Creating Minimum Viable Products in Industry-Academia Collaborations

Jürgen Münch[1], Fabian Fagerholm[1], Patrik Johnson[1], Janne Pirttilahti[2], Juha Torkkel[2], and Janne Järvinen[2]

[1] Department of Computer Science, University of Helsinki,
P.O. Box 68, FI-00014 University of Helsinki, Finland
{juergen.muench,fabian.fagerholm,patrik.johnson}@cs.helsinki.fi
[2] F-Secure Corporation,
P.O. Box 24, 00181 Helsinki, Finland
{janne.pirttilahti,juha.torkkel,janne.jarvinen}@f-secure.com

Abstract. Customer value determines how products and services succeed in the marketplace. Early assessment of customer value is important for software startups, spin-off companies, and new product development in existing companies. Software technology often influences customer value and typically defines the main competitive advantage in both entrepreneurial and intrapreneurial settings. Value-related feedback from real customers is needed during software development and maintenance, and decision-making should be increasingly based on empirical evidence acquired through experiments. Getting such value-related feedback usually requires a so-called minimum viable product (MVP), i.e., an artefact that may be incomplete in functionality or quality, but displays characteristics that allows determining its customer value. In this article we report on a case study which used industry-academia collaboration for creating such an MVP. Our goal was to identify strengths and weaknesses of such an approach to creating MVPs while providing practical recommendations for improvement. The process followed in the case study was found to be very suitable for creating MVPs, reducing company-specific risks when testing customer-value, and advancing university education.

Keywords: Minimum viable product, prototyping, software start-ups, entrepreneurship, intrapreneurship, Lean Startup, Software Factory, case study.

1 Introduction

Software engineering experimentation aims at advancing the knowledge on how software development processes and methods in specific environments impact results [1]. According to Basili et al., experimentation is performed to help us better evaluate, predict, understand, control, and improve the software development process and product [2]. As with any other experimental procedure, experimentation in software engineering follows a cycle of building models for development processes or products, defining and testing related hypotheses, and refining models and hypotheses based on experimental results.

Traditionally, the main focus of software engineering experimentation has been on the technical and managerial aspects; the developer and project manager perspectives have been emphasized. However, since software engineering is a field which is often associated with innovative high technology, a vibrant global community of entrepreneurs, and a customer base with increasing expectations, it should also take into account the customer-perceived value of its products and services. While customer value may be simply defined as "whatever the customer is willing to pay for", it is not simple to assess or measure this concept while designing a product or service. In software projects, multiple stakeholders may have several different needs that affect how they perceive the value of the end product [3]. Rather than attempting to fix a single assessment or measure of customer value, we emphasize the importance of a process of experimentation and learning which allows empirical discovery of customer value in a specific context, guided by analytically derived hypotheses.

This perspective is also important in software engineering education. Students are entering a field which demands not only programming skills, but also the ability to dynamically consider customer value. They should be equipped with suitable knowledge of how to participate in software projects where value considerations drive project activities. It is particularly important that students are exposed to realistic, current problem settings and learn to analyse and understand value-related experiments.

Developing innovative software technology requires early testing of customer-related hypotheses. To make this possible, experimental objects, such as product prototypes, need to be created. Prototyping is used heavily in Agile software development. For example, prototyping is a core technique in the Dynamic Systems Development Method (DSDM) [4]. DSDM recommends four categories of prototypes: business, usability, performance and capacity, and capability prototypes. Early in the development process, business prototypes provide opportunities to conceptualise and communicate the business processes being automated, while usability prototypes allow definition, refinement, and demonstration of the system from a user's perspective. These two types of prototypes are closest to the notion of prototyping used in this paper. However, DSDM does not explicitly consider prototypes as experimental objects for testing the business value of a product idea during development. In Lean Startup terminology, a value hypothesis and a minimum viable product (MVP) need to be developed. Following Ries [5], a value hypothesis "tests whether a product or service really delivers value to customers once they are using it". An example of a value hypothesis is that customers of a specific customer segment will choose to sign up for a service based on a given set of features being offered. An MVP is an experimental object that allows for empirical testing of value hypotheses. According to the Lean Startup method, it should be built with a minimum amount of effort and development time [5].