting to their younger audience who are not wealthy enough to buy the largest cars yet. A similar case exist for the middleclass automakers such as Volkswagen and Toyota, catering to young people with their city cars (ultra small and handy cars fit for urban environments). By doing so they reach out to a much broader customer segment, increasing the probability of acquiring new customers.

Physical Location is a way to expand the business' reach, e.g. if a business deliver commodities and they acquire more and more customers who live further away, it might be profitable to open another business location closer to the new customers. This could be a dependent branch, which is run from the main office, or it could be an independent branch, which is run locally at the new business such that they can make decisions, rules and regulations etc. on their own. Another option could be to open a foreign branch, which is a branch located outside the business' country of origin.

Customer Segment can be changed or extended to reach new customers. This is relevant e.g. if a business only targets male customers but wants to include female customers as well. Changing the customer segment can be achieved in many ways; new channels, re-branding, new products, new commercials, different pricing models etc.

Channels have a significant effect on who the product reaches. E.g. teenagers and younger people tend to use the digital services and media more than older people, which is probably one of the main reasons why traditional newspapers are extending their focus to include digital technologies, or in some cases convert all existing offerings to digital services. This have been seen in many industries, but the most significant are the movie and music industry. They are characterized by aggressive growth of several streaming services that by embracing technology are able to reach a larger uptake than would be otherwise possible by delivering a physical product.

In the case where the product is a software system the above descriptions of how to find new customers does not take into account how this influences the search for new customers.

When working with a well designed software product, following principles such as near-decomposability makes it possible to provide almost the same product to different customers by only changing or adding small amounts of code. We define branching to be the process of expanding the business by making minor modifications to the product to allow it to solve new customers' problems. It is evident that well designed software is superior when it comes to branching, since the effort required to reach a new customer segment is minimal, making the expansion or jump to new customer segments as fast as possible. This builds on our critique of how Ries has defined his notion of an MVP, as discussed in Section 2.5.1.

CHAPTER 5

The ESI Model

As stated in the Problem Statement (Chapter 3), the Entrepreneurial Software Innovation (ESI) model is our suggestion toward bridging the gap between Software Innovation and Software Entrepreneurship, by including concepts from both fields while also focusing on software processes and software design. The foundations of the ESI model presents the concepts that we include from the Software Innovation methodology, Essence, while it also modifies the MVP concept originally defined by Ries. In this chapter we define a concept called Define-Build-Evaluate (DBE), which is inspired by the BML loop defined by Ries, and based on the ETVX model described in Section 4.2. DBE is a central concept in ESI and is used to develop ideas, prototypes, as well as MVPs.

The ESI model is based on the notion of a puzzle, as seen in Figure F5-1, consisting of five unique puzzle pieces. The ESI puzzle is, as DBE, based on ETVX as illustrated by the colors, with the *Product & Business Idea* (PBI) piece being an exception, as can be seen by its color that does not match any color in the ETVX model.

When comparing Figure F5-1 with the figures of ETVX (Figures F4-2 and F4-3), the difference is the pre- and postconditions in ESI. The reason is that in ESI the pre- and postcondition consists of a CT and a BMC. The postcondition serves as a precondition for the next iteration, which is the reason the the different colors of the CT and BMC puzzle pieces.

The ESI puzzle has one strict rule that must be followed:

> *You can only build the puzzle from left to right, laying down a single piece at a time, ensuring orderly progress throughout a puzzle sequence.*

This adds sequence to an otherwise iterative process as an effect of including ETVX. In addition to adding sequence via ETVX, the nature of a puzzle also forces practitioners to lay each puzzle piece in the order intended. However, as later described, sequence is only defined as a way to add operational processes to the puzzle (catering to the critique outlined by Bosch et al. [5]) – the model still follows agile principles, allowing iterative product development for each sequence laid of the puzzle.

The ESI puzzle consists of a number of unique puzzle pieces, that together represent one iteration of the development cycle for the product, project and surrounding business. Section 5.2 describes each puzzle piece as well as the relations between them. The puzzle starts with the PBI piece, which defines an initial CT and BMC for the

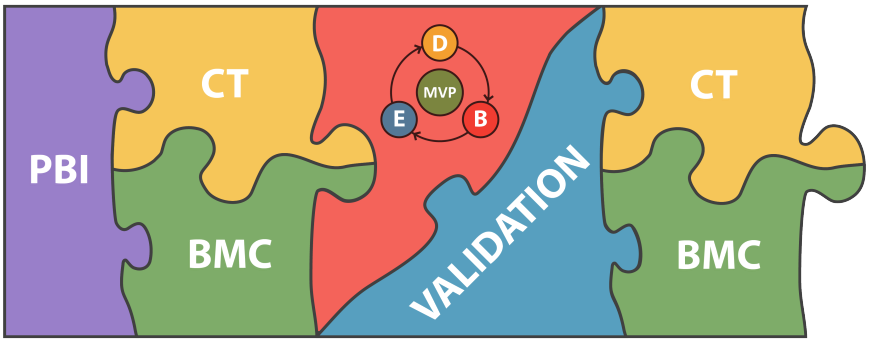


Figure F5-1: A full sequence of the ESI model. Note that the PBI piece only shows one time during the entire puzzle.

overall idea. An iteration (or sequence) starts with a CT and a BMC, and when they are both in place, we can move on to the development of the product. The development is done using Define-Build-Evaluate (DBE) and consists of the *Define* stage, where we choose what to work on in this iteration. This is followed by the *Build* stage, where we develop the product such that it fulfills what we have defined. The last stage is *Evaluation*, where we make sure that we have actually built what we chose to build during the Define stage. The final step of the model is *Validation*, where we validate the MVP that was built, to make sure it matches the problem and challenge from the CT, and that it offers the Value Propositions stated in the associated BMC. If there are any inconsistencies, either in the CT or BMC, or in the correlation between them, it must be handled during this step.

5.1 Define-Build-Evaluate

During our work presented in Kaysen et al. [11], we have practiced the Lean Startup process including ESSSDM by Bosch et al. As outlined in Chapter 2, these models did not fit well for our development of the business and product. We attribute this to the fact that we are building a software-intensive business, and these models only perfunctory treat software and software development.

The input to the DBE loop is a CT and a BMC, that describes the functionality required to solve the problem and realizing the associated BMC. By the definition of Ries, the result of one full iteration of the BML loop is an MVP. In our notion of MVP we replace the BML loop with our DBE loop, defined in this section, such that the output of the DBE loop is an MVP. The MVP, as extended in Section 4.4, either solves the problem defined in the CT (or some parts of it), presents a low fidelity prototype or a new business idea.

By continuing to develop MVPs the team gets closer to answering the challenge, while enabling them to exploit contingencies as soon as possible. This includes, as also described by Ries [14], abandoning features early and including new features based on user feedback in order to improve the product and minimize waste.

Since DBE is based on the principles of the ETVX model, it is fully supported

by both traditional as well as iterative development processes such as Object-Oriented Analysis & Design (OOAD), Scrum, Extreme Programming (XP), Unified Processes (UP), and more as described by Aaen [1]. Figure F5-2 illustrates the iterative workflow of DBE. Its colors are related to the colors in the ETVX model illustrated in Figure F4-2. The *entry criteria* is made in the define stage before the *task* of building the MVP. The *validation* is done during the evaluation stage, resulting in the *post condition* representing the finished MVP.

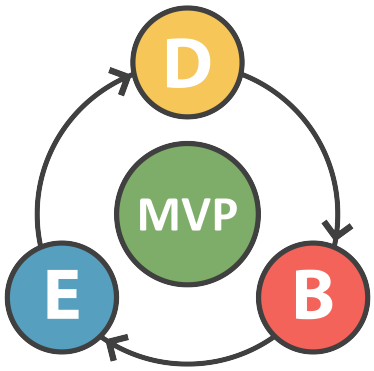


Figure F5-2: The Define-Build-Evaluate loop. The colors of its stages are related to the colors of the ETVX model (Figure F4-2).

Define

This stage is the entry point of the DBE loop and starts by defining a goal for the iteration in order to establish the definition for an MVP. Features defined in the CT are prioritized in collaboration with the customer, such that the MVP to be developed fulfills the most important parts of the problem first. The idea of using the notion of MVPs is that there will always be a version of the product which might only partly answer the challenge, but eventually will answer the whole challenge of the Configuration Table, and thus also facilitate the business model. This can be assumed since you should only be starting DBE when you have a CT and BMC that represent the same project, see Section 5.2 for further description. The incentive for using the MVP is that as development progresses, more experience is obtained, and new ideas and possibilities have appeared based on this experience. Furthermore, MVPs help the creators change direction in case the challenge changes. This is possible since only small amounts of time is spent developing each MVP, effectively ensuring the amount of lost time is minimal, should the challenge suddenly change.

Build

The MVP is built in this stage and should be done using software practices, such as near-decomposability, that allow exploitation of contingencies and incite writing high-quality code (as also described in Section 4.4). In addition, we have found that several practices from XP are valuable in order to minimize waste. An example is pair programming which we recommend using when developing complex components,

as well as when new technologies are introduced. Furthermore, a component should not be declared finished before it is accompanied by unit tests. We found that not having unit tests does not comply with the idea of minimizing waste, since a change in a component that does not have unit tests is likely to introduce bugs, counting as waste since the bugs need to be traced and fixed. Even though test cases need to be corrected when changes occur, we believe this minor overhead is well worth the while, as unit tests help ensuring that the product runs according to its specification. Lastly, we recommend employing Continuous Deployment (Ries [14], Beck and Andres [2]), which means each time a feature is finished, it is immediately deployed to production. This allows the feature to be tested in the production environment very fast, and should a problem in the recently deployed feature be found, the team can perform a rollback to remove the feature. When an error is detected, all other tasks are put on hold until the problem is solved. The feature is first deployed again when the problem has been solved. Section 2.5.3 describes our suggestions toward using best software practices when developing new software products under many uncertainties, as the case with starting a business.

Evaluate

This is the last stage of DBE and its goal is to ensure that the MVP comply with the configuration from the *Define* stage. This includes the customer segments of the BMC, which the MVP must be able to reach. If the MVP does not comply with the configuration after the evaluation, another iteration of the DBE loop must be performed.

The evaluation can be conducted by presenting the MVP to customers and let early adopters try it. It is important to be aware of the feedback, since feedback can be a source of new ideas and features, completely new products and even irritation or annoyance about a feature. When evaluating the MVP on a large scale, analytic and monitoring tools can be used in order to learn from its use. Thus the MVP can be evaluated e.g. by using split tests, making it possible to collect results from two different versions and see if the feature being tested has a positive effect on the group testing it. An advantage from using monitoring tools is that you are able to discover bugs without interacting with users.

5.2 ESI Puzzle Pieces

This section elaborates on each of the puzzle pieces consisting the ESI model. Using the notion of a puzzle makes a clear separation of phases within the model, ensuring that practitioners always are able to identify their progress. The pieces are explained in the order of which they must be laid.

5.2.1 Product & Business Idea

Problem & Business Idea (PBI) is the first piece of the puzzle and is what marks the beginning of a new project, as illustrated in Figure F5-3a. Only one PBI piece exists for each project. At this point neither a CT nor a BMC has been defined. The

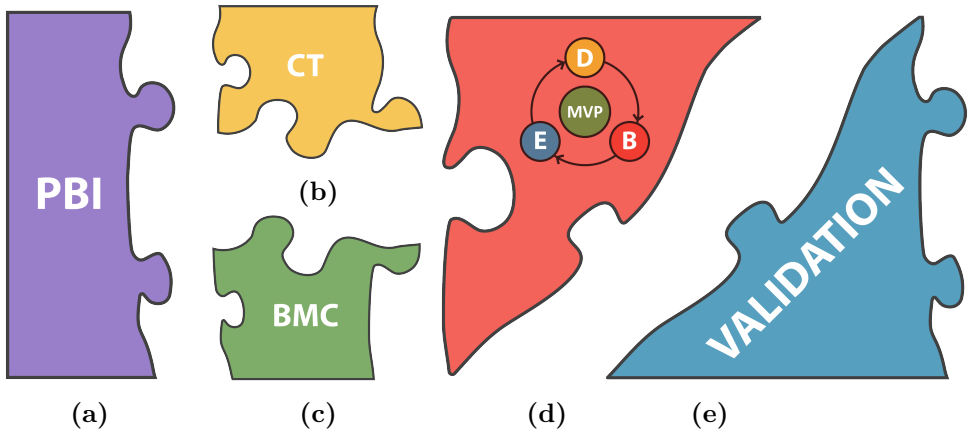


Figure F5-3: Individual puzzle pieces of the Entrepreneurial Software Innovation model.

purpose of this phase is to ensure that a problem worth solving actually exists to enable practitioners to create the initial CT and BMC representing the overall goal for the project. As such, it helps eliminating waste by ensuring that only ideas worth pursuing are initiated. From this point onwards the project is shaped by continuously steering during DBE iterations and validations.

5.2.2 Configuration Table & Business Model Canvas

These two pieces, CT and BMC, are heavily coherent, which is illustrated by each piece having half the connector to the next puzzle piece (DBE). This design means that both pieces must be laid to continue with the next piece – the DBE piece. It should be noted that the CT and BMC pieces can be laid in any order. This step focus not on their specific ordering, but instead that both product and business changes are updated according to recent changes. The Configuration Table is product oriented and focuses on solving a given challenge, while the Business Model Canvas focuses on business aspects. The relationship between the two thus describes the same state of the product, but from two very distinct perspectives.

Configuration Table

This puzzle piece (Figure F5-3b) is a part of the *Entry Criteria* and *Exit Criteria* of the ETVX model described in Section 4.2. This means that in the end of an iteration, the outcome is a revised CT that acts as the Entry Criteria for the next iteration. The Entry Criteria is fulfilled when both the CT and the BMC puzzle piece have been laid.

The CT piece can be laid when it is decided which parts of the CT the current iteration focuses on. Features, components, elements etc. that are postponed to future iterations are written in italics.

Business Model Canvas

This puzzle piece (Figure F5-3c) is a part of the *Entry Criteria* and *Exit Criteria* of the ETVX model described in Section 4.2. This means that in the end of an iteration, the outcome is a revised BMC that acts as the Entry Criteria for the next iteration. The Entry Criteria is fulfilled when both the CT and the BMC puzzle piece have been laid.

The CT piece can be laid when it is decided which parts of the BMC the current iteration focus on. Elements that are postponed to future iterations are written in italics.

Relation Between CT and BMC

As the CT is used to represent the product, the overall ideas, thoughts, and the reasoning behind the product, we use the BMC to represent the business surrounding the product.

In the CT the *Product View* presents Key Technologies, Key Components and Key Features, which together represents the main foundation for the product. In the BMC, the *Value Propositions* present the features that are used as "selling points" when talking to customers, effectively describing the key offerings of the product. *Value Propositions* does not include which technologies are used to create the offerings, relieving the need to always take *Key Technologies* into account. However, for software developers it can help making sure the two definitions, *Product View* and *Value Propositions*, actually describe the same product. Comparing the *Product View* and the *Value Propositions* will identify any discrepancies that must be corrected to maintain the required consistency. It must be ensured that the CT and BMC at all times represent the same state of the product.

5.2.3 DBE

This piece (Figure F5-3d) is solely about developing the MVP, representing the *Task* activity of the ETVX model. When both the CT and BMC pieces have been laid, the problem, or at least when part of the problem to be solved during this iteration has been chosen, the Define phase can begin. The first step in the DBE loop is Define, which is about defining the features that must be developed to solve the problem that have been selected from the CT and from the Value Propositions in the BMC. The Build stage starts immediately after the Define stage has finished. During the Build stage, features decided during the Define stage is implemented in order to build the MVP. Lastly, the Evaluate stage is entered to ensure features that should be implemented during this iteration solves the problem described in the CT, while also corresponding to the Value Propositions defined in the BMC. For further information about the DBE loop, see Section 5.1.

5.2.4 Validation

Validation (Figure F5-3e) follows after the DBE phase, since it is about validating the MVP that has been developed in relation to the CT and BMC. This phase corresponds

Validation Criteria
Does the majority of customers want the problem solved?
Are the majority of customers willing to pay for a solution?
Are the majority of customers willing to pay for the MVP?
Are the majority of customers willing to test the MVP?
Does the majority of customers understand the Value Propositions?
Does the majority of customers accept the pricing model?
Does the business model describe the actual product?
Does the business model correspond with the accompanying CT?

Table T5-1: Suggested *validation criteria* for validating the Business Model Canvas.

to the the purpose of the task activity in the ETVX model from Essence (Section 4.2), ensuring that inconsistencies within the CT are revealed. However, we extend the notion of the validation to also include revealing inconsistencies in the BMC and between the CT and BMC, to be sure that they represent the same product.

Essence proposes validation using PVW Review to validate the CT after each iteration. We use the PVW Review in the validation activity, since the MVP built during DBE may have influenced the CT. The PVW Review focus on maintaining the interrelations in the CT, such that all cells represent the same MVP. The validation phase also focus on reviewing the BMC, such that it represents a business based on the MVP that the CT describes. The reason why the CT and BMC must be validated is that during every phase of the development of both business model and MVP, new knowledge and experiences allow new problems and opportunities to appear. In a design study Schon and Wiggins [17], defines this as seeing-moving-seeing:

> *[...] the designer sees what is 'there' in some representation of a site, draws in relation to it, and sees what has been drawn, thereby informing further designing.*

In order to exploit these contingencies it is important that both the CT and BMC are updated such that they represent the same MVP.

Having suggested that Essence's PVW Review should be used for validation of the CT, we now suggest a set of *validation criteria* for validating the BMC. Inspired by the *exit criteria* from Bosch et al. [5], Table T5-1 presents a list of validation criteria. Some of the criteria are taken from Bosch et al. [5], but we do not define them as exit criteria; we simply use them as questions that can be asked when looking at the MVP and BMC.

The five puzzle pieces defines how to start a project using the ESI model, as well as driving the project forward, while steering through the ocean of new opportunities and decisions that most likely will arise during developments. With the definitions of each piece of the puzzle at hand, the next chapter demonstrate how the ESI model

can be used in practice. This is achieved by first mapping the conducted work on Gymnastbillet into the ESI model, and then expanding it with a new iteration to explore other business opportunities.

CHAPTER 6

Using ESI

This chapter demonstrates how we have used the ESI model, by laying one piece of the puzzle at a time. As we progress through the model, laying the puzzle piece by piece, a figure illustrating a sequence of the ESI model is added and colored according to how we are progressing. Figure F6-1 presents the start state of the ESI model, where no puzzle has been laid.

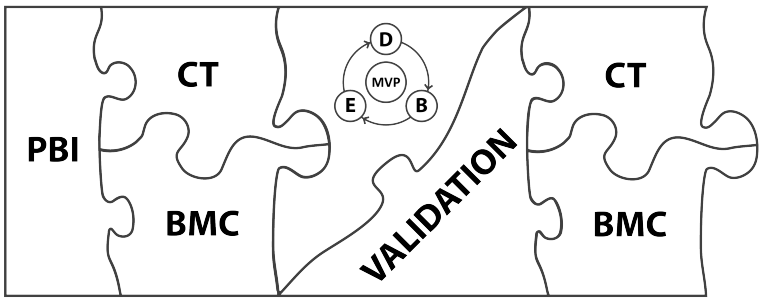


Figure F6-1: Start state of the ESI model with the puzzle with no pieces laid (pieces are colored when they are laid).

In order to demonstrate the use of the ESI model on the case which we started developing in Kaysen et al. [11], we begin by mapping the the development of Gymnastbillet into the ESI model in Section 6.1. Section 6.2 presents a second iteration of the model that investigates the idea of branching the business presented in Iteration #1 (Section 6.1). This is an experiment showcasing how the model facilitates branching in order to reach a larger customer uptake, and thereby also a possible increase in revenue.

6.1 Iteration #1

This section describes the development of Gymnastbillet mapped to the ESI model, from product and business idea to a revised CT and BMC. Thus, a whole sequence of the puzzle, corresponding to the sequence illustrated by Figure F6-1 is laid.

6.1.1 Defining Product and Business Idea

The first step in using the ESI model is to perform the *Product & Business Idea* activity (see Section 5.2.1). The precondition for beginning the DBE phase is to have a CT and BMC that presents the work to be done in the current iteration. To fully understand the problem such that the CT and BMC can be created, a meeting is held in collaboration with DGI to define a set of requirements defining the exact problem to solve. Vision Scenarios [1] are used to explore opportunities and features based on our own ideas and understanding of the problem. This allows us to think of a solution without being biased towards suggestions from DGI. These Vision Scenarios are presented in Figure F6-2.

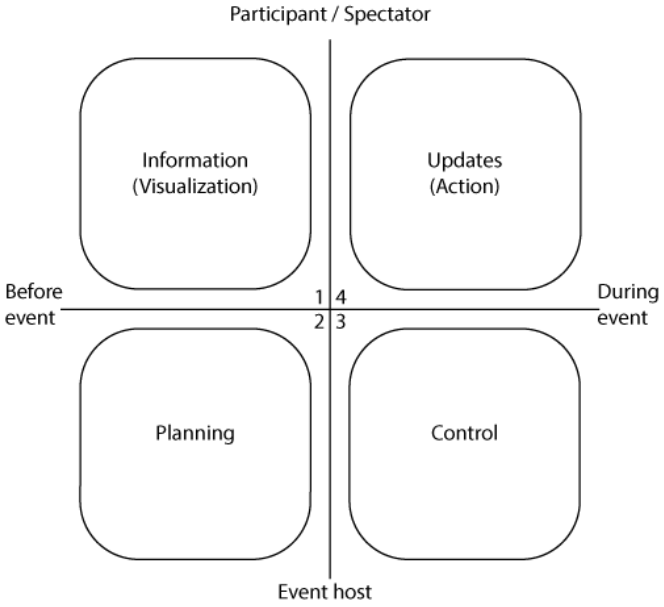


Figure F6-2: Vision Scenarios based on our own ideas and understanding of DGI's problem [11].

The following description of the vision scenarios is taken from Kaysen et al. [11]. The vision scenarios are made using two axes, *Event host* and *Participant / Spectator*, and *Before event* and *During event*. These axes create four quadrants, each representing a general vision scenario. 1) Before an event, both participants and spectators are interested in information about each event they are either attending or might want to go see. Examples of information are dates and addresses for the events, or when and which teams are performing at each event. 2) Before an event, the host would like to be able to plan the event. Planning includes scheduling of activities such as performances during the event and distributing resources to e.g. members of the staff. 3) During an event, the host would like to be in control. This include redistributing staff dynamically or make changes to the schedule, e.g. in case of delays. 4) During an event, participants as well as spectators are interested in updates about the ongoing activities. This for example include notifications in case of delays such that they can plan their day accordingly, and not arrive too early and that participants do not start

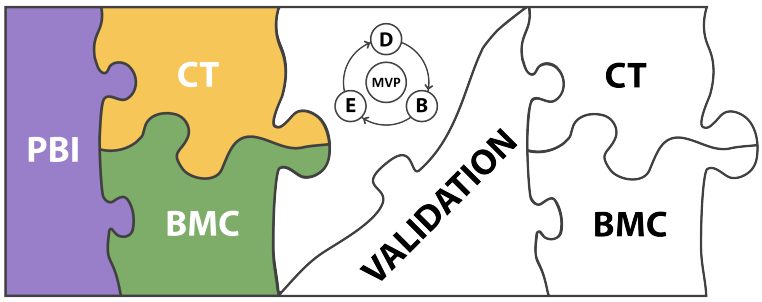


Figure F6-3: The first pieces of the ESI puzzle is laid.

warming up too early.

Using the outcome of the discussions with DGI, we are able to define DGI's problem, and thus also the challenge to solve. This is presented in the Configuration Table in Figure F6-4. Key elements, key components, key scenarios, and key features are all written in italics, as they currently only describe ideas, and not actual features, since development is yet to begin. Based on the same knowledge used to create the CT, a BMC is developed to represent the initial business ideas, see Figure F6-5.

With both the initial CT and BMC constructed, the PBI puzzle piece can be laid, and we can move on to the next step in the puzzle. Following the PBI piece comes the CT and BMC pieces, defining what should be done during developments in the DBE phase. Starting a project, the CT and BMC pieces can be laid immediately after defining the PBI, as it includes an initial definition of both CT and BMC. This also means that the precondition for entering the DBE phase has been met. The precondition states that it must be defined in the CT and BMC what must be developed during the sequence of the puzzle, before entering the development phase (DBE). Figure F6-3 illustrates where in the process we are before continuing to the DBE phase.

View	**Paradigm**	**Product**	**Project**	**Process**
Rationale: Why?	*Challenge* Increase revenue by reducing abuse of free tickets, and statistics for events for R&D for future events. *Problem* Personal free tickets are given to ineligible entrants and a lot of effort goes into distributing content and communication.	*Key Technologies* • QR codes for team identification • Cross platform mobile development • Cloud solution for scalable backend and hosting • *In-app purchases*	*Vision* Digital event management system that supports the organizer before, during and after the event. E.g. by preventing free ticket abuse. *Warrant* Preventing ticket abuse increases revenue from ticket sales. Collecting statistics about entrants will increase future revenue.	*Rationale review:* We can eliminate abuse of free tickets. Thus, we can increase revenue from ticket sales.
Strategy: What?	*Key elements* • *Ticket* • *Organization* • *Association* • *Team* • *Team member* • *Instructor* • *Event* • *Spectator*	*Key components* • *Personalized digital ticket* • *Digital schedule* • *Administration interface* • *Ticket purchase module* • *Notification module: Stay close with entrants using mail/push notifications*	*Justification* • *Backing*: Platform for supporting DGIs needs when hosting gymnastic events. • *Qualifier*: Not every entrant owns or uses a smartphone. • *Rebuttal*: DGI handles corner cases where entrants do not have a smartphone.	*Strategy review* *Expectations*: The key components are necessary and sufficient to implement vision.
Tactics: How?	*Key scenarios* • *Gymnast enrolls for team* • *Team enrolls for event* • *Ticket checking during event* • *Distribution of practical information (schedule, songs, address etc.)* • *Entrant buys ticket(s)* • *There is a change in the schedule of an event*	*Key features* • *Animation on ticket to avoid abuse* • *Ticket checking* • *Approve access to participants* • *See event schedule* • *Send notifications to participants/entrants*	*Key mapping* • An animated personal digital ticket ensures abuse is avoided when the ticket is checked. • Enrolled gymnasts can be approved, through the administration panel. • The administration panel makes the organizer able to attach practical information to an event.	*Tactics review* *Expectations:* The key features reduce the amount of ticket abuses.

Figure F6-4: Initial Configuration Table describing the problem defined by DGI.

Figure F6-5: Initial Business Model Canvas describing the business side of the problem defined by DGI.

6.1.2 Building the First MVP

The process of building the first MVP follows the DBE-loop described in Section 5.1. The first step is to enter the *define* stage, where requirements are defined in collaboration with DGI to know exactly what features the initial MVP needs to consist of. It is therefore ideal to prioritize between features such that only the ones strictly required to solve the challenge defined in the CT are included for each MVP. We do however recognize the potential of adding additional value according to features not specified by the customer. The main features of this MVP consists of the items below.

- Personal digital ticket.
- Distribution of event schedules.

Additionally as requested by DGI, the developed solution includes support for both iPhones and Android smart phones.

With the requirements for the first MVP defined, we are able to move on to the *build* stage of the DBE loop. Since this is the initial MVP of the project, we begin by finding appropriate technologies that can be part of the solution. The mobile applications are supported by a backend developed for the Microsoft Azure Cloud services[1]. The main reasoning behind this choice is to provide automatic scaling, load balancing etc. in order to future-proof the system, and support every possible usage scenario according to the use by DGI's members. This also relieves us from configuring and managing servers, effectively minimizing waste, as such tasks are handled by Microsoft.

The technology choice for developing the mobile applications falls on a cross-platform HTML5 and AngularJS mobile framework, called Ionic[2]. Ionic allow us to develop one application that can support both iOS and Android devices. This is yet another example of how a specific technology can enable the development team to save precious time, in order to get the MVP into the hands of customers as fast as possible. Ionic is an ideal choice for us since we already have experience developing web applications using HTML5 and AngularJS. Another important factor is our focus on developing a high-quality MVP, as this emphasizes how near-decomposability and good software design can speed up development further; having had this focus while developing Juvono has thus allowed some components to be reused for Gymnastbillet.

Entering the *evaluate* stage of the DBE phase, we have spent a day with DGI to present and evaluate the MVP. Part of this evaluation is allowing board members of DGI to try the application for themselves, providing us with essential knowledge about their sentiment toward the solution.

After the demonstration, DGI proposed us to create an easier solution toward how the new digital tickets are managed. This means that another iteration through the DBE phase is required to solve the newly identified problem with the MVP. This is described in the following section.

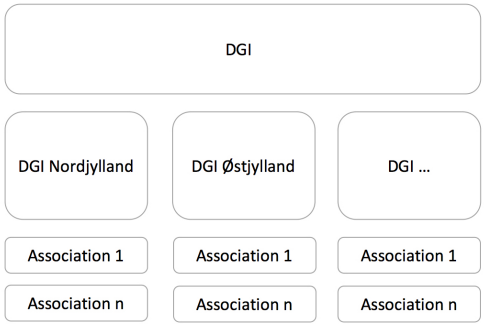
[1] https://azure.microsoft.com/
[2] http://ionicframework.com/

6.1.3 Extending the First MVP

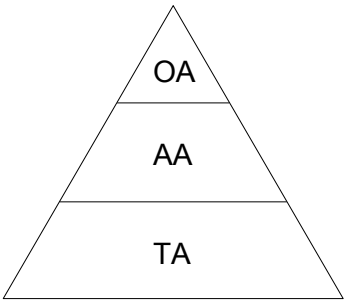
Not having the opportunity to easily manage free tickets for the system requires another iteration of the DBE phase as requested by DGI. We agree that such need indeed exists such that DGI can handle and assign tickets to eligible participants without our intervention. This means that the MVP needs to be redefined in order to prepare for another iteration of the DBE phase. The new requirements for the MVP thus also includes the requirements for the first MVP, as well as an administration system.

It is obvious that the easiest way to develop the administration system is to rely on a platform that builds on the same technologies used to develop the mobile applications. This insight resulted in the administration system being implemented as a HTML5 and AngularJS web application accessible from a browser (as administrative work is usually done from a device that have a browser installed), allowing further reuse of some components already developed for the mobile applications, saving a lot of time and effort in quickly being able to provide a functional MVP. Examples of reused components include modules for handling authentication of users, and for sending and retrieving data between clients and the backend.

To ensure that only users with the necessary responsibility in DGI has access to the administration system, the hierarchical structure of DGI is modeled one-to-one, such that the highest entity is an organization, consisting of one or more associations and each association consisting of one or more teams. Each of these entities can have one or more administrators: organization administrators (OA), association administrators (AA), and team administrators (TA). The hierarchical structure of DGI as well the roles in the system can be seen in Figure F6-6. As these roles are assigned to regular users, one must have either role in order to access the administration system.



(a) The hierarchical structure of DGI. Please note that each association depicted can consist of 1 to n teams.



(b) Hierarchy of the three roles in the system [11].

Figure F6-6: Hierarchy of DGI and roles modeled for users in each part of the hierarchy.

DGI has specified the flow for gymnasts to get a valid ticket to be as follows:

- A TA, typically the instructor of the team the gymnast is enrolled to, distributes a QR code or 6-digit code to the gymnasts on the team.

- Each gymnast of the team then needs to scan the QR code or enter the 6-digit code to request sign-up to the team.
- Then the TAs for the team are responsible for approving only those who are gymnasts on the team, and rejecting everyone else. This step is solely required to avoid ineligible people getting a valid ticket, if they should be able to access the QR code or the 6-digit code. That way the responsibility for avoiding ticket abuse is assigned to the instructor, who is the person closest to the gymnasts and thus the one best suited to accept and decline awaiting participants for the team.

The administration system is also used for importing gymnastics events and teams into the system. This task is usually performed by the OA who typically is a person from DGI with responsibility for the entire gymnastics season. When teams are created in the system, the AA's from each association are responsible for adding instructors to teams in their association. The reason for this delegation is to ensure that the workload is distributed evenly, and because DGI does not know about the individual teams and members of an association; only associations have that information. The persons assigned to enroll teams from an association to events, are automatically assigned the AA role when gymnastics events are imported.

To address the cases where gymnasts are unable to show a ticket, e.g. by forgetting their phone, the phone being discharged etc., we have developed an additional small web application that can search the gymnasts and indicate whether the gymnasts have a valid ticket or not. This is external to the administration system to make it accessible for non-administrators, for example ticket checkers. To access this tool, a ticket checker needs to be handed a 5-digit access code from an administrator.

The *evaluation* stage during this iteration begins with a small-scale test where the system is tested by a gymnastics team, *Vejgaard Unge Piger*. The gymnasts of the team are required to download and then sign-up via the mobile application. Then they enroll to their team such that their instructor (TA) can approve them.

The test revealed a few minor issues. We consider the test a success, as the identified issues were essential to correct before the system being deployed to production. The solution and the test results were presented to the board members of DGI (Nordjylland), who were very happy with the solution. They decided that the MVP fulfills their requirements making the MVP ready to be used for the gymnastics season, from February through April.

They also requested some minor additions to the solution, as described in the following section (Section 6.1.4). This ends the DBE task, allowing us to lay the fourth piece of the puzzle, the DBE piece, as illustrated in Figure F6-7.

6.1.4 Validation of Iteration #1

The evaluation of the DBE activity concludes that DGI are generally happy with the solution, and that they want to use it during their gymnastics season. As such, the solution were used as a test from February through April. Characterizing this as a test is because DGI did not want to commit to the product without knowing about its

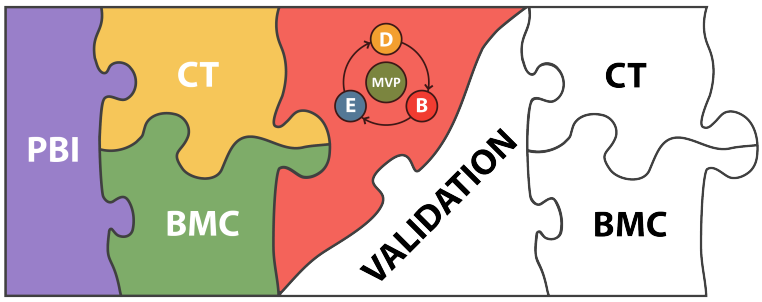


Figure F6-7: Laying the DBE piece marks the completion of the MVP.

effectiveness, and us being able to deliver a product of expected quality. The feedback received from this test period is very positive.

If we review the expectations written in the *Process* column of the CT (Figure F6-4), we can confirm that our expectations of the *Rationale* and *Tactics* rows are fulfilled. These are furthermore confirmed during a meeting held with DGI ultimo April, evaluating the MVP based on this test period. They confirmed that the system solves the problem according to their requirements, while also making it easier for them to spot the now limited free-ticket abuse. We believe this was due to the fact that the amount of physical tickets that had been handed out during this season was significantly smaller than previous seasons.

DGI also partly confirmed our expectations of the *Strategy* cell, although as briefly mentioned in the section above, DGI requested minor changes to the system, namely a staff management feature, such that they can allocate volunteers before an event and reallocate them during the event if needed. This feature has been included in the Configuration Table, and can be seen in Figure F6-9. As such, the vision for next iteration has also been extended.

We are aware that we cannot build a viable business solely on DGI Nordjylland Gymnastics Department. This is already mentioned in Kaysen et al. [11] where we discuss the need for generalizing the product in order to target a larger customer segment. However, another solution is to expand our business within other DGI regions. The fact that we have already created and tested our MVP, with DGI Nordjylland Gymnastics Department, can lead the way for other regions of DGI becoming customers.

During the test period of the MVP, **1,700 users** have registered in system, and a total of **600 teams** were enrolled for the events. Furthermore, the system did not crash or have any downtime during the entire test period. Thus, we can present a system that is stable, supports a large amount of users, and fulfills the requirements specified by DGI Nordjylland. We believe that presenting such results will make it easier to sell the product to more regions in DGI, as well as to other organizations.

At the evaluation meeting with DGI they made it clear that their budget cannot handle paying the amount we imagine the system to cost, effectively putting further development of the system on hold. They have specified that they only want to pay for the system while in use, and thus not a static monthly nor annual fee as proposed by us, forcing a change in the business model. Furthermore they might be interested

in paying for the development of additional features. Leaning on this sentiment, we have to consider how to reach out to more customers, should we want to solely base a business on this product. We facilitate this extension by generalizing our business model to the extent that it supports other customer segments. This is visible in the BMC by changing *DGI* to *Organizers*. The revised BMC supporting these additions can be seen in Figure F6-10.

Now that the MVP, CT and BMC are validated successfully, the last pieces of the puzzle for the first iteration can be laid. Thus, our post condition have been met and we are ready to continue with a new sequence of the puzzle. The state of the ESI model can be seen in Figure F6-8.

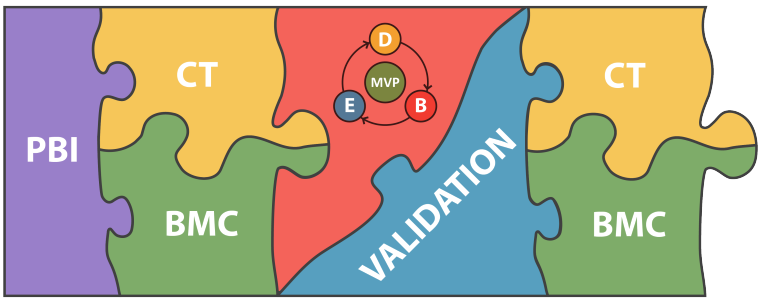


Figure F6-8: The validation and postcondition pieces have been laid.

View	**Paradigm**	**Product**	**Project**	**Process**
Rationale: Why?	*Challenge* Increase revenue by reducing abuse of free tickets, and statistics for events for R&D for future events. *Problem* Personal free tickets are given to ineligible entrants and a lot of effort goes into distributing content and communication.	*Key Technologies* • QR codes for team identification • Cross platform mobile development • Cloud solution for scalable backend and hosting • *In-app purchases*	*Vision* Digital event management system that supports the organizer before, during and after the event. E.g. by preventing free ticket abuse, and ***staff management during the event***. *Warrant* Preventing ticket abuse increases revenue from ticket sales. Collecting statistics about entrants will increase future revenue. Staff management reduces time spend allocating and reallocating staff.	*Rationale review:* We can eliminate abuse of free tickets. Thus, we can increase revenue from ticket sales.
Strategy: What?	*Key elements* • Ticket • Organization • Association • Team • Team member • Instructor • Event • *Spectator* • ***Staff***	*Key components* • Personalized digital ticket • Digital schedule • Administration interface • *Ticket purchase module* • *Notification module: Stay close with entrants using mail/push notifications* • ***Staff management***	*Justification* • *Backing*: Platform for supporting DGIs needs when hosting gymnastic events. • *Qualifier*: Not every entrant owns or uses a smartphone. • *Rebuttal*: DGI handles corner cases where entrants do not have a smartphone.	*Strategy review* *Expectations*: The key components are necessary and sufficient to implement vision.
Tactics: How?	*Key scenarios* • Gymnast enrolls for team • Team enrolls for event • Ticket checking during event • Distribution of practical information (schedule, songs, address etc.) • *Entrant buys ticket(s)* • *There is a change in the schedule of an event* • ***Reallocation of staff during the event***	*Key features* • Animation on ticket to avoid abuse • Ticket checking • Approve access to participants • See event schedule • *Send notifications to participants/entrants* • ***Allocate/Reallocate staff***	*Key mapping* • An animated personal digital ticket ensures abuse is avoided when the ticket is checked. • Enrolled gymnasts can be approved, through the administration panel. • The administration panel makes the organizer able to create an event and attach practical information to an event. ***Also it can manage staff at events.***	*Tactics review* *Expectations:* The key features reduce the amount of ticket abuses.

Figure F6-9: Revised Configuration Table being part of the postcondition for iteration #1.

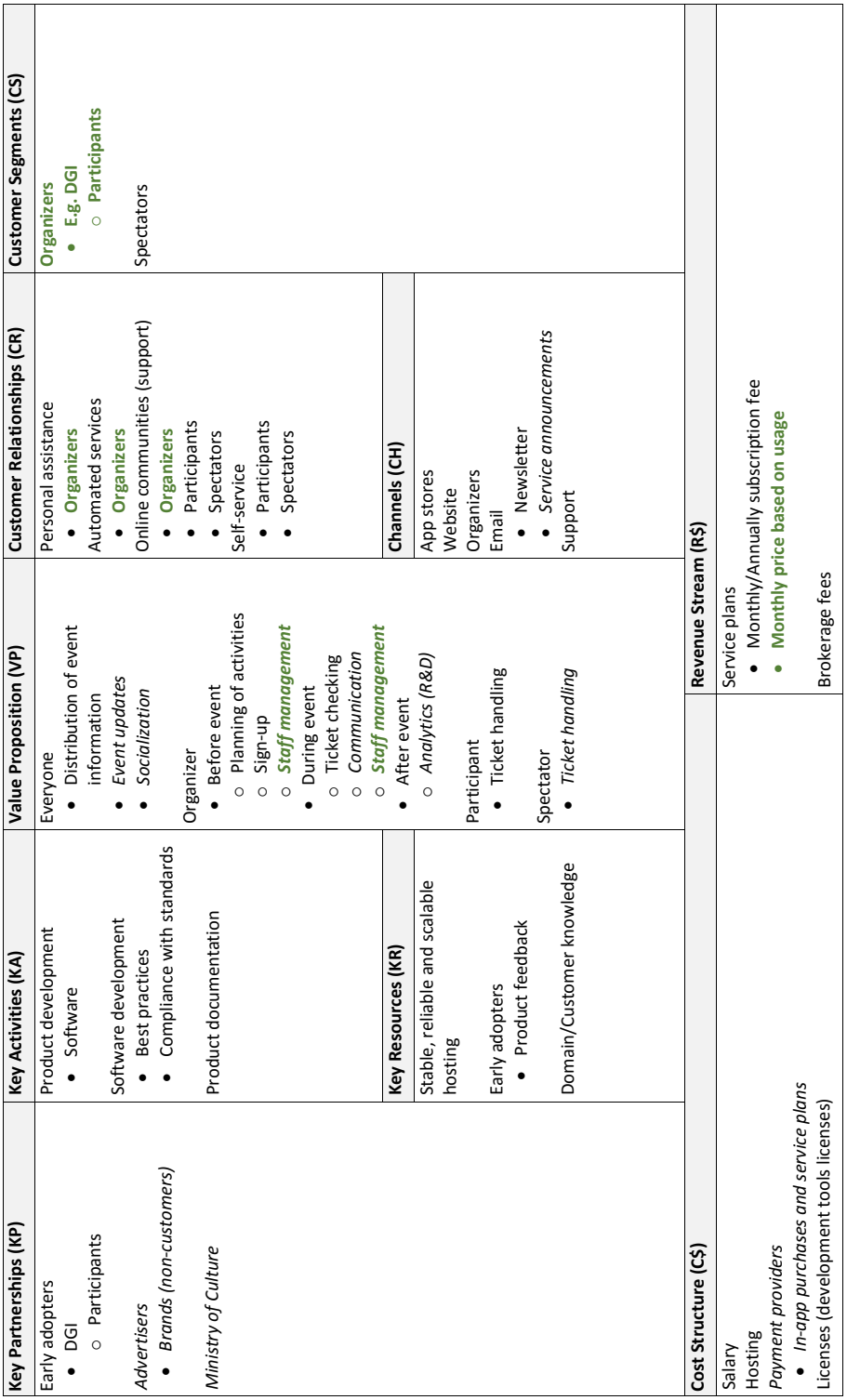


Figure F6-10: Revised Business Model Canvas being part of the postcondition for iteration #1.

6.2 Iteration #2

The first sequence of the puzzle finished with a new Configuration Table and Business Model Canvas that presents the changes from the beginning to the end of the sequence, as seen in Figures F6-9 and F6-10. The generalization of the CT and BMC caters to several customer segments instead of DGI only, making it more attractive to look into branching following this expansion. Branching is about expanding the business, which in our case is done by looking at how to reach new other customers with an existing product, and thereby hopefully be able to create new revenue streams.

This iteration is about exploring the possibility of branching the business, since we cannot charge DGI Nordjylland enough to keep the business afloat. The CT and BMC mentioned above is thus not used for further development of the product, but instead, to examine how the business can be branched and how it will affect the current business and product.

This means that the DBE phase in this sequence focus on expanding our business to reach other customer segments, rather than how to extend the existing product with new features.

Section 6.2.1 examines different options for branching using the theory presented in Section 4.5. The result is validated in Section 6.2.2, where we present a CT and BMC that can be subsequently used during the next sequence based on a branched business.

6.2.1 Branching our Business

To keep the business running in order to provide developers with a monthly salary, we need many small customers or a few large ones. One way to find new customers are to look within the organization of DGI, as they have many departments located around the country, each focusing on a specific branch of sports such as golf or handball. This means we can target all gymnastics departments within DGI, and without much effort extend it to also include other departments. As we already know how to behave under the auspices of DGI, we believe that we with minimal effort will be able to convince key persons responsible for other sports, to use our product. We back up this claim with the fact that a stable and reliable system was delivered to the gymnastics department (as presented in Section 6.1.4). Thus, the potential revenue generated from the whole of DGI, will be much higher when not only considering DGI Nordjylland Gymnastics Department. We estimate it to be enough to run the business solely on DGI, should the entire organization decide to use our product. This is further discussed in Section 8.3.

Another approach to finding new customers is to branch out, as described in Section 4.5. In our case, branching to one or more pricing models might help attracting new customers. For example, by offering a cheaper version of the product, perhaps with optional extra features, would make the product more attractive to people arranging some kind of event without being member of an organization. One possibility could for example be to provide a three tier model, starting with a *basic* version that only provides what is absolute necessary to manage an event while offering other features as an add-on. The basic version can be followed by a *standard* version providing all the main features that are relevant to an event, again with the possibility of adding extra features. At last, if the *luxury* package is chosen, the customer get access to

every feature. Using a pricing model to branch out offers a product that is affordable to different kinds of customers; those who only organize few events as well as those who organize several events per year. This is similar to the examples given regarding pricing in Section 4.5.

The current customer segment is based on DGI being our only customer. This focus can change to include individuals that create private events, or to include organizations similar to DGI that can use the product as is, or with minor changes. Such possible organization is Dansk Boldspil Union (DBU). We imagine our system without much effort can be used to track the results in football tournaments and to provide spectators with information about when and where each match is played. To further maximize revenue it is also viable to consider other similar organizations such as Danmarks Idrætsforbund (DIF), Dansk Golf Union (DGU), Dansk Bordtennis Union (DBTU).

Currently, DGI do not know anything about the members part of their associations. It is therefore not possible to improve their events using knowledge about gymnasts such as targeting members with user-specific advertisement. Since DBU has the same organizational structure as DGI – an umbrella organization consisting of many associations around the country, we believe that DBU has the same issue. Thus we believe our system can be used to collect information about their associations' members, as well as handle their events.

In order to finish the evaluation of this DBE iteration, we have to contact the organizations in question to find out whether our product is interesting to them. Due to the deadline for the project, we do not have time to start this process before handing in the report. As such, we continue to work with branching, based on the assumption that some of the organizations are willing to commit to the product. To evaluate whether branching is profitable or not, organizations must commit to its use in order for us to calculate the expenses compared to the potential revenue. Section 6.2.2 describes the CT and BMC as it would have looked if the organizations mentioned above were committed to use the product, allowing us to branch our business and thus reach new customer segments.

Before our system is ready to be used in a branching context, some development effort needs to be done as some parts are very specific to DGI. This modification is described in the following section.

Extending our Business Model

During the development of Gymnastbillet, we have been using DGI as an early adopter. Because of this relationship, and that the collaboration between us and DGI has been established on the terms that we retain ownership of Gymnastbillet, DGI are not required to pay for current developments. As such, we have proposed a monthly subscription fee for using the system. However, should they be needing any additional features, we have agreed upon an hourly rate covering further development and support. The generalized business model is presented in Validation of Iteration #1 (Section 6.1.4).

Since we cannot run a viable business by having DGI Nordjylland's Gymnastics Department as our only customer, we need to extend our business model to include other parties as well.

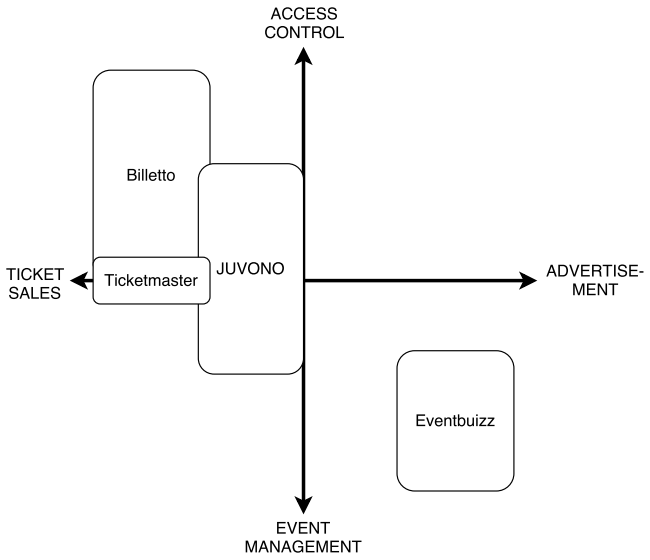


Figure F6-11: Illustration of the *event* market space, placing our extended business model among competitors.

To make our system available to other potential customers, we update our Customer Segment in the BMC by replacing DGI with a more generalized term: *Organizer* in order to provide the Value Propositions to all customers, not just DGI. Furthermore, since our current version of the system only handles tickets, we enter a market with a lot of competition. Competitors such as Ticketmaster[3] and Billetto[4] who are specialized in selling and handling tickets for events, will make it almost impossible to run a business, unless we are able to provide functionality and features that they do not provide. In order to make our product unique, the goal is to extend our focus from only handling tickets to support events from start to finish. Supporting the event itself includes distribution of information such as schedules and other relevant content, planning of each event in advance, retaining users by continuous communication and information using push notifications or emails, and generating statistics for each event to improve future events. We have researched the market and found that other companies also focus on planning; one such example is Eventbuizz[5], but their focus is merely on conferences. Figure F6-11 illustrates the placement of our product in the market space, comparing the business model extension against the competitors mentioned above.

The extension of the business model represents the desire to reach new customers with the existing product. The new BMC is shown in Figure F6-12, presenting the updated Customer Segment. The following section validates the solution and presents the corresponding Configuration Table.

This concludes our DBE activity and thus we are able lay the piece for this as shown in Figure F6-13

[3]http://www.ticketmaster.dk/
[4]https://billetto.dk/
[5]https://www.eventbuizz.com/

Key Partnerships (KP)	Key Activities (KA)	Value Proposition (VP)	Customer Relationships (CR)	Customer Segments (CS)
Early adopters • DGI ○ Participants *Advertisers* • *Brands (non-customers)* *Ministry of Culture*	Product development • Software Software development • Best practices • Compliance with standards Product documentation	Everyone • *Distribution of event information* • *Event updates* • *Socialization* Organizer • Create Event • Before event ○ *Planning of activities* ○ *Sign-up* ○ *Staff management* • During event ○ *Ticket verification* ○ *Communication* ○ *Staff management* • After event ○ *Analytics (R&D)* Participant • *Ticket handling* Spectator • *Ticket handling*	Personal assistance • Organizers Automated services • Organizers Online communities (support) • Organizers • Participants • Spectators Self-service • Participants • Spectators	Organizers • DGI ○ Participants • **DBU** • **DBTU** • **DGU** • **DIF** • **Etc.** Spectators
	Key Resources (KR)		**Channels (CH)**	
	Stable, reliable and scalable hosting Early adopters • Product feedback Domain/Customer knowledge		App stores Website Organizers Email • Newsletter • *Service announcements* Support	
Cost Structure (C$)			**Revenue Stream (R$)**	
Salary Hosting *Payment providers* • *In-app purchases and service plans* Licenses (development tools licenses)			Service plans • Monthly/Annually subscription fee • Monthly price based on usage Brokerage fees	

Figure F6-12: Revised and generalized Business Model Canvas.

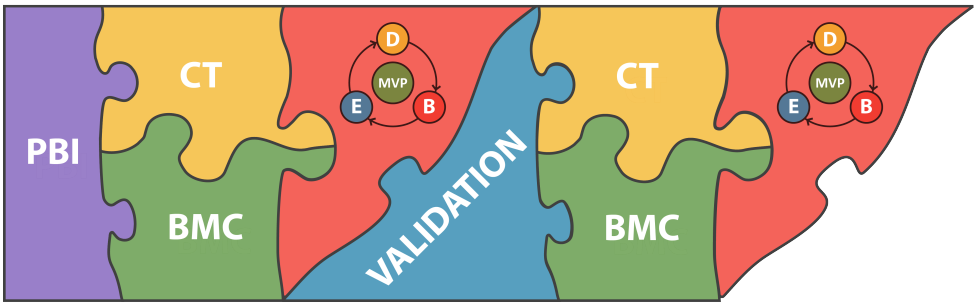


Figure F6-13: Conclusion of the DBE activity in iteration #2.

6.2.2 Validation of Branching

Branching, as described in the previous section leads to a revision of the CT, such that it supports the pursued branching opportunities and matches the business described in Figure F6-12.

In the end of Validation of Iteration #1 (Section 6.1.4) we presented the CT for the MVP that solves the problem with free ticket abuse, according to the specification provided by DGI. They recognize the potential of the product; for example by allowing distribution of content, gathering of information about their members and spectators, as well as receiving statistics e.g. which teams has the most spectators. DGI Nordjylland Gymnastics was not dismissive for the product to be used within other departments in DGI, making us believe that many of the features are also usable to other organizations. This, combined with the generalized BMC, allows us to also change the configuration of the upcoming Prospect. The effect reorders priorities such that problem of ticket abuse will not remain the main focus, but rather become a feature of a larger product. This generalized focus of the configuration makes a lot of changes to the understanding of the problem, namely the *Rationale* row.

The representation of an organization changes as an effect of the generalization. The current representation is based on the structure of an umbrella organization such as DGI, but in order to enable other organizations to use the product, this definition is changed to *group based*. Distinguishing between *organizations* and *group based* allows us to retain support for umbrella organizations by representing departments as groups (such as golf or soccer), while also supporting customers that do not consist of such hierarchical structure.

After this generalization of the Configuration Table, it now matches the business described in the BMC (Figure F6-12) effectively satisfying the postcondition of the iteration. If any of our "assumed" customers choose to commit to this product, the next iteration will be about realizing the generalized product described in this CT (Figure F6-14). This means the current iteration is complete and we are able to lay the final pieces, as shown in Figure F6-15.

View	**Paradigm**	**Product**	**Project**	**Process**
Rationale: Why?	*Challenge* **Sell tickets for events. Manage staff and schedule during event.** Generate statistics for events for R&D for future events. *Problem* **Event organizers does not know much about their entrants and a lot of effort goes into communication and distribution of content.**	*Key Technologies* • QR codes for team identification • Cross platform mobile development • Cloud solution for scalable backend and hosting • *In-app purchases*	*Vision* **Digital event management system that supports the organizer before, during and after the event. E.g. by handling ticket sales, staff and retaining users by notifications and information.** *Warrant* **Selling digital tickets through an app, which at the same time serves as an information and notification hub increases revenue and retains users. Collecting statistics about entrants will increase future revenue.**	*Rationale review:* **We can ease the job of managing events by delivering supporting features in all stages of hosting an event, such as in-app ticket sales.** **Thus, we can save costs of hosting events and increase revenue from ticket sales.**
Strategy: What?	*Key elements* • Ticket • Event • ~~**Organization**~~ • ~~**Association**~~ • ~~**Team**~~ • **Participant** • **Admin** • ***Group (E.g. association or team)*** • *Spectator* • *Staff*	*Key components* • Personalized digital ticket • Digital schedule • Administration interface • *Ticket purchase module* • *Notification module: Stay close with entrants using mail/push notifications* • *Staff management*	*Justification* • *Backing*: **Platform supporting organizers by solving the challenge.** • *Qualifier*: Not every entrant owns or uses a smartphone. • *Rebuttal*: **Entrants not owning a smartphone will be able to get content as print outs.**	*Strategy review* *Expectations*: The key components are necessary and sufficient to implement vision.
Tactics: How?	*Key scenarios* • **Participant enrolls for group** • **Group enrolls for event** • Ticket checking during event • Distribution of practical information (schedule, songs, address etc.) • **Admin creates event** • *Spectator buys ticket(s)* • *There is a change in the schedule of an event* • *Reallocation of staff during the event*	*Key features* • Animation on ticket to avoid abuse • Ticket checking • Approve access to participants • Provide tickets for group • See event schedule • **Create new event** • *Send notifications to participants/entrants* • *Summon/Move staff*	*Key mapping* • An animated personal digital ticket ensures abuse is avoided when entering an event. • Enrolled participants can be approved, through the administration panel. • The administration panel makes the organizer able to attach practical information to an event. Also it can manage staff at events. • **The administration panel makes it possible to create new events.**	*Tactics review* *Expectations:* **The Key Features provide functionality that enables the Key Scenarios to be supported by the system.**

Figure F6-14: Revised and generalized Configuration Table.

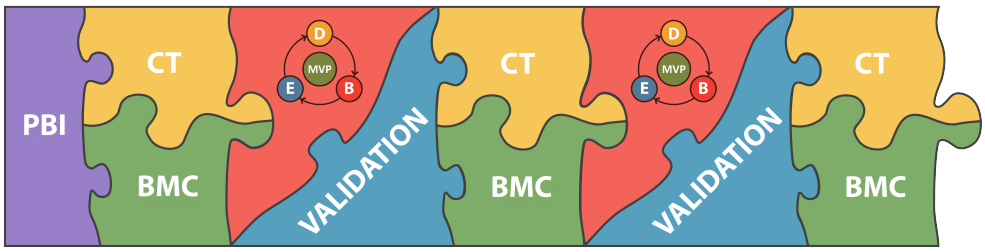


Figure F6-15: Conclusion of iteration #2.

CHAPTER 7

Conclusion

In Chapter 2 we outlined our experiences of using the entrepreneurial process, Lean Startup. Based on our experiences we identified a gap between Software Innovation and Software Entrepreneurship, as described the Problem Statement (Chapter 3). In order to bridge this gap, we have proposed the *Entrepreneurial Software Innovation* model (Chapter 5) which proposes a set of strictly separate activities that allow software entrepreneurs to be both innovative about their business as well as their software product. ESI helps the entrepreneur to develop innovative software products by following concepts of the methodology Essence, while also developing the business and minimizing development time by using concepts from the Lean Startup and Business Model Canvas. However, these concepts do not bridge the gap alone; they simply establish two pillars within ESI, namely Software Innovation and Software Entrepreneurship that are bridged by good software design using established software processes and best practices in software development. In Section 2.5.3 we present the software practices that we find useful in our previous work. Following these principles when developing software in ESI, we found that they allow us to *Branch* our business without making significant changes to the product.

We found that by having a modular (near-decomposable) software product, unit tests and a high quality of code, we were able to expand our customer segment by only making minor changes to the product. This result can be seen by comparing the the post conditions of Iteration #1 (Section 6.1) and Iteration #2 (Section 6.2).

During iteration #2 we expand our customer segment by branching to reach other customers morphological to DGI. When looking at the BMC of the two post conditions, it is clear that there are no great changes except for the list of organizers that has been expanded by similar organizers to DGI. However, when looking at the CTs, the results are rather interesting. As most changes occur in the rationale of the Prospect, this is an effect of the generalization in order to reach similar customers. As such, the vision is no longer specific to DGI, but instead the more generic term *organizers*. The change to the vision causes most of the rationale for the Prospect to change, but the strategy indicating what to build and the tactics of how to do so, does not change significantly.

In our Chapter 3 we define the questions we strive to answer during this thesis:

> *How is it possible to bridge the gap between Software Innovation and Software Entrepreneurship, allowing entrepreneurs to focus on developing a software-intensive product while being able to accommodate changes to the business model and thus support these with minimal effort?*

We believe that *Entrepreneurial Software Innovation* answers this question, since we have succeeded in developing a business by following both Software Innovation and Software Entrepreneurship methodologies. Thus we are able to bridge these methodologies by using software processes and best software design practices, see Section 2.5.3. By following ESI we are able to focus on product development while also focusing on building a business surrounding this product.

CHAPTER 8

Discussion

This section discuss the findings of this thesis. Section 8.1 compares ESI to the mindsets defined by Sarasvathy in order to see it in relation to other established entrepreneurship theories. In Section 8.2 we consider other representations of business models that might prove useful as an alternative to the Business Model Canvas. Section 8.3 provides a reflection as to how the business established with *Gymnastbillet* can be made into a viable business, looking at other possible customer segments as well as the financial aspects involved in doing so.

8.1 The Effect of Good Software Processes

Reflecting on our process and development reveals an interesting finding. Sarasvathy defines two mindsets, Causation and Effectuation [15]. Effectuation is means-driven as it takes origin in who you are, what you know, who you know, and never investing more in the project than you can afford to loose. Causation on the other hand is ideal when you see a great market potential and want to reach mass-market adoption as fast as possible – possibly seeking funding in order to further speed up the process.

By reflecting on our process and development, it is clear that we have followed the *effectual* mindset. We have clearly been means-driven, leaving out any external resources and knowledge. We have focused on controllability as the product and business are developed one MVP at the time. We have been good at tending to the bird in our hands, being able to adapt and learn from Juvono such that Gymnastbillet could be established without extensive development effort. By using the insight obtained from using the Entrepreneurial Software Innovation model, it is possible to further branch out the business to support other customer cases. And as stated in the Conclusion (Chapter 7), it might prove feasible to hunt in the bushes to find additional birds to accompany the one already in our hands.

We thus believe that the Entrepreneurial Software Innovation model is capable of practicing Effectuation, as our process to reach current state includes following the guidelines proposed by Sarasvathy in [15] – Sarasvathy does not display tools for practicing Effectuation. A reason to why we have been able to practice the effectual mindset during software development can be attributed to the fact that we are working with software, an industry characterized by being very means-driven for a software developer, without requiring extensive funding when developing a product. Furthermore, we have knowledge of software processes and practices that support us when branching

and in turn practice Effectuation.

8.2 Reflections on Business Models for ESI

The Entrepreneurial Software Innovation model presented in Chapter 5 is based on our experience working with an entrepreneurial project following the Lean Startup Methodology and ESSSDM. We utilize the CT taken from Essence to present the entire reasoning behind the project and the BMC to represent the business surrounding the product. During our research we have not been able to find any alternative to using the CT. The case is however different when considering alternatives to the BMC, which we present below.

Who-What-How-Why

The *Who-What-How-Why* model [8] presents a simple description of the business by focusing on *Who* the customer is, *What* is offered in the product (Value Propositions), *How* describes how the "What" can be produced and *Why* presents how and why revenue and value is generated in the business. The model, also known as the "Magic Triangle", can be seen in Figure F8-1. The *Who-What* part of the model both describes things external to the business, and *How-Why* what is internal to the business.

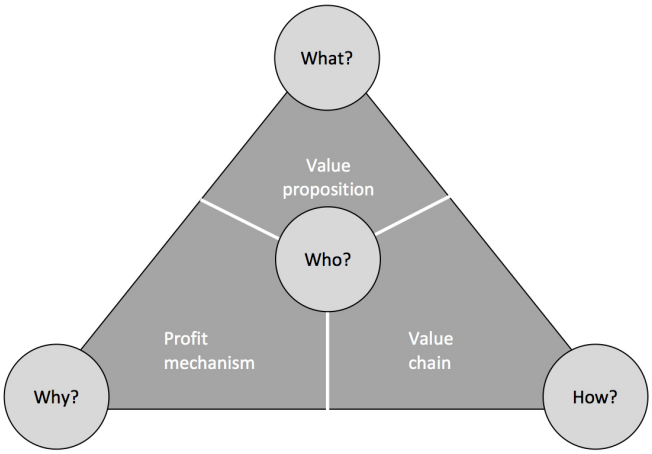


Figure F8-1: The Who-What-How-Why model.

This model can substitute the Business Model Canvas in Entrepreneurial Software Innovation since they both present the same information. Though BMC has a more clear separation of the information due to its nine building blocks opposed to the four of Who-What-How-Why. Thus we find the two models equivalent and we cannot find any reason to use one over the other.

Lean Canvas

Another alternative to the BMC is the *Lean Canvas* (LC) presented by Maurya [12]. It is initially based on the BMC, but with four of the nine building blocks changed, as

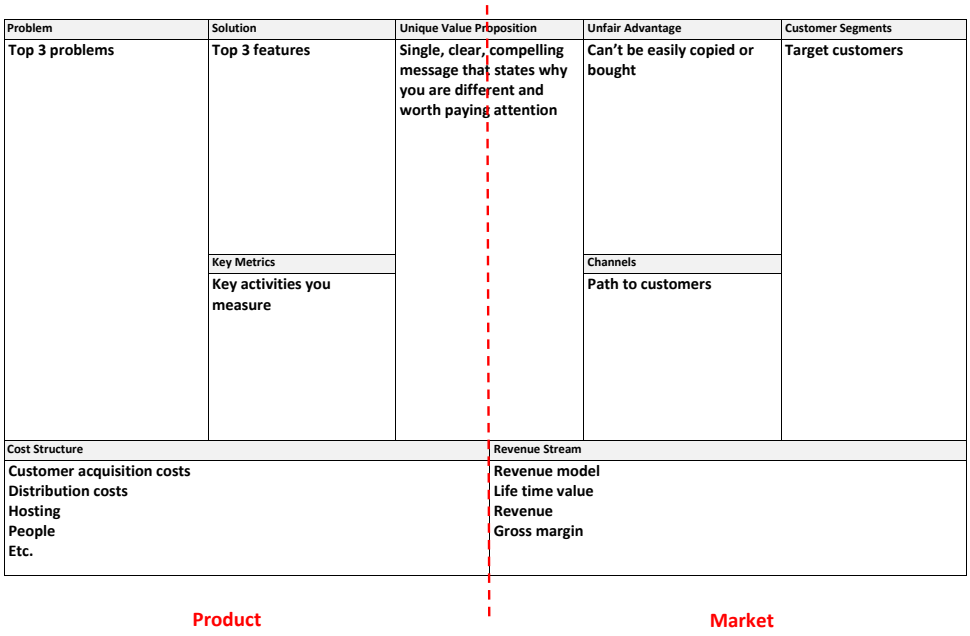


Figure F8-2: The Lean Canvas model.

illustrated in Figure F8-2.

In LC, *Customer Relationship* is replaced by *Unfair Advantage* which is about finding an advantage that differentiates us from potential competition and copy-cats. Maurya argues that every product should be developed with a customer relationship to identify the Channels and Customer Segments that are interesting to the business. Thus the Customer Relationship becomes unnecessary. *Key Resources* is replaced by *Key Metrics* which is a number of activities that are measured. During the project there are a number of activities that are interesting to measure, and each measure is helping steer the project in the right direction to avoid spending valuable time on useless features. This refers to the actionable metrics and minimizing waste presented by Ries. Both Key Activities and Key Resources are argued to be more important to outsiders looking at the business model to understand the business, than to help the entrepreneurs. *Solution* has taken the place instead of *Key Activities*. It is used to describe the top three features in the solution. These features solve the *Problem*, which is the last new element, and replaces *Key Partnerships* in the BMC. This is due to the argument that the majority of new projects do not fall within the category where partners are vital in the startup phase.

Section 5.2.2 describes the connection between the CT and BMC. When using the BMC we find the product to be the connection between the two, but using the LC instead of the BMC actually provides more possible connections to be drawn between the CT and the business model.

We still believe the product is an important connection between the CT and the business model, since software is the central part of both the solution to the problem,

and the business surrounding the product. An obvious connection between the CT and Lean Canvas is the *Problem* since it is described in both of them. The *Solution* in the LC describes the top three features that helps solving the problem, corresponding to the *Key Features* in the CT. These features must make it possible to perform what is described in the Key Scenarios. For us this makes a clear connection between the Solution in LC and Key Features in the CT. The last part of the LC that has a clear connection to the CT is the Key Metrics, which is used to describe the essential parts to measure during development of the product. This corresponds to the Review Expectations in the Process column, found in the Configuration Table.

In further research within the topic of bridging the gap between Software Innovation and Software Entrepreneurship, we find the relation between the Configuration Table and the Lean Canvas essential, because of the similarities and connections between the two. Thus, if we were to continue working on this subject, it would be considered whether to continue to use the Business Model Canvas or switch to the Lean Canvas. Another way to go, could be to combine the Configuration Table and the Lean Canvas into one model that embraces both aspects, such that there is no relation between the models that must be maintained.

8.3 Looking Into the Finances

As mentioned in Section 6.1.4 we know that we cannot build a business solely relying on DGI Nordjylland Gymnastics Department being our only customer. It is therefore natural to consider the opportunity of addressing other sport branches within DGI. To illustrate the potential of doing so with regards to revenue, we look at how much it would cost per member if we should be able to get a salary equal to the SU[1] we recieve while studying at the university. In 2016 the SU is 5,941 Danish Kroner per month, but to make it easier to calculate we round this number up to 6,000, resulting in a total monthly salary of 18,000 Danish Kroner covering three developers. In a year that equals 216,000 Danish Kroner combined.

The gymnastics department in DGI Nordjylland had in 2015 a total of 27,938 gymnasts in their associations. If we were to run the business solely on the revenue from this department, then they would have to pay 7.73 Danish Kroner annually per member. For the naked eye this might not seem unreasonable, but for an association where everything is paid primarily with profits generated from membership fees, it will not be acceptable to raise the membership fee every time a new product is added. Therefore we have to find more users for the system in order to be able to pay ourselves a "decent" monthly salary.

The first step in expanding the use of the system within DGI could be to get more, ideally all, associations with gymnastics in on the deal, effectively providing us with up to 295,632 gymnasts (users) resulting in a cost of 0.73 Danish Kroner per member. If the utopian scenario of having the entire DGI with its 1,524,083 members as users of the system came true, then the same price, 0.73 Danish Kroner per user, would yield a combined revenue in excess of 1.1 million Danish Kroner. The distribution of members

[1] http://www.su.dk/

in DGI is shown in Figure F8-3.

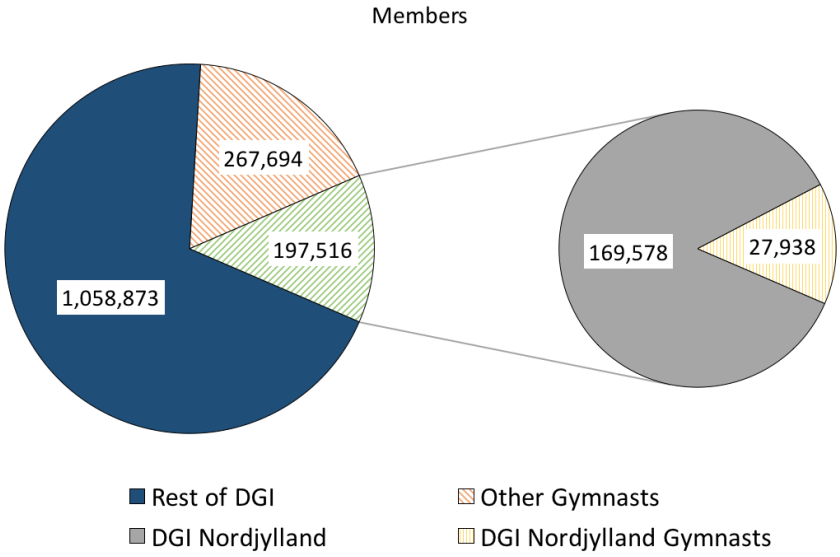


Figure F8-3: Distribution of members in DGI.

Thus our goal of running a business based on the product initially developed for DGI is certainly possible, even if relying solely on DGI, assumed they are willing to pay for and use the system for their entire organization. But our market does not end with DGI. As briefly mentioned in Section 6.2.1, there is a huge potential if we can reach out to several other major organizations in Denmark such as DIF, DBU, DGU, DBTU etc. Their total number of members are shown in Table T8-1.

Potential customer organizations	
Danmarks Idrætsforbund	1,908,867[2]
Dansk Boldspil Union	335,459[3]
Dansk Golf Union	150,916[4]
Dansk Bordtennis Union	8,412[5]

Table T8-1: Number of members in potential customer organizations in 2015.

[2] http://www.dif.dk/da/om_dif/medlemstal
[3] http://www.dbu.dk/oevrigt_indhold/Om_DBU/DBUs%20historie/medlemstal
[4] http://www.danskgolfunion.dk/om-dgu/golfsporten-i-tal
[5] http://www.dbtu.dk/om-dbtu

Bibliography

[1] Ivan Aaen. Essence - Pragmatic Software Innovation. Unpublished book draft. Department of Computer Science, Aalborg University, Aalborg, 2016.

[2] Kent Beck and Cynthia Andres. *Extreme Programming Explained: Embrace Change (2Nd Edition)*. Addison-Wesley Professional, 2004. ISBN 0321278658.

[3] Thirumalesh Bhat and Nachiappan Nagappan. Evaluating the efficacy of test-driven development: Industrial case studies. In *Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering*, ISESE '06, pages 356–363, New York, NY, USA, 2006. ACM. ISBN 1-59593-218-6. doi: 10.1145/1159733.1159787. URL `http://doi.acm.org/10.1145/1159733.1159787`.

[4] Steve Blank. About steve. `https://steveblank.com/about/`. Visited 22/05/2016.

[5] Jan Bosch, Helena Holmström Olsson, Jens Björk, and Jens Ljungblad. The early stage software startup development model: A framework for operationalizing lean principles in software startups. 2013.

[6] New Venture Creation. Welcome to the new venture creation semester. `http://www.nvc.aau.dk/`. New Venture Creation. Visited 4/5/2016.

[7] Entrepreneur. Branching out. `https://www.entrepreneur.com/article/15618`. Visited 27/05/2016.

[8] Oliver Gassmann, Karolin Frankenberger, and Michaela Csik. *The Business Model Navigator: 55 Models That Will Revolutionise Your Business*. FT Press, 1 edition, jan 2015. ISBN 1292065818.

[9] Boby George and Laurie Williams. A structured experiment of test-driven development. *Information and Software Technology*, 46(5):337 – 342, 2004. ISSN 0950-5849. doi: http://dx.doi.org/10.1016/j.infsof.2003.09.011. URL `http://www.sciencedirect.com/science/article/pii/S0950584903002040`. Special Issue on Software Engineering, Applications, Practices and Tools from the {ACM} Symposium on Applied Computing 2003.

[10] Emma Johnson. How to: Branch out to new markets. `http://www.success.com/mobile/article/how-to-branch-out-to-new-markets`. Visited 27/05/2016.

[11] Anders Christian Kaysen, Frederik Bruhn Mikkelsen, and Stefan Mailund Thilemann. Turning an idea into a software-intensive business. January 2016.

[12] Ash Maurya. Why lean canvas vs business model canvas? https://leanstack.com/why-lean-canvas/. Visited 06/06/2016.

[13] Alexander Osterwalder, Yves Pigneur, Tim Clark, and Alan (designer) Smith. *Business model generation : a handbook for visionaries, game changers, and challengers*. John Wiley & Sons, Hoboken (N.J.), 2010. ISBN 978-0-470-87641-1. URL http://opac.inria.fr/record=b1133067.

[14] E. Ries. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Business, 2011. ISBN 9780307887894. URL https://books.google.dk/books?id=r9x-OXdzpPcC.

[15] Saras D. Sarasvathy. Causation and effectuation: Toward a theoretical shift from economic inevitability to entrepreneurial contingency. *The Academy of Management Review*, 26(2):243–263, 2001. ISSN 03637425. URL http://www.jstor.org/stable/259121.

[16] Saras D Sarasvathy. Entrepreneurship as a science of the artificial. *Journal of Economic Psychology*, 24(2):203 – 220, 2003. ISSN 0167-4870. doi: http://dx.doi.org/10.1016/S0167-4870(02)00203-9. URL http://www.sciencedirect.com/science/article/pii/S0167487002002039. The Economic Psychology of Herbert A. Simon.

[17] Donald A. Schon and Glenn Wiggins. Kinds of seeing and their functions in designing. *Design Studies*, 13(2):135 – 156, 1992. ISSN 0142-694X. doi: http://dx.doi.org/10.1016/0142-694X(92)90268-F. URL http://www.sciencedirect.com/science/article/pii/0142694X9290268F.

[18] Olivier Serrat. The SCAMPER technique. *Knowledge Solutions*, 2009.

[19] Mark Spelman. The new business model: Embrace online, branch out. http://www.cnbc.com/2014/01/21/the-new-business-model-embrace-online-branch-out.html. Visited 27/05/2016.