Building Blocks for Continuous Experimentation

Software Systems Engineering Research Group

Alejandro Sánchez Guinea
with Fabian Fagerholm, Hanna Mäenpää, and Jürgen Münch

3 June 2014

RCoSE workshop (collocated with ICSE’14)
Overview

- Motivation
- Research Goal
- Research Approach
- Context of Case Study
- Continuous Experimentation Model
- Case Study & Continuous Experimentation Model
- Summary and Future work
Motivation

- Increasing number of companies are involved in developing software-intensive products and services
  - digitalization in most industry sectors
  - startup software companies
- Challenge: delivering value
  - technical issues not the only main concern
  - most solutions based on guesswork
  - some generic approaches such as Lean Startup methodology (build-measure-learn cycle)
- Framework for Continuous Experimentation is widely missing
Research Goal

**RQ** How can Continuous Experimentation with software-intensive products and services be organized in a systematic way?

- Continuous Experimentation model
- Case Study
Research approach

• Research based on design science research
  • Initial *continuous experimentation model* constructed based on relevant literature and authors’ experience
  • Matching model according to empirical observations and iterative adjustment of the model
• Case study analysis to ground the model in empirical observations
  • Startup company (3 projects)
  • Debriefing session after each project
    • project team, customer representatives, and researchers analyze decisions and actions taken during the project
  • Semi-structured interviews with company representatives
    • gain insights on lessons learned and perceptions concerning the adoption of customer feedback for decision making
Context of Case Study

- Small Finnish startup that develops a video calling solution for the home television set (Tellybean Ltd.)
  - Single-product strategy: delivering a life-like plug and play video calling experience
  - Main qualities marketplace: product affordability, accessibility, and ease of use.
- Three case projects
  - Duration: 7 weeks each, # developers: 4 – 7 students
  - First two projects same business strategy
  - Third project after company had modified strategy and assumptions
Context of Case Study

- In the *Software Factory* at the Department of Computer Science of University of Helsinki
  - Experimental Research and Development Laboratory
Continuous Experimentation Model

- Continuous cycle of experiments
- Repeated Build-Measure-Learn blocks
- Supported by an infrastructure

Learning Cycle

Build-Measure-Learn
Build-Measure-Learn
Build-Measure-Learn

Technical Infrastructure

- time
During analysis, the project data was examined for information relevant to the research question. We evaluated and adjusted our initial model based on the understanding gained from the observations, interviews and how they viewed the development process, especially in terms of software outcomes, learned information, and implications of software-intensive products and services. In some cases, experimentation was conducted with diverse skills, to communicate the assumptions of the roadmap to implement and deploy a Minimum Viable Product (MVP) or Minimum Viable Feature (MVF), and the final product. They first work with the data analyst role, which is usually a team of persons will handle the different parts of the model.

As plans and results accumulate and are stored, they may be reused in the case study and how they inform the di retrospective sessions, and interviews. By continuous experimentation, we refer to a software development approach that is based on field experiments with relevant stakeholders, typically customers or users, but potentially also with other stakeholders such as investors, third-party developers, or software ecosystem partners. The model consists of repeated Build-Measure-Learn blocks, supported by an infrastructure, as shown in Figure 2. The general vision of the product or service is assumed to exist. Following the hypothesis that can be subjected to experimental testing in order to gain knowledge regarding the assumption. Once the hypothesis is formulated, two parallel activities can occur. The hypothesis is used to implement and to deploy a Minimum Viable Product (MVP) or Minimum Viable Feature (MVF), and the final product. They first work with the data analyst role, which is usually a team of persons will handle the different parts of the model.

Experimental plans, which include programmatic features of the user. In the case of a persevere-decision, they work to fully map the areas of uncertainty which need to be tested.

The data analyst designs, executes, and analyses experiments. A data analyst also communicates with a developer and quality assurance engineer to produce proper instrumentation into the front-end system, MVFs, and the final product. They first work with the data analyst role, which is usually a team of persons will handle the different parts of the model. The technical infrastructure, and the information artefacts of such an infrastructure. The roles indicated here will be instantiated in different ways depending on the type of company. Simultaneously, an experiment is designed to test the hypothesis. The experiment is then executed with the necessary instrumentation. Simultaneously, an experiment has the necessary instrumentation. Simultaneously, an experiment has the necessary instrumentation.
**Build-measure-learn block**

![Diagram of the build-measure-learn block]

Figure 2: Build-Measure-Learn Block.
Build-measure-learn block

Figure 2: Build-Measure-Learn Block.
Build-measure-learn block

![Diagram of the Build-Measure-Learn block]

Figure 2: Build-Measure-Learn Block.
Build-measure-learn block

![Diagram of the build-measure-learn block]

Figure 2: Build-Measure-Learn Block.

Learn
were asked to reflect on how the company operated at the time of the vision of the product or service is assumed to exist. Following the and data from the MVP has the necessary instrumentation. Simultaneously, an experiment maximum Viable Feature (MVF), which is used in the experiment and to implement and deploy a Minimum Viable Product (MVP) or Min-
formulated, two parallel activities can occur. The hypothesis is used gain knowledge regarding the assumption. Once the hypothesis is hypotheses that can be subjected to experimental testing in order to
ments. An experiment operationalises the assumption and states a is profitable. However, each assumption has inherent uncertainties.
is connected to the business strategy, which is a description of how Lean Startup methodology [19], this vision is fairly stable and is
Figure 2 illustrates the Build-Measure-Learn blocks. The general conduct experiments, and connect product vision, business strategy,
software.
software-intensive products and services. In some cases, experi-
Figure 1. Conceptually, the model can also be thought to apply not other stakeholders such as investors, third-party developers, or soft-
4. RESULTS

In this section, we first describe our proposed model for contin-
eral model based on the understanding gained from the observations, relevant to the research question. We evaluated and adjusted our ini-
tions.

The back-end system consists of an experiment database which, with diverse skills, to communicate the assumptions of the roadmap are retrieved, analysis performed, and results produced. The results the back-end system. Conceptually, raw data and experiment plans develop or optimise the feature and deploy it into production. Cross-
lyst to produce proper instrumentation into the front-end system, MVFs, and the final product. They first work with the data ana-
are instantiated in di-
Figure 3 sketches the roles and associated tasks, the technical infrastructure, and the inform-
continuous experimentation is needed. Figure 3 sketches the roles different parts of the model.

If the hypothesis was falsified, the decision is to pivot and alter the strategy by considering the implications of the assumption being false. Alternatively, the tested assumption could invalidating the assumption on the strategy level, the decision is to choose strategy. If, on the other hand, the hypothesis was falsified, deployed. The strategic decision in this case is to persevere with the has given support to the hypothesis, and thus the assumption on the other roles working with several of the roles mentioned here. of the strategic roadmap. In order to do so, they consult existing management team, together handle the creation and iterative updating
multiple teams. Five roles are defined to handle three classes of in question. In a small company, such as a startup, a small number be instantiated in di-
and artefacts of such an infrastructure. The roles indicated here will in the experimental activities.

...
Building Blocks for Continuous Experimentation

Infrastructure for Continuous Experimentation

Role

Business Analyst  Product Owner  Data Analyst

Create & Iterate Roadmap  Design, Execute, Analyse Experiments  Develop Product

Technical Infrastructure

API  Experiment DB  Analytics Tools  Instrumentation

Back-end system  Front-end system

Information Artefacts

Raw Data  Experiment Plans  Experiment Results

4.2 Lessons Learned from the Projects

Startup companies operate in volatile markets and under high uncertainty. They may have to do several quick changes as they get feedback from the market. The challenge is to reach product-market fit before running out of money.

“You have to be flexible because of money, time and technology constraints. The biggest question for us has been how to best use resources we have to achieve our vision. In a startup, you are time-constrained because you have a very limited amount of money. So you need to use that time and money very carefully.” (Tellybean founder)

When making changes in the direction of the company, it is necessary to base decisions on sound evidence rather than guesswork. However, we found that it is typically not the product or service vision that needs to change. The change should rather concern the strategy by which the vision is implemented, including the features that should be implemented, their design, and the technological platform on which the implementation is based. Although Tellybean has had to adapt several times, the main vision of the company has not changed.

“The vision has stayed the same: lifelike video calling on your TV. It is very simple; everyone in the company knows it. The TV part doesn’t change, but the business environment is changing. The technology – the hardware and software – is changing all the time.” (Tellybean founder)

“We had to pivot when it comes to technology and prioritising features. But the main overriding is still the same: it's the new home phone and it connects to your TV. That hasn't changed. I see the pivots more like springboards to the next level. For example, we made a tablet version to [gain a distributor partner].” (Tellybean CTO)

In the first project, the new business analytics instrument allowed Tellybean to yield insights on their system's statistics, providing the company a means for feedback. They could gain a near-real-time view on call related activities, yielding business critical information for deeper analysis. The presence of the call data could be used as input for informed decisions. It also allowed learning about service quality and identifying customer call behaviour patterns. Based on the customer's comments, such information would be crucial for decision-making regarding the scaling of the platform. Excess capacity could thus be avoided and the system would be more profitable to operate while still maintaining a good service level for end users. The primary reason for wanting to demonstrate such capabilities was the need to satisfy operator needs. To convince operators to become channel partners, the ability to respond to fluctuations in call volumes was identified as critical. Potential investors would be more inclined to invest in a company that could convince channel operators of the technical viability of the service.

“There were benefits in terms of learning. We were able to show things to investors and other stakeholders. We could show them examples of metric data even if it was just screenshots.” (Tellybean CTO)

The high-level goal of the first project could be considered as defining a business hypothesis to test the business model from the viewpoint of the operators. The project delivered the needed metrics...
Building Blocks for Continuous Experimentation
Building Blocks for Continuous Experimentation

Infrastructure for Continuous Experimentation

Figure 3: Continuous Experimentation Infrastructure.

accessible through an API. Here, these parts should be understood as conceptual; an actual system likely consists of multiple APIs, databases, servers, etc.

4.2 Lessons Learned from the Projects

Startup companies operate in volatile markets and under high uncertainty. They may have to do several quick changes as they get feedback from the market. The challenge is to reach product-market fit before running out of money.

“You have to be flexible because of money, time and technology constraints. The biggest question for us has been how to best use resources we have to achieve our vision. In a startup, you are time-constrained because you have a very limited amount of money. So you need to use that time and money very carefully.” (Tellybean founder)

When making changes in the direction of the company, it is necessary to base decisions on sound evidence rather than guesswork. However, we found that it is typically not the product or service vision that needs to change. The change should rather concern the strategy by which the vision is implemented, including the features that should be implemented, their design, and the technological platform on which the implementation is based. Although Tellybean has had to adapt several times, the main vision of the company has not changed.

“The vision has stayed the same: lifelike video calling on your TV. It is very simple; everyone in the company knows it. The TV part doesn’t change, but the business environment is changing. The technology – the hardware and software – is changing all the time.” (Tellybean founder)

“We had to pivot when it comes to technology and prioritising features. But the main[...]

In the first project, the new business analytics instrument allowed Tellybean to yield insights on their system’s statistics, providing the company a means for feedback. They could gain a near-real-time view on call related activities, yielding business critical information for deeper analysis. The presence of the call data could be used as input for informed decisions. It also allowed learning about service quality and identifying customer call behaviour patterns. Based on the customer’s comments, such information would be crucial for decision-making regarding the scaling of the platform. Excess capacity could thus be avoided and the system would be more profitable to operate while still maintaining a good service level for end users. The primary reason for wanting to demonstrate such capabilities was the need to satisfy operator needs. To convince operators to become channel partners, the ability to respond to fluctuations in call volumes was identified as critical. Potential investors would be more inclined to invest in a company that could convince channel operators of the technical viability of the service.

“There were benefits in terms of learning. We were able to show things to investors and other stakeholders. We could show them examples of metric data even if it was just screenshots.” (Tellybean CTO)
Infrastructure for Continuous Experimentation

Figure 3: Continuous Experimentation Infrastructure.

accessible through an API. Here, these parts should be understood as conceptual; an actual system likely consists of multiple APIs, databases, servers, etc.

4.2 Lessons Learned from the Projects

Startup companies operate in volatile markets and under high uncertainty. They may have to do several quick changes as they get feedback from the market. The challenge is to reach product-market fit before running out of money.

“You have to be flexible because of money, time and technology constraints. The biggest question for us has been how to best use resources we have to achieve our vision. In a startup, you are time-constrained because you have a very limited amount of money. So you need to use that time and money very carefully.” (Tellybean founder)

When making changes in the direction of the company, it is necessary to base decisions on sound evidence rather than guesswork. However, we found that it is typically not the product or service vision that needs to change. The change should rather concern the strategy by which the vision is implemented, including the features that should be implemented, their design, and the technological platform on which the implementation is based. Although Tellybean has had to adapt several times, the main vision of the company has not changed.

“The vision has stayed the same: lifelike video calling on your TV. It is very simple; everyone in the company knows it. The TV part doesn’t change, but the business environment is changing. The technology – the hardware and software – is changing all the time.” (Tellybean founder)

“We had to pivot when it comes to technology and prioritising features. But the main ordering is still the same: it’s the new home phone and it connects to your TV. That hasn’t changed. I see the pivots more like springboards to the next level. For example, we made a tablet version to [gain a distributor partner].” (Tellybean CTO)

In the first project, the new business analytics instrument allowed Tellybean to yield insights on their system’s statistics, providing the company a means for feedback. They could gain a near-real-time view on call related activities, yielding business critical information for deeper analysis. The presence of the call data could be used as input for informed decisions. It also allowed learning about service quality and identifying customer call behaviour patterns. Based on the customer’s comments, such information would be crucial for decision-making regarding the scaling of the platform. Excess capacity could thus be avoided and the system would be more profitable to operate while still maintaining a good service level for end users. The primary reason for wanting to demonstrate such capabilities was the need to satisfy operator needs. To convince operators to become channel partners, the ability to respond to fluctuations in call volumes was identified as critical. Potential investors would be more inclined to invest in a company that could convince channel operators of the technical viability of the service.

“There were benefits in terms of learning. We were able to show things to investors and other stakeholders. We could show them examples of metric data even if it was just screenshots.” (Tellybean CTO)

The high-level goal of the first project could be considered as defining a business hypothesis to test the business model from the viewpoint of the operators. The project delivered the needed metrics
Case Study & Continuous Experimentation Model

Project 1

B Software business analytics system

M Data related to video calls to present to operators

L Operators requirements

Build-Measure-Learn

Project 2

B Software test tool for service infrastructure

M Operational capacity of infrastructure

L Infrastructure limitations

Build-Measure-Learn

Project 3

B Prototype for rapid deployments of software updates

M Continuous deployment capacity

L How to implement continuous deployment

Build-Measure-Learn

pivot

x Technology
x Architectural solutions
x Development methodology
Case Study & Continuous Experimentation Model

Project 1

Figure 2: Build-Measure-Learn Block.
Case Study & Continuous Experimentation Model

Project 1

Figure 2: Build-Measure-Learn Block.
Case Study & Continuous Experimentation Model

Project 1

Figure 2: Build-Measure-Learn Block.
Case Study & Continuous Experimentation Model

Project 1

Figure 2: Build-Measure-Learn Block.
Case Study & Continuous Experimentation Model

Project 1

Figure 2: Build-Measure-Learn Block.

Case Study & Continuous Experimentation Model
Case Study & Continuous Experimentation Model

Project 1

Figure 2: Build-Measure-Learn Block.
Case Study & Continuous Experimentation Model

Figure 2: Build-Measure-Learn Block.

Project 1
Case Study & Continuous Experimentation Model

Project 1

Figure 2: Build-Measure-Learn Block.
Case Study & Continuous Experimentation Model

Project 1
- B Software business analytics system
- M Data related to video calls to present to operators
- L Operators requirements

Build-Measure-Learn

Project 2
- B Software test tool for service infrastructure
- M Operational capacity of infrastructure
- L Infrastructure limitations

Build-Measure-Learn

Project 3
- B Prototype for rapid deployments of software updates
- M Continuous deployment capacity
- L How to implement continuous deployment

Build-Measure-Learn

pivot

x Technology
x Architectural solutions
x Development methodology
Case Study & Continuous Experimentation Model

Project 2

Figure 2: Build-Measure-Learn Block.
Case Study & Continuous Experimentation Model

Project 1
- B Software business analytics system
- M Data related to video calls to present to operators
- L Operators requirements

Project 2
- B Software test tool for service infrastructure
- M Operational capacity of infrastructure
- L Infrastructure limitations

Project 3
- B Prototype for rapid deployments of software updates
- M Continuous deployment capacity
- L How to implement continuous deployment

Build-Measure-Learn

pivot

x Technology
x Architectural solutions
x Development methodology

Department of Computer Science
Alejandro Sánchez Guinea
Building Blocks for Continuous for Continuous Experimentation
Case Study & Continuous Experimentation Model

Project 3

Figure 2: Build-Measure-Learn Block.
Summary

• Continuous Experimentation model supported by empirical evidence
  • Continuous cycle of experiments
    ➢ Experiments are conducted to validate the most important assumptions
  • Infrastructure for the model considers the roles, technical infrastructure, and information artifacts for running large-scale continuous experiments.
• Success factors for a Continuous Experimentation system
  • e.g., rapid and adequate design of experiments, proper instrumentation for collection and analysis of relevant data, feedback loops must be implemented
Challenges and Research Directions

- Expand the model based on more cases and in regard to domain-specific variants
- Answer questions related to specific parts of our model
  - how to build a back-end system suitable for large-scale continuous experimentation?
  - how to properly design experiments in order to reduce uncertainty in strategic assumptions?
  - how to ensure that experiments are trustworthy?
- Set up experimental cycle in hardware and software co-design
  - synchronization issues between hardware and software development schedules
- Experiments conducted with several kinds of stakeholders (e.g., investors, suppliers, etc.)
azsanche@cs.helsinki.fi

- Software Systems Engineering Research Group
  http://www.sserg.org
- Software Factory
  http://softwarefactory.cc
- Software Factory video
  https://www.youtube.com/watch?v=uPAE1uRP65Y
- Twitter
  @Juergen_Muench
Reference of the article


The article can be found online in:
http://www.sserg.org/publications/uploads/fedf90b60c9a480fddf403f127405a248d04ea55.pdf