Key challenges in software startup processes uncovered where do they come from?

Bron, P.C.

Award date:
2018

Key challenges in software startup processes uncovered: where do they come from?

Author

Pieter Cornelis Bron

(S106618/0747139)

Supervisor

Bob Walrave, ITEM

Sharon Dolmans, ITEM

Sjoerd Romme, ITEM

ITEM, Eindhoven University of Technology

Date

December 11, 2017

ABSTRACT

Purpose – This paper reports detailed narratives explaining the relationship between Industry-Specific Knowledge, Product Development Knowledge, and Managerial Experience/knowledge with software startup processes over time, uncovering where key challenges experience in software startup processes come from.

Design/Methodology/Approach – The study employs a grounded theory approach with a preliminary boundary construction phase to gather fabula for a detailed narrative on key challenges software startups face, and where they come from.

Findings – The founding team is the knowledge broker in software startups. For this role, they need various types of knowledge present in the team. The results show that three types of knowledge need to be present. If they are absent, they lead to previously identified key challenges (Wang et al., 2015; 2016) software startups in general face. More specifically, absence of Industry-Specific Knowledge relates to challenges in the customer learning processes and the integration of external information. Missing Product Development Knowledge relates to internal development challenges. Finally, a lack of Managerial Experience and Knowledge relates to challenges in managing the process as a whole.

Originality/Value – The models provide a more detailed understanding on where software startup processes challenges find their origin. Based on this model, practical implications are crafted.

Keywords – Grounded Theory, Process Analysis, Software Startup Processes, Industry-Specific Knowledge, Product Development Knowledge, Managerial Experience/Knowledge

1 Introduction

This world is on the verge of the most exciting and disruptive wave of technology mankind has experienced since the dawn of the PC: artificial intelligence, mixed reality and quantum computing (Nadella, 2017). At its core, these technologies are driven by innovative software products. Software *startups* have shown the potential to bring these innovations into the world. For example, DeepMind used to be a British software startup specialized in artificial intelligence, until it was acquired by Google in 2014 (Medium.com, 2014). They are now better known as the creator of technology that beat a world champion Go player called 'AlphaGo' - a feat that was deemed impossible for a computer (BBC, 2016). DeepMind does not stand alone. Software companies like Facebook, LinkedIn, Spotify, Pinterest, Instagram and Dropbox are all examples of highly successful new software startups. Where it is hard to imagine a world without Facebook or Spotify, these companies did not exist back in 2000 – Facebook was founded in 2004 and Spotify started only in 2006. Software startups are considered one of the key drivers of economic and technological growth in our current era (Dishman, 2015).

The huge potential of software startups, also in terms of financial rewards, have drawn many (experienced) entrepreneurs to take the software path. However, the majority of these entrepreneurs experience significant challenges (Paternoster, Giardino, Unterkalmsteiner, Gorschek, & Abrahamsson, 2014; Sutton, 2000), leading to reported failure rates in excess of 90% (Song, Podoynistyna, Bij & Halman, 2008). What is going on?

This question has also drawn a lot of scholarly attention. In this respect, we know what they key perceived challenges software startups face are, with detailed yet preliminary explanations by Wang, Giardino, Bajwa & Abrahamsson (2015). For example, during the early phases of software startups, such as development and working prototype, startups perceive their biggest challenge as *building the product*. A case study by Wang, Edison, Bajwa, Giardino & Abrahamsson (2016) finds that software startups mention working with new technology as one of the experienced challenges they attribute to the challenge of building the product. However, the state-of-the-art in software startup research does not explain where such key challenges originate from (Unterkalmsteiner, Abrahamsson, Wang & Nguyen-duc, 2016), in view of the entrepreneurial process (Wang, Edison, Bajwa, Giardino, & Abrahamsson, 2016). In other words, *what drives such challenges and when do they occur in the software startup process?* This is the question that drives this research.

As such, in order to better understand what is needed for software startups to succeed, we must first understand the root cause of these key challenges (Unterkalmsteiner et al., 2016). In order to do so, a longitudinal multiple-case study is employed. Using a grounded theory approach (Glaser & Strauss, 1967), qualitative data is used to construct explanations through narrative. The main contribution of this thesis, to the field of software entrepreneurship research, is that it disentangles the intricacies of software startup processes through a rich presentation of what is going on, using current cases recorded in 2016 and 2017.

The rest of this thesis is organized as follows: The next section discusses the known literature on software startup processes. Subsequently, the empirical research design is presented (including a preliminary analysis upon which early concepts and boundaries are constructed for conducting the grounded research). Then, the broad results from the grounded analysis are presented which are translated into an overarching framework—which answers the main research question. I continue with an in-depth discussion through analysis of the overarching framework over the phases of software startup processes. Next, a post analysis with startup experts validates findings. Finally, this thesis concludes with a discussion on the main findings in relation with literature and a conclusion.

2 Background

The field of software development, a subset of new product development (Shalloway, Beaver, & Trott, 2009), enjoys a wide body of knowledge (Da Silva, Santos, Soares, Frana, Monteiro & MacIel, 2011; Kitchenham, Pearl Brereton, Budgen, Turner, Baily & Linkman, 2009). Consider, for example, the main studies on agile software development, covering crystal methodologies (Cockburn, 2004), DSDM (Stapleton, 2003), feature-driven development (Palmer & Felsing, 2002), lean software development (Poppendieck & Poppendieck, 2003), scrum methodology (Schwaber & Beedle, 2001), and "extreme programming" (Beck, 2004). However, in a systematic mapping study Paternoster et al. (2014) found that when it comes to new software *startup* processes, studies are relatively scarce.

The external characteristics in which software startups operate are known. More specifically, they operate in a context relatively similar to regular startups (Giardino, Unterkalmsteiner, Paternoster, Gorschek, & Abrahamsson, 2014; Sutton, 2000). Characteristics include limited resources in terms of economical, human and physical attributes. As such, they rely on third-parties to succeed because of these limited resources. Next, the teams – especially during early stages – are small, with no need for upper management and a founder-centric team structure. Last, they aim to grow rapidly and are highly risky, with failure rates up to 90% (Song et al., 2008). A difference with regular startups is that software startups experience a higher than usual time-pressure due to the fast-paced development speed of software products (Giardino, Unterkalmsteiner, et al., 2014).

While we have knowledge on external characteristics, we lack a thorough understanding of the *internal* processes that drive software startups (Giardino, Wang, & Abrahamsson, 2014; Paternoster et al., 2014). This is problematic, as "self-destruction" rather than competition leads the majority of software startups to fail within their first years of existence (Crowne, 2002).

In order to begin understanding software startup, Wang et al. (2015; 2016) researched key perceived and experienced challenges that software startups face. Perceived challenges are what startups *think* are the key challenges they face, while experienced challenges are what startups *reported* that actually happened in their venture process (Wang, Giardino, Bajwa, & Abrahamsson, 2015). They find that key challenges change over time (see Figure 1), over six distinct from a development perspective: 1) concept phase, 2) development phase, 3) working prototype, 4) full functional product with limited users, 5) full functional with high growth, 6) mature product. Their results show that developing a product in a technological uncertain environment is the biggest perceived challenge during early-stage development - along with assembling a team capable of carrying out the diverse tasks associated with software startup processes. Later on, during the growth phases (i.e., IV and V), customer acquisition and scaling the company become the largest challenges, along with an overburden of things to do.

Next to the development process, software startups go through the customer learning process in parallel (Giardino, Wang, et al., 2014). Blank (2007) proposed a customer learning process model, covering four phases: 1) problem definition, 2) problem validation, 3) solution definition, and 4) solution validation. Wang et al. (2016) continued their research by analyzing how key challenges changed over the course of this model (see Figure 2). Their results show that building the product is the biggest challenge at the start of the process, whereas customer acquisition becomes the core challenge during solution validation, at the end of the process. Building the business model is the second biggest challenge at the start of the customer learning process.

While the placement of the key challenges on the stages of these processes are important for understanding software startup processes, the real question is: why are these the key challenges and how can they be addressed? In other words, where do these challenges find their origin? Wang et al. (2015) themselves briefly responded on the *why* through a multiple-case study. While this is a step in the right direction, their results are not longitudinal. More specifically, they do not position the identified *experienced* challenges over the software startup process. This is a problem, as Giardino et

al. (2014) suggests that very often software startups tend to address challenges at the wrong time in the process. For example: while startups first need to find a problem-solution fit (Blank, 2007), they tend to focus on assessing their solution by trying to find a product-market fit (Giardino, Wang, et al., 2014). This suggests a longitudinal study is required in order to fully understand what is going on. In this respect, Wang et al. (2015) themselves also call for a longitudinal study on the identified key challenges.

Answering this call, this research aims to fill the research gap by finding out where key challenges come from in order to begin understanding the processes that drive software startups to success. Therefore, the main research question is:

RQ: Where do key perceived challenges in software startup processes come from over time?

3 Method

In order to answer the research question, a longitudinal narrative process analysis is adopted (Larty & Hamilton, 2011; Mohr, 1982). Such an approach has offered rich and fruitful new perspectives, including new theory in startup process research, before (Johansson, 2004).

For the construction of theory, Langley (1999) notes that process analysis deals mainly with sequences of "events". These "events" should be linked to "concepts and categories" in order to create theory (Strauss & Corbin, 1998). In this study, since we already have a host of previously identified key challenges (Wang et al., 2016), these can become the concepts for initial research. For example, if the event *attained x customers* is found, it is linked to the challenge *customer acquisition*.

Pentland (1999) identifies events as "stories", explaining how a key challenge was experienced. The culmination of stories construct a "fabula": a generic description of a particular set of events (Pentland, 1999). This study recognizes these "fabula" as general processes related to the key challenges. As a final step, the underlying "generating mechanisms" – the underlying structures that drive the fabula (see: Pentland, 1999) - are identified by analyzing how the fabula are related to each other in order to find a common driver, in line with Van De Ven, Poole, & Poole (1995). These generating mechanisms should theoretically explain how startups can deal with key challenges, as they are the core that drives the events, thus explaining where the key challenges came from.

Multiple cases are used for collecting these fabula, as it allows for in-between case analysis and it has a higher chance of being generalizable (Eisenhardt, 1989). The goal is to find generating mechanisms that are present in a number of software startups, as it improves the ability to draw conclusions about a broader population, given the nature of this research approach (Langley, 1999).

Grounded Theory

In order to perform narrative analysis, the data needs to be structured (Larty & Hamilton, 2011). Langley (1999) defined two general strategies to structure data, through the creation of concepts and categories: inductive Grounded Theory (Glaser & Strauss, 1967), and deductive Alternate Templates Theory (Pinfield, 1986). Since the underlying generating mechanisms explaining where key challenges over the course of software startups come from are currently unknown (Wang et al., 2015; 2016), an inductive approach offers the greatest potential. Thus, the grounded theory strategy is chosen for structuring the data and producing the theory.

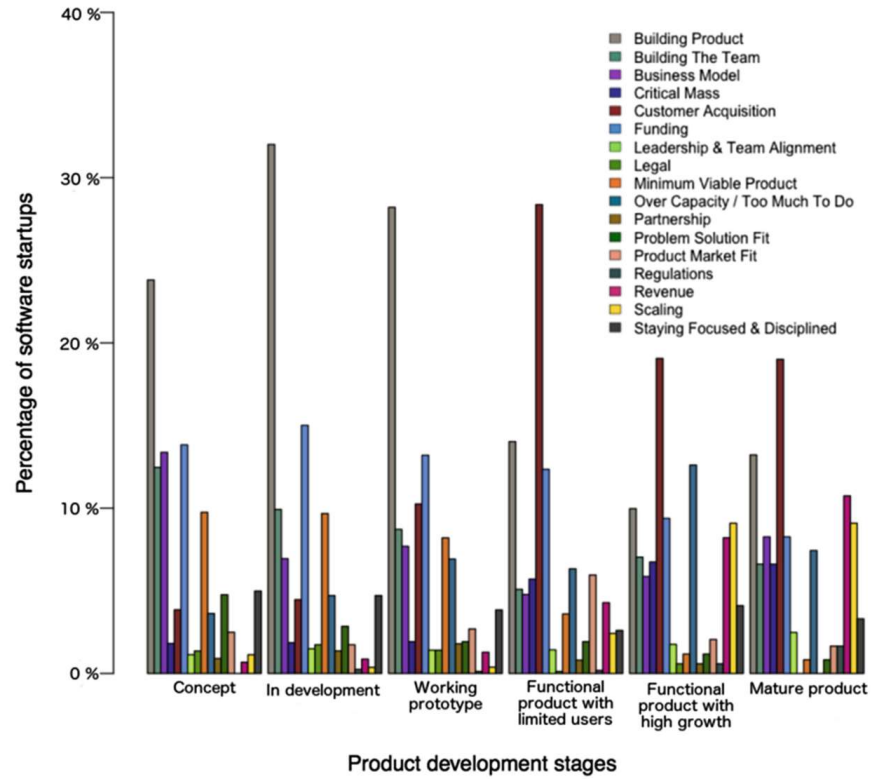

Figure 1: Distribution of key challenges as perceived by software startups, per product development stage. Source: Wang et al. (2016)

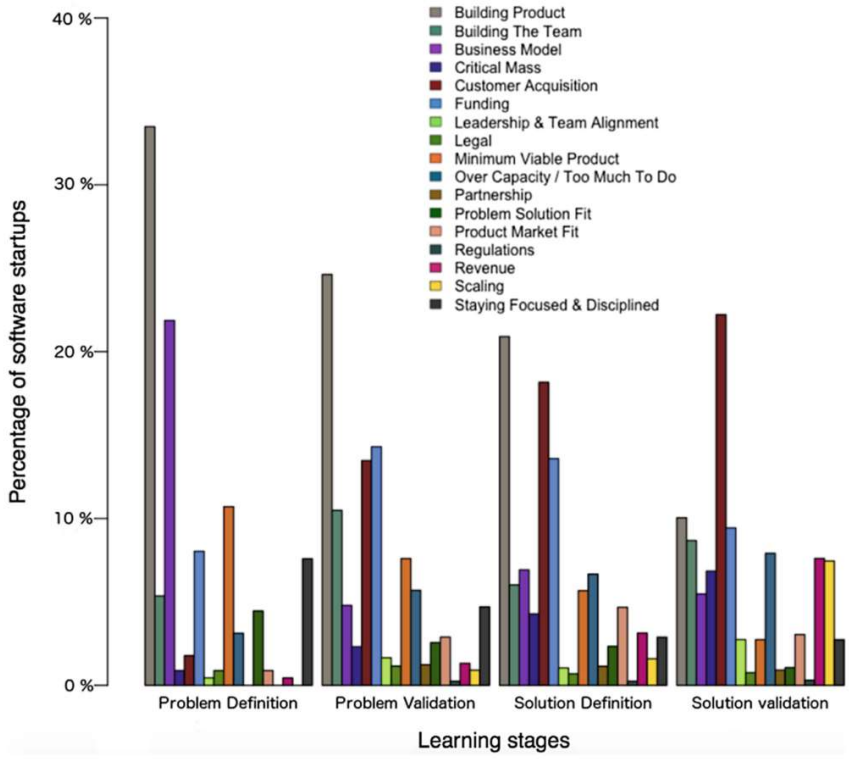

Figure 2: Key perceived challenges throughout the learning stages of the customer learning process by Blank (2007). Source: Wang et al. (2016)

This research uses the grounded theory approach of Strauss & Corbin (1998), in which research questions and boundaries are (partially) pre-determined. This allows for a more effective approach by directing the analyses to the most important concepts (Coleman & O'Connor, 2008). However, given the lack of knowledge on software startup processes (Paternoster et al., 2014), defining such concepts and boundaries is rather challenging. As such, we have a higher chance of finding something more interesting when we look into the data *first*, as it allows for surprises to guide the research (Glaser & Strauss, 1967). As a result, the research is split into two main phases: the first phase uses broad data to construct initial concepts and categories, refine the research question and set boundaries for research. Then, the second phase uses the data and results of the first and continues, by means of additional in-depth data, to construct the generating mechanisms that drive the processes.

Phases

The main source of data are interviews, as they are seen as the best method for collecting qualitative narrative data (Eisenhardt, 1989). Each research was subject to specific (i.e., different) interviews. Data for the first phase was collected - and analyzed - first, after which data collection for the second phase followed. More specifically, Phase I uses a broad interview protocol (see Appendix A1 for more details) – while Phase II uses an in-depth interview protocol (see Appendix A2 for more details). That is, the in-depth protocol is constructed based on findings from Phase I. Notably, both data from round I and round II are used to create a longitudinal dataset from which I construct theory (Figure 3).

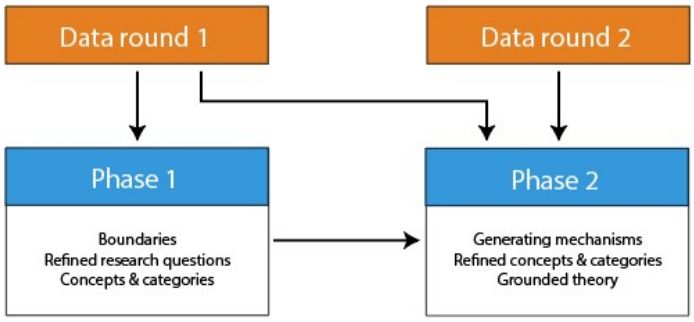

Figure 3: Data used for the research

Phase 1: Study Framing

I first collected data through semi-structured explorative interviews with founding members of six software startups based in the Netherlands. The interview protocol covers fourteen topics that are present in the process of software startups. They provide a holistic overview of the dimensions present in startup processes (Macmillan, Zemann, & Subbanarasimha, 1987). The topics include, among others, team composition, cash flows, product development, intellectual property, and competition (Appendix A1).

Given the exploratory nature of the first phase, the interviews were provided with a certain degree of freedom, with respect to the focal topic of the research. Thus, they were asked to *"just tell their story from start to end."* This allows for tales that include stories and events that are outside the (already wide) protocol, providing an even richer context. This approach also served to prevent "steering by the interviewer", which is known to increase chances of getting biased, subjective data (Eisenhardt, 1989).

After they finished their story, topics from the protocol that were not covered by the original narrative were discussed, in order to maintain maximum comparability between the different obtained datasets. That is, this approach allows for comparison between cases as the data sets, at their core, all cover the same topics (Eisenhardt, 1989; Langley, 1999). Additional data is gathered in terms of business plans,

technical documentations of the product, and general background information of the startup. These data serve triangulation (Yin, 1994).

The preliminary interviews were held between July 2016 and March 2017. The complete database for this phase involves over five hours of narrative. In total, over 50 pages of narrative data is collected with more than 30000 words. Interviews were transcribed one by one, and subsequently coded: key challenges are the concepts to which events belong. Related events are grouped together in fabula. Fabula with similar drivers in turn are grouped together in generating mechanisms (see Figure 4). Using this approach, generating mechanisms are sought for every key challenge.

Following Goulding, (2002), this process continued until no further codes of interest were found. More specifically, after three transcriptions (see Appendix B1, B2.1 and B3), saturation was achieved in terms of concepts and their related events, as no new and unique events were found. Additional data were collected, such as important quotes from the remaining startups' interviews (Appendix B4, B5.1, and B6) – which thus did not provide new insights with respect to events/concepts. However, they serve as supporting evidence that the events occurred in more than three startups.

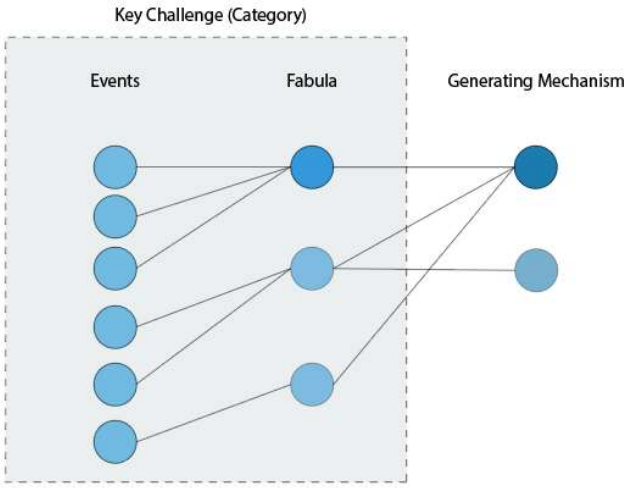

Figure 4: The relationships between events, fabula, and generating mechanisms.

Table 1 provides an overview of all the events that occurred in the processes of the startups, along with their fabula and generating mechanisms (See Appendix C1-C6 for the event tables). This is in line with Pentland's (1999) procedure on narrative analysis (see Appendix D for more details on the method). The table shows that generating mechanisms are based on a lack of or slack in any of the following three types of knowledge: Industry-Specific Knowledge, Managerial Knowledge or Product Development Knowledge. This suggests the knowledge-based view (Grant, 1996) provides a good research lens for the second research phase.

Finalizing Phase I, the available types of knowledge in each founding team are obtained during post analysis. The startups are asked to reflect on what types of knowledge they think their founding team possesses. Table 2 provides a summary of their answers. Product Development Knowledge is further split between having basic knowledge on software architecture, infrastructure and development in general and being able to actually program, as startups remarked that there is an important difference between these variants. Furthermore, managerial knowledge is split between knowledge and experience, as previous research has shown that startup experience seems to drive progress during early phases, while knowledge drives later stages (Bosch, Olsson, Björk, & Ljungblad, 2013).

Table 1: Events, their fabula and their underlying generating mechanisms of the startups, along with the key challenge to which the event belongs.

Event	Fabula	Generating Mechanism	Key Challenge (Concept)
HRComp			
A clear understanding of customer needs in the market	Problem-solution fit	Sufficient Industry-Specific Knowledge	Building the Product
Wrote a successful business plan	Understanding the market and business	Sufficient Industry-Specific Knowledge	Business Model
Development team negotiations stranded	Hiring process	Lack of Product Development Knowledge	Team Assembly
Own team assembly stranded	Hiring process	Lack of Product Development Knowledge	Team Assembly
Year of self-learning in software development	Obtaining product development knowledge	Lack of Product Development Knowledge	Building the Product
Mutual understanding of what had to be done + good technical documentation	Product development	Sufficient Product Development Knowledge	Building the Product
Successful integration and co-development of external API	Product development	Sufficient Product Development Knowledge	Building the Product
Customer needs were met + co-creation for solution evaluation	Customer growth	Sufficient Industry-Specific Knowledge	Customer Acquisition
CardComp			
Process was closely managed, leading to rapid and successful app development	Product development	Sufficient Managerial Knowledge/Sufficient Product Development Knowledge	Building the Product
The founding team does not know how to program: hired developers to develop app	Obtaining product development knowledge, Hiring process	Insufficient Product Development Knowledge	Building the Product
Switched target market, as current target companies do not identify themselves as having the problem CardComp envisioned for their solution	Problem-solution fit, Customer growth	Insufficient Industry-Specific Knowledge	Customer Acquisition
DrinkComp			
Founding team cannot program an app, has no IT knowledge, thus they hired a developer	Obtaining product development knowledge, Hiring process	Lack of Product Development Knowledge	Team Assembly
Re-boot of development after 9 months, as the sole developer blacked out, lost communication. No backup present and no documentation lead to a reboot.	Product development	Lack of Management Knowledge	Building the Product
Product did not fit customer problem.	Problem-solution fit	Lack of Industry-Specific Knowledge	Building the Product
Redevelopment through customer learning processes	Product development	Newly obtained product development/management knowledge	Building the Product
MarketComp			
Very rapid product development as the founding team has programming skills	Product development	Product Development Knowledge	Building the Product
Aborted product, rethought value proposition as product did not catch on.	Problem-solution misfit	Insufficient Industry-Specific Knowledge	Customer Acquisition
ClothComp			
Wrote successful business plan. Based on previous experience, ClothComp was able to assess exactly what the customer is waiting for.	Understanding the market and business	Industry-Specific Knowledge	Business Model
Finetuning the concept with the customer, as he realized understanding the market & business is not enough to find a good problem-solution fit	Problem-solution fit, Product development	Industry-Specific Knowledge	Building the Product
Validate the solution with the customers with a working prototype	Product Development, Problem-solution fit	Industry-Specific Knowledge and Product Development Knowledge	Building the Product
SalesComp			
Hired developer to develop product, as founding team can't program	Hiring process	Lack of Product Development Knowledge	Team Assembly
First customer remarked that this is exactly what they want. Excellent problem-solution fit. Founder got to the core of the problem in the business	Problem-solution fit	Industry-Specific Knowledge	Building the Product
Product easily scalable, rapid expansion as customers are dying to get this product. Good problem-solution fit. Thus: high growth	Customer growth, Problem-solution fit	Industry-Specific Knowledge	Customer Acquisition
Based on previous experiences in running startups, founder learned how to deal with chaotic start: ad-hoc decision making management style	Product development	Management Knowledge (Experience)	Building the product

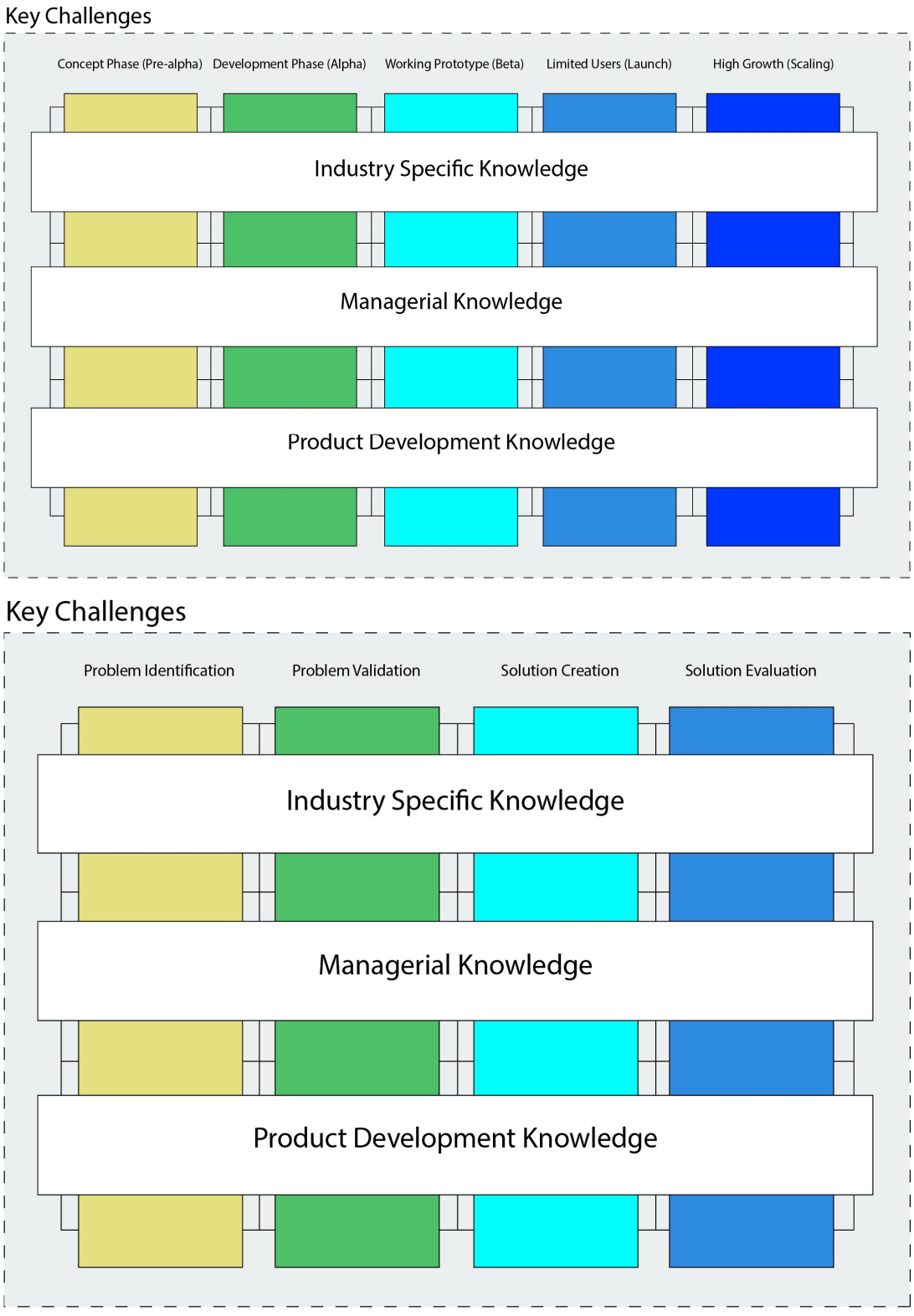

Figure 5: Boundaries used in the analysis.

Table 2: Types of knowledge available in the founding team of the startups.

Profile	CardComp	HRComp	ClothComp	MarketComp	DrinkComp	SalesComp
Industry-Specific Knowledge	-	+	+	-	-	-
Product Development Knowledge: Basic	+	-	+	+	-	+
Product Development Knowledge: Programming Skills	-	-	+	+	-	-
Managerial Knowledge	+	-	+	+	-	+
Management (Startup) Experience	+	-	+	+	-	+

The table shows that the dataset provides startups with a great variety of different background knowledge compositions, offering heterogenetic cases to analyze. The cases range from absolutely no knowledge at all (e.g. DrinkComp) towards full knowledge in all three (five when counting subsets) types of knowledge (e.g. ClothComp). This improves chances of generalizability as the startup cases are vastly different from a knowledge perspective (Yin, 1994).

The analysis in Phase I concludes that three (five when counting subsets) types of knowledge are generating mechanisms of causes related to key challenges in software startups, suggesting a knowledge-based view. The research question is thus refined as following:

RQ: From a knowledge-based view, where do key challenges in software startup processes come from?

The boundaries for the research are set as depicted in figure 5: over the customer learning phases and product development phases, the contribution of the types of knowledge are analyzed in relation with the key challenges, focusing on where they drive key challenges in the process.

Phase 2: In-depth interviews & producing grounded theory

In the second phase, data is gathered through additional in-depth interviews with the startups. These interviews were held between August 2017 and November 2017. At the core, the interviews are similar (see Appendix 2), again for maximizing comparability. However, since the stories of the startups are different, specific questions aimed at their unique processes were added. Also, any remaining questions that arose during the preliminary analysis were discussed in order to confirm assumptions (see Appendix 2.1 for an example). As explained, the interviews were semi-structured (and again exploratory in nature) to prevent bias and steering from the interviewer (Eisenhardt, 1989). Given the resource limitations associated with this project, the in-depth follow-up interviews were conducted with two startups: HRComp and ClothComp (See Appendix A2.2-A2.3 for the protocols), who were upfront determined as most likely key informants.

The data from both Round 1 and Round 2 interviews were combined in order to find generating mechanisms that are present in all researched startups. The results that are discussed are twofold: the broad results (Section 4) discuss overall connections between knowledge and key challenges. The results describe the process as a whole, creating a framework that allows in-depth research between the relationships present in the framework, which is used for the next section.

Next, the in-depth results (Section 5) dive deeper into generating mechanisms by mapping the generic framework onto the startup processes of customer learning (Blank, 2007) and product development

(Wang et al., 2016), uncovering where the generating mechanisms drive what key challenges. From a knowledge perspective, this ultimately explains where key challenges come from, answering the research question.

Lastly, the results and the models that are proposed are validated by one startup (SensorComp, see below) and one expert on entrepreneurship (Section 6). The contacted SensorComp has, from a knowledge perspective, a founding team in line with ClothComp. Their venture has been very successful so far, with a high-growing customer base and a successfully launched product. Their first investment round has resulted into a positive response from investors and a successful funding. Next, the contacted expert has over 30 years of experience in running startups, and is connected to both Utrecht University and Eindhoven University of Technology. His expertise served startups with the types of knowledge, asking him to reflect on success stories with startups he experienced in the past (See Appendix 3.2 for the protocol). By discussing the models from multiple entities in the research field, a validity check is possible from multiple perspectives, improving certainty that the models may be empirically valid (Eisenhardt, 1989).

Descriptive quotes from Interviews

1*: "You see the needs from the market, the wishes, and the problems. Based on your [industry-specific] knowledge you can anticipate on that and formulate a concept."*

2: *"All knowledge as a consumer (with such a background in retail) made me understand the demands from consumers to enjoy a perfect shopping experience."*

3: *"I'd say the business* ***model*** *[is empowered by Industry-Specific Knowledge] as it brings a disruptive solution to the industry just because we saw what everybody else has seen but thought what nobody else has thought"*

4: *"You need to have an "unfair" advantage in comparison to other startups to survive in the software scene. For me, it's my social capital that allows me to converse with large companies [potential customers] others cannot easily get in contact with. Social capital comes from being in the industry."*

5: *"[My business partner] started development by hiring an IT Company to do the development for him. It didn't go well – they got into a fight... In hindsight, he realized he didn't know enough about IT to guide how the product should be built."*

"If you start a company with a development team… then you need to constantly control the [product development] process. You need to document everything, like "Where are we?" "What are we going to do"? You need to plan everything. You need knowledge for that."

6: *"I tried to assemble a team to develop the product myself. Well, like I said I was a true layman, I didn't know anything about software. I didn't know the difference between a WordPress developer and an actual back-end developer. I didn't know who to hire."*

7: *"Having an understanding of and planning the required [product] development platforms, infrastructure, and architecture before you start development makes a far better ground for scaling up later."*

8: *"Management methodologies and principles are impossible to apply. You can use learned negotiation tactics and conversation tactics, however working structurally is very difficult. You just don't know what's going to happen tomorrow."*

9: *"You can't really manage the process during early phases. It's more like guys, we are in this together, and we must all do as much as we can in order to reach the next level."*

10: "*When we have a clear concept and need to develop it to a prototype, your management skills will be the driver of a smooth and efficient development phase. You can't draw a good plan if you don't know how to manage different tasks, resources and ideas and most importantly, how to hire the best team you need for that."*

11: *"If you are able to successfully manage the development phase, leading to an excellent prototype your role from now on is focused almost entirely on managing the company according to its mission, vision and values. The rest is up to each team leader that you have hired at the right time."*

12: *"After a successful product launch and customer acquisition, the process stabilizes and you need to start managing scaling up the company… It becomes more important to apply structure and work far more systematically… You hire more managers."*

4 Results: Broad

The main theoretical framework as shown in figure 6 is a combination of the types of knowledge (generating mechanisms) and the relevant entities found in startups during early phases to which the concepts belong: founding team as knowledge broker, external environment, and the internal developers (categories). In figure 6, the development process concerns the startup process as a whole. The different types of knowledge are considered as generating mechanisms throughout the process that allow for dealing with key challenges.

The different types of knowledge contribute towards the process in different ways:

Industry-Specific Knowledge is the enabler between the external environment and the founding team, aiding with the customer learning process. It helps to address key challenges such as finding a problem-solution fit (quote 1), finding a product-market fit (quote 2), creating the business *model* (quote 3), and acquiring customers (quote 4).

Product Development Knowledge in turn is the enabler between the internal developers and the founding team, aiding the product development process. It helps with addressing key challenges including building the product (quote 5), building the team (quote 6), and scaling up (quote 7).

Lastly, Management Experience in running startups rather than Managerial Knowledge allows a smooth development process during early phases (quote 8, 9, 10). This is in line with previous studies. These studies advocate for ad-hoc decision making and lean strategies during early phases of the startup process rather than applying rigorous management principles and strategies (Bosch et al., 2013; Unterkalmsteiner, 2015).

However, during scaling and maturity phases of the startup process, Managerial Knowledge becomes an important moderator, rather than Experience (quote 11, 12). This is an indicator that from a management perspective, the data needs to be split between early and scaling/maturity phases. Figure 7 is an adaption of the main framework for during these scaling/maturity phases. The founding team transitions into managing the vision and mission of the startup (quote 11), while newly acquired managers working underneath the founding team take on the role of the knowledge broker, essentially creating hierarchy in the work structure. In terms of knowledge, management knowledge plays an increasingly important role during these phases as the process stabilizes and the team grows (quote 12), in line with previous research (Paternoster et al., 2014). The founding team uses external information such as macro and micro-environment factors to guide the startup, aided by Industry-Specific Knowledge.

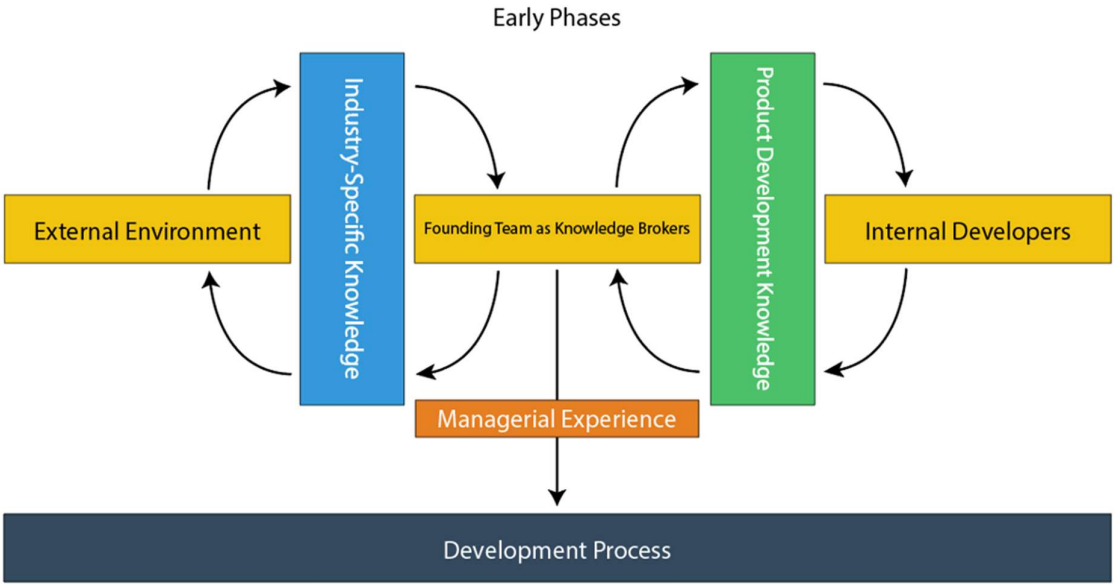

Figure 6: The main theoretical framework for early phases.

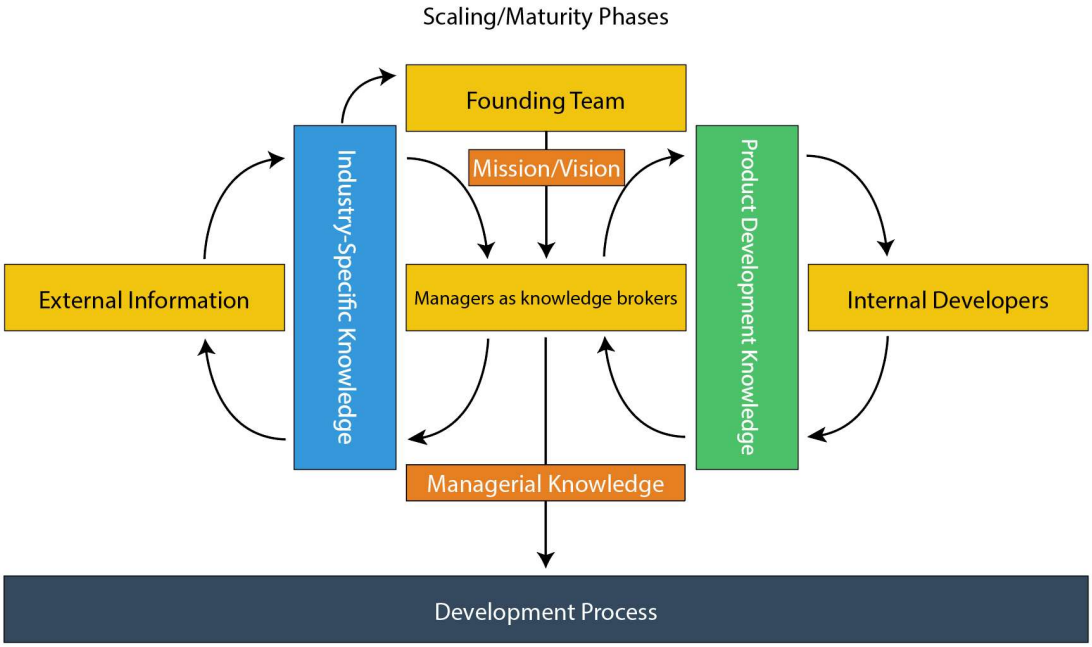

Figure 7: Theoretical framework for scaling/maturity phases.

Descriptive quotes from Interviews

*13: **"**They also need to have the [entrepreneurial] mentality, and many developers do not have that. It's very difficult to find the developers that do."*

***14:** "You are actually trying to sell your dream, your vision, to an external developer and that developer has to transform your dream into a real product."*

***15:** "I invited a savvy developer to organize the idea into a well-structured coding plan… It helped me on the my-brain-to-his-brain-to-his-computer process. It worked perfectly the other way around as well. His knowledge made me change some reasoning wrongly taken out of my coding ignorance."*

***16**: "We **need** to control the process ourselves… So I managed the development process on a weekly basis, like okay, we are now going to build this, we are going to build it like that. I brainstormed with my developers in terms of how we do it and what platform we use. In that sense we developed it ourselves, although I did not write the code. That's where the developers came in."*

***17:** "If you don't understand what the developer is doing you cannot look behind the scenes and understand what is going on. You don't understand the process."*

***18:** "The team needs to have knowledge on setting up a development environment. This requires an understanding of infrastructure such as Microsoft Azure, Hubstaff, Bitbucket (platforms) to welcome our developers."*

***19:** "By the end of April we held an evaluation with our users [after beta launch] which provided feedback that our app was slow. We passed this feedback on to our developer, who proposed to build the product on a different platform."*

***20:** "The developer is working at home. It does not make sense to bring along the developer at every technical meeting with customers as he should be focused on building the product."*

***21:** "A [B2B] customer often requests a specific integration of the product with their systems. You may know that it's technically possible, but [if you lack Product Development Knowledge] you don't know what is needed to do it. You don't know how long it is going to take or the costs that are involved… It becomes very difficult to attract customers if you can't assess that on the spot."*

***22:** "Having Industry-Specific Knowledge helped me to place customer feedback in the context of the market, resulting in a better understanding of how the product should change."*

***23:** "When you speak with people, you speak the same language. Say that I had an entirely different background and I went talking to those HR managers, then we would have two different conversations."*

Founding team as knowledge broker

The key finding in the data that allowed for the construction of the presented framework is that the founding team is the knowledge broker between external and internal information during early phases. While in the past the first developer was usually a software development manager, who would run the development process on its own (Coleman & O'Connor, 2008), that role has changed as these type of developers are not so easily found anymore. A quote by HRComp best describes the problem the startups are dealing with (quote 13): they do *want* to hire software development managers for their founding team, but they are very hard to find. The result is that out of the seven analyzed startups, only two had a software development manager in the founding team. The other five startups employed freelance developers to do the work for them. The difference is that whereas software development managers usually manage the software development process as whole, including the integration of external information (Coleman & O'Connor, 2008), the freelance developers only execute the programming/product development part of the startup process. The founding team then takes on the managing role of the software development manager. The relationship between freelance developers and the founding team is summarized by HRComp (quote 14). ClothComp underlines that the process is a two-way information exchange, in which execution meets management (quote 15).

As such, when a software development manager cannot be hired in the founding team, the founding teams seem to manage development (quote 16). This requires them to have an understanding of product development (quote 17), otherwise knowledge cannot be exchanged. ClothComp best describes what types of knowledge the founding team should have at least have for successfully managing product development (quote 18): they should know how to setup the development platform in terms of infrastructure and architecture upon which the future product is built. Without creating a proper base before starting development, the process is doomed to fail. When the developers only execute the process, the founding team *needs* to have this Product Development Knowledge.

Turning the lens outwards, the startups remark that the members of the founding team are responsible for gathering customer information for developing the product (quote 19, 20). They report that they usually do not bring their developers to customer meetings, as that would cost a lot of valuable time while the developer should be focused on developing the product (quote 20). While this seems like a reasonable strategy, this also requires that the founding team knows about the properties of the software product that is being built. Without this knowledge, they cannot integrate customer feedback into their product, as they interpret and pass on this information. An example scenario that underlines this notion is found in a B2B scenario by HRComp (quote 21). Lastly, the startups remarked that having Industry-Specific Knowledge enables smoother conversation with potential customers. They were able to better understand feedback they receive (quote 22). Also, they were able to "speak their language", increasing a mutual understanding of the market (quote 23).

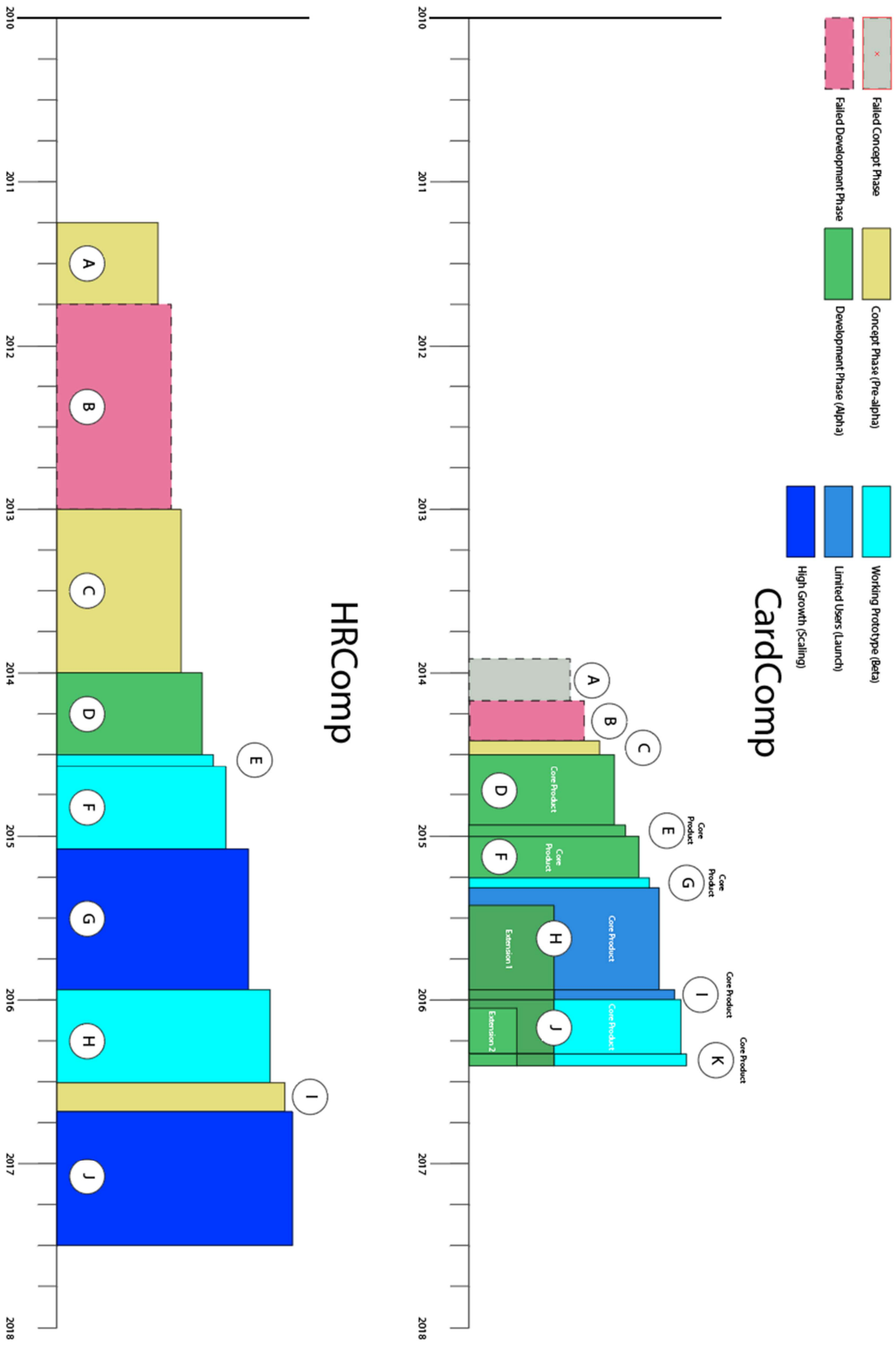

Figure 8: Startup processes of HRComp and CardComp.

Descriptive quotes from Interviews

24: *"Our key partner contacted us that they wanted to stop collaborating with us. We had grown to be a large threat for their own existence."*

25: *"In January/February [2017] we realized that our first step on the market was too big. We must focus on something smaller, like events… That's when we invented the dashboard, as we had nothing to offer for our new target market."*

26: *On their website: "We are gone for now, but we will be back with a new website and app and a new value proposition. To fix this we have to quit for now. See you later!"*

27: *"When we first contacted three private brands, showing it [the product], that's when we got input and reactions from outsiders. We used the feedback to further develop our working prototype. We wanted to impress people from these brands, so we made specific ads for them."*

28: *"Only talking about your idea does not convince someone to dedicate time to it. Only when you have something solid you can contact people to partner up. "*

29: *"Planning, planning, planning, before anything else. Listening is very important too."*

30: *"There are so many events that could set a startup back a few steps. It is inevitable that it will happen."*

Non-linear phases

Next, it was found that no startup went through the development phases in a linear manner. As an example, Figure 8 shows the startup processes over time of CardComp and HRComp in terms of development phases. CardComp never experienced high growth, while HRComp was forced back into a concept phase after a partnership was discontinued. CardComp also showed a simultaneous development phase with a launched product phase, as they were developing two products at the same time.

There are many external events that may put a startup back into previous phases that cannot be controlled by the entrepreneurs. For instance, HRComp's event that a key partner stopped collaborating put them back in a concept phase. The founders could not anticipate this beforehand; it was a risk they had to take (quote 24). CardComp experienced a negative reception of their first product that was launched in the market. This forced them to reposition their value proposition and target market, after which they had to develop new products that would fit the new market (quote 25). This put them back into a concept/development phase. MarketComp experienced a similar event, in which they had to take their entire offering offline to re-think what they wanted to offer to their customers (quote 26). This set them back to the concept phase. ClothComp remarked that they were set back to the development phase from a working prototype phase three times because of customer feedback (quote 27). This seems to be a common thing to do, as startups report that in order to convince partners to collaborate or to obtain feedback, a working product is needed instead of just an idea (quote 28). As such, they take their working prototypes to their customers, which in turn tell them that they may need to change aspects of their product, essentially sending them back to the drawing boards.

In order to deal with setbacks and surprises, ClothComp remarks that planning before starting out is key for development, suggesting that it is the combination of being comfortable to deal with chaos and maintaining overview and planning throughout the early process is what drives success during early phases of startups (quote 29). In line with effectuation principles, ClothComp notes that during early phases, entrepreneurs should focus on what they *can* control (Sarasvathy, 2001).

Concluding, it must be taken into account that actual startup processes do not necessarily go through phases in a linear manner. Set-backs are the norm rather than the exception (quote 30).

5 Results: In-Depth

Several fabula are found in the data per type of knowledge. These are discussed here, per knowledge type, over the phases.

Managerial Experience/Knowledge

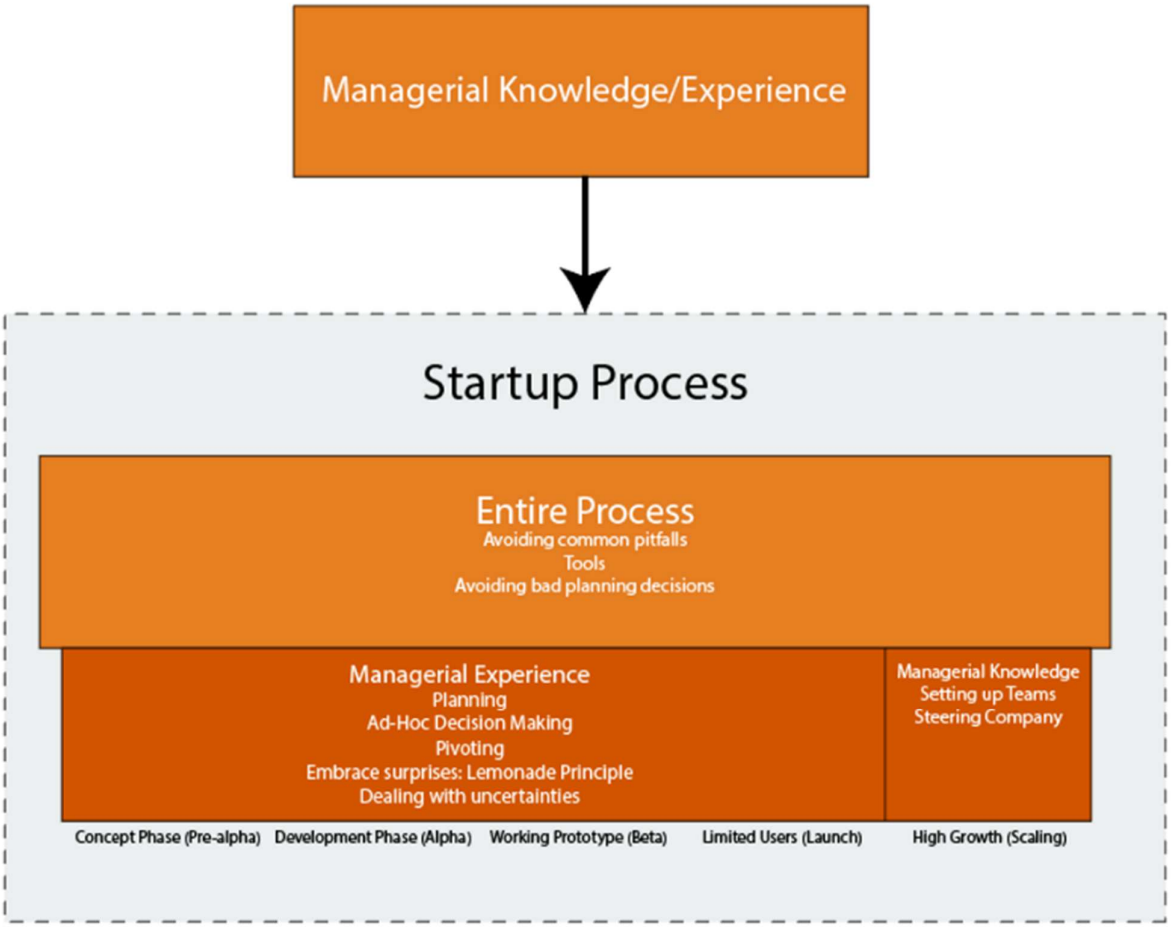

Figure 9: Managerial Experience and Knowledge aids the development processes of startups as a whole.

Managerial Experience and Knowledge seems to help managing the startup process as a whole (See Figure 9). As previously discussed in the generic results, previous *experience* in startup processes help during early phases of the development, up to high growth/scaling, as opposed to Managerial Knowledge. Managerial Knowledge becomes important during high growth/scaling phases, as the team grows and the process stabilizes.

As a whole, Managerial Experience and Knowledge influence the tools the founding team uses for managing the process. In similar fashion as Coleman & O'Connor's (2008) findings on software development managers, founders reported that they use the tools they have used in previous ventures and roles (quote 31). Furthermore, bad *startup* experiences shape their management styles as they try to avoid them in the future (quote 32, 33, 34). ClothComp noted that they have over 30 years of accumulated experience as managers in the industry, yet the one experience with a previous startup is what made him avoid pitfalls they fell into previously (quote 32).

Descriptive quotes from Interviews

31: *"Management skills made it far too easy building up an efficient plan using Microsoft Office 365 as we are all far (geographically) from each other on a daily basis."*

32: *"[Startup experience] helped me avoiding bad [management] decisions as the first one didn't go well. Planning, planning, planning, before anything else. Listening became very important also."*

33: *"Based on the previous experience that my business partner had with developing an app with an external IT company [which failed]... [We realized] we need to control the process ourselves… So I managed the development process on a weekly basis, like okay, we are now going to build this, we are going to build it like that."*

34: *"After three false starts I realized that if I want to successfully run development in a startup, I need to be on top of my developers."*

35: *"Management methodologies and principles are impossible to apply… You just don't know what's going to happen tomorrow."*

36: *"[Ad-hoc decision making] It's more gut feeling than applying methods. You can only improve your gut feeling by* ***experiencing*** *startup processes."*

37: *"You learn about management in early startup processes by* ***doing it****."*

38: *"You can use learned negotiation tactics and conversation tactics, however working structurally is very difficult."*

39: *"We've been running startups for the past 25 years."*

40: *On their website: "We are gone for now, but we will be back with a new website and app and a new value proposition. To fix this we have to quit for now. See you later!"*

41: *"I started with entrepreneurship more than seven years ago, and I've been in multiple successful startups."*

42: *"The product changed over time, based on feedback that I received directly from the customers. I changed the product based on feedback from my customers. I use their feedback to evaluate my solution to see whether the value proposition fits for them."*

43: *"After the process has stabilized and the startup grows, more people join the company and you need to hire managers. You need to delegate more. Then it becomes important to work systematically and structured."*

44: *"If you start a company with a development team… then you need to constantly control the process. You need to document everything, like "Where are we?" "What are we going to do"? You need to plan and document everything."*

45: *"The developer brought in the technical know-how, and we created the strategy, the business model, making money, the commerce. That combination worked really well. Unfortunately, he got a blackout and he was gone. As we did not keep track of documentation, we lost everything. No technical know-how, no documentation. No plan."*

Experience

There are a couple of fabula explaining how experience helps and why knowledge does not during early phases.

First of all, the consensus of the startups is that applying management principles is impossible because these phases are too unstable to manage (quote 35). On the other hand, experiencing startup processes help with creating a gut feeling that improves ad-hoc decision making skills and pivoting (quote 36, 37).

Second, HRComp notes that previously obtained skills such as negotiation tactics and conversation tactics can be utilized (quote 38). Such soft skills come from experience, not from theoretical knowledge (Engeström, Miettinen, & Punamäki, 1999), supporting the proposition.

Third, a common pitfall is that inexperienced entrepreneurs do not want to deviate from the original plan. Previous research found that novice entrepreneurs tend to see their product as their baby, reluctant to change their ideas. On the other hand, experienced entrepreneurs are not afraid to change their value proposition into something that sells (Nijssen, 2014). This seems to be in line with what is found in this research: *experience* in startups seems to be what drives good decision making for a successful startup.

The difference is seen in the researched startups: in the case of CardComp, changing the target market led to a novice entrepreneur leaving the company, as he was unwilling to deviate from the original plan. Another is example is related to development speed. DrinkComp consists of three young entrepreneurs running their first startup. They needed rigorous coaching by a startup mentor to make them realize their original value proposition was not working. They were reluctant to change, costing them a lot of time before they actually switched.

On the other hand, MarketComp consists of three seasoned entrepreneurs (quote 39). Once they realized that their value proposition was not working, they pulled their product offline and focused efforts on rebuilding the product (quote 40). Another example is SalesComp. The founder created multiple software ventures in the past decade (quote 41) and he showed a clear trend in his ad-hoc decision making style: he led feedback guide the product's properties (quote 42). Both startups show an openness to change, as opposed to the novice entrepreneurs in CardComp and DrinkComp.

Knowledge

Managerial Knowledge becomes more important when scaling up the company. Then, management processes and principles become relevant as the team grows and the process stabilizes (quote 43). The startups tend to create management hierarchy during these phases, which they deem require a more structured and systematic management style.

Controlling the process

The startups report that while management principles do not apply, the process still needs to be controlled. HRComp, ClothComp, CardComp,

and SensorComp note that they successfully managed the early phases because they controlled the process by creating documentation and a plan before and changing the plan during the process. The planning mostly involves awareness of the startup's position in the industry and pivoting possibilities – e.g. directions (quote 44). As for planning frequency, CardComp remarked that they were on top of the process, managing it on a weekly basis in order to keep grip on the process (quote 33). On the other hand, DrinkComp failed to create a smooth development process because they did not document development during the process; they only created a strategy at the start. They had to restart development as everything fell apart (quote 45).

Conclusion
Whereas during early phases up to growth Managerial Experience aids entrepreneurs the most, Managerial Knowledge becomes important during high growth/scaling phases as the team grows and the process stabilizes. During early phases experience helps avoiding bad decisions made in previous ventures, such as not planning out the future. It helps with creating gut feeling that helps with ad-hoc decision making and with pivoting. The startups also reported that it helps with learning to see opportunities and dealing with unexpected events. As a whole, experience/knowledge helps with picking the tools to manage the project.

Product Development Knowledge

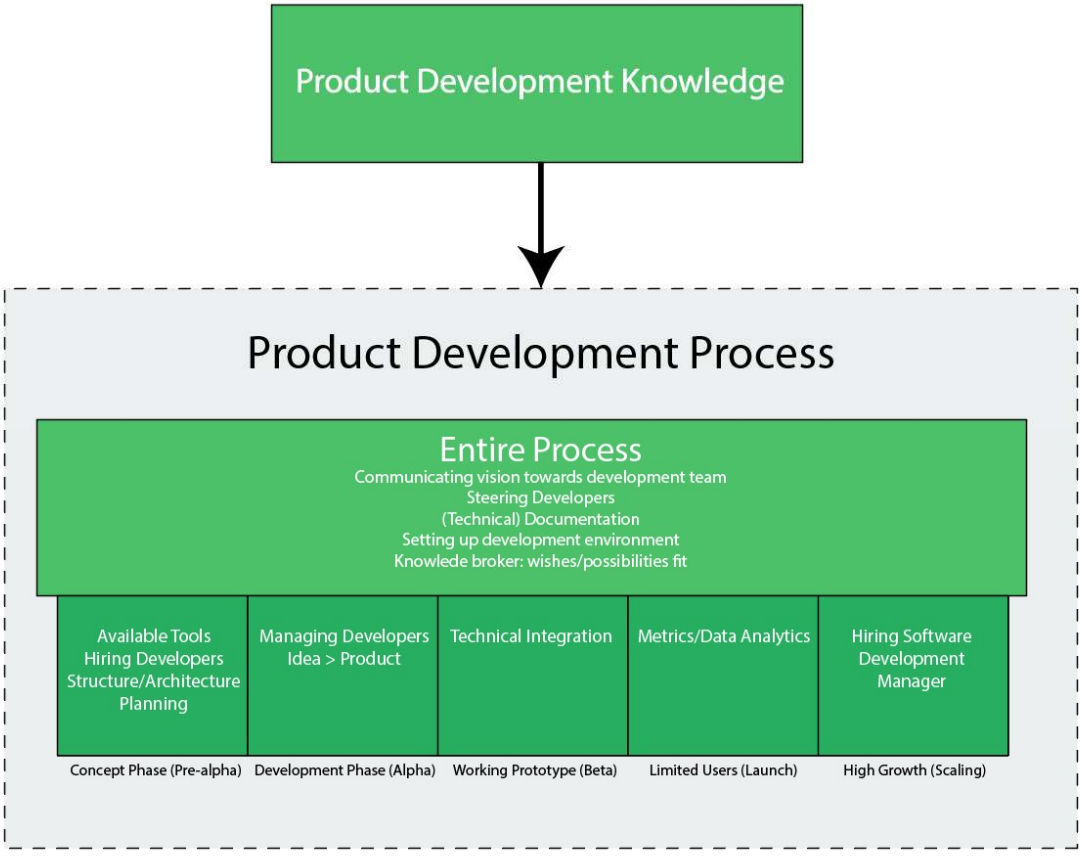

Figure 10: Product Development Knowledge aids startups with the product development process.

Figure 10 summarizes the fabula related to Product Development Knowledge. Having Product Development Knowledge in the founding team enables communication with the internal developers. Overall, the startups narrate that product development with freelance developers is a process that needs to be closely managed (quote 46). Essentially, the founders try to communicate their vision towards the development team, and the team has to translate their ideas into a product (quote 47). Without

knowledge on software development, the startups note that communication becomes very difficult (quote 48).

Furthermore, having Product Development Knowledge allows the founding team to create technical documentation themselves. As previously remarked, the startups seem to agree that documenting everything by themselves is of critical importance for a smooth startup process (quote 44). As an example, HRComp and DrinkComp experienced problems transferring knowledge to a new developer when their old developer left, as he did not write documentation, and neither did they (quote 49). Creating documentation as a founding team minimizes the risk of losing such important information in the startup. ClothComp remarks that they experienced similar issues with previous startups, and they learned that the founding team should set up a proper development environment with documentation and tools for proper development management (quote 50).

During the concept phase of development, having Product Development Knowledge helps by finding the right tools the internal developers can use to develop the product. This relates to the previous mentioned need to know how to build the development environment (quote 50). Furthermore, it helped HRComp in terms of human resources. HRComp's lack of Product Development Knowledge lead to a failure to assemble their own development team, as the founder simply did not know who to hire to do the job (quote 51). CardComp and SalesComp also remarked that having Product Development Knowledge is what helped them understand what developers were needed for the task.

During the development phase Product Development Knowledge helps with guiding the idea into a product. Steering and adhering to the roadmap while documenting the process is enabled through Product Development Knowledge (quote 48).

When the product reaches a working prototype stage, startups tend to start using their prototype for showing their idea to customers and potential partners. The reason they do not do contact them earlier is that they experienced that a product sells, not an idea (quote 52). Since early customer adoption greatly increases the chances of venture success (Nijssen, 2014), a fast-paced development speed to reach the working prototype stage is important. This creates a relation between early customer adoption and product development knowledge: more knowledge creates a smoother (and faster) development process, which in turns provides a higher chance to attain early customers as the working prototype stage is reached faster.

After the working prototype stage is reached, software startups often have to integrate their product. Out of the seven startups, six startups' success relied on successful technical integration with partners. Technical integration seems to require knowledge on product development in the founding team, as the startups tend to not bring developers during technical meetings (quote 53), while they do need to discuss technical details (quote 54).

Descriptive quotes from Interviews

46: *"I was on top of our developers every day. Testing every day. Checking progress every day."*

47: *"You are actually trying to sell your dream, your vision, to an external developer and that developer has to transform your dream into a real product."*

48: *"If you don't have development knowledge yourself, you cannot look behind the scenes and understand what is going on. You don't understand the process."*

49: *"[We had a developer]. Unfortunately, he got a blackout and he was gone. No technical know-how, no documentation."*

50: *"The founding team needs to have an understanding about the development platforms, infrastructure, and architecture of software products in order to set up a development environment for the developers."*

51: *"I was a complete layman in software development. I didn't know who to hire. At one point I had hired some developers that didn't turn out to be real developers. They were WordPress developers."*

52: *"You need to have something physical to show. It is very hard to sell an idea with nothing to show for it."*

53: *"It does not make sense to bring the developer to every technical meeting. He should be at home, focusing on developing the product."*

54: *"A [B2B] customer often requests a specific integration of the product with their systems. You may know that it's technically possible, but [if you lack Product Development Knowledge] you don't know what is needed to do it. You don't know how long it is going to take or the costs that are involved… It becomes very difficult to attract customers if you can't assess that on the spot."*

55: *"We were closely listening to customer feedback, we monitored conversion rates and analyzed usage statistics and what they [the customers] said about the product."*

56: *"If you are able to successfully manage the development phase, leading to an excellent prototype your role [as a founding team] from now on is focused almost entirely on managing the company according to its mission, vision and values. The rest is up to each team leader that you have hired at the right time."*

Next, after the product is launched, MarketComp remarked that it greatly helped them to have Product Development Knowledge as they had experience in data analytics and metrics (quote 55). Furthermore, CardComp and ClothComp both hired data analysts to monitor customer reception in order to improve their product. Data analytics, as MarketComp remarks, is used to measure their customer conversion rates and customer clicks to see how well the market responds to the product, which in their eyes is vital for fine-tuning the product.

During scaling phases of the startup, Product Development Knowledge becomes less important. By now, most startups had hired a dedicated software development manager, now in charge of developing the product. Having Product Development Knowledge did help them by building up an environment that is capable of scaling up. However, the core activities of the founding team seems to be that they now focus on guiding the startup according to its mission and vision, while the managers take over development roles (quote 56).

Conclusion

Overall, Product Development Knowledge allows communication between the founding team and the development team. It helps with steering developers and with managing and creating (technical) documentation. Furthermore, the founding team needs to know how to build a development environment for their developers that is also future proof for scaling, as freelance developers only execute the programming part of the software development.

Industry-Specific Knowledge

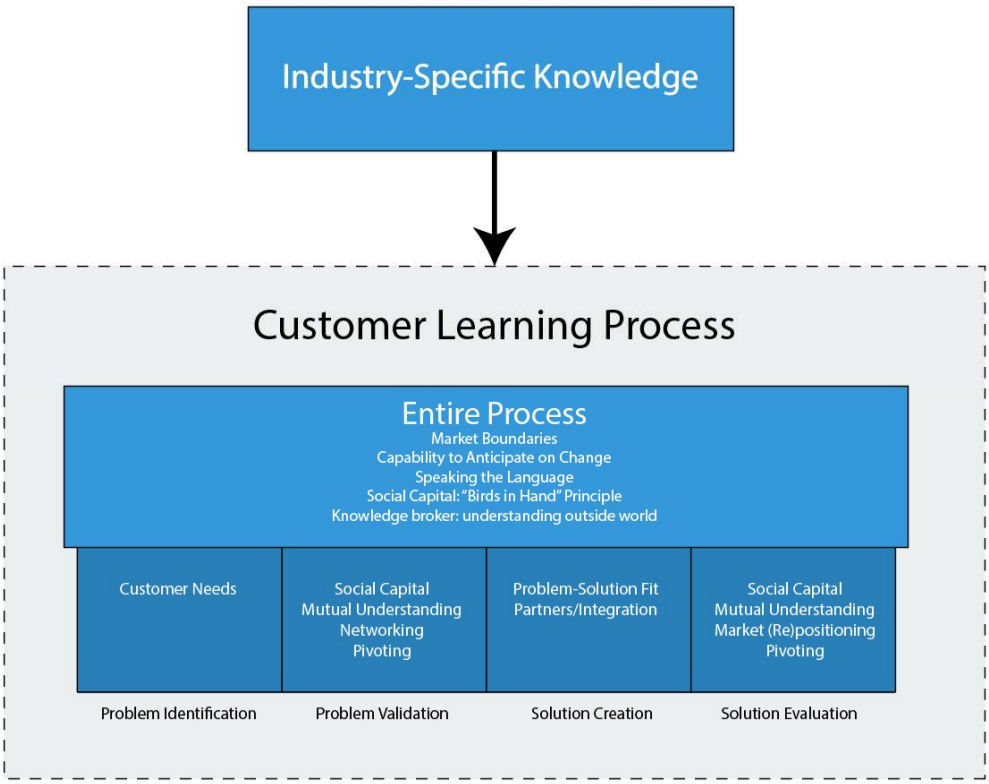

Figure 11: Industry-Specific Knowledge aids the customer learning process.

Figure 11 summarizes the fabula related to Industry-Specific Knowledge. Industry-Specific Knowledge helps with understanding the market in which the product is going to operate. As a whole, it helps with customer development as it enables software startups to understand the market

Descriptive quotes from Interviews

57: *"Throughout my 14 years' experience in retail, first behind the counter as a shop assistant then moving on to shop manager, shop owner, franchisee, master franchiser and ending up in the last 8 years as a regional manager for the same international brand I started with 14 years before, made me realize that the whole retail industry was obsolete and needed a global fixing. In my mind, retail needed the same solution as Uber brought to taxis and Airbnb to accommodation."*

58: *"Despite that a market is always changing, the basis remains roughly the same. With Industry-Specific Knowledge, you can put changes in the right context. You can anticipate what is going to happen as you understand how the market is changing."*

59: *"[With Industry-Specific Knowledge] you speak the samen language. Had I conversed with HR managers with a commerce background, we would essentially have had two different conversations at the same time."*

60: *"Without my industry background I would never have found this problem."*

61: *"My experience in the industry made me realize there was something very wrong. My years of experience made me crystallize the core problem."*

62: *"we saw what everybody else has seen but thought what nobody else has thought"*

63: "*Throughout development we are frequently in touch with potential customers (brands and retailers) and with potential users (shoppers). All inputs are taken into consideration and help correct any misleading idea we had initially assumed as correct. Also helps us validate and adding value to the solution. Yes, we had those moments [where we assumed too much] but seeking advice and listening to those influencers helped us avoid any down-term around the corner."*

64: *"We announced to our friends that we are building this product, and then you also share your vision. Then you hear from them what they would like. So yes, we checked, but not a fully-fledged market research."*

65: *"We first launched our product in April. We obtained a lot of feedback on our product. Things like that it loaded a bit slow and that we had to improve that."*

66: *"[After launch] In January/February [2017] we realized that our first step on the market was too big. We must focus on something smaller, like events… That's when we invented the dashboard, as we had nothing to offer for our new target market."*

67: *"You need to have an "unfair" advantage in comparison to other startups to survive in the software scene. For me, it's my social capital that allows me to converse with large companies [potential customers] others cannot easily get in contact with. Social capital comes from being in the industry."*

68: *"Industry-Specific Knowledge helped me understand customer feedback loud and clear."*

69: *"Thanks to my extensive knowledge in the field, I can easily put customer needs in their context and truly understand what is needed."*

boundaries, its movements and its quirks (quote 57). Furthermore, seems to allow startups to anticipate changes in the market more easily (quote 58). Startups also remarked that having Industry-Specific Knowledge allows them to speak the language of fellow people operating in the market, making networking/social interaction easier (quote 59).

Industry-Specific Knowledge is a double-edged sword. While during early problem identification startups have shown that it helps them enormously with recognizing a problem in the market (quote 60, 61, 62), it also seems to make startups overconfident that what they think is true. They seem to ignore problem validation. Instead they rush to the market with their product. The successful startups like ClothComp are aware of this problem. They listened very carefully to what their customers had to say (quote 63). The not so successful startups indeed rushed to the market, without proper validation whether the problem they are trying to solve is actually there. DrinkComp remarked that they did verify whether their identified problem is actually there, however what they were doing is an evaluation of their solution (quote 64). CardComp also developed a product first. During their beta launch, they obtained feedback on the usability of the product, not whether or not people actually believe they would find value in the product (quote 65). They went on to process this feedback and launch the product. This resulted in a change of target market after their first actual product launch as their core target market was not yet interested in their product (quote 66).

Startups that realize that assumptions need to be validated can greatly benefit from their Industry-Specific Knowledge. As SalesComp remarks, their extensive knowledge in the field gave them an unfair advantage as they had a large social capital (quote 67). Furthermore, Industry-Specific Knowledge helps processing feedback during problem and solution evaluation. Startups that have Industry-Specific Knowledge narrate that they understand the feedback loud and clear (quote 68), and they can place it in context more easily (quote 69). This also seems to help with pivoting and positioning the product in the market.

Lastly, creating opportunities for the startup is deemed more important than applying rigorous management principles during early phases of the startup, as mentioned earlier. Since being experienced in the field yields a higher social capital (quote 67), Industry-Specific Knowledge also contributes towards the creation of opportunities, as it provides a larger network of people operating in the field.

Conclusion

Industry-Specific Knowledge provides knowledge on the market boundaries, as well as an increased capability to anticipate on changes in the market. Mutual understanding with customers is possible because the founders can "speak the same language" as their customers. Having experience in the industry also seems to unlock a larger social capital, with plenty of networking opportunities. Lastly, it enables knowledge brokering with the outside world.

6 Post Analysis

In order to validate findings, the broad and depth models are discussed with an expert in entrepreneurship and a successful software startup. In general, the expert and the startup agree with the models, contributing to the validity of the models (Eisenhardt, 1989). There are a couple of specific subjects on which the expert elaborated, based on his own experience. These are discussed below.

Most notably, the expert stresses the importance of having startup experience during early phases of the process. In line with what the startups remarked, he notes that the process is too chaotic to manage using structures principles. He argues that being comfortable with "riding chaos" is indeed core to success. However, he notes that "riding chaos" does not entail blindly going through the process. In line with effectuation principles (Sarasvathy, 2001), he notes that entrepreneurs should control what they can control, and let go of structuring things they cannot control. He sees the process of planning the process beforehand as incredibly important (quote 70).

Descriptive quotes from interview

70: *"I always suggest startups to add another phase before everything: planning, planning, and planning. While being comfortable with riding chaos is core to success, planning is an incredibly important second. The better you plan and prepare beforehand, the higher the chances of a smooth development process. For example, there is plenty of research showing why software projects with governments are such a drama: these show that it often they already fail with planning, rendering successful development almost impossible."*

71: *"A roadmap is a must. The founders have to plan and setup a development platform that is future proof. That is, it should not be built on an architecture that does not allow scaling, for instance. There are many projects in which they fail to think ahead, making a successful end-product impossible."*

72: *"Understanding what a developer needs and understanding how software development works are two different things. A basic understanding of programming, logic skills and software development greatly aids with managing the developers as only then you truly understand what is going on."*

73: *"Having Product Development Knowledge helps preventing a misfit between what the customers want and what's technologically possible. As a founder you should be able to communicate that at customer meetings."*

74: *"Coaching startups is also part psychology. I have experienced being in startups, so I know what it's like to be there. Any person that wants to run startups has to experience being in one in order to create proper skills for decision making. Running startups is a skill."*

Turning our lens to product development, the expert remarks a few points. As previously discussed, ClothComp notes that the founding team should have the knowledge to create a development environment for their freelance developers to operate in. The expert confirms this finding, stating that he has seen many development processes fail because they did not create a viable development environment for their developers. He notes that the founders *must* create a development roadmap in order to successfully create a development environment, as many problems are down the road (quote 71).

Furthermore, another misconception is what leads some startups to failure: the expert experienced that while many startups understand what a developer needs, such as a platform to develop the product on, often they do not actually know how software *works*. A telling example of this phenomenon is what happened with HRComp: the startup knew that the developers needed a software development platform. However, they hired developers for a platform that was not viable for their envisioned end-product, as they did not understand critical core differences between software platforms. The expert vouches that a basic understanding of software development is needed to properly manage the developers and prevent issues such as experienced by HRComp (quote 72). His remark synergizes with the previous remark about setting up a roadmap: understanding how software development works allows the creation of a roadmap, as the team then understands what is going on down the road.

As a last remark on Product Development Knowledge, the expert elaborates on the importance of having this knowledge while consulting with partners. He agrees with the role as knowledge broker of the founding team, vouching that a founder should be able to communicate technical aspects in order to prevent a misfit between what the customers want and what is technologically possible (quote 73).

Finally, from a startup coaching perspective, the expert notes that having software startup experience does not only help as an entrepreneur, but also from his position as an expert. He believes that only coaches that have actually experienced being in a software startup can properly help entrepreneurs, due to the unique psychological environment these entrepreneurs operate in (quote 74).

7 Conclusion & Discussion

This research has addressed where key challenge in software startups come from, and has shown that a knowledge-based view uncovers several generating mechanisms that help dealing with them. Whereas previous research has already shown that Industry-Specific Knowledge, Product Development Knowledge and Managerial Experience/Knowledge contribute to increased chances of new venture success in general (Jo & Lee, 1996; Kakati, 2003; Song et al., 2008), these studies do not show the relationship between key challenges and these types of knowledge. As found in this study, Managerial Knowledge does not contribute much towards the startup process during early stages of the process as ad-hoc decision making is the norm and the process is extremely chaotic, rendering knowledge on structured management principles irrelevant. Managerial Experience does help during early stages, in line with Unterkalmsteiner et al.'s (2015) research. However, while Unterkalmsteiner et al (2015) argue that this type of knowledge is relevant due to the intense competition software startups are facing, this research also finds that successful startups learned the principles of pivoting, embracing change and planning ahead through previous startup experiences, not from theoretical knowledge. Most importantly, they let loose of tightly structured development processes and used previous startup experience instead to guide them through their processes.

The identified knowledge broker role of the founding team has important implications from a knowledge-based view. Especially during early stages, the team becomes the knowledge integrator between external information and their development team. Once the process has stabilized, the founding teams of successful software startups let go of the knowledge broker role and transitions into a leadership role, in which they do start to structure the process and apply management principles. The managers underneath the founding team then take on the role of the knowledge broker. This study finds that having both Industry-Specific Knowledge and Product Development Knowledge is of vital importance in order to successfully integrate external and internal information, and suggests that the biggest key challenges with regards to the customer learning process as well as the product development process come from a lack of either two types of knowledge in the founding team, answering where key challenges in these processes in software startups as identified by Wang et al (2015; 2016) come from.

Zooming in on software developers in software startups, the findings in this study contrast previous findings. Conversely, while Coleman & O'Connor (2008) found that software startups tend to hire software development managers to guide their development, this study finds that startups nowadays mostly hire freelance developers. The shift from hiring expensive software development managers to lead development towards developers that take on an executional role may be attributable to the software environment we currently operate in. During Coleman & O'Connor's (2008) research, developing a software product was only for the people educated in the field. However, the advent of mobile applications fueled by the launches of iOS in 2007 and Android in 2008 spurred the development of tools and platform that allows anyone to develop a software product (Nadella, 2017). This results in a landscape where cheaper alternatives than a software development manager become available for development, as shown in the narratives. Freelance developers have become an option. With the recent spur of effectuation principles as a mainstream way of entrepreneurship (Perry, Chandler, & Markova, 2012), hiring freelance developers instead of full-deck software development managers seems a logic choice, considering the affordable loss principle of effectuation: minimize costs as much as possible (Sarasvathy, 2001). While this shift in developers helps in minimizing investment for startups during early phases, the trade-off is that the founding team needs to have Product Development Knowledge and Industry-Specific Knowledge for a successful software product development. Startups should be aware of their new role as a knowledge broker, as it entails that they need to have knowledge of both the external environment and internal development in their founding team. They essentially mimic the role of a sales knowledge broker (see: Van Den Berg et al., 2014) in larger companies.

Finally, we focus on Industry-Specific Knowledge. Next to greatly contributing towards the identification of customer problems, this study shows that Industry-Specific Knowledge helps with building a large network of contacts within the market a startup aims to operate or in other words, a large social capital (Sarasvathy, 2001). Successful entrepreneurs are aware of this opportunity and leverage it accordingly. Having a social capital helps them with getting in contact with potential customers, suppliers, partners and other entities in the market. However, Industry-Specific Knowledge is found to be a double-edged sword. On the one hand it unlocks the power of having a social capital and understand customer needs, but on the other hand, a common pitfall of software startups is that their extensive knowledge creates a blind spot for things they assume to be true and things that are true. Entrepreneurs should be aware of their limits of their knowledge, and validate often which assumptions are valid, and which are not.

Limitations of the study & Future Research

Using grounded theory as a research method, paired with interviews, collects data that is centered on the insights and opinions of the interviewed respondents (Glaser & Strauss, 1967). Since these insights and opinions are subjective per interviewee, the responses may be at odds with reality. While researchers must accept the accuracy of what respondents narrate during interviews (Hansen & Kautz, 2005), the researcher opted for additional validity of the models. This is why the models presented in this research are validated by a software expert and a software startup outside the datasets, as it gives a fresh, objective perspective from respected entities in the software startup field that are unrelated to the software startups used for the construction of the frameworks.

Given the heterogenetic nature of software startup processes (Davidsson, 2016), generalizability is always a question in process analysis (Langley, 1999). However, the original dataset upon which the focal lens was built for constructing the grounded theory consisted of a group of startups that, within this knowledge-based view, was as heterogenetic as it could be. Startups that had none of the types of knowledge at all were present in the dataset, as well as startups that had all types of knowledge. Furthermore, while Eisenhardt (1989) suggested to have at least two cases to study per startup phase, this research used six.

Lastly, using process theory at its core means to approach data in a way that cannot be captured using variance analysis, as variance theory does not allow for temporal sequences (Abbott, 1990). However, the results from process analysis may produce alleyways that *can* be researched using variance analysis. Whereas the results offer rich narratives on where key perceived challenges come from, they are not of statistical significance. This is to be expected for a process study (Mohr, 1982). A suggestion for future research is thus to triangulate this process research with variance research in order to create empirically solid results (Van de Ven & Poole, 2005). The fabula narrating the various relationships between the types of knowledge and the entities related to software startups may be researched for statistical significance, in order to produce a list of statistically significant contributors to new venture success. This provides a more detailed list than what is currently available in software startup literature (Paternoster et al., 2014).

References

Abbott, A. (1990). A primer on sequence methods. *Organization Science*, *1*(4), 375–392.

BBC. (2016). Google Achieve AI Breakthrough by Beating GO Champion. Retrieved from http://www.bbc.com/news/technology-35420579

Beck, K. (2004). *Extreme Programming Explained: Embrace Change*. Addison-Wesley.

Blank, S. G. (2007). *The Four Steps to the Epiphany: Successful Strategies for Products that Win*. *Evolution* (3rd ed.). Quad/Graphics.

Bosch, J., Olsson, H. H., Björk, J., & Ljungblad, J. (2013). The Early Stage Software Startup Development Model: A Framework for Operationalizing Lean Principles in Software Startups. *Lean Enterprise Software and Systems*, *167*, 1–15. https://doi.org/10.1007/978-3-642-44930-7

Cockburn, A. (2004). *Crystal Clear: A Human-Powered Methodology for Small Teams*.

Coleman, G., & O'Connor, R. V. (2008). An investigation into software development process formation in software start-ups. *Journal of Enterprise Information Management*, *21*(6), 633–648. https://doi.org/http://dx.doi.org/10.1108/MRR-09-2015-0216

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

Da Silva, F. Q. B., Santos, A. L. M., Soares, S., Frana, A. C. C., Monteiro, C. V. F., & MacIel, F. F. (2011). Six years of systematic literature reviews in software engineering: An updated tertiary study. *Information and Software Technology*, *53*(9), 899–913. https://doi.org/10.1016/j.infsof.2011.04.004

Davidsson, P. (2016). Developments in Entrepreneurship Research. *Contemporary Entrepreneurship: Multidisciplinary Perspectives on Innovation and Growth*, 17–28. https://doi.org/10.1007/978-3-319-28134-6

Dishman, L. (2015). The State of the American Entrepreneur in 2015. Retrieved October 27, 2017, from https://www.fastcompany.com/3046773/the-state-of-the-american-entrepreneur-in-2015

Eisenhardt, K. M. (1989). Building Theories from Case Study Research. *Academy Management Review*, *14*(4), 532–550. Retrieved from http://intranet.catie.ac.cr/intranet/posgrado/Met Cual Inv accion/Semana 3/Eisenhardt, K. Building Theories from Case Study Research.pdf

Engeström, Y., Miettinen, R., & Punamäki, R. L. (1999). *Perspectives on activity theory*.

Giardino, C., Unterkalmsteiner, M., Paternoster, N., Gorschek, T., & Abrahamsson, P. (2014). What do we know about software development in startups? *IEEE Software*, *31*(5), 28–32. https://doi.org/10.1109/MS.2014.129

Giardino, C., Wang, X., & Abrahamsson, P. (2014). Why early-stage software startups fail: A behavioral framework. *Lecture Notes in Business Information Processing*, *182 LNBIP*, 27–41. https://doi.org/10.1007/978-3-319-08738-2

Glaser, B. G., & Strauss, A. L. (1967). The Discovery of Grounded Strategies for Qualitative Research. *Observations*, x, 271 . Retrieved from https://books.google.ca/books?id=rtiNK68Xt08C&printsec=frontcover&dq=Glaser,+BG.,+%26+Strauss,+A.+(1967).+The+discovery+of+grounded+theory.+Chicago:+Aldine.&hl=en&sa=X&ved=0ahUKEwiegsyV6MbPAhVr6YMKHZToC9kQ6AEIHDAA#v=onepage&q&f=false

Goulding, C. (2002). *Grounded Theory: A practical guide for management, business, and market researchers*. Thousand Oaks, CA: Sage.

Grant, R. M. (1996). Toward a knowledge-based theory of the firm. *Strategic Management*, *17*, 109–122. https://doi.org/10.1002/smj.4250171110

Hansen, B. H., & Kautz, K. (2005). Grounded Theory Applied - Studying Information Systems Development Methodologies in Practice. *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, *0*(C), 1–10. https://doi.org/10.1109/HICSS.2005.289

Jo, H., & Lee, J. (1996). The relationship between an entrepreneur's background and performance in a new venture. *Technovation*, *16*(4), 161–171. https://doi.org/10.1016/0166-4972(96)89124-3

Johansson, A. W. (2004). Narrating the Entrepreneur. *International Small Business Journal*, *22*(3), 273–293. https://doi.org/10.1177/0266242604042379

Kakati, M. (2003). Success criteria in high-tech new ventures. *Technovation*, *23*(5), 447–457. https://doi.org/10.1016/S0166-4972(02)00014-7

Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering - A systematic literature review. *Information and Software Technology*, *51*(1), 7–15. https://doi.org/10.1016/j.infsof.2008.09.009

Langley, A. (1999). Strategies for theorizing from process data. *Academy of Management Review*, *24*(4), 691–710. https://doi.org/10.5465/AMR.1999.2553248

Larty, J., & Hamilton, E. (2011). Structural approaches to narrative analysis in entrepreneurship research: Exemplars from two researchers. *International Small Business Journal*, *29*(3), 220–237. https://doi.org/10.1177/0266242611401796

Macmillan, I. C., Zemann, L., & Subbanarasimha, P. N. (1987). Criteria distinguishing successful from unsuccessful ventures in the venture screening process. *Journal of Business Venturing*, *2*(2), 123–137. https://doi.org/10.1016/0883-9026(87)90003-6

Medium.com, P. A. B. (2014). The Last AI Breakthrough DeepMind Made Before Google Bought It For $400m. Retrieved from https://medium.com/the-physics-arxiv-blog/the-last-ai-breakthrough-deepmind-made-before-google-bought-it-for-400m-7952031ee5e1

Mohr, L. B. (1982). *Explaining organizational behavior*. Jossey-Bass.

Nadella, S. (2017). *Hit Refresh: The Quest to Rediscover Microsoft's Soul and Imagine a Better Future for Everyone*.

Nijssen, E. J. (2014). *Entrepreneurial Marketing: an effectual approach* (1st ed.). Routledge.

Palmer, S. R., & Felsing, J. M. (2002). *A Practical Guide to Feature-driven Development*. Prentice Hall, Upper Saddle River, NJ.

Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., & Abrahamsson, P. (2014). Software development in startup companies: A systematic mapping study. *Information and Software Technology*, *56*(10), 1200–1218. https://doi.org/10.1016/j.infsof.2014.04.014

Pentland, B. T. (1999). Building process theory with narrative: From description to explaination. *Academy of Management Review*, *24*(4), 711–724. https://doi.org/10.2307/259350

Perry, J. T., Chandler, G. N., & Markova, G. (2012). Entrepreneurial Effectuation: A Review and Suggestions for Future Research. *Entrepreneurship: Theory and Practice*, *36*(4), 837–861. https://doi.org/10.1111/j.1540-6520.2010.00435.x

Pinfield, L. T. (1986). A field evaluation of perspectives on organizational decision making. *Administrative Science Quarterly*, *31*(1), 365–388.

Poppendieck, M., & Poppendieck, T. (2003). *Lean Software Development - An Agile Toolkit for Software Development Managers*. Addison-Wesley, Boston.

Sarasvathy, S. D. (2001). Causation and Effectuation : Toward a Theoretical Shift from Economic Inevitability to Entrepreneurial Contingency Authors (s): Saras D . Sarasvathy Source : The Academy of Management Review , Vol . 26 , No . 2 (Apr ., 2001), pp . 243-263 Published by. *The Academy of Management Review*, *26*(2), 243–263. https://doi.org/10.5465/AMR.2001.4378020

Schwaber, K., & Beedle, M. (2001). *Agile Software Development with Scrum*. Prentice Hall, Upper Saddle River, NJ.

Shalloway, A., Beaver, G., & Trott, J. R. (2009). *Lean-Agile Software Development: Achieving Enterprise Agility*.

Song, M., Podoynitsyna, K., Bij, H., & Halman, J. I. M. (2008). Success Factors in New Ventures. *The Journal of Product Innovation Management*, *25*, 7–27. https://doi.org/10.1111/j.1540-5885.2007.00280.x

Stapleton, J. (2003). *DSDM: Business Focused Development*.

Strauss, A., & Corbin, J. (1998). *Basics of qualitative research techniques*. Sage publications.

Sutton, S. M. (2000). The role of process in software start-up. *IEEE Software*, *17*(4), 33–39. https://doi.org/10.1109/52.854066

Unterkalmsteiner, M. (2015). Software Business, *210*(June). https://doi.org/10.1007/978-3-319-19593-3

Unterkalmsteiner, M., Abrahamsson, P., Wang, X., & Nguyen-duc, A. (2016). Software Startups – A Research Agenda, *10*(1), 89–123. https://doi.org/10.5277/e-Inf160105

Van de Ven, A. H., & Poole, M. S. (1995). Explaining Development and Change in Organizations. *Academy of Management Review*, *20*(3), 510–540. https://doi.org/10.2307/258786

Van de Ven, A. H., & Poole, M. S. (2005). Alternative Approaches for Studying Organizational Change. *Organization Studies*, *26*(9), 1377–1404. https://doi.org/10.1177/0170840605056907

Van Den Berg, W. E., Verbeke, W., Bagozzi, R. P., Worm, L., De Jong, A., & Nijssen, E. (2014). Salespersons as internal knowledge brokers and new products selling: Discovering the link to genetic makeup. *Journal of Product Innovation Management*, *31*(4), 695–709. https://doi.org/10.1111/jpim.12156

Wang, X., Edison, H., Bajwa, S. S., Giardino, C., & Abrahamsson, P. (2016). Key Challenges in Software Startups Across Life Cycle Stages. *Extreme Programming*, *48*, 385–386. https://doi.org/10.1007/978-3-642-13054-0

Wang, X., Giardino, C., Bajwa, S. S., & Abrahamsson, P. (2015). Key Challenges in Early-Stage Software Startups. *Lecture Notes in Business Information Processing*, *212*, 52–63. https://doi.org/10.1007/978-3-319-18612-2

Yin, R. (1994). *Case study research: Design and methods*. Beverly Hills.