Sami Hyrynsalmi, Krzysztof Wnuk, Maya Daneva,
Tuomas Mäkilä, Andrea Herrmann (Eds.)

Proceedings of

From Start-ups to SaaS Conglomerate: Life Cycles of Software Products Workshop (IW-LCSP 2013)

Hosted by 4th International Conference on Software Business (ICSOB 2013)
in Potsdam, Germany, June 11, 2013

Addresses of the editors:

Sami Hyrynsalmi
University of Turku
FI-20014 Turun yliopisto, Turku, Finland
sthyry@utu.fi

Krzysztof Wnuk
Lund University
P.O. Box 118,
221 00 Lund,
Sweden
krzysztof.wnuk@cs.lth.se

Maya Daneva
University of Twente
PO Box 217
7500 AE Enschede,
The Netherlands
m.daneva@utwente.nl

Tuomas Mäkilä
University of Turku
FI-20014 Turun yliopisto, Turku, Finland
tuomas.makila@utu.fi

Andrea Herrmann
Daimlerstr. 121
D-70372 Stuttgart,
Germany
herrmann@herrmann-ehrlich.de

Contents

Organizing Committee

Krzysztof Wnuk — Lund University, Sweden
Tuomas Mäkilä — University of Turku, Finland
David Callele — University of Saskatchewan, Canada
João M. Fernandes — University of Minho, Portugal
Timo Knuutila — University of Turku, Finland
Sami Hyrynsalmi — University of Turku, Finland
Antero Järvi — University of Turku, Finland
Andrey Maglyas — Lappeenranta University of Technology, Finland
Maya Daneva — University of Twente, The Netherlands
Andrea Hermann — Herrmann & Ehrlich, German

Program Committee

Krzysztof Wnuk — Lund University, Sweden
David Callele — University of Saskatchewan, Canada
João M. Fernandes — University of Minho, Portugal
Andrea Hermann — Herrmann & Ehrlich, German
Andrey Maglyas — Lappeenranta University of Technology, Finland
Maya Daneva — University of Twente, The Netherlands
Timo Knuutila — University of Turku, Finland
Tuomas Mäkilä — University of Turku, Finland
Peter Buxmann — Technische Universität Darmstadt, German
Sami Hyrynsalmi — University of Turku, Finland
Antero Järvi — University of Turku, Finland
Arho Suominen — VTT Technical Research Centre of Finland, Finland

Software Ecosystems: From Software Product Management to Software Platform Management

Slinger Jansen, Stef Peeters, and Sjaak Brinkkemper

Department of Information and Computing Sciences
Utrecht University, the Netherlands
{s.jansen, s.brinkkemper}@cs.uu.nl; stefpeeters86@gmail.com

Abstract. Presently, it is impossible to use software product management practices and tools for software platforms that operate in software ecosystems. The extensive and mature Software **Product** Management Competence Model cannot easily be applied in this context. In this paper the Software Product Management Competence Model is ported towards keystone players in software ecosystems, to create the new Software **Platform** Management Competence Model. If a keystone player implements the capabilities described in these new tools, it can manage stakeholders, know and align their interest, and thereby further enable value creation by itself and ecosystem members.

Keywords: Software Product Management; Software Ecosystems; Keystone; Partners; Directed Approach; Software Platform Management

1 Introduction

When software products grow larger, external parties may wish to extend products to create niche solutions for niche markets. The product grows to a platform on which third parties build extensions, components and applications. Iansiti and Levien (2004) define a platform as a set of standard services, tools, and/or technologies that function as resources for other members. In this case, the software company and its platform provide core technology that function as a basis for a Software Ecosystem (SECO). Jansen, Finkelstein and Brinkkemper (2009) have defined a SECO as: "*a set of actors functioning as a unit and interacting with a shared market for software and services, together with the relationships among them.*"

If a software producing organization is going to manage its product *as a platform* it is taking a directed SECO approach. Cusumano (2010) presents several reasons for software companies to take a directed SECO approach, such as when (potential) customers come from a plethora of niche markets. Due to time constraints and limited R&D investment budget, satisfying different types of customers with software that fit their needs cannot be solely done by one keystone software provider. Expanding the product to a platform on which externally created components or applications can be built is an instrument for creating the required niche functionality and is a basis for the creation of a SECO.

Software Product Management (SPM) plays a key role in product software organizations in creating successful products. Organizing SPM well increases the quality of the developed products. Researchers have instantiated many initiatives to improve the activities with regard to SPM. The results of these studies have led to an approach to organize the management of requirements, releases, products and portfolio of products; i.e. a complete framework with all relevant management practices (van de Weerd, Brinkkemper, Nieuwenhuis, Versendaal, & Bijlsma, 2006).

Taking a SECO approach affects how companies look at their SPM. Those software companies that are able to manage their software platform, relationships and surrounding environment in a successful manner create a sustainable and profitable business for themselves and their stakeholders. The current SPM framework already contains practices that take into account external parties, such as *"Monitor Partner Network"*. The current model, however, does not sufficiently accommodate companies that take a directed SECO approach, as it lacks, for instance, the identification of SECO player types, the use of multiple distribution channels (app stores), and the certification of partners to develop for and on top of the platform.

According to Bosch (2009) the keystone (i.e. platform developer) can take a SECO approach ranging from directed to undirected. If a keystone takes a directed approach it identifies specialized market segments (i.e. niche markets) to offer solutions for and collaborates with third parties to develop domain specific functionality. In the undirected approach every external party that wishes to build new functionality on or with the offered platform can do so without permission of the keystone. This study focuses on SPM for software producing organizations with a directed SECO approach. To determine how SPM with a directed SECO approach (i.e., Software Platform Management) needs to be organized, we investigated the adequacy of the instruments of the state of art of SPM, and if it is not, what needs to be changed. The instruments are:

- The SPM Competence (SPMC) Model: a framework which presents an overview of all important areas and practices for SPM (Bekkers, van de Weerd, Spruit, & Brinkkemper, 2010; van de Weerd et al., 2006).
- The SPM Maturity (SPMM) Matrix (Bekkers & van de Weerd, 2010): a maturity matrix in which all practices of the SPMC Model are presented in a best practice order for implementation. The SPMM is not presented in this paper for reasons of brevity.

We have found several clues that suggest that the model and matrix are not adequate for application in a company that is a keystone player. First, partner requirements in SECOs may be of a higher priority than customer requirements. While a customer only represents the value of one customer, partners usually have many customers and thus represent a larger possible value for the keystone organization. Second, it may affect release planning because the configurations of features in new releases have consequences for the externally created components. To synchronize releases with partners, the creation of widely accepted release definitions is essential. Third, besides a roadmap for the platform of the keystone organization each partner may have a roadmap for its solution(s). If the roadmaps in the SECO are not aligned correctly, it can lead to major problems. For example, the vision for the platform

might not be reconcilable with solutions created by partners. On the other hand, the platform company does not want to get in a 'release stasis' as well, in which it does not dare to innovate its platform. Fourth, the externally built solutions may compete with other products built by the organization. These problems and many more may arise when taking a directed SECO approach, which are explored.

The paper continues with a description of how design science was used for the research design. Furthermore, interviews and a questionnaire were used to find legitimate candidate changes for the tools and models. In section 3 and 4 related literature on the topics of SPM, the SPMC Model, the SPMM Matrix and SECOs is described. In section 5 the results of the study are presented, with an emphasis on the SPMC Model. Section 6 discusses the results and shows the weaknesses in the research approach in terms of validity. In Section 7 is concluded that the SPlaM is future proof and enables keystone players in software ecosystems to better manage their platform within the complete ecosystem. Furthermore, future research topics are suggested that further develop the maturity models in the domain of software platform management.

2 Research Approach

For this study is chosen to use the Design Science Research Cycle as described by Takeda, Veerkamp, Tomiyama, and Yoshikawa (1990). Five steps are defined that represent the design research process. The five steps are: 1) awareness of the problem, 2) suggestion, 3) development, 4) evaluation and 5) conclusion.

First, during the step 'Awareness' a relevant problem to conduct a study was found. Second, during the step 'Suggestion' was suggested what solutions could possibly solve the stated problems. A literature study on the topics of the SPMC Model, the SPMM Matrix, SPM, SECOs, and activities related to these research domains was performed.

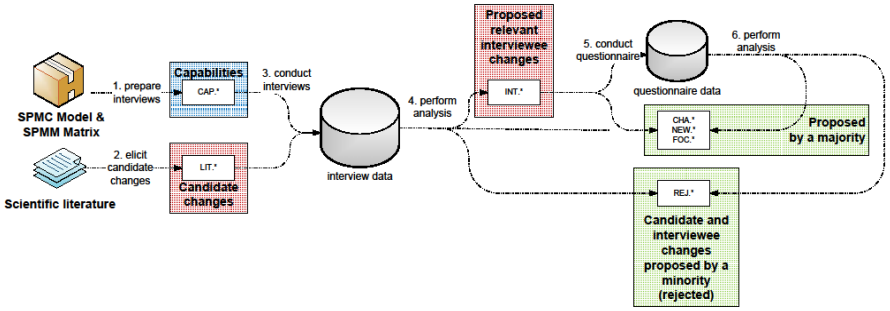

Fig. 1. Research steps in the development of the SPlaM.

The literature study was used as the knowledge base on which eleven structured interviews with Dutch product managers were conducted. The interview data were classified per topic within the SPM and SECOs literature, as to collect as many facts about specific SPM practices. The interviews consisted of an introduction, some generic questions about software platforms and platform management, a deep-dive into the SPM model, and a wrap-up. During the discussions surrounding the SPM model, proposed changes were documented carefully. Interviews took an average of 2 hours, mostly because of the lengthy discussions about the SPM model. Interviews were typically interrupted by a coffee break.

Table 1. Interviewees and experience.

#	Experience in				
	RM	**RP**	**PP**	**PM**	**Gained at**
i1	x	x	x		Ordina, Infra Design, Unisys
i2	x	x	x	x	PinkRoccade
i3	x	x	x	x	Everest
i4	x	x			NetAspect
i5	x	x	x	x	ThinkWise
i6	x	x			AFAS Personal, Yunoo
i7	x	x	x	x	Everest
i8			x	x	ANVA, Cap Gemini
i9	x	x	x	x	BackBase, SDL Tridion, Pallas Athena, Data Distilleries
i10	x	x	x	x	Everest
i11			x	x	Everest
Sum	**9**	**9**	**9**	**8**	

Third, during the step 'development' the data of the interviews was used to create an overview of all proposed changes from literature and from the interview data. This resulted in several pages of proposed changes, which were used to create a draft version of the SPlaM. In those cases where proposed changes were conflicting, the interviewees were contacted to iron out differences. Practically all definitive changes in the resulting model were made based on what the majority of the product managers wanted to see changed. However, some minor and relevant changes (e.g. expanding the examples in the description of a SPM activity) were performed without a majority vote from the software product managers.

Fourth, during the step 'Evaluation', the previous interviewees were asked to evaluate the new model in a pen-and-paper survey where the researcher was present. Each of the proposed changes was evaluated. Furthermore, the main evaluation questions were:

- Which candidate changes are not relevant for a directed SECO approach?
- Which candidate changes still require additions or clarifications? Are they made in the right places of the SPM model?
- Which candidate changes are still missing in your opinion?

As the model had been discussed before with the interviewees, the evaluation did not lead to heated discussions. Finally, during 'Conclusion' step, the study is concluded by presenting the study results to the scientific community. An overview of the interviewees' experiences in the different fields can be found in Table 1 and shows an even spread across the four disciplines in SPM.

3 Software Product Management

The capacity of a software product to satisfy the needs and expectations of stakeholders determines its quality (Berander, 2007). Therefore Berander (2007) states, a software producing organization needs to gather, select and plan the right set of features for a product to find the highest value for all stakeholders. However, SPM covers more responsibilities. SPM concerns the definition of a strategy for, distribution of, launching of, providing support for, and phasing out of a product; i.e. all phases of the life-cycle of product software (Ebert, 2007). The same author says about SPM that it assures ´winning´ products by implementing business cases, agreeing on and implementing marketing, creating functional and technical roadmaps, managing product life-cycles, and aligning and optimizing the organization' product portfolio.

The SPM Competence Model (Bekkers, van de Weerd, Spruit, & Brinkkemper, 2010; van de Weerd et al., 2006) gives an overview of the key areas of SPM. Its objective is threefold: aiding software companies in organizing and enhancing their SPM, structuring education on SPM and structuring research on SPM. The model consists of four business functions: Portfolio management, Product planning, Release planning and Requirements management. Its structure is chosen because a software producing organization possesses a portfolio of products, which consists of one or more products, which has multiple releases, and a release represent a set of requirements. Development activities are not part of the model, simply because it is not part of SPM. Each business function consists of a highly cohesive group of focus areas, which in turn represent a group of highly cohesive capabilities (i.e. relevant SPM practices). The external stakeholders are the: market, customers, and partners. The internal stakeholders are the (business units): company board, sales, marketing, research & innovation, development, support, and services.

The SPM Maturity (SPMM) Matrix (Bekkers, van de Weerd, Spruit, & Brinkkemper, 2010) is based on the SPM Competence Model; the matrix has the same structure (i.e. the business functions) and the same components (i.e. the focus areas and capabilities). However, the capabilities on which the focus areas are based are spread in a best practice order for implementation over several maturity levels. In this way, product managers and software organizations can determine how mature their SPM organization is (i.e. the level corresponding to their capabilities) and what the areas of improvement are (i.e. the missing capabilities corresponding to the desired level).

4 Software Ecosystems

The term Software Ecosystems (Jansen et al., 2013) and their underlying theory are based on biological ecosystems. The biological ecosystem is the result of the interactions between its members and the physical environment (Dhungana, Groher, Schludermann, & Biffl, 2010). SECOs involve the organization of its members (i.e. software vendors, third-party developers, suppliers and users) and its platform (J. Bosch, 2009).

Taking a directed SECO approach implies that the platform developer takes a community perspective, in which internal and external stakeholders are taken into account (Fricker, 2010). A well-known example of a successful ecosystem is Apple with its Appstore. The large quantity and quality of the product software (apps) offered in this store could not be devised and produced by Apple on its own. The success of this and other ecosystems lie in the opportunity for a large set of developers to use the platform to create and distribute software.

Two key types of members in SECO literature are recurring (van der Schuur, Jansen, & Brinkkemper, 2011); the keystone and the niche players. Iansiti and Levien (2004) state the keystone is: "…*a benevolent hub in the network that provides benefits to the ecosystem and its members*." They say it typically gives other members (i.e. niche players) the necessary space to grow and prosper and niche players (i.e. the other members of the network) do not try to compete with the keystone. They leverage the resources of the network to create solutions which are targeted at niche markets (i.e. specialized segments of the market). Thus, the keystone creates and delivers a keystone product (i.e. the platform) and surrounding services, which enable niche players to create and deliver niche solutions.

5 Software Platform Management

Many authors use the term platform when they talk about the keystone its product (and surrounding services) that is provided to enable other members to create value (e.g. in Geir K., 2011; Iansiti & Levien, 2004; Iyer, Lee, & Venkatraman, 2006; Kilamo, Hammouda, Mikkonen, & Aaltonen, 2012). The keystone opens its product to external entities to create a platform by which business and SECO objectives can be reached. Thus, management needs to be focus on how the keystone and other members of the SECO can create value. Kittlaus and Clough (2009) named this approach platform planning. Companies that conduct successful platform planning will realize several benefits (Robertson and Ulrich, 1998), they will:

- have a greater ability to create niche products for niche markets or customers;
- lower the costs to reach niche markets or customers;
- create niche products that more closely meet the needs of them.

We define *platform management* as consisting of four processes: Portfolio management, Product planning, Release planning, and Requirements management. Since

these platform planning processes are similar to those in the SPM framework, the framework under development is called the Software Platform Management (SPlaM) framework.

5.1 Candidate Changes

Based on the literature candidate changes for the SPMC Model are defined, classified in the following topic groups: foster the sharing of resources, manage the involvement of partners, manage the communication of requirements, raise the quality by means of certification, initiate and manage (new) partner relationships, create a healthy SECO product portfolio, and initiate and manage (new) SECO (sales and distribution) channels. A large number of candidate changes were identified. For example, based on literature on the topic of "foster the sharing of resources", a candidate change is made: the *identification of core assets* process is expanded to include core assets created by third parties as well (see Table 2). Imagine, for instance, the role of the Facebook application on the iPhone, which can be considered a core asset in the iPhone ecosystem, even though it was not released and developed by Apple.

Table 2. Candidate change core asset identification.

focus area	candidate change	processed as
Core asset roadmapping	Common components/functionality (core assets) is systematically identified among the ecosystem's products and deliverables surrounding these products.	Expand capability b. Core asset identification with this candidate change or add it as a new capability.

5.2 SPlaM: The New Model

As explained earlier the current model and matrix consists of four business functions: Requirements management, Release planning, Product planning and Portfolio management. No (new) business functions are added or removed. Although the high level business functions are not changed, new focus areas and capabilities are added and existing focus areas and capabilities are changed. The following tables (**Table** and **Table**) describe what changes were made to the model. In the 'part' column is described what is changed; it can be the name of a new or changed capability or the description of a focus area. In the 'c/n' column is described if it is a new (i.e. 'n') or changed (i.e. 'c') capability or focus area. For the sake of brevity **unchanged** capabilities, **unchanged** focus areas and (**changing**) maturity levels are not presented. For more information on the unchanged capabilities and focus areas, please see Bekkers and van de Weerd (2010). All changes made to (the description of the) focus areas are based on the changes made with regard to new and changed capabilities.

A final remark has to be made about the *Partnering & contracting* focus area of the current SPMC Model. Due to the fact that nine new capabilities are added to it, it is split up into three new focus areas. The three new focus areas are *Contracting*, *Partnering* and *Channel development*. Three of the five capabilities of the 'old' *Partnering & contracting* focus area are unchanged and therefore not described in the

following tables. The unchanged capability *Intellectual property management* is added to the *Contracting* focus area and the unchanged capabilities *Establish and evaluate pricing model* and *Investigate distribution channels* are added to the Channel development focus area. The interviewees mentioned several interesting observations while proposing changes to the SPMC model.

One specific aspect that was frequently discussed is partner trust: "*There exists a significant danger in opening up the requirements database to anyone."* Furthermore, it was added that the type of access is relevant: *"Even if you intensely collaborate with partners, you don't want them to be able to delete requirements from your database.".* Also, trust does not only play a role between the organization and its partners, but also partners among themselves: *"It's utopian to think that partners will not compete amongst themselves. You can try all you want, there is always competition and you do not want to be an arbiter in an endless fight."*

Another aspect mentioned frequently is transparency: *"Sometimes you don't even know your customer base is interested in a specific feature until one proposes it and others get to comment on the feature. Sometimes the customers themselves don't even know until they see the idea from another customer."*

The most positively evaluated addition to the SPMC model is the certification of partners: *"It provides stakeholders with an unmatched transparency. Customers will know that this is a trusted party and that their extensions are at least ok'd by us."* Certification is also seen as a marketing tool: *"Partners can go through several phases: from unknown to registered to certified to preferred. They can even use this for marketing purposes, every promotion is a sign that the partner is a little higher on the ladder to partnerhood."*

Table 3. Part one of the performed changes table.

focus area	part	c/n	description
Requirements gathering	[description]	c	Expanded with the sharing of requirements with relevant and authorized external stakeholders.
	Opening central database	n	The central database with incoming requirements is opened for relevant and authorized external stakeholders. It must foster the sharing of resources between SECO members.
	Stakeholder involvement	c	A combination of three pre-existing capabilities; i.e. the *Internal stakeholder involvement*, *Customer involvement* and *Partner involvement* capabilities. In it all relevant internal and external stakeholders are involved by gathering their requirements. Plus, per product or release is determined which stakeholder involvement is most important. Leading to the involvement of the right stakeholders.
	Requirements communication flows	n	The requirements communication networks are modeled and analyzed to determine the proper communication tactics for requirements.
Requirements identification	*External feedback*	n	Extra feedback on product requirements is gathered from external stakeholders. It raises the quality of product requirements by enriching its content.
Requirements organization	[description]	c	Expanded with sharing the gathered requirements with all relevant internal and external stakeholders.
	Requirement organization	c	Requirements are organized on shared aspects and requirements for externally build products are recognized and communicated to the specific external developer (i.e. partner). In this way, partners get all the relevant information they need to improve their niche solutions.
	Opening requirements history log	n	Make the history log accessible to relevant and authorized external stakeholders. Requirements will be reusable for other external projects and thereby fosters the sharing of resources within the SECO.
Requirements prioritization	[description]	c	The prioritization of requirements is only performed by relevant stakeholders.
	Internal stakeholder involvement	c	Still all relevant internal stakeholders indicate the requirements that should be incorporated in future releases. However, for each stakeholder is determined how important their involvement for the product is.
	External stakeholder involvement	c	A combination of two pre-existing capabilities; i.e. the *Customer involvement* and *Partner involvement* capabilities. In it, all relevant external stakeholders are involved by prioritizing the requirements and per product is determined which stakeholder its involvement is most important. Leading to the incorporation of the right requirements.
Release definition	*Communication*	c	The original *Internal communication* capability is expanded with external communication (i.e. communicating the release definition to external stakeholders as well). Now, partners will know what will be developed and they can prepare and/or improve their niche solutions based on the new or changed features.
Release definition validation	[description]	c	Expanded with external parties.
	External validation	n	The release definition is checked by external stakeholders as well. It creates a better alignment with externally created products, increases its quality, and generates awareness among the external stakeholders.
Roadmap intelligence	*Legislation*	n	Continuously an overview needs to be created with regard to changing legislation for the organization its product industry in order to keep compliant with laws and regulations.
Core asset roadmapping	[description]	c	Widened to the whole SECO.
	SECO core asset identification	c	The original *Core asset identification* capability is expanded to all products created in the SECO. Core assets are systematically identified among and surrounding the deliverables of SECO's products, because it increases and simplifies the reuse and maintenance of SECO its core assets.
	Make, buy or co-creation decision	c	The original *Make or buy decision* capability is expanded with co-creation decisions. A process needs to be in place to actively investigate make, buy or co-creation decisions, because costs can be reduced and time can be saved by using and/or co-create with external parties.
Product roadmapping	*Theme identification*	c	Release themes are identified and maintained together with relevant internal and external stakeholders for internal and external creation. It is expanded with external stakeholders, themes and creation, because external parties (i.e. partners) are going to develop the new value in a SECO.
	Consultation	c	The original *Internal consultation* capability is expanded to external stakeholders. Relevant internal and external stakeholders are consulted for the creation of a product roadmap. To have SECO wide acceptance of the product roadmap, to use the knowledge of all relevant members and to create richer and more realistic product roadmaps.
	Long-term roadmap	c	A long-term roadmap is created that spans a time period of maximum two years. The time span is shortened, because the software industry is changing so fast it is not possible to create valuable roadmaps that span more than two years.
	Roadmap procedure	n	A decision procedure has to be defined to make partners aware what will happen if no consensus is reach between the keystone and them in the future plans for the platform.

Table 4. Part two of the performed changes table.

Market analysis	*Market trend identification*	c	There is an active search for market opportunities to expand existing or create new products for, by doing market research at all kinds of places (e.g. related markets and visiting conferences). It is expanded by adding market research via the use of information gathered from partners, because in SECOs the keystone closely collaborates with the partners.
	SECO customer win/loss analysis	c	The original *Customer win/loss analysis* capability is expanded to all products in the ecosystem. A win/loss analysis is performed to determine why customers (of partners) did or did not choose to buy SECO products (i.e. the sales process is reviewed). To learn more about how to generate more customers by tuning the development of the platform.
Contracting	[description]	n	Focuses on establishing relations with external stakeholders by creating proper and clear agreements with them.
	Service level agreements	c	SLAs are set up for customers and partners. It is expanded to partners since they will ask for specific services on which agreement have to be made.
	Contract negotiation process	n	A contract negotiation process is set up in which (e.g.) realistic objectives, agreements on earnings, intellectual property rights, termination clauses, penalties for bad performance and arbitration procedure are determined.
	Determine information profiles	n	Information profiles are determined for each (type of) partner(s) (according to their role), it makes clear which partner has access to which information to simplify the sharing of information.
Partnering	[description]	n	Focuses on managing relations with external stakeholders and supporting them in creating the biggest possible value for the ecosystem.
	Register partners	n	All partners are registered in a central database which all (relevant) internal stakeholders can access, to create an overview of all partners and share knowledge (e.g. best practices and experiences) with regard to the partners.
	Set up partner network	c	The original *Monitored partner network* capability is split up in this capability and the new capability *Partner performance analysis*. Partner networks and/or portals are used to regulate and promote partnering.
	Cluster partners	n	Partners are clustered into groups with specific goals, functions, etcetera to simplify the management of them.
	Coordinate partner alliances	n	Partner(s) (alliances) are coordinated to avoid conflicts and to foster synergy to create a stronger and more coherent SECO.
	Partner performance analysis	c	The original *Monitored partner network* capability is split up in this capability and the new capability *Set up partner* network. A partner analysis is performed on an organizational level to analyze what partners have to offer, what their strengths and weaknesses are, and are going to offer. To create a clear and correct picture of the performance of partners which is the basis on which decisions can be made about maintaining or ending partner relations.
	Certify partner	n	Partners are certified divided over different ranks with different obligations and privileges to make clear what is expected to raise quality.
	Certify external components	n	Certify external created components on standard quality rules to raise the quality of niche solutions.
Channel development	[description]	n	Focuses on establishing and managing distribution channels.
	Common delivery channel	n	Set up a common delivery channel (e.g. the Apple Appstore) to enable partners to sell their products to a large customer base. It makes the SECO more attractive for new partners and customers.
	Model the SECO	n	Model the SECO at its different levels (e.g. within and between SECOs) to identify distribution channels, main competitors and potential partners.
Product lifecycle management	[description]	n	Widened to the entire SECO
	SECO product lifecycle analysis	c	The original *Product lifecycle analysis* capability is expanded to the whole ecosystem and by using information from external stakeholders as well. At least once per year the current life phase of each product in the SECO is determined based on technical and financial aspects. Plus, information is gathered from internal and external stakeholders. It is important to determine if the keystone still want to support the creation of certain niche solutions and therefore external stakeholders have to provide information on externally created products.
	SECO portfolio scope analysis	c	The scope of the original *Portfolio scope analysis* capability is widened to all products in the SECO. A product scope analysis is performed to identify overlaps and gaps between the products in the whole SECO, because it is important to create a healthy (i.e. diverse) SECO product portfolio.

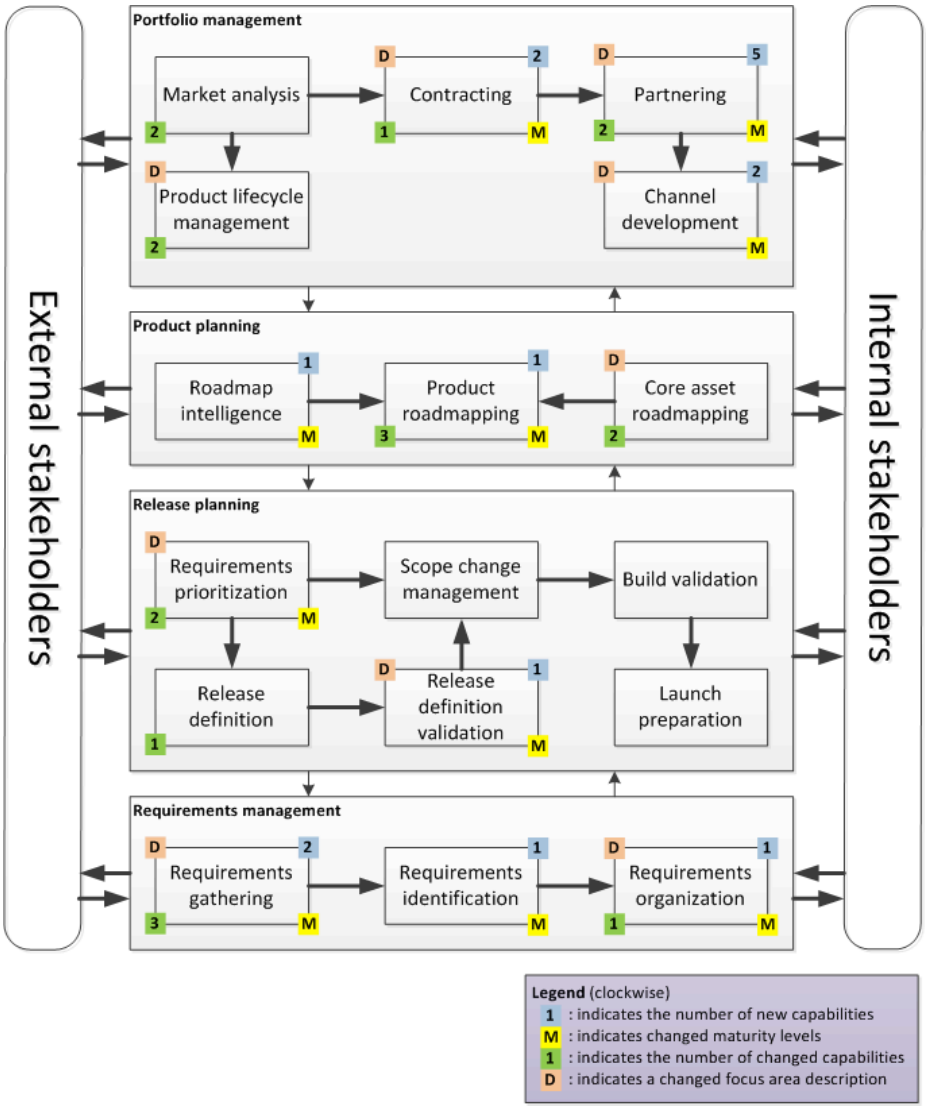

Fig. 2. The Software Platform Management Competence (SPlaM) Model.

In **Fig.** 2 the SPlaM Model is presented. The only changes that are visible in this figure are the three new focus area *Contracting*, *Partnering* and *Channel development*. The three focus areas are a substitution of the *Partnering & contracting* focus area in the old model. Per focus area is indicated if changes are made with regard to its description, new capabilities, changed capabilities and maturity levels. First, at the upper left corner is indicated in orange with the letter 'D' if its description is changed. Second, at the upper right corner is indicated in blue how many new capabilities are add to it. Third, at the lower left corner is indicated in green how many capabilities

are changed. Fourth, at the lower right corner is indicated in yellow if maturity levels of capabilities are changed.

6 Discussion

To solve validity issues, tactics with regard to content validity were studying the instruments of the state of the art of SPM (i.e. the SPMC Model and SPMM Matrix), performing a literature study on the topics of SPM, SECOs, and activities related to these research domains, to now every relevant facet of the object under study. Tactics with regard to construct validity were using structured predefined approaches in which no room for own interpretations was possible during the gathering and analysis of the data. To ensure the product managers knew if and how they had to make changes, an introduction on the model, SECOs and the research objective was given with an introduction that was read to each interviewee. Tactics with regard to external validity are the use of multiple product managers from multiple companies as the developers and validators of the new model. A reduction of external validity results from the fact that only Dutch product managers are used as developers and validators. Thus, the question remains if the result of this study is generalizable to other countries and/or cultures. Fourth, the result is reliable because for every activity a predefined and structured approach is used. Thus, if this study were to be replicated, we expect it to lead to the same output.

During the interviews some of the product managers questioned whether there should be so much emphasis on the management of partner relationships within SPlaM. They indicated partnering activities are more appropriate at a partner management department. We have chosen to include these activities for the following reasons. First, other product managers did add new capabilities in regards to partnering activities. Furthermore, the current model already contains partner management capabilities. Thirdly, partners of keystone organizations with a directed SECO approach add value and knowledge to the ecosystem. Thus, a product manager needs to be deeply involved in partner management activities. Also, Fricker (2010) states: "*Software product management establishes and maintains a software ecosystem by managing stakeholders and studying and aligning their interests.*" Bekkers, van de Weerd, Spruit and Brinkkemper (2010) also indicate that the product manager is located at the center of the company; from its position she needs to keep contact with every relevant stakeholder to collaboratively reach goals derived from (business) strategy. Finally, Ebert (2007) describes that the product manager needs to find a balance between the needs and wishes of external entities (i.e. customer, markets and stakeholders) and guide them in the right direction.

7 Conclusion and future research

The interviews resulted in fourteen new capabilities, sixteen changed capabilities, nine focus areas with changed maturity levels and nine focus areas with changed de-

scriptions. During the interview analysis, a selection was made of changes suggested by two or more product managers. These changes are presented to and assessed by the product managers by means of the questionnaire. The questionnaire resulted in two new capabilities, three changed capabilities and one focus area with changed maturity levels. The new, changed and unchanged capabilities and focus areas describe how keystone software organizations should organize their SPlaM practices with a directed approach. Thus, a Software Platform Management Competence (SPlaMC) is developed and validated.

The limitations of this study can be the initiation for further research. First, new studies could focus on what the relevant SPlaM practices are for software vendors that take an undirected SECO approach. Second, future research could focus on the matrix specific maturity levels and prerequisites of capabilities of the SPlaMM Matrix.

References

Bekkers, W., & van de Weerd, I. (2010). *SPM maturity matrix* (UU-CS-2009-015). Utrecht: Utrecht University.

Bekkers, W., van de Weerd, I., Spruit, M., & Brinkkemper, S. (2010). A framework for process improvement in software product management. In A. Riel, R. O'Connor, S. Tichkiewitch & R. Messnarz (Eds.), *Systems, Software and Services Process Improvement* (pp. 1-12). Berlin Heidelberg: Springer.

Berander, P. (2007). *Evolving prioritization for software product management.* Blekinge: Blekinge Institute of Technology Doctoral Dissertation Series.

Bosch, J. (2009). From software product lines to software ecosystems. *Proceedings of the 13th International Software Product Line Conference,* San Francisco, CA, USA, 111-119.

Cusumano, M. A. (2010). *Staying Power: Six Enduring Principles for Managing Strategy and Innovation in an Uncertain World.* Oxford University Press, USA

Dhungana, D., Groher, I., Schludermann, E., & Biffl, S. (2010). Software ecosystems vs. natural ecosystems: Learning from the ingenious mind of nature. *Proceedings of the* 4*th European Conference on Software Architecture,* Copenhagen, Denmark. 96-102.

Ebert, C. (2007). The impacts of software product management. Journal of Systems and Software, 80(6), 850-861.

Fricker, S. (2010). Requirements value chains: Stakeholder management and requirements engineering in software ecosystems. In R. Wieringa, & A. Persson (Eds.), *Requirements Engineering: Foundation for Software Quality* (pp. 60-66). Berlin Heidelberg: Springer.

Geir K., H. (2011). A longitudinal case study of an emerging software ecosystem: Implications for practice and theory. *Journal of Systems and Software, 85*(7), 1455-1466.

Iansiti, M., & Levien, R. (2004a). *The keystone advantage*. Boston: Harvard Business School Press.

Iyer, B., Lee, C. H., & Venkatraman, N. (2006). Managing in a small world ecosystem: Some lessons from the software sector. *California Management Review, 48*(3), 28-56.

Jansen, S., Finkelstein, A., & Brinkkemper, S. (2009). Business network management survival strategy: A tale of two software ecosystems. *Proceedings of the 1st Workshop on Software Ecosystems,* Falls Church, Virginia, USA, 34-48.

Jansen, S., Cusumano, M., Brinkkemper, S. (2013) Software Ecosystems: Analyzing and Managing Business Networks in the Software Industry. Edward Elgar Publishers, 350 pages.

Kilamo, T., Hammouda, I., Mikkonen, T., & Aaltonen, T. (2012). From proprietary to open source—Growing an open source ecosystem. *Journal of Systems and Software, 85*(7), 1467-1478.

Kittlaus, H. B., & Clough, P. N. (2009). *Software product management and pricing: Key success factors for software organizations*, Berlin Heidelberg: Springer.

Robertson, D., & Ulrich, K. (1998). Planning for product platforms. *Sloan Management Review, 39*(4), 19-31.

Schuur, H. W. v. d., Jansen, R. L., & Brinkkemper, S. (2011). The power of propagation: On the role of software operation knowledge within software ecosystems, *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*, San Francisco, CA, USA, 76-84.

Takeda, H., Veerkamp, P., Tomiyama, T., & Yoshikawa, H. (1990). Modeling design processes. *AI Magazine, 11*(4), 37-48.

van de Weerd, I., Brinkkemper, S., Nieuwenhuis, R., Versendaal, J., & Bijlsma, L. (2006). Towards a reference framework for software product management. *Proceedings of the 14th IEEE International Requirements Engineering Conference,* Minneapolis/St. Paul, MA, USA, 319-322.

Lean Product Development in Early Stage Startups

Jens Björk, Jens Ljungblad and Jan Bosch

Software Engineering Division
Chalmers University of Technology
Göteborg, Sweden
[jensbjljenslj]@student.chalmers.se, jan.bosch@chalmers.se

Abstract. Software startups are more popular than ever and growing in numbers. They operate under conditions of extreme uncertainty and face plenty of challenges, underlined by their high failure rate. Using Design Science Research, these challenges were investigated. A literature study showed that in recent years, several authors have suggested ways to increase the odds of succeeding as a startup, such as customer focused development, fact based decision making, pivoting and agile/lean thinking. Interviews with industry professionals showed that few used these Lean Startup practices: many found them difficult to implement in practice. In response, we developed the Early Stage Software Startup Development Model (ESSSDM) for managing early stage software startups by applying Lean Startup principles. The model is novel in that it supports managing a portfolio of product ideas and provides clear criteria for when to move forward with product ideas, when to abandon product ideas as well as recommends what concrete techniques that can be used and when, in order to achieve this. The process was instantiated and evaluated on a startup project in an incubator setting.

Keywords: startup companies, lean product development, software process.

1 Introduction

New software companies are started each day, and emerging technologies such as smartphones, cloud infrastructure platforms and enhanced web development tools have made it even quicker and easier to get started. However, contrary to what the media portraits, far from all startups succeed, i.e. only one in 58 turn out successful [13]. Several authors [1][11][15] argue that this is not only attributable to fierce competition, but to how startups are typically run.

During the early 2000's, agile software development methods [4] gained traction in the software development community. This was followed by increased attention towards metrics-driven development [10] where techniques such as A/B testing are used to base decisions on data instead of opinions. During the same time, Lean Startup concepts gained popularity in the startup community [1][5][13][15]. However, our interviews with nine startups shows that applying these principles in practice proves to be difficult. In this paper, we present the Early Stage Software Startup Development Model (ESSSDM) as a solution. The model has been validated in a startup setting.

The contribution of this paper is twofold. First, it presents a validated process model that manages a portfolio of ideas, whereas existing approaches focus on one idea. Second, the model provides a detailed approach to handling individual ideas with clear stage gates and exit criteria that provides much more guidance than existing approaches.

This paper is structured as follows. First, we outline the research method, followed by background and related work with respect to product development in early stage startups. Then we present the findings from interviews with nine existing startups and the problem statement, followed by the section that presents the ESSSDM model that presents the key contributions of the paper. Next we present the validation of the model using a case study, followed by the conclusion of the paper.

2 Research Method

Design Science Research (DSR) was chosen as the primary framework for the research. DSR differs from traditional research in that it focuses on learning through design, i.e. the construction of artefacts. The act of designing is, within DSR, used as a research method or technique [17].

Takeda, et al. [17] describes a model of the iterative design cycle. It comprises five phases. (1) Awareness of problem. Research proposal and research questions are formed. (2) Suggestion. Abductive reasoning, drawing from existing knowledge and theory within the field, leads to a suggestion of how to solve the problem. (3) Development. The suggested solution is realized in the form of an artefact. (4) Evaluation. The artefact is evaluated according to defined criteria. (5) Conclusion. When the artefact performs to satisfaction according to evaluation criteria, iteration stops and conclusions are drawn.

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

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

2.1 Research methodology instantiation

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

Suggestion. A generic literature review was conducted, focusing initially on agile practices and in later iterations on Lean Startup theory. In addition, semi-structured interviews with industry professionals were carried out. Although an interview guide with template questions was written, structure was kept loose so that discussions were free to go in new and interesting directions. Abductive reasoning based on the literature and the interviews led to a set of problems (see chapter 5) and a suggested solution in the form of a process.

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

3 Background and Related Work

Emerging technologies such as smartphones, cloud infrastructure platforms and enhanced web development tools have made starting a company very easy. However, only one in 58 newly started companies is successful [13]. To understand these challenges, we need to understand what constitutes a software startup. A popular definition, by Eric Ries [15], states that ”a startup is a human institution designed to deliver a new product or service under conditions of extreme uncertainty.” For the purpose of this paper, we narrow the definition by including the following characteristics: startups have limited resources in terms of people and funding. Consequently, startups run on tight schedules and are exploratory in nature, initially lacking clear requirements, customers and even business models.

3.1 Agile software development

Over the last decade, several agile software development methods have been developed, but Scrum [16] is one of the most popular agile development processes, and is founded on empirical process control theory. Empiricism states that knowledge comes from experience and that decisions should be made based on what is known, not on what is believed. Empirical process control theory is a way to deal with ”imperfect processes that generate unpredictable and unrepeatable outputs” by

prescribing frequent inspection and adaptation. In Scrum, inspection and adaptation is applied not only to the software product in development, but to the process as well.

3.2 The Lean Startup movement

Agile development processes are solution focused. That is, they are mainly applied in situations where the problem is well known/understood but the solution is not. In a startup context, however, uncertainty is even greater: both problem and solution are typically not well understood. For the software engineer working in a startup, however, being focused on the solution is often not enough. A product is more than a solution, it is a business model, and in a startup the software engineer is often involved in both business and technical development efforts.

This customer and problem focused thinking has been advocated in the past by people such as Steve Blank [1], John Mullins and Randy Komisar [13], but has in recent years gained traction because of Eric Ries [15] and the Lean Startup movement. Ries noticed that, because of solution focused thinking, a lot of software startups were failing, including his own. Turns out, many were spending time and money developing products that people were not interested in. He calls this achieving failure: successfully executing a bad plan. While projects may have been delivered on time and on budget, and with good design to boost, nobody wanted the product. This underscores the importance of understanding the problem before the solution. While Ries can be credited with coining the term Lean Startup and bringing the word to the masses, his work is heavily influenced by, in particular, that of Steve Blank, who outlined the Customer Development Model in 2005 [1]. Others, such as John Mullins and Randy Komisar, contributed greatly to the field before Ries with Getting to Plan B in 2009 [13]. Likewise, Jason Fried and David Heinemeier Hansson touched upon many similar concepts with the book Getting Real [5].

3.3 Customer Development

In his book The Four Steps to the Epiphany [1], Steve Blank presents the Customer Development Model, which is further explained in his 2012 follow-up The Startup Owner's Manual [2]. Blank argues that the highest risk in building a business is not building the product, but finding people to pay for it. Startups generally do not lack products; they lack customers. Therefore, the traditional product centric development model, where a product is thought of, developed, beta tested then launched is flawed because it ignores customers up until product launch. The Customer Development model, on the other hand, considers customers from the start. It is a structured process for testing business model assumptions (or hypotheses) about markets, customers, channels and pricing. The model consists of four steps, where the first two mark the search for the business model, and the last two its execution.

3.4 The Lean Startup

Ries published The Lean Startup in 2011 [15], wherein he states that entrepreneurship is a form of management. It is fundamentally different from traditional management in that the unit of progress is learning about customers and what they want. And because agile methodologies are not enough for this purpose, Ries brings in Steve Blank's Customer Development Model to fill the gap. Another central concept within The Lean Startup is The Pivot, which is the term Ries uses for when a startup changes direction, but stay grounded in what they have learned (about customers) so far. He claims having pivoted is the most frequently occurring commonality among successful startups: they seldom end up doing what they initially set out to do. By reducing the time between pivots, it is possible to increase the odds of success, before running out of money.

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

Ries suggests treating this process of validated learning as if one were a scientist: by applying the scientific method and thinking in terms of learning experiments. By formulating falsifiable hypotheses (statements that can be proven wrong by empirical data) learning objectives can be defined up front. By running experiments, hypotheses are validated (proved valid/invalid), by analyzing the data typically leading to the formulation of new hypotheses.

The Lean Startup suggests many techniques for speeding up the BML loop time. One of them is building a Minimum Viable Product (MVP). An MVP is typically the first version of a product released to customers, and should contain only the absolute minimum in terms of features and design for it to become viable to the customer, i.e. it solves the customer's problem.

3.5 Running Lean

Running Lean [11] is a rigorous process and handbook for creating Lean Startups based on [1][15]. The process is divided into three steps: (1) document Plan A, (2) identify the riskiest parts of the plan and (3) systematically test the plan. Documenting the initial plan is done in the form of a Lean Canvas, which is Maurya's version of Osterwalder's Business Model Canvas [14]. The Lean Canvas captures and focuses on the entire business model, not only the product/solution. The canvas is a living document, and is continuously updated as the plan iterates from Plan A to a plan that works.

4 Interview Results

Nine software startups in the Gothenburg region were interviewed using a semi-structured format, both from a business and a software perspective. The purpose of these interviews was to get a good understanding of how software startups typically work in the early stages, and if any patterns, processes or best practices could be observed.

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

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

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

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

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

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

Startups are run in many different ways and there are many different types of startups. On the software development side, all companies adhere to agile methods,

but on the business development side, few agreed upon best practices could be observed. What all can agree upon, however, is that it is difficult to work in an organized and structured manner in an early stage software startup.

5 Problem Statement

Lean Startup principles significantly reduce the uncertainty that surrounds startups and increasing their success rate. These principles, however, are rather philosophical in nature and hard to put into practice. This was confirmed by the interviews: entrepreneurs either were not familiar with the concepts or had a hard time implementing them in their companies. Although some authors, e.g. [11], claim to provide a guide for implementing Lean Startup principles in practice, we have identified some key areas where improvements are needed:

1. Existing processes and theories do not adequately support working on, or investigating, multiple product ideas in parallel, as part of an idea portfolio.
2. Existing processes and theories provide insufficient validation criteria for moving product ideas forward through process stages.
3. Existing processes and theories give no clear guidance on when to abandon a product idea.
4. Existing processes and theories provide insufficient suggestions of what techniques to use and when, while validating product ideas.

6 ESSSDM

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

The overall process is comprised of two major levels. On the topmost level, managing product idea portfolio, one develops hypotheses concerning potential problems to solve, drafts up rough business models and documents these in a prioritized product ideas backlog. On the second level, a product idea is picked from the backlog and is validated systematically using the Build- Measure-Learn (BML) loop until the product is deemed scalable, put on hold or abandoned.

6.1 Level 1: Managing product idea portfolio

Typically when creating a startup, a product domain is selected and frequently the team has at least one product idea. Independent of the specific initial product idea, the first task is to generate additional promising product ideas. Ideas can be crude; the rest of the process is designed to iterate over an initial (crude) idea and improve upon it. Often, it is worthwhile to investigate multiple ideas in parallel as it allows for more than one person involved in the startup and because it decreases the risk of an overly emotional connection with a specific idea. This increases the odds of finding an idea worth pursuing.

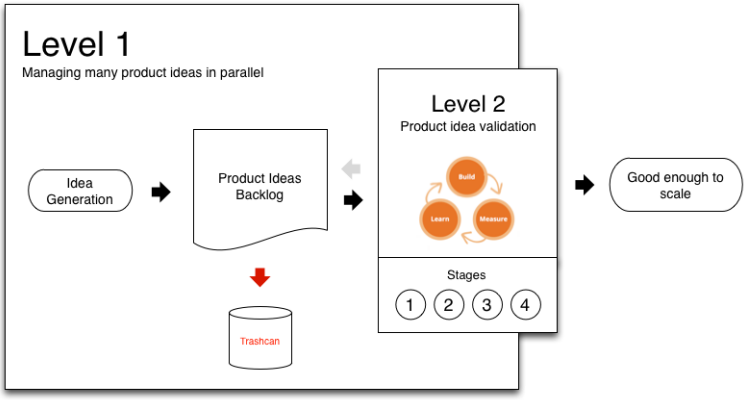

Figure 1: Overview of ESSSDM

6.2 Step 1: Product idea generation

Typically, ideas spawn instantly or emerge over time. A product idea contains, as a minimum, a problem or collection of problems that need solving. One of the primary techniques is to organize exploratory interviews with potential customers in the domain. The selection of interviewees should be influenced by:

- Areas where the entrepreneur has domain knowledge
- Copying and tweaking an existing (successful) product
- Problems experience by the entrepreneur (scratch your own itch) [5]

6.3 Step 2: Documenting product ideas as business models

In order to compare product ideas, it is important to document these in a comparable format, e.g. Lean Canvas [11]. Working on multiple product ideas (as part of an idea portfolio) in parallel provides several advantages, one being that there is always something in the pipeline to work on when one idea is on hold: waiting for interview session dates to arrive or waiting for feedback or other data. However, if working on multiple ideas in parallel, it is important to enforce a limit on how many ideas can be worked on simultaneously. Our validation showed that the number of concurrent product ideas pursued should be below 50% of team size. The product ideas backlog is prioritized continuously, with the most promising ideas actively worked on. Eventually, as one idea gains traction and demands more resources, other ideas should be put on ice until resources become available.

6.4 Level 2: Product idea validation

When an idea is picked from the product ideas backlog, it undergoes systematic validation. This process can be described as a feedback loop comprising risk prioritization and BML looping. The product idea moves through four stages, each with its own activities and defined exit criteria. The four stages are: (1) understanding the problem; (2) defining the solution; (3) qualitative validation; (4) quantitative validation. Each stage is associated with different sets of risks. Risks are prioritized and put on a risk backlog. With risks identified, assumptions (falsifiable hypotheses) can be formed and tested using the BML loop. The learning that is gained from validating the assumptions is fed back into the product idea and the risk backlog. By doing this, risks can be dealt with one by one, through the stages, until the product is deemed scalable, or until a risk becomes blocking and the product idea invalidated.

6.5 Stage 1: Understand the problem

During the first stage of testing the plan, focus lies on gaining a deep understanding about the problem and who experiences it. The key risks during this stage are: (1) do we have a problem worth solving?; (2) who experiences this problem?; (3) what competition is there? Before proceeding to the next stage, at least half of potential customers should give a positive indication of the product idea. Also, the team should identify a promising customer segment, find at least one problem that customers want solved, as well as build understanding of how customers currently solve the problem.

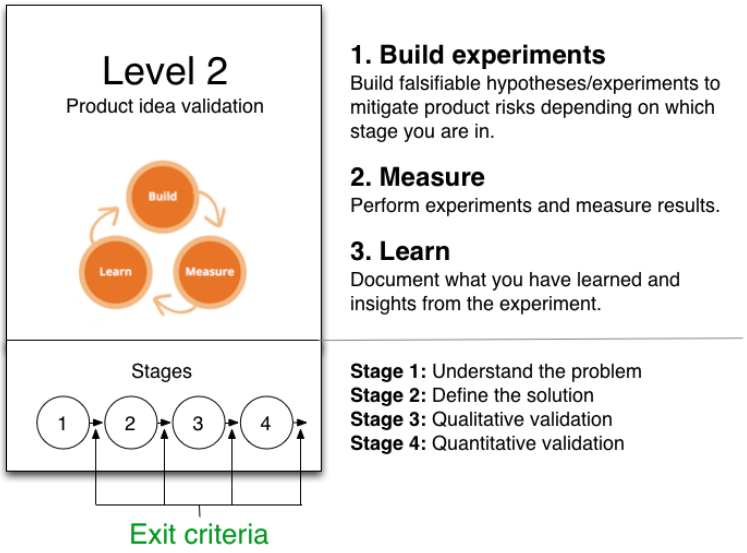

Figure 2: ESSSDM Level 2 process

6.6 Stage 2: Define the solution

In this stage, the purpose is to define a potential solution and communicate this to potential prospects. When defining the solution, the team focuses on the simplest possible solution to solve the problem: the MVP. The key risks during this stage are: (1) what is the minimum feature set for the MVP?; (2) who are the early adopters?; (3) what would customers pay for the solution?

In order to move to the next stage, the team interviews a minimum amount of potential customers, has identified a typical early adopter, has defined a MVP, has confirmed that customers are willing to pay for the product, has verified that the solution is feasible to implement within a realistic time horizon, as well as has secured a test user base for the product.

6.7 Stage 3: Qualitative validation

The purpose of the third stage is to develop an MVP and launch it to early adopters in order to verify that it solves their problem and that they are willing to pay for it. Key risks are: (1) Does the MVP demonstrate a Unique Value Proposition (UVP)?; (2) Are early adopters willing to pay?; (3) How to sustain an influx of early adopters?

6.8 Stage 4: Quantitative validation

During the fourth and final stage the product is launched to a larger group of people, including non-early adopters. The key risks are: (1) Do people want the product?; (2) How to reach customers through inbound channels?; (3) Will the business model hold? This stage marks the beginning of the scaling phase.

7 Validation

ESSSDM was evaluated as part of a startup project at the School of Entrepreneurship at Chalmers University of technology. The following evaluation criteria were considered:

- The process must be perceived as practical by project members
- The process must support working on, or investigating, multiple product ideas in parallel, as part of an idea portfolio
- The process must provide clear guidance on when to move product ideas forward through process stages
- The process must provide clear guidance on when to abandon a product idea
- The process must provide clear guidance on what techniques to use and when, while validating product ideas

7.1 Project context

The project group consisted of five students: three business developers and two software engineers, working in an incubator setting. The incubator provides office space and business advisors. As of this writing, the project has been running for five months, and will continue to run for four more. The focus of the project was to find a promising product idea in the small business segment and turn it into a company.

7.2 Process evaluation

At the beginning of the project, no ideas existed. Doing exploratory interviews with potential prospects made the team both knowledgeable with how small businesses operated (the targeted market segment) and provided many ideas. As more exploratory interviews were conducted, a picture of how small businesses operated in general began to emerge and the team began to focus on certain promising leads. When 40+ exploratory interviews had been conducted, the value of each new interview became less and less, with no new learnings gained, and focus shifted from problem understanding to solution defining.

Project members perceived the process as practical. Based on the amount of people in the team, the team always had ideas to work with and never lost its momentum. Having a cap of three simultaneous ideas worked out well for the size of the team due to the fact that one business developer was responsible for one idea each. When dividing responsibility this way, the importance of having a comparable format (Lean Canvas) became apparent during idea prioritization sessions. Furthermore, using canvases made the team think of the product not as a solution, but as a business model. This gave the team a broader perspective and made it easier to spot the potential in each product idea.

The process gave clear guidance on when to move product ideas forward through process stages. Using well-defined exit criteria for each stage contributed to the team's good momentum and allowed each business developer to work independently.

The process gave clear guidance on when to abandon product ideas. The team constantly evaluated whether exit criteria had been reached or not. When additional interviews resulted in no more learning, and there was no clear path towards fulfilling the criteria, the team took a decision: pivot, persevere or abandon. If there was no obvious way to pivot, the team usually opted to abandon the idea in favor of another one from the product ideas backlog.

The process gave clear guidance on what techniques to use and when for validating product ideas. The proposed techniques proved efficient; no technique took more than two working days to apply. This was valuable in order to push the project forward and get fast feedback from customers.

Concluding, initial validation in the project context suggests that ESSSDM overcomes the challenges discussed in the problem statement.

8 Conclusions

Software startups are more popular than ever and growing in numbers. They operate under conditions of extreme uncertainty, and face plenty of challenges, underlined by their high failure rate. In this paper, these challenges were investigated through a literature study of the Lean Startup community as well as through interviews with industry professionals. The result of this investigation showed that few practitioners apply Lean Startup methods because these were found too vague and imprecise to implement in practice.

In response to the investigation, we developed the Early Stage Software Startup Development Model (ESSSDM) that addresses the identified challenges:

- The process supports working on, or investigating, multiple product ideas in parallel, as part of an idea portfolio
- The process provides clear guidance on when to move product ideas forward through process stages
- The process provides clear guidance on when to abandon a product idea

- The process provides clear guidance on what techniques to use and when, while validating product ideas

In future work, we intend to provide more validation especially to stages 3 and 4 of level 2 of ESSSDM and to apply to the process to additional startups.

References

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

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

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

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

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

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

[7] Gartner, Inc. (2011) Forecast: Software as a Service, All Regions, 2010- 2015. http://www.gartner.com/it/page.jsp?id=1791514 (14 Sept. 2011).

[8] Gustafsson, A., Qvillberg, J. (2012) Implementing Lean Startup Method- ology: An Evaluation, Chalmers University of Technology.

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

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

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

[12] McClure, D. (2007) Startup Metrics for Pirates: AARRR! 500 Hats. http://500hats.typepad.com/500blogs/2007/09/startup-metrics.html (26 Sept. 2007). 56

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

[14] Osterwalder, A. and Pigneur, Y. (2010) Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers, Wiley.

[15] Ries, E. (2011) The Lean Startup: How Constant Innovation Creates Rad- ically Successful Businesses, Penguin Group, London.

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

[17] Vaishnavi, V. and Kuechler, W. (2004). Design Science Research in In- formation Systems. January 20, 2004, last updated September 30, 2011. http://www.desrist.org/design-research-in-information-systems/

Requirements Engineering as a Surrogate for Business Case Analysis in a Mobile Applications Startup Context

David Callele[1], Aubrie Boyer, Kent Brown, Krzysztof Wnuk[2]
and Birgit Penzestadler[3]

[1]Department of Computer Science, University of Saskatchewan
Saskatoon, Saskatchewan, Canada
callele@cs.usask.ca
[2]Department of Comptuer Science, Lund University
Lund, Sweden
Krzysztof.Wnuk@cs.lth.se
[3]Software & Systems Engineering, Technische Universität München
München, Germany
penzenst@in.tum.de

Abstract. Mobile application development operates in a market characterized by low barriers to market entry, short time-to-market and the need for rapid return on investment, making it suitable for exploiting the potential of open innovation. Technology-driven entrepreneurs often diverge from the standard practice of antecedent business case analysis. We report here upon the result of a six-month empirical investigation of this question, performed within an incubator setting, and our analysis of the results indicates a reasonable probability of success, at least for ventures with access to experienced requirements practitioners. Our results indicate that incorporating RE techniques from the beginning of the venture has the potential to reduce the risks associated with the missing business case analysis. The field observations have also identified requirements engineering challenges in this domain worthy of further investigation. In particular, the relative impact of business requirements upon the technology requirements is extreme and requirements methods must respond not only to agile development processes but function even when a pivot (an instantaneous and complete change) in business focus occurs.

Keywords: startup, entrepreneurship, business case, product definition, agile requirements

1 Introduction

Mobile applications development operates in a market characterized by low barriers to market entry, short time-to-market and the need for rapid return on investment (especially when operations are self-financed, a scenario distinct from large-scale efforts financed by venture capital, for example, where the goal is to first build a customer base then attempt to monetize that customer base). The innovations in this market are often acquired or based on freely available frameworks, *e.g.* Google Android, setting these efforts within an open innovation context. Technology-driven entrepreneurs in this domain often diverge from the standard practice [2] of performing business case analysis before beginning new ventures – proposing new ventures based on an inferred market demand that is justified by the project proponent's intuition and belief. Our experience is that business analysts are rarely members of a startup team, particularly in the mobile application development space, and the lack of an antecedent business case analysis can lead to significant investments with low probability of commercial success.

Identifying a viable customer value proposition continues to be a significant challenge in this domain. For example, a recent market study [1] notes:

> **Lack of customer understanding in lean mobile applications development.** We find it remarkable that only 24% of developers in our sample plan their apps based on discussions with users, a figure which does not change with development experience or proficiency. This indicates that the bottleneck of the build-measure-learn cycle of lean development is the "measuring", or understanding customers.

A business analyst is typically responsible for the initial customer contacts (not those contacts that occur during development), assessing the viability of the business case and ensuring that the product requirements represent market needs [2]. Requirements engineers are responsible for translating the product requirements into a requirements specification suitable for guiding a development effort.

We have noted that technology development teams usually receive (at least rudimentary) requirements engineering training during their software engineering courses. We posit that extending RE practice to applying requirements engineering techniques to at least some of the issues usually addressed in the business case analysis would enhance the viability of these entrepreneurial endeavors. When compared to the scenario where these issues are not addressed, this approach enables improved value proposition definition, improved compliance of the product definition with the value proposition and improved focus of development efforts upon the business goals. We expect the process change will improve customer understanding, addressing (at least in part) the issue noted in the market research above.

An empirical investigation was performed in the context of a mobile application entrepreneurship camp held in the summer of 2012 in Saskatchewan, Canada. Entrepreneurship camps are often used by universities and business incubators as a mechanism to foster new business opportunities. The low barriers to market entry in the mobile application space have led to a significant increase in mobile application startups within these camps. The camp reported upon here was conceived and directed by the lead author and grew to become a collaborative effort by the members of the local innovation and technology commercialization ecosystem. Given the diversity of the stakeholders and their goals, the lead proponents made the deliberate decision to apply RE activities to the tasks of stakeholder identification, requirements elicitation, negotiation, prioritization and triage as a management tool for controlling the camp's design – in essence treating the design of the camp as if it were a product design.

We present our experience report on the suitability of using those RE activities noted above (plus requirements scoping, tracing and validation) as a surrogate for business case analysis in the mobile application entrepreneurship context. We found these well-established RE techniques [18][19] to be an acceptable surrogate when business analyst resources were unavailable, at least when guided appropriately via an experienced RE mentor. Our experience also identifies that this environment and application scenario (mobile application startup in an incubator environment) has cases of relatively extreme pressure on the requirements processes and we present our observations and practitioner guidance in Section 7.

In Section 2 we describe the camp and our experience in transforming the camp from concept to reality while Section 3 describes the participant selection process. In Section 4, we describe related work and in Section 5 we describe the camp's design and implementation. Section 6 describes the milestone methodology used in the camp and Section 8 captures our observations and practitioner guidance. We close with our conclusions and directions for future work in Section 9.

2 Mobile App Entrepreneurship Camp

The mobile application entrepreneurship camp focused on providing business skill training to technology-trained aspiring entrepreneurs. The camp advocates used two requirements brainstorming sessions to (1) identify the goals for the camp and (2) identify potential external camp stakeholders (potential camp supporters). Triage was performed on the range of potential camp stakeholders; the stakeholders on the resulting short-list were those with a strong interest in promoting economic development within the community and region.

Despite the common interest in regional economic development, the short-listed external stakeholders had never worked together on a single project before (but subsets had worked together in the past). Achieving their endorsement for the camp was challenging because each entity has divergent mandates. We approached achieving support in this context in the same manner as we would approach requirements negotiation, generally following an iterative process wherein we would make a proposal based on a best-effort synthesis of positions, receive stakeholder feedback, then update the proposal before submission for further stakeholder review.

The external stakeholders began to support the camp once they were solicited for their input both to the test-bed design and to the type of data to be captured throughout the camp. Each stakeholder provided in-kind instructional support on specific topics that were aligned with the mission statement of the instructional stakeholder, providing participants with access to wide-ranging domain expertise. Industrial stakeholders also provided their domain-specific perspectives to the participants and direct participant mentoring as their time and resources permitted.

The final list of camp supporters included six service provider entities related to either the local university or the innovation support community and three local businesses that provided philanthropic support. The service providers included the technology commercialization entity that hosted the camp, an entrepreneurship support and business case development entity, an incubation and office space provider, the local university industry liaison office, the provincial organization responsible for export development and the local office of the federal business regulation compliance and commercial entity support. The local businesses included a mobile game developer, a business strategy consulting firm and a business operations consultant.

3 Camp participants

Camp participants were recruited via a competitive call for participation that was circulated at two local universities and one local college. Applicants were required to pitch themselves as entrepreneurs and their product concept before an evaluation panel. The panel made their decisions based on an assessment of the applicant's ability to meet the following prioritized requirements (most important first).

1. Ability to learn and utilize the course materials, work and education history: Did the applicant have the necessary prerequisites to comprehend the material as delivered? Many applicants simply did not yet have the necessary technical skills and practical business experience to complete the very intense first month of the camp – typically they were too early in their academic career or the evaluators felt that the camp would not be able to provide the requisite level of support to these applicants.
2. Oral and written communication skills: The intense pace of the camp created a risk of failure for participants who could not understand the materials as presented or could not succeed in the 'sales' elements of entrepreneurship.

3. Entrepreneurial potential: Did the applicant provide some evidence that they had an entrepreneurial attitude, the perseverance and the determination necessary to succeed?
4. Product concept: Did the applicant follow the provided guidelines and attempt to present the product in the context of market demand or did they choose to present in the context of a technology push?

Twenty-five projects were proposed by the applicants and four were chosen for the camp. Three of the four projects were led by single participants: a mature developer with college training, a 3rd year electrical engineer and computer science combined program student and a 2nd year computer science student. The 2nd year computer science student also acted as the graphic designer for the camp. The final project was led by a group consisting of three recent graduates from Bachelor and Master computer science degree programs (each with less than 3 years industrial experience). All members of the three-person group were actively employed by third parties throughout the duration of the camp.

Participants in the camp committed to meet the primary requirement to deliver a product that was ready for deployment (or as close as possible) within the appropriate mobile application store and within the constrained time and resources. The primary requirement was supported by a secondary requirement: to develop a supporting business plan and a pitch for third-party investment.

The camp began with product definition techniques that included workshops on clarifying the product value proposition and identifying the minimum viable product. Product definition and feature identification, storyboarding for use-case and scenario analysis, and project scoping followed. Later workshops focused on more business focused aspects such as conducting a commercial opportunity assessment, identifying market segments, general business planning, intellectual property and revenue generation models.

The camp was not an academic exercise. While two of the six participants were still attending university, all projects were real entrepreneurial ventures undertaken in an industrial setting. The total budget for the camp exceeded $100,000 including preparation of instructional materials, workshop delivery, facilities, mentoring, and project investments.

4 Related Work

Given that this work is set in the context of a startup environment, we constrained the literature review to publications that investigated requirements engineering in a startup context or publications that address the delineation between requirements engineering and business analysis roles and how the roles complement each other. We were unable to identify specific related work in this area. The literature review was broadened to include related work for the different aspects of the presented paper with a focus in the interaction of RE with business-related issues. In the area of software startup studies, Ruokolainen and Igel [3] and Burgel and Murray [5] focused on economic success while Mann *et al.* discussed legal issues [3], but none discuss requirements engineering in the startup context. Startups within an academic incubator were discussed by Barbe *et al.* [6].

Seyff *et al.* [7] and Vogl *et al.* [8] present methods for RE in mobile application development, but without consideration of business analysis issues. Aranda *et al.* analyses RE applied in small companies [14], and Gordijn *et al.* explored RE applied to innovative e-commerce ideas [15] and their evaluation [16]. Koivisto and Rönkkö

[9] explored entrepreneurial challenges faced by rapidly expanding small companies, of which startups are an extreme example. Daimi and Rayess [10] argued that the undergraduate software engineering curricula needs to be extended and the need to focus on promoting computational thinking as a source of entrepreneurship, a position supported by Morrogh [11]. Carnegie Mellon's software management Masters program is also focusing on technical leadership within existing companies or within entrepreneurial ventures [12]. Despite these initiatives, there has been little focus on entrepreneurial skills in requirements engineering training.

5 Meeting stakeholder requirements

The stakeholder requirements for the camp were met by an intensive four-month program focused on developing the business skills necessary for entrepreneurial success. The basis for the camp's content was derived from the stakeholder representatives' combined practical experience. The camp was presented as a series of workshops and participant deliverables after each workshop were directed toward advancing their entrepreneurial endeavor. The camp did not provide participants with specific application development technology training, the participants required a minimum technical skill level to participate, but mentors did provide the participants with on-demand technical support and mentorship in mobile applications development.

The participants successfully completed a condensed business plan and the first release of their mobile application by the end of the four month camp. Each participant demonstrated their abilities in the following areas:

- Business startup process, accessing and utilizing business resources
- Requirements elicitation and prioritization
- Finance and accounting
- General and mobile app marketing
- Legal aspects of intellectual property protection
- Project management
- Business and technical presentations
- Public speaking

While each participant began the camp with a product concept, these concepts were at various stages of maturity but all of the concepts were scoped in excess of what the given resources could accomplish. An intensive and heavily mentored effort was undertaken to identify the intended customer (stakeholder identification) as well as their wants and needs (as much as possible given resource constraints) followed by developing a clear description of the associated customer value proposition. An intensive feature prioritization and triage effort was performed by each of the participants, again supported by the mentors and instructors.

Unlike traditional RE where requirements are elicited from a known customer, the camp required the definition of a new product for a projected market. Hundreds of potential product requirements were proposed, reviewed and prioritized and each project postulated numerous use-cases and user scenarios in an attempt to identify core customer needs. These models were evaluated against market segmentation information. Did the requirements (value proposition) hold together for the projected market? Was this a market willing to pay (was there a known pain)? Was the customer able to pay? These techniques supported the development of a minimum viable product definition, discussed further in Section 7.

These initial efforts identified the core customer needs, enabling the participants to define the minimum viable product for their markets, successfully completing an entire

requirements scoping cycle – from elicitation to scope commitment. The process used by the participants assumed that the available resources were sufficient to deliver the revised product since they were working toward a minimum viable product definition. In each case the proposals were reviewed by experienced practitioners to ensure that there was a reasonable probability of success. Finally, the participants captured the requirements for their product in a context-appropriate manner, usually using structured lists and rich text formats, only occasionally using formal statements for major requirements that *must* be met.

Approximately two months into the four month camp, the camp organizers recognized a need to extend the course content to meet the investment readiness goal. Requirements for a two-month supplementary program, focused on preparing an entrepreneurial opportunity for third-party investment, were gathered and the content developed. The supplementary program was focused on providing the participants with the ability to communicate their entrepreneurial goals via a third-party pitch for investment. The three singleton entrepreneurs completed the supplementary program.

6 Project management and methodology

The camp extrinsically motivated participants to meet educational and performance objectives via a milestone payment structure, designed to be similar to disbursement models used by investors in early-stage startups. Each milestone was comprised of several deliverables and participants would only receive a monetary stipend after successfully completing that milestone. Each deliverable within a milestone was introduced to the participants as a requirement. Deliverables for a business analysis element, such as defining the customer's value proposition for the marketing plan were treated no differently than functional requirements for a product or service specification.

The following seven milestones were defined prior to the start of the camp and a summary of their deliverables were presented to the camp participants, in task format, as presented here. The specific RE activities utilized in each milestone are enclosed in parentheses at the end of the description.

M1: Define the Product Identify the market wants and needs and provide a clear definition of the value proposition for the app. Describe the features and explain how they relate to the value proposition. Using persona techniques [13], define a model of the user as an exemplar of the target market. Develop the functional requirements that meet market requirements and prioritize them in a manner that meets the dominant market needs. Identify the resources needed to deliver the project and develop a high-level project plan. (elicitation, negotiation, prioritization, triage, representation)

M2: Refine the Product Further define the target user and the target market. Develop estimates for the size of the target market including market segmentation data and develop first estimates of the revenue potential. Begin development of the revenue model. Finalize the scope of the app and develop an initial task list for both business and software development goals. Develop low fidelity prototypes of the app user interface and begin user testing. Define a software architecture for the app and identify high-risk development tasks. (scoping, prioritization)

M3: Proof of Concept Develop a functional prototype of the user experience for the app. Identify nonfunctional requirements and business constraints such as regulatory compliance and ensure that the app complies as necessary. (quality requirements via elicitation from mentors, research into regulatory requirements)

M4: Complete ALPHA Complete internal testing, mobile applications must be ready for focus group testing with external candidates. Locate and fix design flaws, validate and verify that the product meets the functional requirements and works as intended. (requirements validation using focus groups, traceability to requirements as they have evolved over milestones M1 through M3)

M5: Complete BETA Perform focus group testing and adjust usability as necessary, report on focus group findings. Complete marketing plan and develop sales model, finalize revenue model. (continuous requirements verification and validation)

M6: Delivery Demonstrate that app is ready for submission to app store. (continuous requirements verification and validation)

M7: Investment Pitch Prepare and deliver a third-party investment pitch targeted at the Angel investment community. (elicitation, negotiation, prioritization, triage, representation)

7 Guiding Project Evolution

A project within a startup support environment (such as an incubator) is expected to be a commercially oriented product (or service) with a well-defined customer value proposition. We initially observed that the technology focused entrepreneurs proposed a technology push to a perceived customer problem instead of focusing on a market pull. They expected to use agile development to deliver a rapid prototype, perform customer testing and obtain test market feedback.

We observed customer test plans in these projects that were focused on evaluating functionality and usability but the technology-trained camp participants did not consider investigating the customer's willingness to pay, or customer's ability to pay, during their customer interactions. In the absence of business analyst resources, we recommended to the camp participants that they perform as much primary market research (customer interviews) with the test group as possible.

Project refinement is incremental if the project appears to have the potential to meet a real customer value. However, the project may, for example, be rejected in market testing, as experienced during the camp, resulting in a *pivot* where the fundamental nature of the business changes. Interestingly, the team members usually stay together after the pivot, even when their skills may not be as well-suited to the new business direction as the old.

The observed operational pattern followed by the camp participants is described as "build it, ship it, fix it, monetize it." In other words the entrepreneur's intent is to make a significant development investment in an effort to ensure minimal delay before market entry and it is often the case that a monetization focus does not even begin until after market entry [17]. The described pattern is typical of those reported in the popular press and across the internet but it is a significant financial gamble – the developed product may be "as intended" but a viable monetization model may not exist. We attempted to reduce this risk within the camp by requiring parallel development of the technology and monetization plans.

Many mobile applications expect to generate revenues, not from active monetization via an initial sale of the product or service itself, but by charging small amounts for various extensions as the customer becomes committed to the product. Alternatives for passive monetization include selling access to the customers for advertising and performing data mining upon the users and their information then selling the results to third parties. Unfortunately, in the absence of the antecedent

business case analysis the development efforts may be in vain for, even though the intended product was successfully developed, the market may simply not exist as expected and all monetization attempts essentially fail.

Fig.1 illustrates the observed and desired behavior patterns (the diagram is simplified and many iteration paths are not shown). The right-hand side of the diagram (with graphic elements in black text on white fill) captures the technology-driven pattern observed in all projects on entry to the camp. The project proponent has either a perceived problem or a technology innovation that they believe solves a customer problem (referred to as a technology push) and a solution concept is generated. A prototype is iteratively developed and tested by internal users until such time as the quality is sufficient to test in the market, following the agile paradigm that all projects are prototypes until they ship. Market feedback is obtained and further iterative cycles are performed until product release.

The left-hand side of Fig. 1, with graphic elements in white text on black fill, captures the business case analysis process. A solution concept is proposed for a given market need, supported by evidence that justifies financial investment. The value proposition for the product is iteratively refined until there is sufficient evidence of customer ability to pay and customer willingness to pay. The product definition is revised as necessary before undergoing a competitive analysis. Market position is defined and estimates of the return on investment are generated.

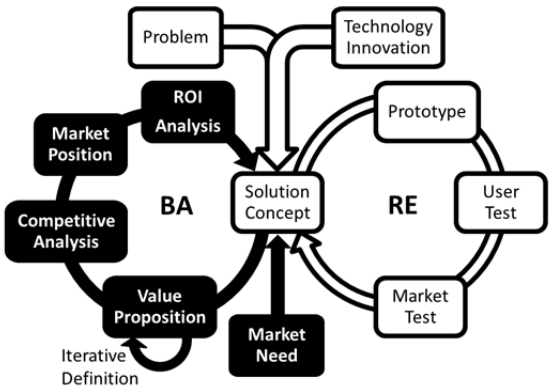

Fig.1. Observed and desired participant behaviors

Iterative refinement of the value proposition leads to the definition of the minimum viable product, also known as the least salable unit. The minimum viable product is intended to capture the core of the customer value proposition, rather than attempting to identify the most accurate representation of the product requirements, and as such represents the minimum subset of possible features for which the customer is willing to pay. This approach requires extensive RE activities to exhaustively identify the requirements for which a customer *might* be willing to pay. These requirements are then analyzed to determine the minimum set of the most important requirements that need to be included in the product for a customer to actually pay for the product/service. Considering a large body of requirements provides some degree of confidence that the global perspective for the minimum viable product has been achieved rather than a minimum viable product for only a specific market segment. These product definition methods are similar to those advocated by Ries [17].

The minimum viable product approach is a natural outgrowth of the intense time-to-market pressures for mobile applications and their low probability of successfully generating revenue. By definition, the minimum viable product requires the minimum viable development effort, and therefore the minimum development investment, maximizing the probability of an overall positive return on investment.

The process for obtaining the value proposition appears to be functionally identical to an extremely rigorous requirements prioritization and triage effort. Rather than using factors such as technological uncertainty and development risk for requirements prioritization, the value proposition investigation intensely cycles between customer ability to pay and customer willingness to pay. Integration of this financial focus into requirements engineering practice may yield substantial customer satisfaction benefits and is an area of interest for further research.

8 Observations and Practitioner Guidance

8.1 When BA is not available, can RE be used as a surrogate for BA?

In a mobile applications startup environment it is likely that a single person will be responsible for performing both BA and RE tasks. In a technology driven startup, it is more likely that RE skills are present therefore we propose the use of traditional RE techniques [18][19] as a surrogate that focus on business viability for the proposed product, particularly in circumstances with significant time-to-market pressures.

In our case, the participants performed RE on various aspects of the business process where in some cases a BA would be better suited. But in this case it would have taken too long to train them to do the work using a BA point of view. Instead, it was more time efficient and effective for participants to use tools that they have already used in the past as a (likely non-optimal) improvement over not performing any type of business case analysis.

As noted earlier, we have observed that the techniques used by business analysts and requirements engineers are quite similar, at least in an abstract sense. However, we have found that applying the RE techniques to the BA domain has significant challenges. The first challenge lies in the area of domain specific terminology – in the same way that business analysts may not understand the technology details we find that requirements engineers are equally challenged by business terminology.

This language barrier is compounded by a lack of subject matter expertise. For example, participants in the camp found the requisite financial analyses to be challenging even though the underlying mathematics were considerably simpler than what they were used to using in their traditional practice; what was once familiar was suddenly challenging. We also observed that the participants had difficulty understanding the concept of market segments and target markets even though these concepts are similar to stakeholder groups. One-on-one coaching was required with each participant in addition to the weekly instructional courses. After much repetition, the participants finally understood what a market segment was and how to identify them.

While observing experienced requirements practitioners performing their mentoring, we noted that they exhibited the characteristics and domain knowledge that we also associated with business analysts. When the participants worked with their mentor we observed significant improvements in their output quality and reductions in

their training time, illustrating the importance of domain knowledge and providing further evidence of the cross applicability of the techniques between domains. Some of the mentors observed that entrepreneurs might be able to successfully use task checklists, constructed by experienced personnel, to guide their efforts and complete these tasks if experienced personnel are not available.

Another challenging area is the validation of business requirements. When validating a requirement, particularly when using agile methodologies, practitioners simply query the customer representative. Entrepreneurial new product development efforts do not have a customer representative; successfully introducing an innovation that meets market needs, in the absence of the customer, is uncertain at best. For example, the iPhone had no real customers but was a dramatic market success – very few gambles like this provide such a positive return on investment. Business constraints, such as regulatory issues, also proved challenging for there was no obvious way to capture and represent these nonfunctional requirements in a lightweight manner.

Finally, defining the functional requirements for a product differs greatly from validating the business case for a proposed product concept in a proposed market. The participants' ability to perform functional or feature definition was relatively strong whereas their ability to identify or quantify the market value proposition was relatively weak. We recognize the difficulties associated with developing a market value proposition and are concerned that reliably performing this task may require significant practical experience

8.2 Which is more important – the business case or the product requirements?

Both requirements engineering and business analysis are important and necessary elements of the entrepreneurial process. However, it is our opinion that a sound business case analysis significantly improves the probability of delivering the desired product to the customer. Requirements engineering can reliably deliver a valid product specification, but what if the customer does not want the specified product? Without customer validation a technology push effort is only an educated guess. It follows that having a validated value proposition as the basis for the requirements effort will lead to a greater probability of commercial success.

In a fast-paced market such as mobile applications, business case analysis can begin at the same time as prototype development used to elicit market feedback (non-functional prototypes used to evaluate customer response to product or service concepts, used with caution due to the possibility of loss of control over the underlying intellectual property). If the results of the business case analysis are positive then the parallel start on prototype development delivers a jumpstart on the overall development process. However, if the business case is negative then significant expenses and lost opportunity costs have been incurred. The feedback received during the camp can be summarized as follows: the participants have good presentations skills and well understood market propositions and well-crafted business models. The participants were weak on their financial analyses.

Pursuing a startup venture in the absence of a valid market value proposition significantly increases the probability of a pivot. In the camp studied here, two of the four projects were pivots on entry into the camp. These pivots were deemed necessary as a result of the analysis done between the time of participant selection and their entry into the camp. One pivot maintained the general nature of the product (stop-motion animation) but the target market(s) were completely redefined. The original target

market was neither willing nor able to pay for the proposed product and failing to pivot would have resulted in a well-defined product that no one was interested in purchasing. In the second pivot, the original concept was discarded and an entirely new market opportunity was pursued.

We have been unable to identify a plausible scenario that justifies not performing a business case analysis. The analysis does not have to be done before development begins. It can be done in parallel with development if the associated risks are acceptable, but the evidence before us suggests that it needs to be done. We cannot conclude that the business case is more important than the product requirements but an antecedent business case analysis can greatly reduce the risk that the product requirements define a product that no one wants. We note that, in the same way that customer willingness and ability to pay should constrain RE efforts, so should technology constraints be considered in the business case analysis.

8.3 Validation

The camp participant selection process was strongly biased toward selecting candidates with entrepreneurial tendencies that were also believed capable of learning the materials and completing their app within the allotted time. As such, the chosen candidates were the elite of the applicant population and were intended to represent a sample of the entrepreneur population and not the general population.

The camp participants were drawn from a technology-trained population. Their observed behavior in Section 7, Figure 1 may have been biased toward the right-hand (technology-focused) cycle rather than the left-hand (market-focused) cycle. Further investigation is needed to determine how prior training and experience affects behavior in this area.

9 Conclusions and Future Work

This work has investigated the feasibility of extending RE practices by applying requirements engineering techniques to the investigation of commercial viability for proposed products and services. None of the camp participants had more than superficially considered commercial viability of their products before camp entry. Usually addressed through business case analysis, these RE-based efforts enhance the entrepreneurial endeavor's viability through improved value proposition definition, compliance of the product definition with the value proposition and provide focus upon the business goals for development efforts. Our results were generally supportive of the practice, successfully applying requirements elicitation, negotiation, prioritization, triage, scoping, tracing and validation to business case analysis tasks, particularly when guided by an experienced mentor. We do not consider this a best practice, but our initial results indicate that using RE to perform business case analysis does benefit the project. For the camp participants, two of the four projects performed significant and successful business pivots that addressed real (and not just assumed) markets by the application of these techniques.

Perhaps the most significant challenge to success is domain specific terminology and knowledge. Asking requirements engineers to perform business analysis tasks requires them to become subject matter experts, at least to a degree, in a whole new discipline and this is not something that can occur quickly. We suggest that academic training could include entrepreneurial concepts and greater strength in the

fundamentals needed to perform due diligence in a business case analysis. While perhaps unnecessary on well-rounded teams, this domain knowledge would facilitate communication and provide insurance for smaller teams.

Four business cases were developed in the camp and in all cases we began business case analysis at the same time as prototype development. Our experience indicates that a business case analysis is a significant element within a risk reduction strategy and that entrepreneurs should prioritize this analysis as much as possible to determine the viability of the venture as quickly as possible. The techniques that we applied in the camp are widely used outside of RE and it appears that traditional RE practice assumes that they have already been performed by other members of the team. Simplistically, for commercially motivated endeavors, the RE practitioner should first identify that there exists an identifiable customer population that has a willingness to pay for the new product or service. Then the practitioner should confirm that there exists a sufficiently large subset of this customer population that also has the ability to pay for the new product or service – only then does a viable business case for sufficient ROI possibly exist and only then should intensive RE efforts begin.

We note that the condensed timelines associated with mobile application development appear to be aligned with the needs of academic research scheduling and resource allocation. Incubators may be a rich source of case studies for combined academic-industrial research.

Feedback was received from all stakeholders and we have learned many practical lessons from delivering this camp. The next generation of the camp has been adapted as much as possible, within resource constraints, and the next session of the camp is eagerly awaited by the local community.

References

[1] Vision Mobile, “Developer Economics 2013”, http://www.visionmobile.com/product/developer-economics-2013-the-tools-report/

[2] International Institute of Business Analysis (IIBA). (2006). "Business Analysis Body of Knowledge (BABOK ®) v1.6."

[3] Ruokolainen, J., Igel, B.: The factors of making the first successful customer reference to leverage the business of start-up software company — multiple case study in Thai software industry. Technovation, 24, 673-681 (2004)

[4] Mann, R. J., Sager, T. W.: Patents, venture capital, and software start-ups. Research Policy. 36, 193-208 (2007)

[5] Burgel, O., Murray, G.: The International Market Entry Choices of Start-Up Companies in High-Technology Industries. Journal of Intl. Marketing, 8, 33-62 (2000)

[6] Barbe, D., Thornton, K, Green, J., Casalena, T., Weinstein, M., Ghavam, B., Robertson, B.: Hinman CEOs student ventures. In: Conference Proceedings 2005 ASEE Annual Conference and Exposition, pp. 13255-13267, (2005)

[7] Seyff, N., Ollmann, G., Bortenschlager, M.: iRequire: Gathering end-user requirements for new apps. In: 19th IEEE Int. Requirements Engineering Conference, pp.347-348, IEEE Comp. Society, (2001)

[8] Vogl, M., Lehner, K., Grunbacher, P., Egyed, A: Reconciling requirements and architectures with the CBSP approach in an iPhone app project. In: Proceedings of the 19th IEEE International Requirements Engineering Conference, pp. 273-278, (2011)

[9] Koivisto, N., Rönkkö, M.: Entrepreneurial Challenges in a Software Industry. Lecture Notes in Business Information Processing, vol. 51, pp. 169-174 (2010)

[10] Daimi, K., Rayess, N.: The role of software entrepreneurship in computer science curriculum. In: Proc. of the 2008 International Conference on Frontiers in Education: Computer Science & Computer Engineering (FECS 2008), pp. 332-8 (2008)

[11] Morough, P.: Is software education narrow-minded? A position paper. In: Intl. Conference on Software Engineering, pp. 545-546 (2000)

[12] Bareiss, R., Mercier, G.: A Graduate Education in Software Management and the Software Business for Mid-Career Professionals. Proc. 23rd IEEE Conference on Software Engineering Education and Training (CSEET), pp. 65-72 (2010)

[13] Aoyama, M.: Persona-and-scenario based requirements engineering for software embedded in digital consumer products. Proc. 13th IEEE Int. Conf. on Requirements Engineering, pp. 85 – 94 (2005)

[14] Aranda, J., Easterbrook, S., Wilson, G.: Requirements in the wild: How small companies do it. In: 15th IEEE International Requirements Engineering Conference, pp. 39-48 (2007)

[15] Gordijn, J., Akkermans, H.: Value Based Requirements Engineering: Exploring Innovative e-Commerce Ideas. Req. Eng. Journal, 8, 114—134 (2001)

[16] Gordijn, J., Akkermans, H.: Designing and evaluating e-business models. IEEE Intelligent Systems, July/August 2001

[17] Ries, E.: The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses, Random House Digital Inc. (2011)

[18] Sommerville, I., Kotonya, G.: Requirements engineering: processes and techniques. John Wiley & Sons, Inc., 1998

[19] Sommerville, I., Sawyer, P.: Requirements engineering: a good practice guide. John Wiley & Sons, Inc., 1997

Game Development Accelerator – Initial Design and Research Approach

Antero Järvi, Tuomas Mäkilä, Sami Hyrynsalmi

University of Turku, Turku FI-20014, Finland
{antero.jarvi, tuomas.makila, sthyry}@utu.fi

Abstract. Start-ups and game development are trending topics. There are established methods for both, but these are not suitable as such for starting game companies developing their first commercial game product. In this paper, a design for a series of accelerator programs, targeted for the first-time game developers, and an accompanying research approach are discussed. The goal of the approach is to combine quality research with relevant, imminent results, which help game start-ups to raise the success probability and lower the investors' risks. Initial ideas of the accelerator design are presented to activate discussion with other researchers and practitioners planning or doing similar experiments.

Keywords: game business, lean start-up, start-up accelerator, game development

1 Introduction

During the last few years, entrepreneurship has become a mainstream trend and there are now start-up accelerators not only in traditional start-up hubs like Silicon Valley but in almost all major university cities around the world. Similarly, game development as a hobby and as a career choice is gaining interest among students with background from computer science to art and more humanistic disciplines.

As mentioned, there is a wide variety of accelerator programs like Y Combinator[1] and Seedcamp[2] to name some of the most well-known ones. However, only a few accelerators targeted to starting game developers exist, although the needs of a starting game company differ from the needs of more traditional start-ups. For example, monetization, marketing, and distribution – as well as the product life-cycle – all have specific characteristics in the game industry. A game start-up usually targets to a single, well-segmented, intangible *game product* whereas traditional start-ups nowadays concentrate on service concepts or wider product lines.

This paper describes our research approach and initial ideas on a game-specific accelerator program. In this paper, we raise a discussion on the following topics:

[1] http://ycombinator.com/
[2] http://www.seedcamp.com/

1. Can first-time commercial game developers benefit from a game-specific start-up accelerator program?
2. How such a program should differ from existing, general start-up accelerators?
3. Can a popular and field-tested Lean Start-Up Method (LSU) be used as a basis for such an accelerator program?

It should be noted that the established game companies commonly use agile practices, which are closely related to the lean start-up methodology, in organisation of their daily development work. They also apply business and product development principles that are very close to the lean start-up practices. However, the key motivation in this research is to disseminate this knowledge to inexperienced game developers and train them to utilize these apparently good practices.

The paper is organized as follows. Section 2 presents game development and start-up methodologies in general, and challenges of the game development more specifically. Section 3 presents our planned research approach. Section 4 discusses the initial ideas of the design of a game-specific accelerator program. Section 5 defines the next steps of our research and concludes the paper.

2 Background and Motivation

Although computer and electronic games origin decades ago from basements of the universities and the existing studies on various aspects of computer games is exhaustive (see e.g. Smed & Hakonen 2006), the research on computer game start-ups and game business is, to the authors' knowledge, rare.

In the following, we will first shortly present three perspectives on game development and use these to state the motivation for this research. Then we will present different game production concepts from literature in order to understand the product development process of games and how it differs from traditional software development. It is followed by a brief discussion on the domain of software start-ups and challenges faced by the game developers.

2.1 Motivation: Three perspectives on game development

Hakonen et al. (2008) identified three perspectives for making of computer games: Humanistic, Construction and Business perspectives. The first perspective addresses how games affect gaming communities, players, and society at the large. The second focuses on the building of the game with a technical point-of-view. The last one concerns the economics of the computer games including e.g. productization and competition strategies.

We use this division as a baseline. However, as our approach is more pragmatic, we narrow the scope of the perspectives and rename them as 1) Game design, 2) Game building, and 3) Game business. These perspectives are not separated; instead, they are highly intertwined as presented in Figure 1. The first, *Game design*, address actual design issues such as control mechanism, gameplay, story, artistic style and

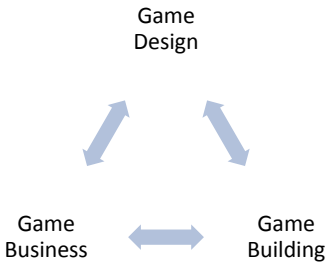

Figure 1: Three perspectives for a computer game production

graphics as well as the social dimension of the game. That is, the Game design perspective focuses on the gameplay experience.

Game building includes the software and audio-visual engineering viewpoint of game production. For example, this perspective includes such issues as handling of hardware and platform, skills and formation of the team, company's organization and project's schedule. We include, furthermore, in this perspective also other important issues, that should be taken into account in the game development, such as billing, tutorials and menu systems *i.e.* the productization of the game.

The third perspective, *Game business*, focuses on the economic side of computer games. It includes such issues as customer and partner identification, monetization plan, organization of the game launch as well as discovery and growth plans.

The three perspectives help to identify different issues that need to be thought through during the actual game development. However, it should be noted that the perspectives and issues are highly interdependent. For example, game design issues might either enable or prevent the use of in-game payments. Similarly, the decision to use a *Freemium* -based business model might set requirements to the game design (e.g. premium content) and game building (e.g. player engagement).

Looking more closely to the presented three perspectives, we can notify the complexity of the computer game domain. When a new, inexperienced, team starts to develop a new game, they face several relevant questions. From the presented three perspectives, we can easily highlight questions such as 'Where to start?', 'What influences on what?', and 'What do we know and what we should know?'. This complexity acts as the driving force for our research.

2.2 Game production

Computer game production differs from more traditional software product development in that a game production process often includes several multi-disciplinary areas such as game and story design, graphics design and implementation, sound engineering and level design (Mäkilä et al. 2009). A few scholars have discussed about the generic game development models, e.g. Chandler (2006), Larsen (2002), and Manninen et al. (2006). In the following, we will review the models of Manninen et al.

(2006) and Chandler (2006). Manninen et al. (2006) divides the creation of computer games into six phases:

1. **Concept** phase in which the conceptual design of the game is drafted.
2. **Pre-production** phase consists of creation of a working prototype. The objective of the phase is to "plan, test and evaluate everything possible".
3. **Production** phase contains all tasks, from programming to graphics and sounds, and integration needed in game creation.
4. **Validation and testing** phase includes functional testing as well as quality assurance of gameplay, user interfaces etc.
5. **Launch** phase consists of releasing the game and supporting activities.
6. **Maintenance** phase includes bug fixing and upgrades development.

In comparison, in Chandler's (2006) generic model has only four phases: Pre-production, Production, Testing, and Wrap-up. The two models are very similar. The former, however, emphasis more post-release activities. Hakonen et al. (2008) compared these two models to a general software product development model by Hohmann (2003) and noted only minor differences. They stressed the natural co-operation of several disciplines in game production which is rare or non-existent in a software product development.

Electronic games, however, have one clear difference: the users are seldom able to choose which desktop software they use, unlike game players who do not have to play games that they do not like. Furthermore, in addition to the requirements of being easy to use, the games are required to challenge the users (Weinschenk & Barker, 2000). That is, the game is required to be both entertaining and challenging; we call this simply as a fun factor and address its design later in the paper.

2.3 Software Start-up Process

During the last years, the software start-up practice has been revolutionized mainly by two business development frameworks: Customer Development model by Steven Blank (2005) and Lean Start-up methodology (LSU) by Eric Ries (2011). These tools aim to create manageability and measurability into the start-ups; they are meant to change the way products and companies are built and launched. We will quickly present these methods and refer interested readers to Blank (2005), Ries (2011), and Cooper & Vlaskovits (2010) for further details.

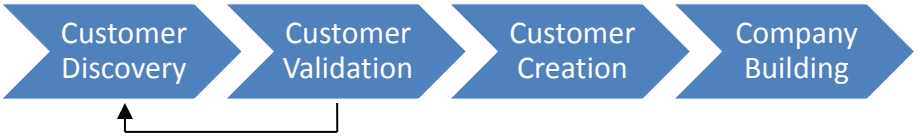

Figure 2: Customer Development model (adopted, Blank 2005)

Blank (2005) describes Customer Development methodology, illustrated in Figure 2, in four steps: Customer Discovery, the first step focuses on identifying the customers and how they value the problem that the start-up is trying to solve. This step tries to establish the Problem/Solution Fit, i.e. a validation that the real customer problem is found. The second step, Customer Validation, aims to prove that the start-up has found a market which reacts positively to the product. In practice, this includes e.g. verifying the size of the market, pricing strategy and repeatable sales model. At the end of this step, the start-up has established the Product/Market Fit.

The third step of the Customer Development model is Customer Creation (Blank, 2005). In this step, the aim is to scale execution by creating and driving customer demand. For example, some start-up companies join to the market populated by the rivals while others create markets for their products. In the last step, Company Building, the goal is to transform the company from learning and discovery organization to a well-oiled execution machine for the business.

Blank (2006) emphasizes that in contrast to the traditional product development model, Customer Development is an iterative process and going backward should not be treated as a failure. Furthermore, he underlines the importance of getting out of the building and meeting the customers. In top of these principles, Ries (2008, 2011) started to build his own Lean Star-up methodology.

The Lean Start-up model was first presented with three pillars (Ries, 2008): 1) the use of open-source and free software or low cost development platforms, 2) the use of agile development methodologies (see e.g. Larman, 2003), and 3) the use of Customer Development. Cooper and Vlaskovits (2010) added the fourth pillar to LSU: the use of cheap and effective measurement and analysis tools. Ries (2008) stated that his belief is that using these pillars will lower development costs, shorten time-to-market, and improve the quality of products.

LSU has since been evolving and it now utilizes the principles of previously mentioned pillars in more general context than in software start-up development. The fundamental activity is the Build-Measure-Learn loop (Ries, 2011). The loop, illustrated in Figure 3, aims to eliminate uncertainty and help to work smarter, not harder. The central concept in the loop is a Minimum Viable Product (MVP), defined as “a version of a new product which allows a team to collect the maximum amount of validated learning about customers with the least effort” (Ries, 2009).

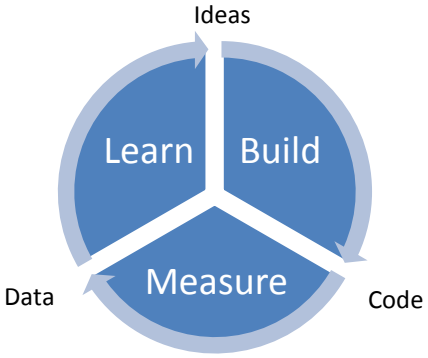

Figure 3: The Lean Build-Measure-Learn –loop (Ries, 2011)

3 Research Goals and Approach

This paper describes initial thoughts on how a game startup can be helped to build their first game product by taking steps to approximately right direction. We see that there is an opportunity to successfully combine the systematic investigation methods of academic research with hands-on learning-by-doing activities to construct relevant guidelines for the first-time commercial game makers. The goal of this endeavor is to 1) give game startups better chance of success with their first game products, 2) make work of investors who finance these start-ups easier and less risky and 3) simultaneously do high-quality academic research.

We propose a research approach where a suitable start-up accelerator design is developed through a series of real-life game startup accelerators, which are analyzed using qualitative and quantitative case study methods. The scientific data and results are used to iteratively improve the design and ultimately make it scalable and to be used more broadly. Lean principles are utilized in this research by testing the design as early as possible and by adjusting it based on participant feedback.

The planned steps to achieve the research goals are listed below:

1. First draft of the accelerator design and adjustments based on the interviews of the game companies;
2. First batch of the accelerator analyzed as a case study;
3. Adjustments to the design based on the first case study;
4. Second batch of the accelerator analyzed as a case study; and
5. Dissemination of the research results as a pragmatic handbook and a permanent accelerator program.

Utilization of empirical strategy in this research work is justified by the uncertainty in the accelerator design and real impacts on the game business development. The feasibility of initial acceleration design is verified rapidly, and the decision on continuing the research can be done before wasting significant resources.

The success of a specific accelerator is heavily influenced by two factors: 1) How good (skill-set, team dynamics etc.) are the teams and 2) How seasoned mentors participate in the accelerator. The accelerator can also be seen as a learning experience and if all teams do not succeed with their first game product, they have better odds to do so with their following games.

4 Accelerator Design

The inspiration for the design of the game accelerator is the LSU methodology. It is widely field-tested in business software start-ups, but less applied to game start-ups. There are only a few early ideas and experiences outside the academic field (see e.g. Vining, 2011; York, 2012).

4.1 Lean Start-up Concepts in Game Development

The main LSU principles do not carry over to game projects as such, but need to be reinterpreted for the game development domain. Thus we briefly discuss how these principles are reflected into game development.

Context – LSU is meant to be used when developing something new under the conditions of extreme uncertainty. This is not the case in all game projects, as some aim for the replacement game market, i.e. essentially copy an existing successful game concept by modifying it without major innovations. Thus, we require that the game concept involves something new that is untested in the targeted game market. Another factor that increases uncertainty is the lack of experience in the team. This is why we prefer first time commercial game developers, as we expect to generate the highest benefits for this group. To emphasize: in order to maximize the achievable benefits of LSU process we decided to exclude game clones and "me-too" versions of the games, as well as experienced teams.

Minimum Viable Game (MVG) – The development should as early as possible aim for a minimum playable game that implements the core game mechanics leaving out everything else. After this point the game should be kept playable at all times. However, a major challenge in using minimum viable games to test hypotheses about the gameplay is that games are holistic products and it is not trivial to know what contributes to the players experience and what can be left out of the game. This is different from minimum viable business software products, where validating a customer need can be simply done by adding a feature to see how the customer values it.

Build-Measure-Learn – The loop works similarly in game production than in plain vanilla version of LSU. It should be noted that the scope of the LSU loop includes all three areas – Game Business, Game Building and Game Design – whereas the traditional 'Play testing' frequently done in game development only focuses on Game Design, i.e. finding the fun factor.

Validated Learning – Testing in LSU must be done scientifically, i.e. experiments are designed for a specific purpose and metrics are chosen to measure the outcome of

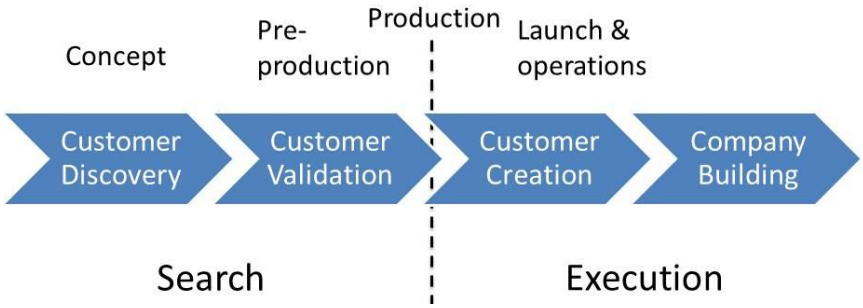

Figure 4: The mapping of game production phases to Customer Development model

the experiment. Both test design and used metrics are very game specific and part of domain experience. LSU emphasizes the use of these methods with discipline.

Information Accounting – In LSU it is vital to keep track on what hypotheses about the business being built have been validated and what still remains uncertain. This is typically done with a business model canvas (Osterwalder & Pigneur, 2010). The uncertainties in game development, discussed previously as the three perspectives in game production, differ from those captured in commonly used canvases. Thus a game specific business modelling canvas should be crafted. To the authors' knowledge, there are currently no game specific business canvases, and as part of this research a canvas will be developed.

The overall progress in the LSU methodology is structured along the four phases in Customer Development as shown in Figure 4. The four phases cannot be directly matched to game business development phases, instead the division to *search* and *execution* modes is more important. In the original Customer Development method, first two phases are about searching a repeatable and scalable business model, whereas the latter two phases are about growing the business and creating an established company. In game development projects, we find similar modes. Concept creation and pre-production are about searching for a game that is worth making as well as a business model to monetize it. In the production phase and later, final game is produced, polished, launched and supported, thus it is about execution. Traditionally, in pre-production, before crossing over to production, the game developers need to fix all major decisions about the game, since changing them would be too expensive once the game is in production. In short, bulk of the creative and innovative work is done before entering the production phase. Therefore we limit the scope of the accelerator into the concept and pre-production phases.

Some of the known applications of LSU thinking into games are for latter phases, especially live operation of an online game. The focus here is on optimization of the game – the balance of its mechanics, monetization etc., and although uses similar techniques than LSU, is fundamentally very different.

4.2 Key Game Development Decisions

The key activities in developing a game and making it commercial are the approximately the following:

1. Build the fun
2. Find the market i.e. the players
3. Choose monetization mechanisms
4. Ensure growth of game audience

Build the fun, i.e. finding the customer value, is the cornerstone of a successful game. This is, as any creative design work, difficult to do in a strictly forward process. The Mechanics-Dynamics-Aesthetics (MDA) (LeBlanc, 2008) model captures the nature of game design: mechanics are the rules of the game, dynamics is what happens when the game is played and aesthetics is what the player experiences, the fun. The aesthetics is the value that is sought after, but the designers can directly only affect the mechanics. A common way to deal with this is to use game testing and prototyping extensively in game design (Schell, 2008). However this differs from the Build-Measure-Learn loop in LSU since the prototypes are typically tested by the game developers themselves, not the intended customers. Furthermore, this design is strongly guided by the vision that the developers have about the game; it is more about realizing an anticipated customer need than learning what the customer needs.

Finding the customer and eventually the market early is one of the fundamental principles of LSU. When developing games it is crucial to understand what audience the game is targeting. This affects everything: what the game should be like, how the game should be monetized, marketed, distributed, what is the size of the business opportunity, etc. However, new game development teams tend to ignore this question and simply develop the game for themselves, or even worse, to everybody. Early analysis of the game audience will be highlighted in the accelerator and possibly validated using LSU methods.

Monetization is more complicated than just setting the price and selling the game. In many game platforms, and especially in mobile gaming, monetization is done increasingly using the free to play model with small monetary micro-purchases during the game play. This model requires deep understanding on what the player tries to achieve in the game and hooking the micro-purchases directly to this. This is an example of how tightly Game Business and Game Design are connected.

Growth of a business can have three different drivers (Ries, 2011), all of which are applicable to game businesses: sticky-, viral- and paid engines of growth. In sticky mode the growth comes from keeping the players as long as possible and generating revenue either via micro-purchases or some subscription model. In viral mode growth relies on players bringing in new players via some social media connections or multiplayer game mechanics. Third mode, paid growth, simply means more traditional marketing driven sales. Depending on the case, only one or all of the growth engines can be involved. Using viral and sticky engines typically entail decisions that are suitable for validation by the LSU loop.

It should be emphasized that the above issues, especially monetization mechanisms and viral or sticky growth models, cannot be added later onto otherwise finalized game. Instead, they are directly connected to designing the core game play and building the fun, thus they must be taken into account early as the game concept is formed and the game designed. This fact will be taken into account in the accelerator design, by using the LSU method to validate these important business decisions as early as possible.

4.3 Practical Implementation Issues

As we have chosen the lean start-up method as the guideline for our first accelerator trial, we will choose the participating game projects so that they maximally fit the lean start-up 'sweet spot'. The accelerator aims to teach the business aspects of the game production to technically skilled participants. In practice this means that the teams are inexperienced first time *commercial* game developers, however they have sufficient skills in coding, game design, media production and other needed development skills. This is to ensure that the focus of the accelerator program remains on developing the Game business aspects, not learning basic development skills. The games are small enough that playable MVG's can be built in reasonable time, that the game play testing can be organized with reasonable effort, and the development platform allows fast development and publishing.

In the first accelerator trial we aim at six teams since it is manageable, leaves room for one or two teams dropping out and yields enough cases for the research. The duration will be approximately two months which should be enough for releasing several sequential minimum viable games, and force the teams make decisions on critical business issues in addition to developing the game.

The teams will be supported by weekly mentoring session by seasoned experts on various topics in Game Design, Game Business, and Game Building. At the end of the program the game projects will be presented to investors to get their opinion if the projects are easier to evaluate or more mature compared to a normal first time commercial game project.

5 Future Work and Conclusion

In this paper, we raised a discussion on a game-specific start-up accelerator program for first-time commercial game developers, and how such a program should differ from existing, general start-up accelerators. Furthermore, we ask can a popular and field-tested Lean Start-Up Method (LSU) be used as a basis for such an accelerator program.

We have described above a research approach to develop a game start-up accelerator program for game developers doing their first commercial production. In addition to the systematic research approach, we have presented our initial thoughts on the pragmatic design of such an accelerator program.

Our intention is to start the research activities and the first round of the program during the year 2013 in Turku, Finland. We will run and develop the accelerator design simultaneously with the research activities, which will provide us objective data on the program results. We expect to publish a more detailed description of the accelerator design on the end of the year 2013.

As mentioned, most of the work will be done in the Turku region. The region is fertile for this work since there are currently lots of enthusiastic game development hobbyists, but only a few, small professional game companies. Turku is a fast growing game development site in Finland. The authors are heavily involved in the game development training and the start-up development activities in the area. Finally, we encourage other researchers and practitioners with same kind ideas to consider our thoughts and, if the concept seems sensible, boldly adopt and adapt the accelerator design in their own experiments.

References

Blank, S. The Four Steps to the Epiphany: Successful Strategies for Products that Win. Second edition. Cafepress.com, San Mateo (2005)

Chandler, H. M.: The Game Production Handbook. Game Development Series. Charles River Media (2006)

Cooper, B., Vlaskovits, P.: The Entrepreneur's Guide to Customer Development. Cooper-Vlaskovits, USA (2010)

Hakonen, H., Mäkilä, T., Smed, J., Best, A.: Learning to Make Computer Games: An Academic Approach. TUCS Technical Report No 899. Turku Centre for Computer Science (2008)

Hohmann, L.: Beyond Software Architecture: Creating and Sustaining Winning Solutions. Addison-Wesley Signature Series. Addison-Wesley Professional (2003)

Larman, C.: Agile & Iterative Development: A Manager's Guide. Addison-Wesley (2003)

Larsen, S.: Playing the Game: Managing Computer Game Development. Technical Report International Edition. Version 1.1. Blackwood Interactive (2002)

LeBlanc et al. MDA Framework, http://www.cs.north-western. edu/~hunicke/MDA.pdf. Accessed April 2013.

Manninen, T., Kujanpää, T., Vallius, L., Korva, T., Koskinen, P.: Game Production Process — A Preliminary Study. Technical Report v1.0. Ludocraft/ELIAS-project (2006)

Mäkilä, T., Hakonen, H., Smed, J., Best, A.: Three Approaches Towards Teaching Game Production. In Kankaanranta, M., Neittaanmäki, P. (eds.) Design and Use of Serious Games. Intelligent Systems, Control, and Automation: Science and Engineering Volume 37, pp 3-18, Springer Netherlands (2009)

Osterwalder, Y., Pigneur, Y.: Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers. First edition. Wiley (2010)

Ries, E.: The lean startup. (2008) Accessed on April 7, 2013 http://www.startuplessonslearned.com/2008/09/lean-startup.html

Ries, E.: Minimum Viable Product: a guide. (2009) Accessed on April 7, 2013 http://www.startuplessonslearned.com/2009/08/minimum-viable-product-guide.html

Ries, E.: The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. First edition. Crown Business, New York (2011)

Schell, J. (2008) *The art of game design: A book of lenses.* Amsterdam : Elsevier.

Smed, J., Hakonen, H.: Algorithms and Networking for Computer Games. John Wiley & Sons, UK, Chichester (2006)
Vining, N.: Lean Startups, part II: Some Games That Suck (And Why That's Not A Bad Thing). Gaslamp Games Inc. (2011). Accessed on April 9, 2013 http://www.gaslampgames.com/2011/01/26/lean-startups-part-ii-some-games-that-suck-and-why-thats-not-a-bad-thing/
Weinschenk S., Barker D.T.: Designing effective speech interfaces. John Wiley & Sons, Inc., New York (2000)
York, T.: Making Lean Startup Tactics Work for Games. Gamasutra – The Art and Business of Making Games (2012)- Accessed on April 9, 2013 http://www.gamasutra.com/view/feature/168647/making_lean_startup_tactics_work_.php

Research on "Non-Issues" - Difficulties of Empirical Research on the Requirements Engineering & Management Process at the Client's Site

Some Notes from an Explorative Study

Rüdiger Weißbach

Hamburg University of Applied Sciences (HAW Hamburg)
Faculty Business & Social Sciences
Berliner Tor 5
D-20099 Hamburg
ruediger.weissbach@haw-hamburg.de

Abstract. This paper reports the difficulties of recruiting participants at the client's site in an empirical project on requirements engineering & management. The author discusses the origins of the problems and some stated reasons for the participation. These results lead in a short discussion about the improvement of empirical research on the requirements engineering & management process at the client's site.

Keywords: Qualitative research, requirements engineering, requirements engineering & management, user oriented research

1 Introduction

Known studies on software project processes seem to be mostly conducted on the producer's site, especially in cooperation with software producing companies. The reason is obvious: Software projects are a core business for software producing companies, who are interested in improving their processes. But business information systems are systems whose success is related to the usage by the clients. Therefore research at the client's site should be important. On the client's site the research focus seems to be the (business) success of information systems, according to the DeLone & McLean information systems success model (Urbach et al. 2008; Urbach et al. 2009). Research on processes on the client's site is rare and is concerned from special influencing factors.

This paper will show and discuss some problems, observed in an empirical study that was conducted in 2010/2011 in Germany in companies outside the IS business (Weißbach, R. 2013). The paper starts with a short description of the study and the recruitment of participants (Chapter 2), followed by a chapter on the stated difficulties for participation (Chapter 3) and a chapter on triggers for participation (Chapter 4).

Chapter 5 presents and discusses the conclusions. Chapter 6 will show some ideas for further work.

Acknowledgements. I like to thank the reviewers of this paper for their valuable advice to work out the ideas more precisely.

2 The Study and the Recruitment of Participants

In 2009/10, the author started a study on the participation of business department staff in the requirements engineering & management [RE&M] process. The aim of this study was to get a more differentiated view on the RE process in business information systems projects and in non-project work. The study was focused on (but not exclusively limited to) small and medium enterprises [SME].

RE&M in general is a topic that is covered by many textbooks and empirical research. The importance of the RE&M process for the success (or the failure) of projects is generally accepted in literature (overview in Herrmann, A. et al [eds.] 2013). But the research on RE&M focuses on the main actors in software engineering: requirements engineers, project managers, developers. Business department staff is commonly seen as object in the requirements elicitation process. The active participation of business department staff in contrast is disregarded in literature and obviously ignored in research (checked against the summarization in (Cheng, B., Atlee. J. 2007).

In this situation the author wanted to conduct 25 semi-structured personal interviews in the area of Hamburg, Germany, as a pilot study. The project was staffed by the author and a student assistant. Five participants had been found by personal contacts. To get the other 20 participants we collaborated with a regional entrepreneurs' association ("Bundesverband Mittelständische Wirtschaft", Hamburg). We thought to contact 50 member companies by personal telephone calls to get a relevant, but not ex ante quantified number of participants and added a call for participation on the website of this association. To clarify: The interviews should be conducted personally, the telephone calls should only be used to arrange the interviews. In return, the participants have been announced to get the results of the study and to get an invitation to a free workshop on RE.

No company accepted the personal invitation and only two companies responded to the presentation on the website. But both of these companies had been software companies, who were interested in the *result* of the study.

Therefore we decided in 2011 to make personal telephone calls by the student assistant to make appointments. The student assistance had experience in acquiring participants for marketing research studies. We picked telephone numbers from public telephone directories and asked for responsible persons in IS and/or business departments. The telephone agent worked from the university's site, so that the university's official telephone number was transmitted. To get 18 participating companies it was necessary to contact ca. 900 companies (multiple calls counted only as one contact), equivalent to a response rate of 2%. This response rate seems to be very low, but we did not find information about response rates in comparable situations. Typical re-

sponse rates for telephone interviews with companies in Germany are 20-30% (Koll, C. 2006)

In 4 of the 25 participating companies, the interviews had been conducted simultaneously with 2 interviewees working in the same company, either in the same or in different departments.

3 Arguments for Non-Attendance

3.1 Introduction

Many companies mentioned the lack of time or a privacy policy as reasons to their non-attendance. But these have not been the only arguments. Focusing on the main research topic, we did not record and count the answers explicitly at that time. Therefore the following aspects should be seen as indicators, not as clear and complete results.

3.2 General Lack of Interest in Research

Companies in the IT branch are interested in market research and in improving processes. For these companies the benefit of participating in research projects is obvious. But what could be the interest for companies in other branches to participate in IT research projects?

The focus on IS research at the user's site is the success in IT projects. This topic is an accepted research topic (DeLone and McLean, see Urbach et al. 2008, Urbach et al. 2009). While this topic addresses the management in general, research on RE&M processes is a very specialized topic. Referring to the "rigor vs. relevance" discussion (Benbasat, I., Zmud, R. 1999, Lyytinen, K. 1999) it seems that RE&M is not relevant for management.

3.3 Research Topic is a "Non-Issue"

Many of the asked companies told us, that they are not interested in the research topic, because it is not relevant to them.

This argument could be interpreted in different directions:

- Unknown vocabulary: The term "requirements engineering" (or the German translation, "Anforderungsanalyse") is not known. Anticipating this danger, the telephone agent paraphrased the problem additionally.
- Lack of awareness: The importance of this topic is not realized. - Or: The importance of this topic is realized, but it is no problem in praxis. - Or: The topic seems to be not relevant, because IS are not seen as important for the core processes of the business.

3.4 Empirical Research has no Direct Benefit

Empirical Research has no direct benefit to the participants: One reason is that there is no direct output of the research action. The other is "difference in timeframes of action between academics and practitioners" (Kuechler, W., Vaishnavi, V.. 2011: p. 127). A typical result of empirical research is a benchmark that could be useful for the participants. But this benchmark refers to a former situation that must not be valid any more.

In Design Science Research (Vaishnavi, V.,Kuechler, W. 2012) the benefit for the participant is more concrete and immediate.

4 Triggers for Participation

4.1 Own Academic Background

Some people agreed to participate because they remembered their own university background. They wanted to support the university in general or the scientists and they wanted to get back in contact to the university to discuss and reflect their positions.

This argument was produced by participants in companies with a relative low proportion of academic staff.

4.2 Professional Awareness

Some people had been interested in the research topic due to their professional education. These people were computer specialists, programmer, technicians, regardless of an academic degree.

4.3 Own Experiences

Some participants acquired awareness of RE&M by own experiences in projects. Most of them reported problems in the project due to poorly conducted RE&M, including a complete project failure. Other participants realized early enough the importance of RE&M in larger projects.

4.4 Desire for Reflection

Some participants had been interested in the reflection and discussion of their practice, as a kind of consulting. They were interested in the results of the research for improving their own knowledge.

4.5 Mouth-to-Mouth References / "Snowball Principle"

One participant gave us the phone number of a colleague who was interested in this research topic, too.

5 Conclusions

Starting the project we thought that the effort for recruiting participants would be less. According to (Benbasat, I., Zmud, R. 1999) who specified applicability (= utility), currency and interest to professionals as important, we thought to have a research topic well fitting to the companies' needs.

But it seems that research at the client's site is more difficult than at the producer's site, because the benefit for the participants will often not be directly recognizable. Also, results of empirical research will need longer time for dissemination than the design of artifacts.

Due to the lowly estimated importance of empirical research on RE&M processes at the client's site, it seems to be difficult to establish new research directions. Therefore we will depend on qualitative research – e.g. case studies, grounded theory – to understand the diversity in RE&M processes especially on the client's site.

6 Validity Discussion

This paper analyses an RE&M research project with an untypical research question and a heterogeneous group of potential interviewees. Therefore the observations could not be seen as valid for other research questions in Software Engineering in general.

The response rate for typical research questions with a homogeneous group of potential interviewees will be higher.

7 Ideas for Further Work

Regarding the research experiences described in this paper and the preliminary results described in Weißbach, R. (2013), we will state a lack of understanding of internal processes and of collaboration processes between internal and external staff on the client's site in the RE&M process. To work out a framework for how RE&M processes are conducted at the client's site, grounded theory and case studies will be the first valuable approach. This framework could be enhanced with quantitative empirical research.

8 References

1. Benbasat, I., Zmud, R.: Empirical Research in Information Systems: The Practice of Relevance. In: MIS Quarterly Vol. 23 No. 1, pp. 3-16/March (1999)

2. Cecez-Kecmanovic, D.: Critical Research in Information Systems: The Question of Methodology. In: *ECIS 2007 Proceedings.* Paper 150, pp. 1446-1457 (2007). URL: http://is2.lse.ac.uk/asp/aspecis/20070023.pdf (retrieved 2013-04-06)^
3. Cheng, B., Atlee, J.: Research Directions in Requirements Engineering. In: *FOSE'07: Future of Software Engineering*, Minneapolis (2007)
4. Herrmann, A., Knauss, E., Weißbach, R. (eds.): Requirements Engineering und Projektmanagement. Berlin: Springer (2013)
5. Hevner, A., March, S., Park, J., Ram, S.: Design Science in Information Systems Research. In: MIS Quarterly Vol. 28 No. 1, pp. 75-105/March (2004)
6. Jabar, M. et al.: An Investigation into Methods and Concepts of Qualitative Research in Information Systems Research. In: Computer and Information Science Vol. 2 No. 4, pp. 47-54/November (2009)
7. Koll, C.: Möglichkeiten und Grenzen der CATI-Methode bei Betriebsbefragungen. In: Buchwald, C. (ed.): Das Telefoninterview – Instrument der Zukunft? pp. 22-41. Forschungsberichte aus dem zsh 06-3. Halle (2006)
8. Kuechler, B., Vaishnavi, V.: Promoting Relevance in IS Research: An Informing System for Design Science Research. In: Informing Science: the International Journal of an Emerging Transdiscipline Vol. 14, pp. 125-138 (2011). URL: http://www.inform.nu/Articles/Vol14/ISJv14p125-138Kuechler570.pdf (retrieved 2013-04-06)
9. Lyytinen, K.: Empirical Research in Information Systems: On the Relevance of Practice in Thinking of IS Research. In: MIS Quarterly Vol. 23 No. 1, pp. 25-27/March (1999)
10. Urbach, N., Smolnik, S., Riempp, G.: A Methodological Examination of Empirical Research on Information Systems Success: 2003 to 2007. In: Proceedings of the Fourteenth Americas Conference on Information Systems AMCIS, Toronto, ON, Canada August 14th-17th (2008)
11. Urbach, N., Smolnik, S., Riempp, G.: The State of Research in Information Systems Success – A Review of Existing Multidimensional Approaches. In: Business & Information Systems Engineering 4, pp. 315-325 (2009)
12. Vaishnavi, V. and Kuechler, W.: Design Science Research in Information Systems. January 20, 2004, last updated November 11, (2012). URL: http://www.desrist.org/design-research-in-information-systems/ (retrieved 2013-04-06)
13. Wu, Y.: Implications of Case Study Research in Information Systems in Supply Chain Management. In: 16th EDAMBA Summer Academy Soreze, France July (2007). URL: http://www.edamba.eu/userfiles/Yi%20Wu.pdf (retrieved 2013-04-06)
14. Weißbach, R.: How Business Departments Manage the Requirements Engineering Process in Information Systems Projects in Small and Medium Enterprises. In: Issues in Informing Science and Information Technology Volume 10 (2013). URL: http://iisit.org/Vol10/IISITv10p539-549Weissbach0093.pdf (retrieved 2013-05-06)

Workshop Notes: From Start-up to SaaS Conglomerate: Life Cycles of Software Products, IW-LCSP 2013

Tuomas Mäkilä and Krzysztof Wnuk

University of Turku, 20014 Turku, Finland
tuomas.makila@utu.fi

University of Lund, P.O. Box 118, 221 00 Lund, Sweden
krzysztof.wnuk@cs.lth.se

Abstract. The first international workshop *From Start-up to SaaS Conglomerate: Life Cycles of Software Products, IW-LCSP 2013* was held in June 11th 2013 at Potsdam, Germany. In the workshop five papers were presented with topics varying from product management in ecosystems and challenges in empirical requirements engineering research to start-up methods, incubators and accelerators. In these workshop notes the key findings of the presentations and lessons learned from workshop discussions are presented.

Keywords: start-ups, ecosystems, requirement engineering, game development

1 Introduction

From Start-up to SaaS Conglomerate: Life Cycles of Software Products International Workshop (IW-LCSP 2013) was held in June 11th 2013 at Potsdam, Germany. The workshop was hosted by the 4th International Conference on Software Business (ICSOB 2013).

A rationale behind the workshop was to provide a forum for researchers to present novel approaches and case studies, which they needed peer feedback on. The focus of the workshop was on emerging topics on the interface of the software engineering and business. These topics included software product lifecycles, especially building economically sustainable software, and software start-ups. Special interest of the workshop was to discuss the research approaches and how the results of the empirical research would benefit the software industry. Out of eight submissions, five papers were presented in the workshop.

2 Presentations

As mentioned, the workshop had five presentations. Timeframe for each presentation was 15–20 minutes followed by about 10 minutes discussion. After all presentations

an discussion session was held, where the presenters and the audience could exchange thoughts on the workshop topics. The workshop and the followed discussion session were facilitated by *Krzysztof Wnuk* and *Tuomas Mäkilä*. The key points for each presentation are analyzed in this section.

Software Ecosystems: From Software Product Management to Software Platform Management (2013) was presented by Slinger Jansen. Jansen introduced an addition to the existing Software Product Management model (SPM), where the effects the related of ecosystems is taken into consideration during the product development lifecycle. The model was developed based on the interviews of Dutch product managers. The paper triggered an interesting discussion regarding the interplay between software product management and ecosystems, and regarding the possible extensions in the software product management body of knowledge.

Lean Product Development in Early Stage Startups (2013) was presented by Jens Björk and Jens Ljungblad. The presenters had participated an experiment where a team of students developed several start-up ideas in parallel. The main principle was that the team would focus on the most prominent idea and would prepare to change the idea under development based on the feedback. A model for executing this kind of parallel approach was also presented. The model was based on the lean start-up methodology but was enhanced with the parallel business development aspects. The paper ignited discussion about how many possible solutions can a software start-up afford within limited resources.

Requirements Engineering as a Surrogate for Business Case Analysis in a Mobile Applications Startup Context (2013) was presented by Krzysztof Wnuk. He went through a case study where an incubator was held for technically oriented mobile applications developers. A model for gathering user requirements and taking business aspects of the applications using both business analysis and requirements engineering principles was presented. Wnuk argumented that the requirements engineering approach had basically the same goal as the business analysis approach, but would be easily adopted by technology people because of previously familiar terms and vocabulary.

Game Development Accelerator – Initial Design and Research Approach (2013) was presented by Tuomas Mäkilä. Mäkilä showed a plan for a research design, where a game development accelerator will be iteratively developed by actually running the accelerator and analyzing the results using scientific techniques. The accelerator would be based on the lean start-up methodology, which has to be adapted to suite the needs of the first time commercial game developers. Workshop participant lively discussed the differences of game development and traditional software development, and its implications on applying game development accelerator in practice.

Research on "Non-Issues" – Difficulties of Empirical Research on the Requirements Engineering & Management Process at the Client's Site (2013) presented by Rüdiger Weißbach. He briefly introduced a research setting where the relevance of the requirements engineering for the business functions of companies was investigated. However, the main contribution of the presentation was to analyze the reasons behind the low interest to participate the research interviews. Image of irrelevant research topic and lack of interest usually lead to the denial of the interview request,

while own research background and hands-on experience on the research topic lead to participation. The presentation created discussion about re-using classical business pitch techniques when getting industry involved.

3 Lessons Learned

The discussion between and after the presentations was lively. Especially the presenters and professors *Pasi Tyrväinen*, *Tiziana Margaria* and *Jan Bosch* participated actively to the ending discussion session. In this section lessons learned are summarized from the discussions and from the workshop presentations in general.

The challenge of successful vertical integration while extending software product management by software ecosystem concepts was extensively discussed during the ending session of the workshop. Furthermore, many of the presentations discussed about requirements engineering in a one way or another. Also, the lean start-up methodology, mentioned in several presentations, is based on understanding the core requirements and needs of the start-up's target customers. It can be said that one common factor of the presentations was *to find ways to understand external actors* like users and ecosystem partners *better during the product development life cycle*.

Another common factor between presentations was close relationship to the software industry. The presented research endeavors were inspired by software industry needs, were done near or with software industry and, hopefully, are relevant to software industry.

In general, the workshop was successful and the presentations were solid. During the discussion it became apparent that more research is needed on the organization of software start-ups and on the application of the requirements engineering techniques in modern software development and business context.

Workshop Papers

Jansen, S., Peeters, S., Brinkkemper, S.: Software Ecosystems: From Software Product Management to Software Platform Management. Proceedings of IW-LSCP 2013. CEUR-WS.org (2013)

Björk, J., Ljungblad, J., Bosch, J..: Lean Product Development in Early Stage Startups. Proceedings of IW-LSCP 2013. CEUR-WS.org (2013)

Callele, D., Boyer, A., Brown, K., Wnuk, K., Penzestadler, B.: Requirements Engineering as a Surrogate for Business Case Analysis in a Mobile Applications Startup Context. Proceedings of IW-LSCP 2013. CEUR-WS.org (2013)

Järvi, A., Mäkilä, T., Hyrynsalmi, S.: Game Development Accelerator – Initial Design and Research Approach. Proceedings of IW-LSCP 2013. CEUR-WS.org (2013)

Weißbach, R.: Research on "Non-Issues" – Difficulties of Empirical Research on the Requirements Engineering & Management Process at the Client's Site. Some Notes from an Explorative Study. Proceedings of IW-LSCP 2013. CEUR-WS.org (2013)