CONTINUOUS EXPERIMENTATION COOKBOOK

An introduction to systematic experimentation for software-intensive businesses

Myriam Munezero, Sezin Yaman, Fabian Fagerholm, Petri Kettunen, Hanna Mäenpää, Simo Mäkinen, Juha Tiihonen, Leah Riungu-Kalliosaari, Antti-Pekka Tuovinen, Markku Oivo, Jürgen Münch & Tomi Männistö

DIMECC
RESULT PUBLICATIONS 3/2017

DIMECC PUBLICATIONS
SERIES NO. 15

Continuous Experimentation Cookbook

An introduction to systematic experimentation for software-intensive businesses

DIMECC
RESULT PUBLICATIONS 3/2017

DIMECC Program
NEED FOR SPEED

Authors |Myriam Munezero, Sezin Yaman, Fabian Fagerholm,
Petri Kettunen, Hanna Mäenpää, Simo Mäkinen, Juha Tiihonen,
Leah Riungu-Kalliosaari, Antti-Pekka Tuovinen, Markku Oivo,
Jürgen Münch & Tomi Männistö

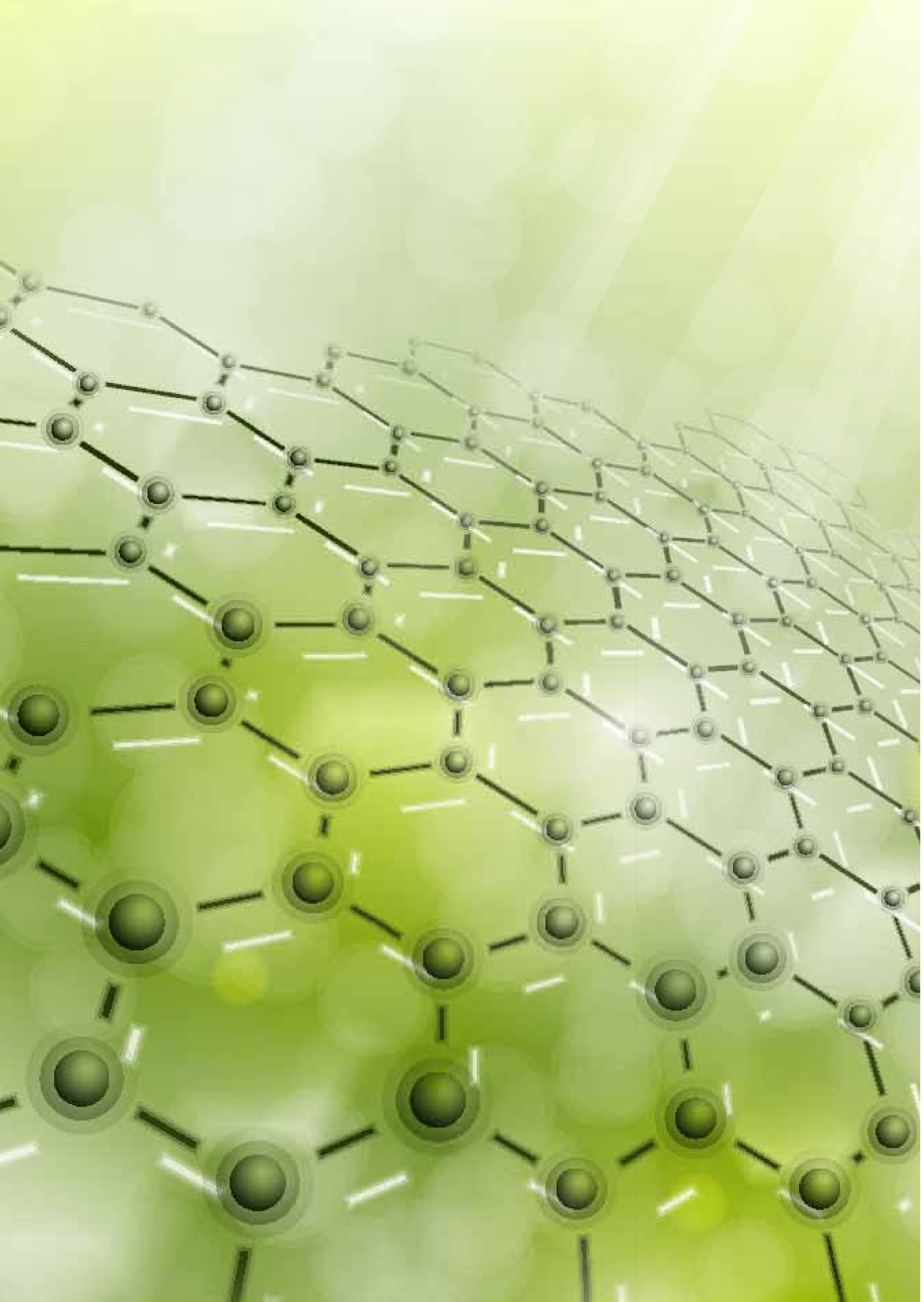
Cover | Marja Hautala, Muuks Creative
Images | iStock
Drawings | Simo Koivunen, Cultural Cooperative Vehrä
Design | Katja Kuuramaa, Cultural Cooperative Vehrä

Publisher | DIMECC Oy
Korkeakoulunkatu 7
33720 Tampere
www.dimecc.com

ISBN 978-952-68735-1-0

Helsinki, Finland 2017

Content



Executive Summary

If you only have time for one chapter, this is the one.

An increasing number of companies are involved in building software-intensive products and services – hence the popular slogan "every business is a software business". Software allows companies to disrupt existing markets because of its flexibility. This creates highly dynamic and competitive environments, imposing high risks to businesses. One risk is that the product or service is of only little or no value to customers, meaning the effort to develop it is wasted. In order to reduce such risks, you can adopt an experiment-driven development approach where you validate your product ideas before spending resources on fully developing them. Experiments allow you to test assumptions about what customers really want and react if the assumptions are wrong.

This book provides an introduction to continuous experimentation, which is a systematic way to continuously test your product or service value and whether your business strategy is working. With real case examples from Ericsson, Solita, Vaadin, and Bittium, the book not only gives you the concepts needed to start performing continuous experimentation, but also shows you how others have been doing it.

Continuous experimentation is also a means of expanding the viewpoint of product and service development. The focus should shift from just identifying and solving technical problems to identifying the right customers, identifying their relevant problems, and offering valuable experiences for them. While this can be based on experience, personal opinions, and guesswork, approaching the issue in a more systematic way will result in improved software products and services. The main idea is using customer behaviour data to test assumptions about products or services and to support decision-making. Figure 1 depicts the overall model for continuous

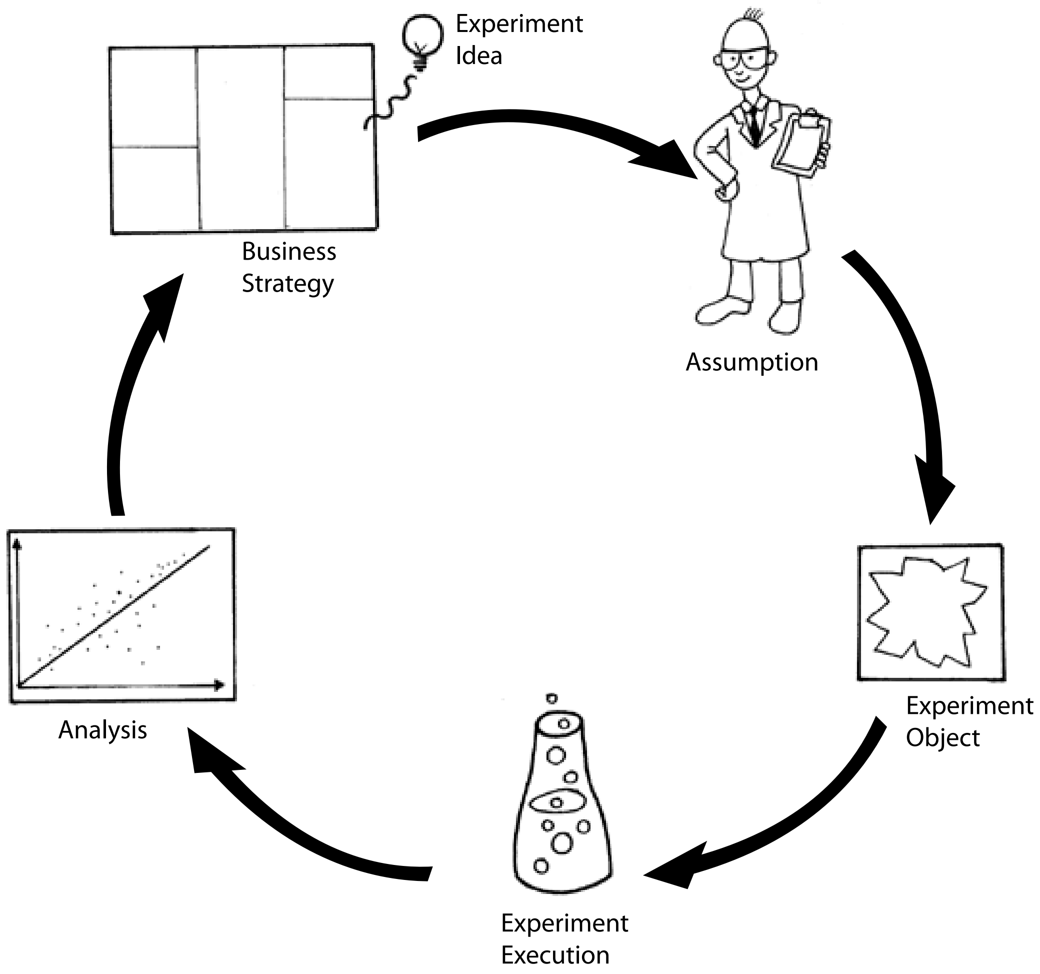


Figure 1. Continuous experimentation cycle.

experimentation from idea to explicated assumption and finally with data obtained form the experiment to decision-making and next experiment.

Development of software-intensive products and services increasingly occurs by continuously deploying product or service increments, such as new features and enhancements to customers. Each increment creates a slightly different customer and user experience. Businesses must continuously find out what their customers need in order to provide improved value. Business people and developers continuously collect direct customer feedback and observe usage behaviour, before the actual development project and even after delivery of the product or service to customers. In the web world, this is already the norm, but the examples and models in this book show that it can be done in other software-intensive industries as well.

Continuous experimentation, the approach presented in this cookbook, is a systematic approach for linking customer behaviour to development decisions. In addition to collecting data from product or service usage, one proactively introduces changes to the product or service as experiments in order to learn how the customer reacts to them, possibly changing the customer's behaviour. Collecting data on these actual usage changes allows informed decision making. Doing this continuously means that companies stay up to date with the direction of the market as they learn more and more about who their customers are and what they want – even as they change.

Experiments rely on small, fast, and cheap probes to create a more complex and dynamic strategy for the future. They can be conducted as a way to gain deeper customer insight, identify means to increase customer loyalty, product feature usage behaviour and in identifying the best business model. However, in order to conduct successful experiments, a few key elements should be understood, such as how to conduct them systematically and what the pitfalls are.

This cookbook gives an introduction to continuous experimentation and offers actionable recipes and tools to guide its adoption by businesses. The book offers concrete and practical advice with a solid theoretical background needed to conduct systematic experiments. The guidelines, tools, models, experiences, and examples given in the book have been developed in collaboration with researchers and software companies that are part of the Need for Speed research program. The company cases from Ericsson, Solita, Vaadin, and Bittium provided in this book showcase experiences from adopting and conducting experiments.

This book primarily targets practitioners involved in the development of software products and services who would like to adopt continuous experimentation as a way to improve the development or business processes. It is also useful for all those interested in the subject generally.

The book is divided into five main chapters. The first chapter gives an introduction to continuous experimentation and its main elements. Case examples, demonstrating experiments conducted by companies, are given in Chapter 2. Chapter 3 presents a collection of recipes to guide you when starting the continuous experimentation cycle. These are linked to the case examples to show their practical application. As conducting experiments does come with pitfalls, a checklist of things to avoid is provided in Chapter 4. Chapter 5 provides a sample of tools and models that can be used to structure the experimentation activity and reach results faster.

1 Introduction to Continuous Experimentation

Before attempting to make bold product or service development or business decisions, it is often best for companies to make sure that knowledge and data – and not just assumptions or opinions – are guiding the process. Experiments are an efficient way to obtain this knowledge and data, and bring objective assessments of initiatives and plans in business or development decisions. Experimentation brings knowledge and data from the real environment of products and services to back up decisions on guiding the future of the development.

Continuous experimentation is a development approach where the R&D and business process is guided by constantly conducting systematic experiments and collecting user feedback. It requires a company to utilise empirical evaluation of their offerings, e.g., features and products, in order to avoid unnecessary product risks. This helps to make data- and knowledge-driven decisions and to ensure that the development is focused on features that provide real value for customers.

Particularly in the software development domain, experimentation can support a wide variety of decision-making situations. It can provide answers to software development questions such as: Does the product or a feature solve users' real problems and thus provide value? Which of the alternative implementations do users like best? Have the customers changed their behaviour? Does the product (still) fit the market or a segment?

Haphazard or ad-hoc experimentation can produce interesting data, but may fail to reveal the reliable and valuable knowledge required to make good decisions. Systematic experimentation requires the ability to identify areas where experiments are needed, would be beneficial, and would be worth the effort. In addition, before conducting any experiments, stakeholders must agree on how they are going to interpret the results once they become available, in order to avoid a biased interpretation. Experiment results must not be ignored, even when they contradict the assumptions or intuition of management. Decisions should also not be made blindly out of the data, but usually needs a human to make an informed judgement.

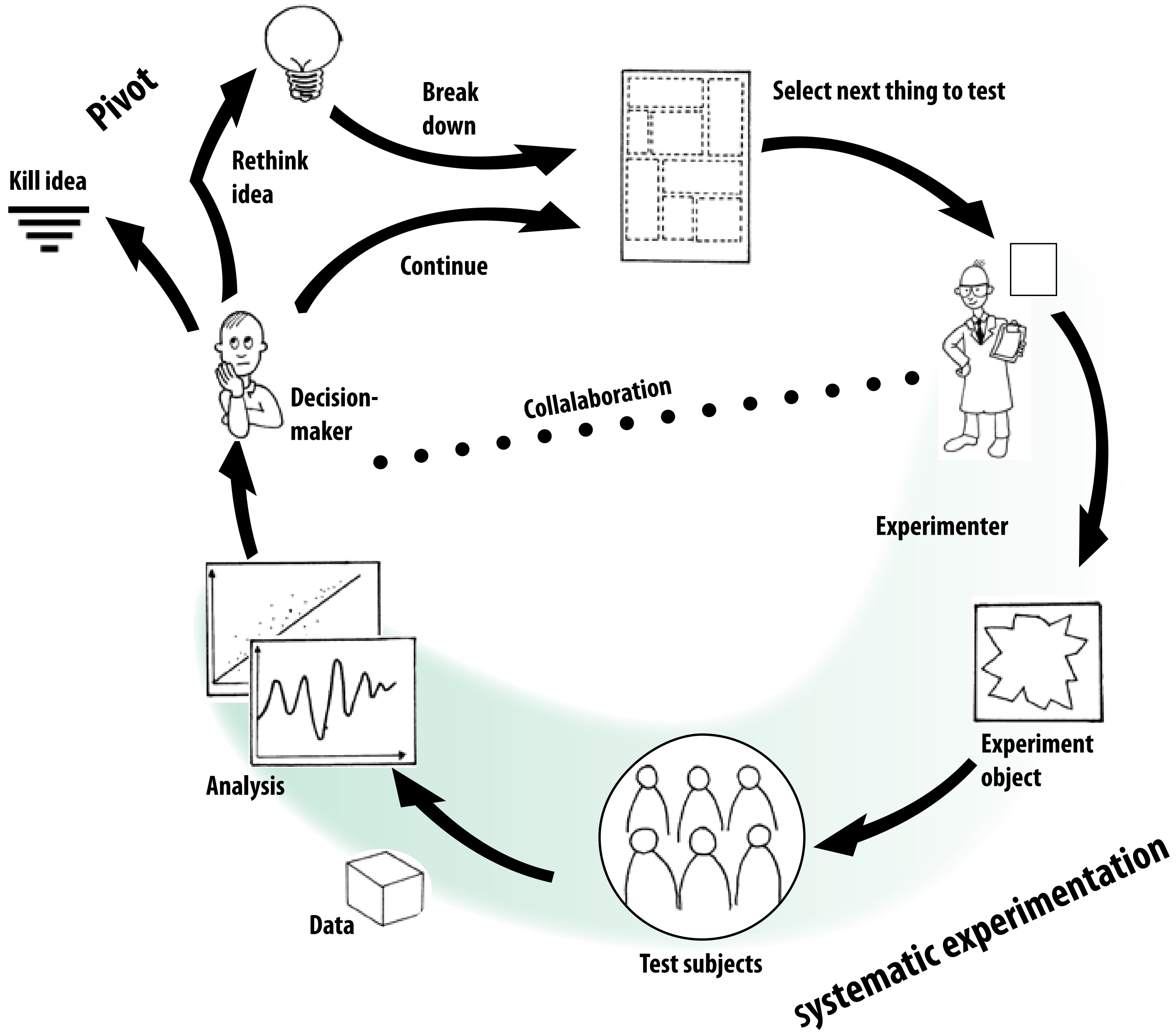


Figure 2. Continuous experimentation links decision-making to systematic experimentation. The systematic part ensures that there is a repeatable way of getting reliable data for decision-making.

The detailed cycle of continuous experimentation is shown in Figure 2. Continuous experimentation starts by identifying an idea that needs to be validated. The idea can come from various sources, but is often a part of the product strategy or roadmap. Once an idea has been chosen, it should be broken down into a set of assumptions. These assumptions are what the idea rests on – if they don't hold, the idea is invalid. The set of assumptions can be prioritised and the most important assumption is then selected for a more detailed, systematically designed experiment where the assumption is turned into a testable hypothesis. The experiment is designed so that after seeing the results, it is possible to make a decision:

- Kill the idea – the entire idea is so flawed that it makes no sense to continue with it.
- Rethink the idea – there is something wrong with the idea but the experiment indicates it could work in a different form.
- Continue – the experiment shows the idea is promising and the next assumption should be tested.

For conducting meaningful experiments, experimenters need to identify and separate an independent variable (the presumed cause) from a dependent variable (the observed effect) and hold all other potential causes constant. Then the independent variable can be manipulated through a treatment, in order to study changes in the dependent variable. Carefully conducted experiments can yield insights in company operations and test assumptions of which variables cause which effects. Experimenters need to have a clear purpose for the experiment and have a good hypothesis to test. For instance, a weak hypothesis (such as "we can increase our sales") doesn't specify a particular independent variable to test on a specific dependent variable, which makes it difficult to either support or reject that hypothesis. A good hypothesis helps delineate those variables.

One critical aspect of an experiment is how to achieve the manipulation at low enough cost and quickly enough in order to justify the experiment. In software development, the manipulation includes the creation of an experiment object, such as a minimum viable product (MVP) or minimum viable feature (MVF) that can be given to users. It is highly important that the experiment object is designed to represent the critical aspects of the final product or service so that the results achieved in the experiment can actually be used in decision-making.

The release of the experiment object is followed by careful observation, data collection and analysis, which yields insight into the relationships between cause and effect. There are many questions involved in designing and running the experiment. For example, choosing the right number of users to involve in the experiment not only has an impact on the statistical validity of the experiment, it also has an impact on the cost of the experiment. Deciding on how long to observe the users further depends on the variables and the success criteria included in the hypothesis, i.e., the criteria at which the hypothesis is falsified or validated.

The main challenges of continuous experimentation are how to form proper hypotheses, make good experiment designs, and utilise experiment results in decision making. As many companies start or continue to discover the benefits of conducting experiments, the next steps include creating value, which comes from analysing the data and utilising the results for decision-making.

2 Continuous Experimentation: Four Cases

Continuous experimentation includes many challenges that need to be solved for each situation. This chapter provides intuition for how to implement continuous experimentation through four company cases. Each case illustrates the complex details that need to be taken into account, and shows examples of how to tackle the challenges you may face when starting to do continuous experimentation.

Case Ericsson: First experiences in structured experimentation

The Ericsson case presents experiences from two teams within a large company that took their first steps towards adopting continuous experimentation in their development process.

Company and product description

Ericsson is a multinational communications technology corporation that provides equipment, software and services that enable the transformation towards a networked society.

One of their products enables telecom operators to offer connectivity management and billing services for their enterprise customers. One of its components is the Activity Log tool which provides the overview information about mobile subscription events, such as when a SIM card is registered on the network, a data transfer occurs, or an SMS message is sent. The tool is used by Ericsson's operator customers to troubleshoot problems with mobile subscriptions.

Aim of the first experiment

The Activity Log tool was a promising target for experimentation as many questions regarding its design were open. The product's feature requests were analysed and formulated as behaviour-driven development (BDD) stories. One feature was selected by the development team's technical coach to be the subject for experimentation.

The experiment aimed at testing options for feedback messages that a user of the Activity Log tool receives after clicking on a "reconnect" button which flushes a SIM card registration, meaning that the mobile device must reconnect in order to resume normal operation. A good feedback message was to inform the user on the current state of the connection and to provide instructions on what to do next so as to avoid the case where a user would continuously click the reconnect button, and clogging up the network.

Design of the experiment

Based on the selected feature, reconnect button, and its BDD story, the underlying assumption was identified: users will be able to know what is happening once they click the reconnect button. Based on the assumption, a hypothesis to be tested was formed. Subsequently, an experiment plan was drafted, detailing how to run the experiment in order to validate the hypothesis (see Table 1).

Two rounds of experiments were conducted with internal employees of Ericsson acting as test subjects. The experiments were run by the technical coach of the development team and two members of the user experience (UX) team. In the first round of the experiment, the new feedback messages were found to be unclear and misleading to the test subjects. At the same time, the people conducting the experiment had difficulties determining whether a user had succeeded in answering accurately to the criterion outlined in the hypothesis (see Table 1). Based on this information, the messages were updated and the experiment was run again. In this second round, the product's original feedback message was included in the set.

Table 1. The first experiment.

BDD Story	As an Activity Log user, I want to flush network memory for a subscription so that I can be sure that there is no mismatched information and next I can see when a device connects to the network.	
	First run	**Second run**
Hypothesis	We believe that with the right feedback message, users should be able to tell 1) what the state of the device's connection is and 2) what the next action is.	We believe that with the right feedback message, users are able to tell: 1) what is the next action to take, 2) what is the state of device's connection, 3) what to do if the device does not connect to the network.
Experiment plan	In order to validate this, users will be shown a set of feedback messages and will be asked to provide answers to the above two criteria. The message with the most yes answers for each criterion will be the best message and will be selected.	In order to validate this, users will be shown a set of feedback messages and will be asked to provide answers to the above three criteria. The message with the most yes answers for each criterion, especially for criterion 1, will be the best message and will be selected.
Experiment object	Five mock-ups (PowerPoint) with different feedback messages	Seven mock-ups (PowerPoint) with different feedback messages
Test subjects	Three internal company employees invited by the experimenters based on availability	Seven internal company employees invited by the experimenters based on availability
Experimenters	One person from the development team and one from the UX team	One person from development team, one from UX team and an additional observer from the UX team
Data collection	Yes or no scores for each test subject according to each hypothesis criterion as well as experimenters observations of test subjects during the experimentation.	
Total duration of running experiment	60 minutes	120 minutes
Data analysis	Ranking of the feedback message scores in order to identify the best message.	

Results

In line with the original hypothesis, it was observed that the test subjects were able to complete their task best when presented with a particular type of a feedback message. Based on the results of the experiment, the development team was able to select the best feedback message, which was included in the next product release.

Lessons learned

The development and UX teams mentioned that the process of experimentation had benefits on its own: "We have not done any structured experimentation before. Now we have the structure". In light of this, they plan to spread the experimentation culture within other development teams as well. Table 2 presents additional learnings gained by starting to conduct a structured experiment.

Table 2. Learnings from starting to conduct structured experiments.

Quotations	Learnings
"We have to start experimenting with something small, more importantly we have to start now. Practice will make it perfect."	It's possible to start with small teams and small-scale experiments.
"Piloting the experiment was important."	It's beneficial to do a pilot run before a real experiment.
"If there is a mistake in the experiment design, we should not dwell on it. We fix the experiment in the best way we can do and run it again. We learn so much with each experiment, over-planning (after certain amount of time) would be pointless."	It's important to realise when the design of the experiment is not appropriate and in consequence modify the design and rerun the experiment.
"Experimentation really moved along when BDD user stories were prepared and introduced."	It was useful to prepare the BDD stories as they were able to capture the user requirements and the underlying assumptions.
"It was possible to run experimentation with a simple [PowerPoint] mock-up."	It's possible to run an experiment effectively without much development effort and with low costs.
"Experimentation made it clear to the team that there is no need to debate between opinions as you can quickly test them with an experiment."	It's better to make decisions based on data than opinions and assumptions.

Case Solita: Setting a baseline for learning

The Solita case describes a project transitioning an existing product towards a new technological infrastructure, simultaneously enabling new business. The case aimed at understanding how a key feature in this product was used.

Company and product

Solita is a Finnish company that specialises in aiding the digitalisation of businesses and services. One of their products being developed for a client, a media monitoring service (MMS), allows real time analysis of online content, helping the client's customers to e.g. follow trends relating to their own and competitors' businesses. The MMS is based on customer specific search profiles that are trained by identifying and ranking relevant content items. Alerts of new content that match the client's or their customer's interest criteria can be received by email and the retrieved content can also be viewed from a web based reporting.

Need for experimentation

Solita had devoted significant development efforts on the web based reporting tool. However, it was uncertain whether the current customers of the MMS were content with accessing their results only as email alerts. If email was sufficient, the new report web-based tool would lack users. However, the report tool provided users with more features such as commenting, rating and sharing of reports. Hence, the company wanted to use experimentation to form an understanding of the role the email alerts play for their customers.

Design of the experiment

The email alert was the experiment object. As the email alert showed the media monitoring results as a summary with links to the source of the original content, the idea of the experiment was to design the email alert so that it redirected users to the reporting tool instead of the original media source and thus also increase the number of users using the report tool. Table 3 presents the details of the experiment.

Table 3. Experiment details.

Assumption	Linking the email alerts to the online report tool will drive more users to the online reporting tool.
Hypothesis	We believe that sending users an email alert linked to the report tool will increase the number of users coming to the tool from the email alert by 90%.
Experiment plan	In order to validate this, we will run a two week experiment where users will receive email alerts with links to the report tool and we will measure the users that come to the report tool through these alerts.
Experiment object	Email alert that is linked to the report tool and collects the users' movements from the email alerts to the report tool.
Test subjects	Ten beta testers of the MMS.
Execution and duration of running experiment	A developer at Solita implemented the MVF and instrumentation for data collection. The experiment was run for two weeks.
Collected data	The raw data form is a collection of timestamped events that are associated with a specific user and a specific report (except for logins that did not originate from a report link) and contains a data item. Data was collected in a database.
Data analysis	For each user, calculate ratio of "number of email alerts that brought the user to the report tool" to the "total number of email alerts sent to user" in the two-week period.

Results

Determining whether the experiment had been successful was not easy as the company had no previous usage data that the analysis results could be compared to. Therefore in the end, product management would need to make a decision on what the ratio results would mean for them and whether to keep the email alert which directs users to the report tool or change it. Performing an interview with users would assist in the decision making.

Next steps

As this was the first structured experiment the company performed, participants from Solita stated "having a structured and academic context ensured that the company participants thought through all the details, which probably would have been skipped in everyday operative work". Next steps include running the experiment again and using the current results as a baseline for setting a new hypothesis.

Case Vaadin: Conducting experiments to better understand customer needs

Vaadin improved their development platform using experiments with their APIs.

Company description

Vaadin builds and provides open source tools and components that make building web applications easy. Their tool framework includes APIs for developers and businesses.

Experimentation at Vaadin has been adopted as an approach to trying out new technologies and products, and for gaining business insight "fairly early" on. Vaadin tries to prototype and pilot all their products and services to get quick feedback from customers and try out different ideas. Through this, more information about the customer needs can be gained and the market understood better. This then allows them to focus on viable products and features and drop the ones that do not gain traction.

Need for experiment

Vaadin sought to improve their development framework by collecting and analysing API usage data in order to understand what APIs developers are using and also explore any potential issues in the APIs. In particular, Vaadin was interested in improving two key metrics in their framework: API quality and hit-rate. The API quality refers to the manner in which an API is used by developers, usage patterns of APIs and issues involved; hit-rate refers to how often APIs are being used. Although there were known issues with these two metrics, Vaadin was interested in finding unknown issues, for instance, boilerplate code or antipatterns (workarounds that developers are forced to use due to lack of support from the technology or API) of usage.

Purpose of the experiment

Collecting API based information would help Vaadin steer their development efforts to the right parts of API development, give them the ability to use their limited resources wisely, address any issues or fix bugs with the most impact first, fix API issues that affect the most users, and thus consequently improve API quality and hit-rate. In return, user satisfaction for the framework and the APIs could improve as well.

Conducting the experiment

The experiment was conducted in collaboration with Codetrails and Åbo Akademi University. Codetrails is a company which creates tools for Eclipse that aid a programmer with, for instance, better code completions, knowledge transfer, and API usage analysis for enhancing development efforts. Vaadin and Åbo Akademi co-operated and initiated experiments to see what could be done for Vaadin programmers, and what usage information could be collected. The experiment design was exploratory in nature as the main aim was to identify how API usage data can be used as a decision making tool to assist API developers to make better decisions about API development.

The experiment was started by collaborating with Codetrails. With Codetrails, Vaadin developed a plugin prototype called Vaadin Insights that could be enabled for usage data collection. The prototype development took 3–4 calendar months.

The API experiment was conducted internally in two separate trials with a limited amount of test subjects. The first trial lasted for about one month, and the second one was a snapshot from several large projects (i.e., they analysed the API usage at specific points of time). By first conducting experiments internally, they were able to fix multiple issues and to develop the analysis further.

In addition to the data collection plugin, Vaadin had developed a result analysis tool using Python data mining libraries as part of their tool chain. This also included a report generation tool that was developed internally at Åbo Akademi. Both of these tools were used for the analysis of the collected usage data. Analysis was performed both quantitatively and qualitatively. The quantitative data was provided by the tool chain. It provided information on the number of patterns identified as unexpected and actionable.

Results

The tool chain provided 144 interesting patterns, such as methods commonly used together, from 400+ KLOC (Thousand Lines of Code) to the developers in an automatically generated report, out of which 20 actionable items, i.e., product improvement actions, were identified by the Vaadin experts. Based on the analysis from Vaadin experts, the plugin and insights tool were updated accordingly. For example, they identified needs and ways to get more representative data with the tools. The experts identified what action related to API structure needed more attention, and they also gave their opinion on how the collected information can be used for better decision making.

Lessons learned

By conducting experiments fairly early on, Vaadin is able to get more information about their customer needs and understand the market better. As Vaadin’s key account manager Pekka Perälä puts it, experimentation allows them to “focus on viable products and features and drop the ones that do not gain traction.”

Conducting the API experiment has helped Vaadin pay more attention to how their users behave and to understand how important it is in product and service development. They have now started, and plan to continue, to include data gathering as part of their development process. For instance, they have started performing structured usability experiments on all their products.

As Vaadin has experienced, conducting experiments does have some challenges, many of which they have managed to solve. Their biggest challenge however has been making experimentation as part of the daily work and in the backlog of the development teams. Changing the culture of doing things at Vaadin has also been a challenge. Solving these is still work in progress at Vaadin.

Case Bittium: Experimenting towards innovation

Bittium incorporated continuous experimentation in their innovation process to make it more agile and lean, and to improve its performance. This is in contrast to previous experimentation cases where experimentation was used to guide software product development.

Company description

Bittium is a B2B provider of embedded systems for the wireless communications industry with more than 500 employees in four countries. The company had used the traditional stage-gate model for their ideation for over thirteen years. In the spring of 2014, they decided to adopt a more experimental approach to their idea harvesting, focusing, and validation stages in order to speed up their innovation process, gain better fit of ideas to the company's business targets and reach more radical business innovations. More specifically, the company wanted to meet four targets: 1) Harvest more ideas within the company, 2) Grow ideas faster into business innovations, 3) Capture ideas with better fit for purpose, and 4) Improve the participation of various company stakeholders in the innovation process.

Need to incorporate experimentation in the innovation process

As an established technology company operating in an environment where the technology push and market pull are high, Bittium needed to maintain their current business and simultaneously develop new business ideas. Before incorporating experimentation in their stage-gate innovation process, ideas were focused only on creating intellectual property rights. The processing of ideas involved only a limited number of experts so the ideas did not spread to others. There was also a gap between management running the business cases and specialists who had the ideas. Mutual visibility was missing. As a result, collected ideas seldom supported the business goals.

As a solution, continuous experimentation and transparency were incorporated into the stage-gate innovation process.

How experimentation was incorporated in the innovation process

Bittium's idea screening and feedback process includes continuous iterations of idea harvesting, focusing and validation activities with established practices. Experimentation focuses on developing early and low-cost demos of ideas and collecting feedback for early validation, to figure out what ideas work and have potential. The light demos (i.e., experiment objects) create a common understanding of an idea within the company. When all participants understand an idea similarly, collected feedback is more relevant and constructive. The plan of each light demo is visible and iteratively maintained in a centralised system. This supports continuous learning about and feedback on ideas. Idea screening is frequent and transparent, which leads to early idea validation and short process lead times. As a result, many ideas reach the maturity level required for business decisions much faster than before. Ideas visualised with light demos, collected feedback, and screening decisions give the management validated information for making better informed business decisions. Therefore, Bittium management sees experimentation as a significant part in the innovation process: give ideas an opportunity to enter the demo stage, and learn what works and does not work by trying things out.

Continuous experimentation is a way for Bittium to get early validation for ideas without high expenses, as well as to give many ideas an opportunity to show their potential. Bittium recognises though that it is not reasonable to expect to create an innovation, especially a disruptive innovation, with a single experiment. Instead, a series of experiments are needed to evolve the novel solution.

Lessons learned

An established technology company like Bittium needs to be simultaneously efficient and profitable in their current business and also flexible in developing their future business opportunities. Incorporating continuous experimentation and transparency into the stage-gate model allowed Bittium to succeed in both. Good practices of the stage-gate model were combined with the experimental approach and free idea evolution. The stage-gate model has systematic practices that support idea validation and idea growth in a way that connects ideas to business targets. Furthermore, practices supporting long-term strategic planning were retained – innovation experimentation cycle does not include corresponding practices; therefore conducting experiments only is not viable. The inclusion of experiments allowed them to achieve rapid and radical business innovations and it was found to support the company's radical thinking of ideas and early collection of feedback.

3 Recipes

In this chapter, we provide recipes to guide you in actionable and concrete terms when starting to conduct experiments. Each recipe includes information on where and when it should be used, how to perform it and how to present the outcomes, as well as particular pitfalls and cautions to be considered. Recipes offered in this section are in the natural order of a systematic experimentation cycle (see Chapter 1) and they are enriched with examples, especially referring to the cases presented in Chapter 2.

Recipe 1

Setting clear goals for the experiment

Before starting an experiment, you should know why you're doing it! How will the experiment results contribute to the high-level goals you have for your product or service idea, business strategy, ways of working, etc.?

As an example, Airbnb's high-level goal was to increase its revenue and it does this by increasing the chances that its renters will have more customers. Airbnb had an assumption they wanted to test (i.e., 'apartments with high-quality photos are rented more often'). This has the crisp goal of increasing the number of nights a listing is rented which is directly connected to the higher-level goal of increasing revenue. Based on the experiment results, Airbnb hired photographers to take professional pictures.

Identifying clear and relevant goals is not trivial and requires domain knowledge and expertise.

Where to use this recipe	Apply this recipe when the development of your product/service/business calls for objective data. Objective data can be needed to make a decision on whether to apply an idea or which alternative idea to adopt, to verify an assumption, to settle a difference of opinions or to resolve an uncertainty.
What you need	• Product roadmap, vision or strategy • Decision maker(s), • An idea that you wish to test
Directions	• Place the idea you want to test, which could be an uncertainty or assumption on the roadmap, vision or the business strategy you have. If this is not possible, determine the items on the roadmap that will be affected by the experimentation. Or, determine what other major aspects such as user experience or non-functional properties will be affected. • Consider, what implementing that idea, settling that uncertainty or testing that assumption would mean for your product and business vision. • Together with the decision maker(s), write down how the results, whether negative or positive, will be implemented.
What to expect	A clear sense of what running the experiment will mean for the product, service, or business development.
Presentation	Document or display the crisp goals of each experiment and how they are aligned with the higher-level business goals and strategy of the company where relevant people can see it.
What next?	Forming a testable hypothesis to guide drafting the experiment plan.
Pitfalls	• Yielding goals that do not relate to company's strategy and vision and with no connection to the product or service roadmap. • Avoiding to agree early on what will be done with experiment results.
Case example	Case Solita provides an example of having a crisp goal for an experiment, i.e., increasing the number of users for a product which will increase product value.

Recipe 2

Defining a concrete hypothesis

An effective hypothesis is the core and kernel of a successful experiment. A hypothesis is a tentative, testable answer or "an educated guess" to a scientific or business question. A relevant hypothesis provides an answer to a question of interest. A well-formed hypothesis makes it possible to plan experimentation systematically.

This recipe shows an example of forming hypotheses.

Where to use this recipe	You need to have a hypothesis to get started with experimentation and finally get the results for supporting "go/no-go" decisions or determining the best alternative.
What you need	• An assumption • Clear goals (Recipe 1) • Success criteria
Directions	• First, take your assumption and clarify it. • Start with writing down the following statement: • We believe that [insert assumption, .i.e., (Airbnb) travelers will book more properties because of professionally photographed listings], which will [insert goal i.e., improve the property bookings], by [insert success criteria i.e. X%]' • Revise it until it is concrete enough to be tested.
What to expect	A clear measurable hypothesis ready to guide designing the experiment object (Recipe 3) and the experiment plan (Recipe 4).
What next?	Take the hypothesis formed with this recipe as input for Recipes 3 and 4.
Pitfalls (caution)	An unclear and unmeasurable hypothesis will lead to inability to make decisions or to make wrong decisions based on experiment results. Thus set it carefully! A hypothesis has to be specific and measurable – avoid using hedging words like "maybe", "better", or "some" when forming the hypothesis.
Case example	Case Solita Hypothesis: "We believe that sending users an email alert linked to the report tool will increase the number of users coming to the tool from the email alert by 90%."

Recipe 3

Designing and using the experiment object

Designing the experiment object means planning the intervention or treatment that you are interested in testing, and setting things up so that you can observe how the treatment causes changes in the outcome you are interested in.

Where to use this recipe	After drafting your hypothesis and are starting to draft the experiment plan (Recipe 4)
What you need	• An hypothesis (Recipe 2) • Metrics to collect • A product, service or feature
Directions	Based on the metrics you need to collect to validate the hypothesis, take your product, service or feature, (e.g., the web user interface in the case of Airbnb experiment mentioned in recipe 1), then: • Implement the intervention, (e.g., user interface with better pictures) • Implement also a data collection mechanism, (e.g., recording the number of clicks on the 'book' button) (Recipe 6 and 7) • Decide where to store the collected data (e.g., in a database or text file) (Recipe 7)
What to expect	An experiment object ready to be used for Recipe 5, 6 and 7.
What next?	Use the experiment object to start executing your experiment on.
Pitfalls	Not instrumenting the experiment object correctly to collect the needed metrics to validate the set hypothesis.
Case example	In the case of Solita, the experiment object was the email alert feature, which was implemented to collect user clicks and link those clicks to the report tool. Note that in the case of Ericsson, they made use of a prototype of the experiment object, which is also a possibility if carefully designed.

Recipe 4

Setting up a systematic experiment plan

So you have a hypothesis and a planned experiment object – but how should you test the hypothesis on the experiment object in order to validate or falsify it? This takes some creativity and experience. Consult data analysts and other relevant people in your company to confirm the metrics to be collected and measured, and also developers on the feasibility of collecting the needed data. Planning an experiment is an iterative activity, and you need to be able to switch between brainstorming and checking the logic of your plan.

Where to use this recipe	When you have selected the assumption you want to test and have formed your hypothesis (Recipe 2).
What you need	• A hypothesis (from Recipe 2) • An experiment object (from Recipe 3) • Test subjects • Other resources
Directions	• Start by taking the hypothesis and think carefully what metrics you need that would allow you to (in)validate the hypothesis. Often the hypothesis already includes some hints regarding what metrics are needed. For instance, in the Airbnb hypothesis, property bookings is an obvious metric. • Then decide how long to run the test such that will be sufficient for collecting enough data. For example, would 1 week or 1 month be better to measure a change, for instance, in property bookings? • Take your experiment object and modify it so as to allow it to collect the set metrics if you have not done it yet. Here decide also where to store the data and in what format. See Recipe 3 and 8 for ways to do this. • Note that if you have existing data (see Recipe 6), then you might not need an experiment object. • You need test subjects to use your experiment object, thus you need to decide who these are and how many of them you need. These could be real users/customers, or proxy users such as internal employees (see case of Ericsson). • Set the data analysis strategy and how the results will be presented. • Document the activities and results along the steps, in an iterative way.

What to expect	An experiment object ready to run.
What next?	Use the experiment object to start executing your experiment.
Pitfalls	Not instrumenting the experiment object correctly to collect the needed metrics to validate the set hypothesis.

Recipe 5

Testing your experiment using mock data

As designing an experiment plan is a creative process, there are times when you would like to test your experiment plan before you run it. Using mock data is a useful and low-cost way of achieving this. When using mock data, you generate one or more sets of plausible results that allow you to think through the analysis. These can allow you to then discover flaws in your experiment design and fix them before you run it for real.

Where to use this recipe	It is always useful to test your experiment before running it. Especially in cases where the analysis is difficult or complex.
What you need	• An experiment plan or at least an initial one (Recipe 4). • A way to generate mock data – preferably automatic, especially if you expect to have a large amount of data in your experiment.
Directions	• Start by listing different possible outcome scenarios of how the experiment could turn out. Remember, you don't know the result at this point, but you can list many possible outcome scenarios. • Once you have the scenarios, ask yourself what data would result in each outcome. • Then, generate data of that kind. If you expect to have only a small amount of data, you could simply construct it by hand using a spreadsheet or text editor. If you expect to have a large amount of data, you can either make a program that generates it, or you can use existing data (see Recipe 6) and alter it to fit your scenarios.
What to expect	Running the experiment with mock data allows you to test the experiment before spending time and other resources on running it. If done right, you will increase the chances of getting a correct and valid result, and can eliminate potential biases.
What next?	You can alter the plan based on the results received at this stage and go forward to executing the plan for real.
Pitfalls	You have to be sure that your mock data is identical in form to the final data that you will use. Otherwise you have tested the wrong thing. Note that if you test the experiment on mock data based on existing data, you should not run the actual experiment on the same data because you may have tuned the experiment to give you the result you expect using the mock data.

Recipe 6

Using existing data to test hypotheses

You may already have the data you need for your experiment. Web server logs, authentication system logs, and other runtime information from an existing system may allow you to conduct your experiment on existing data. Before spending time on collecting new data, identify whether you already have suitable data and ensure that it really can be used to validate your hypothesis.

Where to use this recipe	After you conduct experiments on an existing system – even a prototype – and/or whenever the users you are interested in are already using a system you can access.
What you need	• Experiment plan (Recipe 4)
Directions	• If you suspect you already have suitable data, you should start by listing the data you already have. • Match this list against the metrics you need to validate your hypothesis. • Extract a small sample of existing data that matches your metrics and test if it really is as you expect.
What to expect	You may be able to reduce the cost of an experiment by using existing data. However, do not underestimate the cost involved with extracting and preparing the data.
What next?	Extract a bigger sample of the identified data, and move into the analysis phase based on your experiment plan.
Pitfalls (caution)	• Using existing data may be easy, but is it really suitable? Events in the past may have caused effects in the data that may skew your experiment results. Also, the data may be from users who are different than the ones you are interested in. Make sure you know the data you are using, and think about the validity of the data in advance. • Looking at the data before designing the experiment. If you look at the data first, chances are you will become biased by it and design your experiment to confirm your prior belief rather than learning anything from the data. • Do you have permission to use the data for your experiment? When users signed up, did they agree for their data to be used in the way you would like? Ensure that you are not overstepping the legal or ethical boundaries of data use.

Recipe 7

Collecting the right data in the right format

To enable efficient data analysis, the data needs to be collected and stored in such a way that it is easy to get the desired results with enough accuracy.

The data can be collected in various ways. You may have an automatic system which can collect information about events in the system, you may ask your users to perform specific actions which you record, or you may even collect the data by hand while observing your users. For these and all other scenarios, make sure you think through what form of the data you will be collecting, and how it will be stored.

As an example, let's take a polling mechanism for determining duration of use: The experimenter wants to know how long users spend in a specific section of the system. The developer implements this as a polling mechanism which inserts a log event with some interval that doubles every time. Although this saves resources, analysing this event data is difficult and the accuracy may be less than what was assumed for the experiment. In this example, if the user closes the system just before the next polling event, the timing will be as inaccurate as the polling interval, meaning the analysis of the experiment data will be harder and the accuracy may not be enough to make a decision.

Where to use this recipe	When you need to collect new data for your experiment and when you need to know what data to store and how
What you need	• Experiment plan (from Recipe 4) • Metrics to collect
Directions	• If you are collecting data automatically from system events, you need to analyse your system and decide what format to collect the data in and where to store it. This depends completely on your experiment plan and your system. • If you are asking your users to perform specific actions which you record, you need to make sure that recording points are designed well so that you get the data you need. If you are collecting the data manually while observing the user, you need a data entry form that allows you to quickly record what you see. • Have the necessary information available and not in a too difficult form or organisation for the later analyses.
What to expect	Good-quality and relevant raw data.

Presentation	The last phase of collection is to gather the data you have collected in one place. Usually, all of it should be in a single set of files or in a database.
What next?	Once you have the data, move into the analysis phase.
Pitfalls (caution)	You might end up with bad data because you forgot to include something like a timestamp, or information that links a sequence of events. If you are collecting data manually, you may be overloaded while trying to both observe and record at the same time.

Recipe 8

Preparing for decision-making

Analysis of collected data can be done in various ways depending on context. For instance, in the Solita case, the ratio of "number of email alerts that brought the user to the report tool" was calculated for the data analysis. After that, decisions based on the analysed data must be made with respect to the hypothesis and set goals. If the analysis results shows that the hypothesis is falsified, i.e., do not meet the success criteria, this implies that the tested assumption was wrong and not based on any actual data. Therefore a decision could be made to redesign the experiment plan to collect more data, change and re-test the hypothesis or to kill the idea altogether. However, if the hypothesis is validated, then the idea can go forward and be implemented.

Where to use this recipe	Use this recipe after having analysed the collected data.
What you need	• Experiment results • Initial set goals (Recipe 1) • Success criteria specified in the hypothesis • Domain experts
Directions	• Take your experiment results and compare them with your success criteria. • With domain experts, judge how to implement the decision that was discussed before the experiment started, whether it was to kill the idea or implement it, or take a risk and do something different.
What to expect	You make decisions that are empirically justifiable.
Presentation	Put the results in a proposal form or report for the rest of your team members to enjoy. Present the final results and or decision to relevant parties, include the impact, and your learnings as well for a true knowledge-sharing party.
What next?	Implement the insights gained from the results, either to continue development or pivot the product.
Pitfalls (caution)	• Not ensuring that there is the ability to properly define decision criteria and act on experiment results. • Overconfidence and reliance on experiment results. There is a responsibility to making decisions and that responsibility should not be outsourced to experiment results alone.
Case example	In the Ericsson case, a decision was first made to rerun the experiment after results were found to be wrong and another decision to implement the idea based on valid results.

4 Pitfalls

The following list includes pitfalls typical in continuous experimentation. Some pitfalls apply more generally, while others are very specific. Read the list and check the box when you know you have avoided falling in the particular pit.

□ **Lack of vision**

Software quality requirements and baseline architecture goals should be addressed before starting development. The lack of a clear vision of the product can hinder

appropriate development work and increase the overall cost of the innovation effort. In addition, it can lead to misunderstandings and slow development due to unnecessary specification changes and variance in the project scope.

□ **Prophecy vision**

Sometimes the product vision is only a fantasy which is not in line with the development team's ability to produce software. While this could provide a valuable learning experience for the customer, it may lead to exhaustion of the development team. The vision must be clearly articulated and understandable by everyone involved.

□ **Vision not properly translated into usable guidelines**

In addition to a product vision, more specific and usable guidelines must exist to steer concrete development and experimentation tasks. Otherwise the development team may start making uninformed interpretations and the created product will not match the original vision. A lack of usable guidelines may also create confusion in task and feature prioritisation, and can lead to impaired collaboration between stakeholders due to misunderstandings.

□

We know what is good for our customers

Insufficient voice-of-customer work in product design contributes to low customer acceptance. Establishing software design on non-validated assumptions about the end-user or letting technological solutions shape the product risks the project creating software that nobody wants. Neglecting validation of the product concept or features with real end-users contributes to poor user acceptance.

□

Experimenting without figuring out and explicating the right hypothesis

Some things are good to know, but not necessarily that valuable. As experimentation is about testing a hypothesis, it is very important that it is linked to a real goal. It should be clear where and how testing the hypothesis will provide value.

□

Ambiguous hypotheses

When it comes to systematic experimentation, the devil is often in the details. That is, with the slight variation in the wording, you may actually run an experiment on something else, thus potentially yielding useless results and wastage of effort and resources.

□

Testing ideas from data that you already have without a clear hypothesis

Companies often collect a lot of data in the hope that analysing it will lead to something useful. However, when what you would like to answer is not defined beforehand, you might end up realising that you collected all that data for nothing or even you did not collect the right data. Moreover, starting from data places a lot of emphasis on the data scientist's ability to analyse and interpret the collected data in some way that would be meaningful for the business.

□

Buying or building a tool for an unvalidated purpose

Buying or building a data analysis tool without defining the purpose beforehand may lead to wastage of effort and resources. It is often easier and clearer to buy a tool, as you know what you are getting, but it may be that the tool does not actually contribute to solving the problem you are about to address.

□

Not ensuring that you are able to act upon experiment results

It is crucial that as a company or team, you have the ability to properly define decision criteria and act on experiment results. The pitfall is failing to make sure that the results will actually be taken into use. In particular, you should have a means to handle situations when the results contradict the assumptions or intuition of experts, or other forms of organisational resistance, e.g., when challenging long-standing practices or conventional wisdom. This however, does not mean, you should blindly trust the experiment results.

□

Running experiments in parallel without considering how they interact

Having several experiments run in parallel presents a particular challenge. Statistical interactions between experiments should be considered in order to assess the trustworthiness of the experiment results. For this reason, it is important to coordinate the design and execution of experiments so that correct inferences are drawn. If experiments are run in different parts of the organisation, one experiment may influence another without the experimenters realising it. More generally, the issue of validity becomes important when the entire R&D process is experiment-driven.

□

Overconfident reliance on data or experiment results

One should take certain care when interpreting the results of an experiment (as with any data). More importantly, there is a responsibility to making decisions and that responsibility should not be outsourced to data or experiment results alone. Decision-making should remain with the actual decision makers. (For an example of data-driven decision-making and its potential dangers in comparison to using data to inform and support your decisions, see Sebastian Wenicke's (MIT) TED talk: http://www.ted.com/talks/sebastian_wernicke_how_to_use_data_to_make_a_hit_tv_show)

□

Jumping to conclusions too fast

Experimentation with customers is inherently about working with a moving target, as the behaviour of customers is subject to change as a result of introducing new products, services or features. It might take time to notice a change among the customers. Thus, there is a risk of stopping the experiment too early, before collecting enough data to observe the change, which will result in misleading experiment results.

□

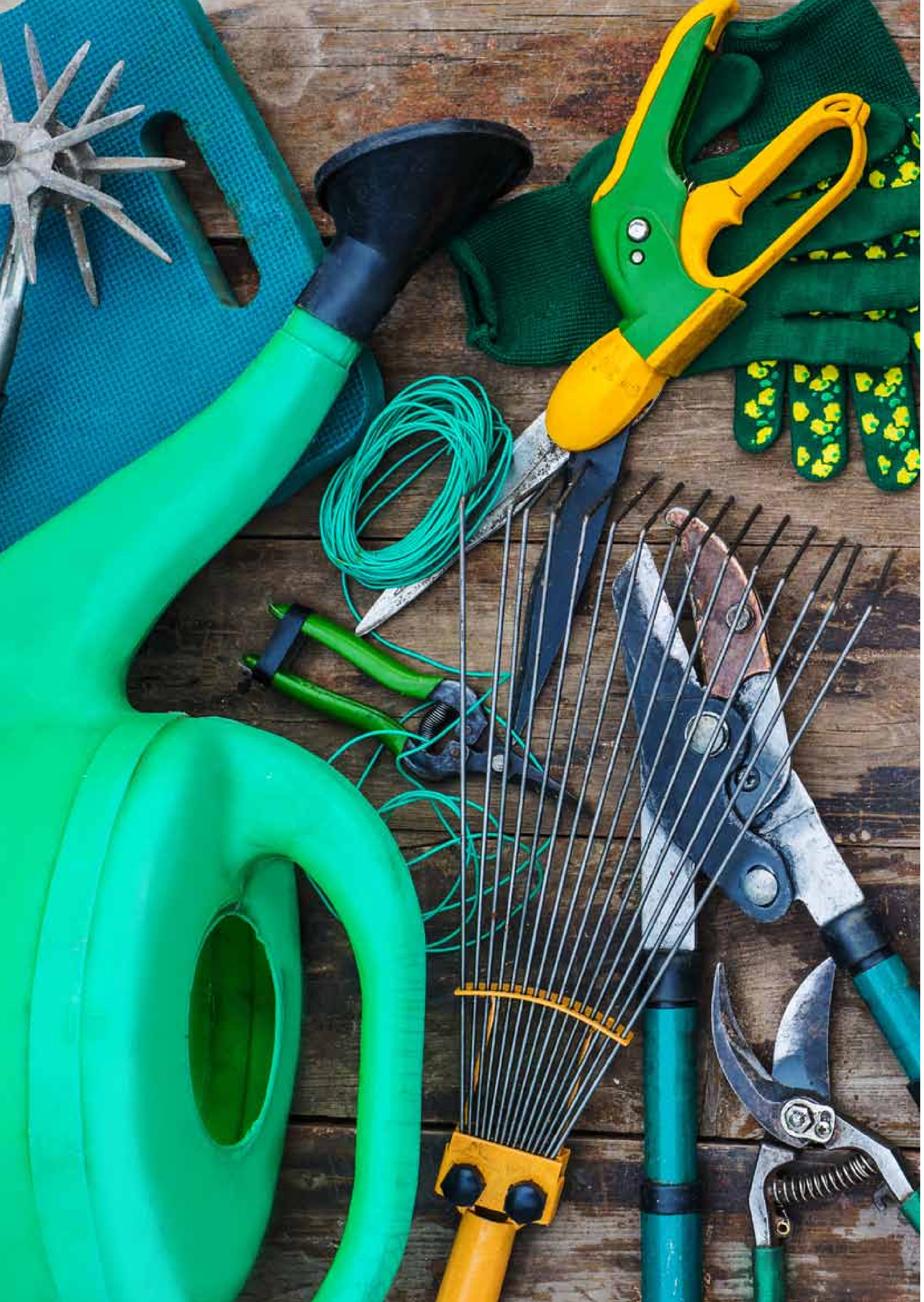
Confusing ad hoc and systematic experimentation

Continuous experimentation is a systematic approach for understanding customers and acquiring the data needed for guiding product and service development decisions. However, not every trial you may conduct or thing you test should be called an experiment, nor are they continuous. The hallmarks of systematic experiments are the presence of a clearly formulated, up-front hypothesis and a logically solid experiment design where cause and effect are separated.

□

Using experiments for everything

Continuous experimentation is only one approach for understanding customers and acquiring the data needed to guide product and service development decisions. There are other means that are called for in different situations, such as usability tests, service design methods, market research tools, etc.



5 Tools and Models

This section presents tools that can be used to support different parts of the continuous experimentation cycle. In broad terms, the focus is on discovering, describing, and modelling customer value, making business models, and understanding the technical infrastructure and architecture needed for automated, large-scale field experiments with real users.

The Wheels of Value Model

The Wheels of Value Model is a strategic tool for quickly identifying and documenting the value mechanics of business models with many actors. Instead of identifying assumptions for each element of a business model it generates closed value chains among the right actors and ensures that important links are not missed. By doing this, critical assumptions will be unearthed and visualised.

Organisations are increasingly operating in complex business environments with many actors. One of the reasons is that innovative products and services often need to be created and provided together with an interdependent network of partners. All actors have their own needs and the satisfaction of these needs depends on the interplay between the actors. Such complex business environments are often referred to as ecosystems. Companies in such environments frequently struggle with deciding on how to connect and interact with other actors. The big picture that visualises the connections and reveals the success-critical assumptions is often missing.

The Wheels of Value Model allows you to invent, design, sketch, change, analyse, and validate a multi-actor ecosystem with the aim of creating a successful and sustainable business. It is a simple and fast tool. It visualises multiple actors as well as their needs, behaviours,

The Wheels of Value Model is a new tool that integrates and synthesises many important components of product management, lean thinking, and business model generation. The Wheels of Value Model allows quick visualisation of the value mechanics in multi-actor business models. It focuses on the value chains that are most important in creating a good customer experience and a profitable business. Sketching a Wheels of Value Model reveals assumptions that are present in the business model and helps to identify risky assumptions that should be tested first. Testing the assumptions using continuous experiments can help to avoid flaws in the business model and reach a working business model faster. The Wheels of Value Model was designed by Jürgen Münch and has been successfully used with companies of different sizes in the context of the Finnish Need for Speed research program. It will hopefully evolve with further applications. Your help is appreciated: try it out and see how it works in your environment. For sharing your experience and feedback please contact Jürgen Münch (j.muench@computer.org).

and capabilities and integrates them into a big picture. It helps to unlock business and user value and to identify important business model assumptions. The Wheels of Value Model can also help you to get a shared understanding of an ecosystem and its business dynamics.

The Wheels of Value Model can help you, in complex business environments, to unearth critical value-related business assumptions and to make informed decisions on what to develop.

Sketching Wheels of Value Models

A Wheels of Value Model is a visualisation of all relevant actors in a business environment and their value-related connections. It is typically developed and sketched collaboratively by considering the following elements:

Actors

Actors have needs and demands for satisfying these needs. Actors can also offer capabilities that contribute to satisfying needs of other actors. Actors are individuals, organisations, or groups. Examples for actors are "health care professional", "fashionable women getting married", "delivery team", or "operator of mobile services". The following questions can help to find actors:

- Who has unfulfilled needs?
- Who could offer capabilities that support the satisfaction of these needs?
- Who else could contribute to satisfying these needs?
- Who could obstruct the satisfaction of these needs?

Needs

Needs describe what actors want or value. Needs are motivating forces that require action for their satisfaction. We focus on underserved needs, i.e., needs that are important and currently not adequately met. We distinguish business needs such as "increase revenue" and user needs such as "explore the online world safely". The description of a need does not include how it can or should be satisfied and does not impose constraints on a solution. The following questions can help to find needs:

- What tasks are actors trying to achieve?
- Why are actors using products, services, or features?
- Why are actors offering products, services or features?

Behaviour Changes

Behaviour changes are modifications of the behaviours of actors. Behaviour changes refer to either start doing something that is new or familiar, or to doing something differently (e.g., to increase or decrease a behaviour), or to stop doing something. Actors need to change their behaviour in order to satisfy their needs. For instance, to get healthy a user might need to run more often; to increase revenue a company might need to start selling something new. The following questions can help to find behaviour changes:

- How can actors change their behaviours to satisfy their needs?
- How can actors change their behaviours to help other actors satisfy their needs?

Capabilities

Capabilities are entities that are offered to support behaviour changes of other actors in order to help them making progress towards the satisfaction of their needs. Examples of capabilities are features, epics, services, products, data, or knowledge. For instance, a feature that reminds runners to schedule a new run (capability) might help them to run more often (behaviour change) and in consequence to get healthier (need). If a capability does not support a behaviour change that drives the satisfaction of a need it will probably not create value. The following questions can help to find capabilities:

- How can we trigger a desired behaviour change?
- How can we support a desired behaviour change?

Assumptions

Assumptions are statements that are taken for granted. For instance, an assumption could be: if a delivery team provides a certain feature then users will start using it. Assumptions might be wrong. Therefore, assumptions that are critical for a business model should be seen as working hypotheses to be tested. The following questions can help to find assumptions:

- Who is delivering value and who is receiving the value?
- How does a capability impact a behaviour change of an actor?
- How does a behaviour change impact the satisfaction of a need?
- Who is capturing value and from whom?

The Wheels of Value model characterises each actor with his needs, capabilities, and behaviour changes (see Figure 3). For each actor, at least one need must be identified.

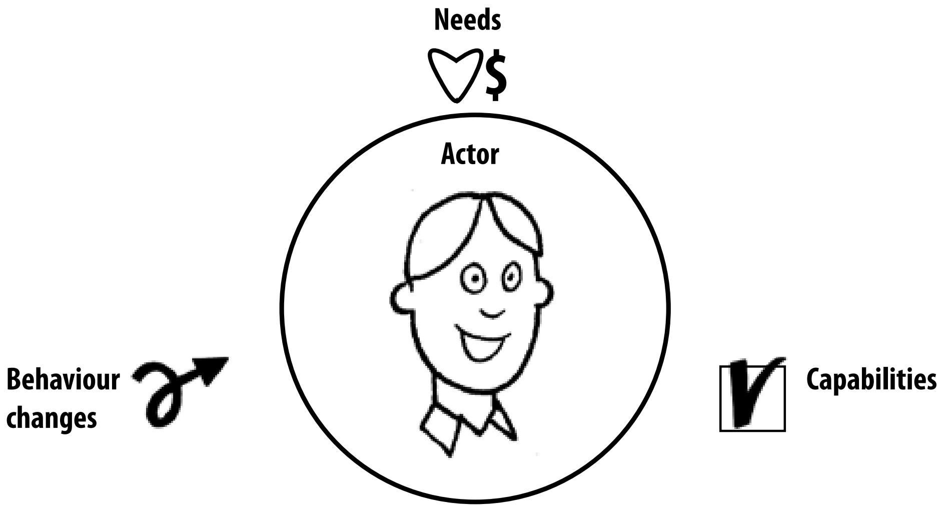


Figure 3. Actors are described in terms of capabilities, behaviour changes, and needs.

The Wheels of Value model communicates assumptions about the connections between capabilities, behaviour changes, and needs (see Figure 4). These assumptions are visualised as arrows. Consider the following example: an overweight person might have the need to lose weight. One option to satisfy this need is changing his or her eating behaviour, e.g., by reducing the daily calorie intake. In order to do this, the person might need a capability to calculate calories. An app developer could, for instance, develop a new app with a calorie counter. The business need of the app developer could be to increase revenue from the users. In this scenario, the overweight person uses the calorie counter of the app (capability) to change his or her eating behaviour (behaviour change) in order to lose weight (user need). In return the user pays for the app (behaviour change) and thereby increases the revenue of the app developer (business need). This example is based on the following value-related assumptions: The new feature is used by the overweight person and leads to a reduction of calorie intake (value delivery), the reduction of the calorie intake leads to a decrease in weight (value creation), and the user pays for the app (value capture). If one of these assumptions is not true, it is highly likely that the business mechanics will not work.

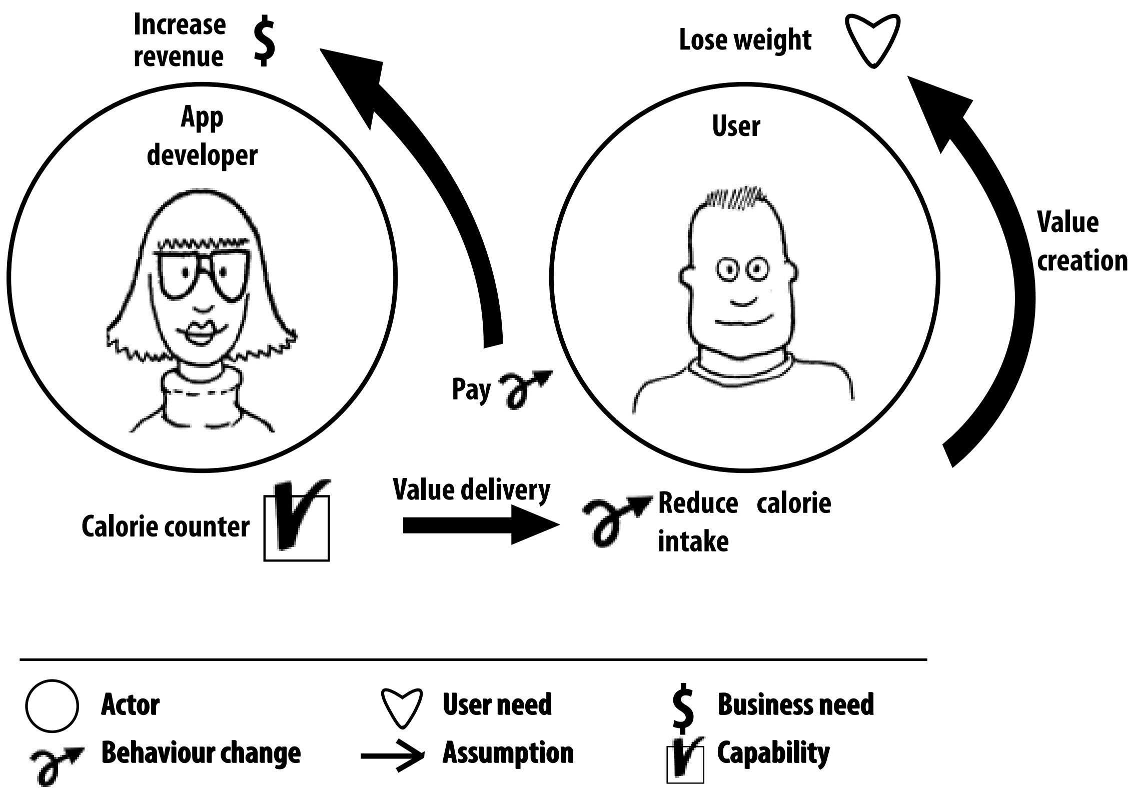


Figure 4. Describing the relationships between actors' capabilities, behaviour changes, and needs.

Complex business environments often involve many actors. Without a clear understanding of the needs of the different actors and the necessary behaviour changes to satisfy these needs it is very difficult to determine which capabilities to develop. Developing capabilities without understanding the value mechanics implies a high risk that these capabilities will not create value and that the effort for developing the capabilities will be wasted.

Example: Digital Marketing

Let's look at an example in the skin care business to illustrate the Wheels of Value Model. A company selling skin care products such as sun protection lotion for kids wanted to extend its business. Selling sun lotion for kids involves different actors such as parents, children, and the product manager responsible for sun lotion.

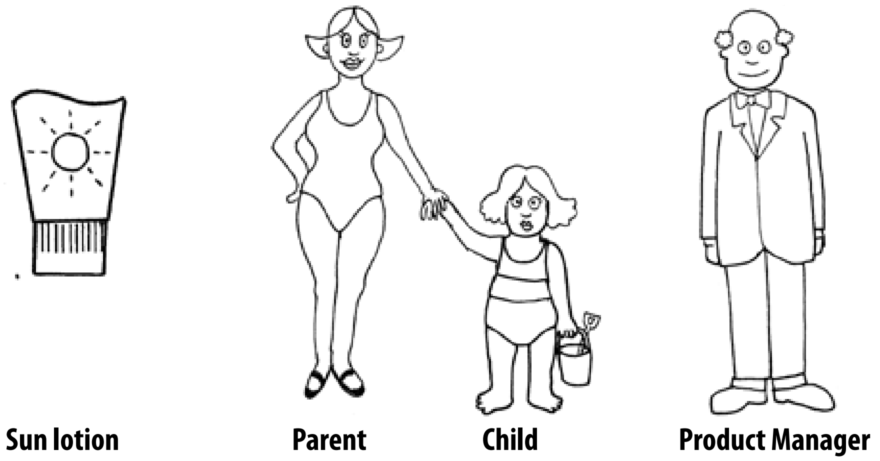


Figure 5. Product and actors.

The vision of the company was to be the number one skin care company in the world. The company was not number one in all markets. It was, for instance, not market leader in Rio and therefore the product manager for sun lotion got the task to improve the business with sun lotion for kids in the Rio market.

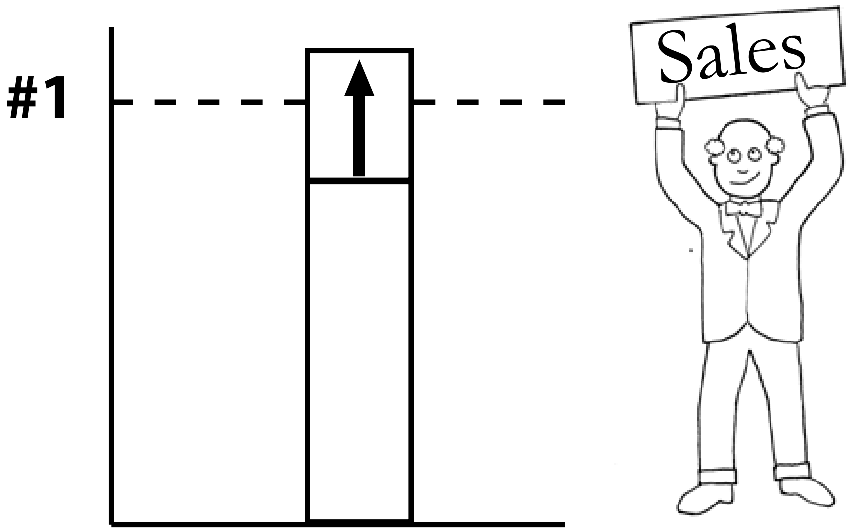


Figure 6. Example improvement target based on company vision.

The product manager decided to first identify the user needs in order to come up with new options to satisfy these needs. Therefore, the product manager asked parents in Rio why they want sun lotion. This revealed that they want sun protection to avoid sunburn of their kids. Further asking the "why" question revealed that the parents want that their children can play in the sun and do not need to stay at home. Another "why" question uncovered that parents want to enjoy the beach (e.g., sunbathing) while the kids are protected. Relaxing at a beach without worrying about their children can be compared to a happiness state and represents a need of many parents in Rio.

After identifying the need, the product manager thought of other options besides sun lotion to satisfy this need. He thought about the questions: "Are there other solutions to satisfy the need? Are there other options besides sun lotion to protect children at the beach? Is there another layer of protection that could be offered to potential customers?" (see Figure 7).

The product manager gained the insight that losing a child at a crowded beach in Rio is a serious problem. Protecting children from getting lost at the beach by keeping them close to the family would contribute to the satisfaction of the need of parents to relax at the beach without worrying about their kids. Developing a solution for this problem might help the company to reinforce the product's main attribute, i.e., protection. Through the principle of association this could in turn help to increase sun lotion sales.

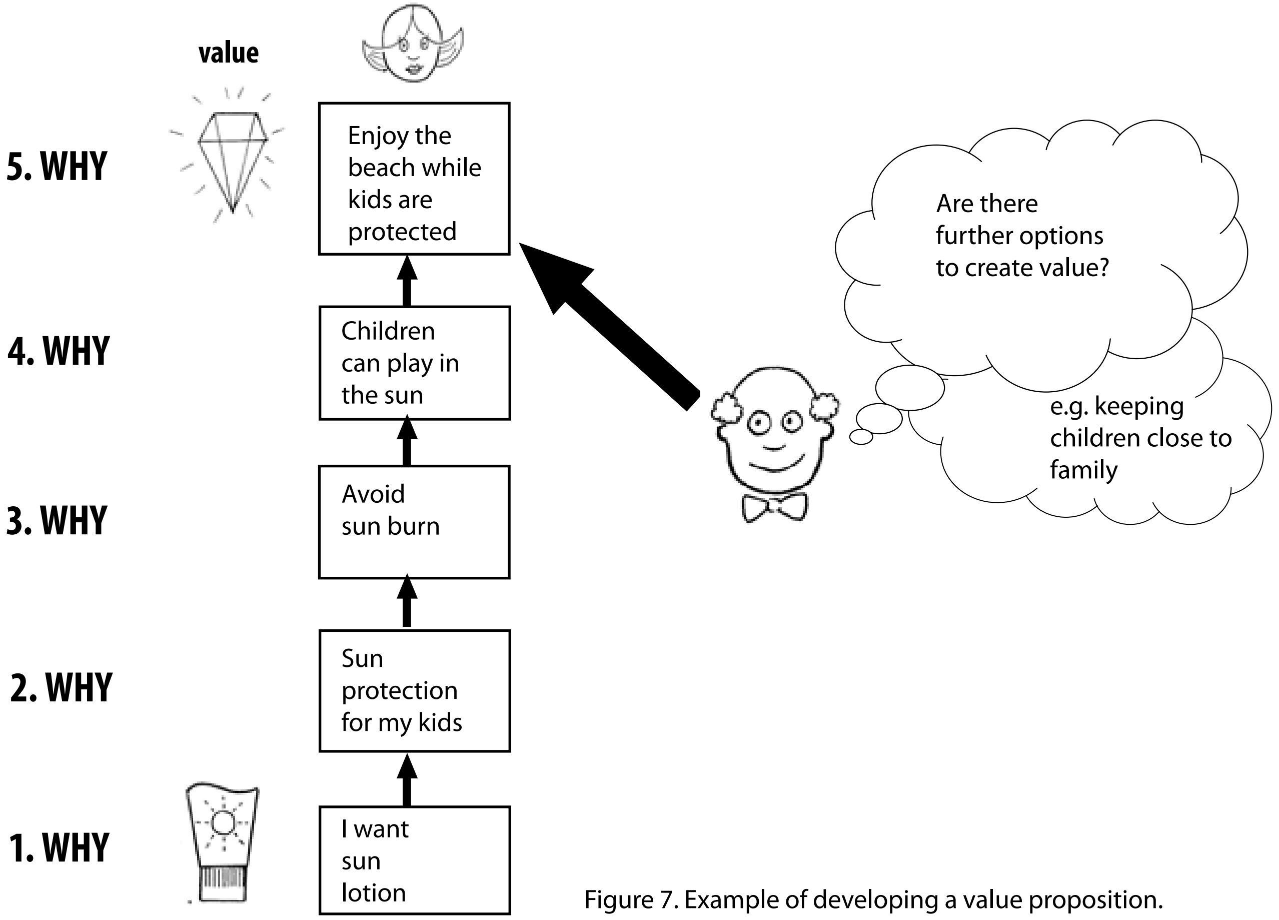

value
5. WHY
Enjoy the beach while kids are protected
4. WHY
Children can play in the sun
3. WHY
Avoid sun burn
2. WHY
Sun protection for my kids
1. WHY
I want sun lotion
Are there further options to create value?
e.g. keeping children close to family

Figure 7. Example of developing a value proposition.

Together with an advertisement agency and a development team, the product manager developed a solution for keeping children close to the family at the beach. He used Internet of Things technology to implement the solution (see Figure 8). The solution consisted of a bracelet to be placed around a child's wrist and an app for the parent's smart phone that links with the bracelet to track the movements of the child. If a child wanders beyond a pre-set maximum distance, an alarm will start and the app helps to find the child by indicating if the parent is approaching or moving away from the child.

In order to distribute the bracelets to parents, the company created an ad for magazines that included the bracelet. The bracelet could easily be detached from the ad and given to a child. The most important weekly magazine was chosen for the ad.

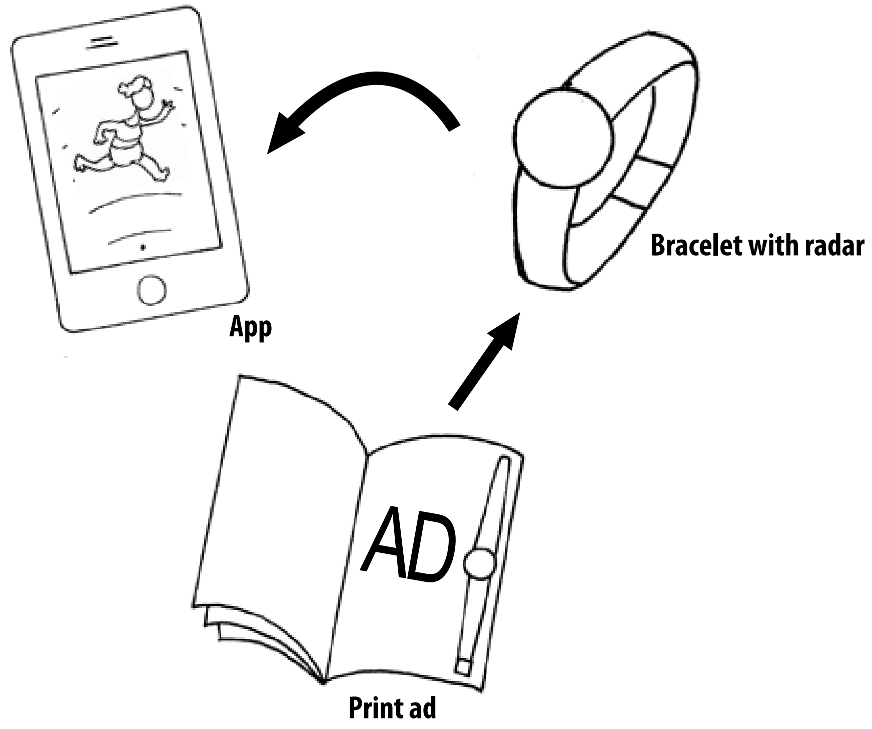


Figure 8. Example solution using Internet of Things technology together with traditional paper media and smart phone app.

The Wheels of Value model can be used to uncover the value mechanics and the most important assumptions (see Figure 9). First we identify the needs of the actors: parents want to keep children safe while relaxing at the beach, kids want to play freely, and the company wants to increase the sales volume with its sun lotion. Different behaviours are necessary to satisfy the user needs: children need to use the bracelet while playing at the beach (and not refusing to wear it or throwing it away). Parents need to download the app and to wrap the bracelet around their child's wrist. In order to support these behaviour changes the company provides the app and the bracelet (as part of the ad) to the parents. In addition, the principle of association needs to be strong enough so that the protection power of the bracelet solution is associated with the protection attribute of the sun lotion and that this association increases sales of the sun lotion. This in turn would contribute to the satisfaction of the business need of the company.

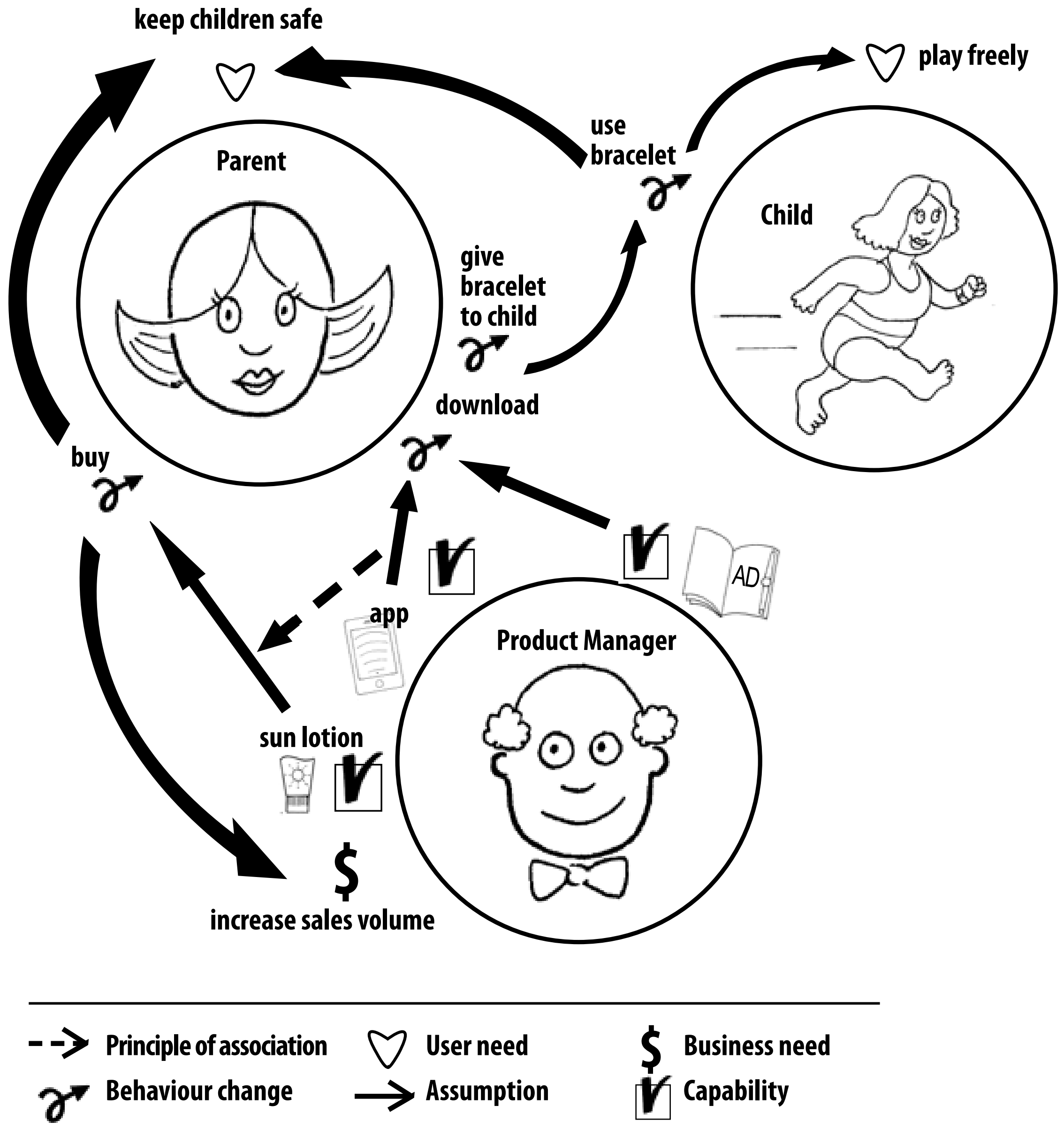


Figure 9. Example Wheels of Value model.

Understanding the relations between the actors helps to understand the value mechanics of the business model and to identify important assumptions. Important assumptions are visualised as arrows in the Wheels of Value model (see Figure 10). For instance, an assumption is that the parents who see the ad download the app and give the bracelet to their children when they are at the beach. Another assumption is that the children actually use the bracelet at the beach and do not take it off and go elsewhere. One more assumption is that using the bracelet in combination with the app really keeps children safe from getting lost. A bracelet not being humidity resistant or an unstable connection between the bracelet and the smartphone could invalidate this latter assumption. There are several more assumptions that can easily be identified from the Wheels of Value model.

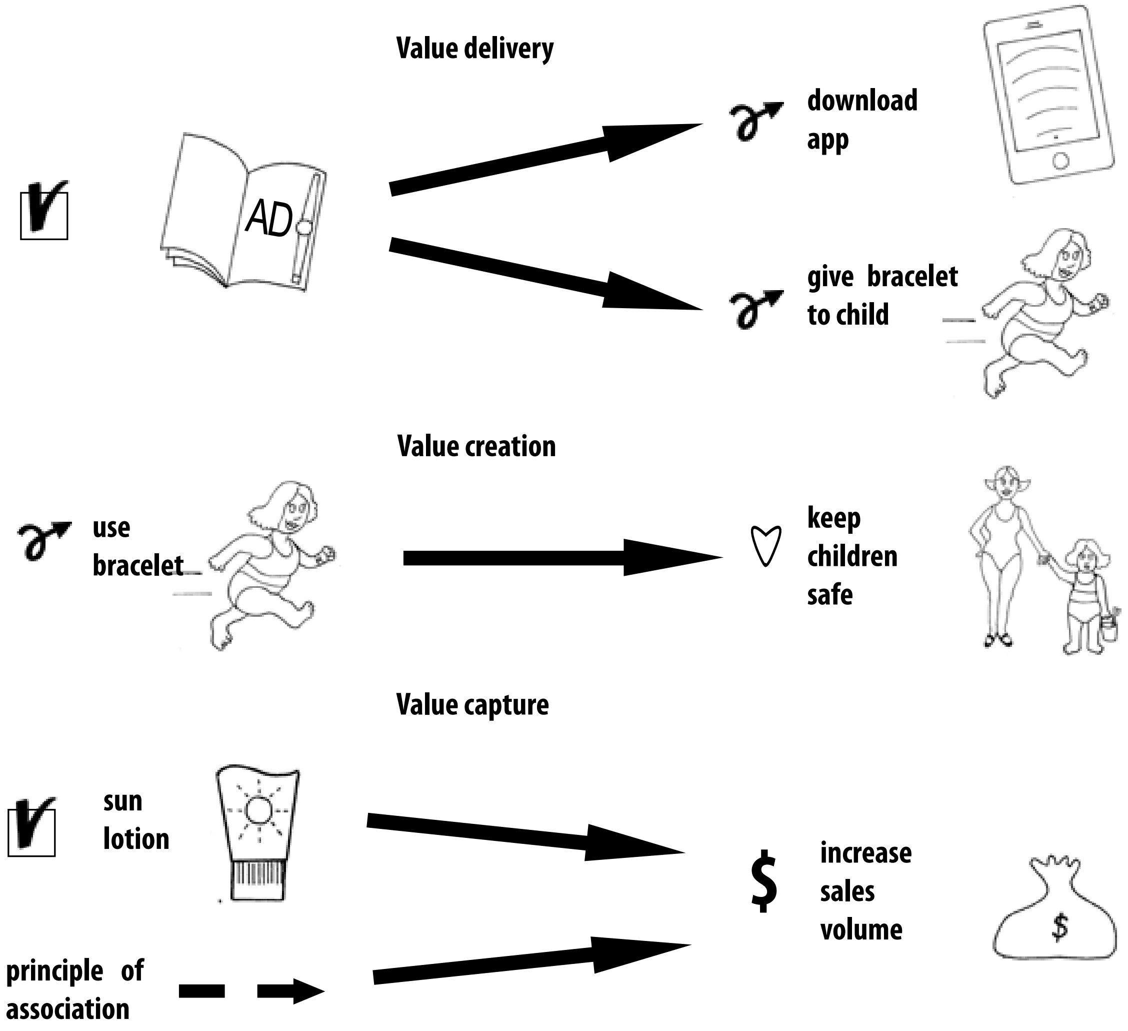


Figure 10. Example assumptions identified from Wheels of Value model.

If there is a high uncertainty whether an important assumption is true or not, it is recommended to test this assumption. Testing assumptions can be done by experiments and should be done early in the development process, ideally before the implementation of the solution.

Testing important assumptions can help to reduce risks and to increase the odds of success. If a test shows, for instance, that nearly nobody is downloading the app after seeing the ad, an analysis about the reasons should be made. A consequence could be that the ad gets improved. Such a test would not require an upfront broad distribution of the ad or a full implementation of the components of the system.

Continuing in the example, the product manager decided to develop the system in a lean way without wasting development effort and by focusing on value creation for customers (see Figure 11). Based on the vision and the identification of the main actors and their needs the company defined a big goal (i.e., being market leader in the Rio market). This big goal was subsequently refined into smaller goals. For each smaller goal, build-measure-learn cycles were conducted. For instance, in a first test of the ad the product manager aimed at 80% conversion rate for downloads of the app but only reached 5%. This led to an improvement of the ad.

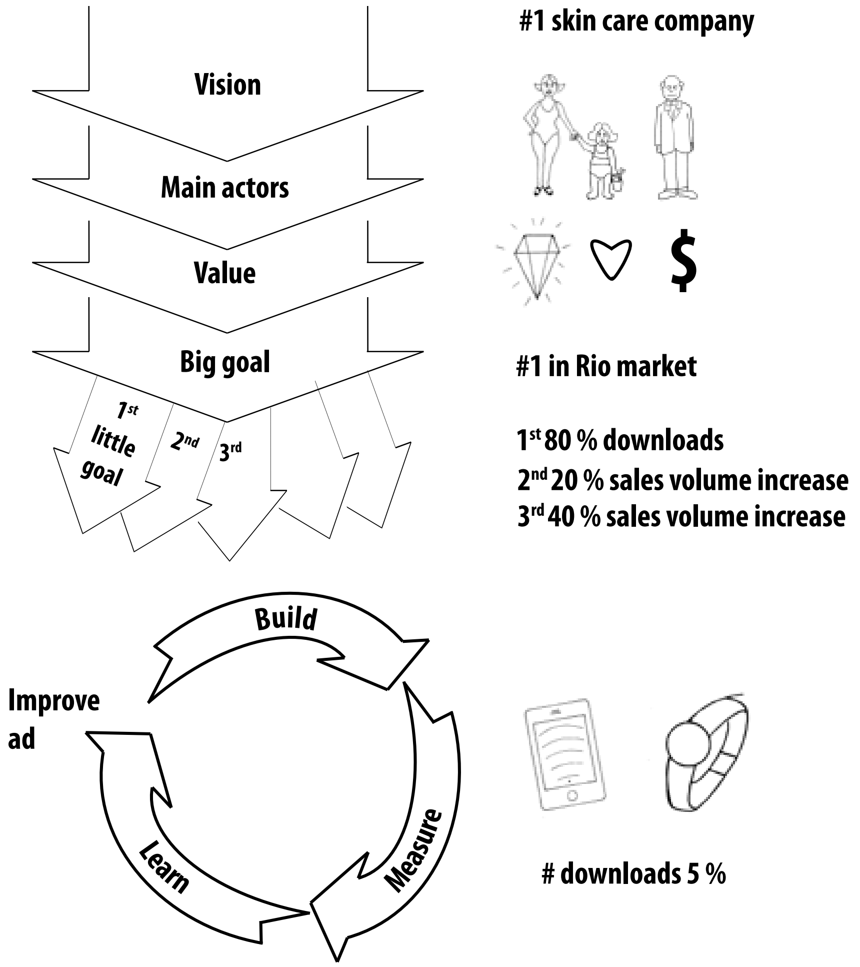


Figure 11. Large goals are broken down and assumptions tested using continuous experimentation cycles (build-measure-learn).

Let us finally check the prerequisites for establishing a business model that works. One way of doing this is to check if two equations are met; the value equation and the sustainability equation (as defined by Maurya in his book "Scaling Lean", see Figure 12).

The value equation requires that the perceived value created for a customer is higher than the captured value. In the example one would need to check if the new layer of protection (i.e., keeping children close to the family at the beach) is perceived as a higher value than the cost for buying sun lotion. Although the bracelet and the app are free, on average customers need to buy more sun lotion.

The sustainability equation requires that the value captured back is at least as high as the cost to deliver the value to the users. In the example the accumulated cost for the app, the bracelet, and the distribution should not exceed the profit from the increased sales. If these two equations are not met in the long run, a sustainable business model cannot be established.

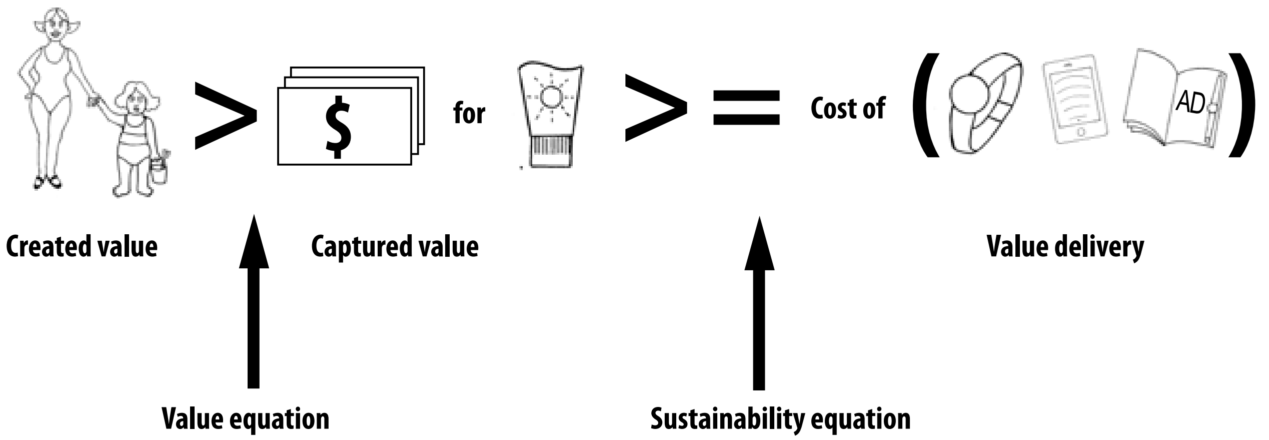


Figure 12. The value equation and sustainability equation are prerequisites for a working business model.

The results of the company's efforts were impressive: 80% of those who saw the ad downloaded the app. The company became the market leader for sun protection for the first time. The sales volume increased by 62% and surpassed the main competitor by 13%. As a side effect, the company significantly improved its innovative perception in the target market. Due to the success, the company considers to give the bracelet away with each sun lotion packet or to sell the bracelets as a separate product.

Making happy customers is not about giving the users products or features. It is about results (see Figure 13). Therefore, the Wheels of Value Model focuses on satisfying and balancing needs, i.e., it focuses on how actors can make progress towards their outcomes. In the example, the parents can relax at the beach without worrying about their children. The children can play freely and have fun. In return, the product manager grows his business.

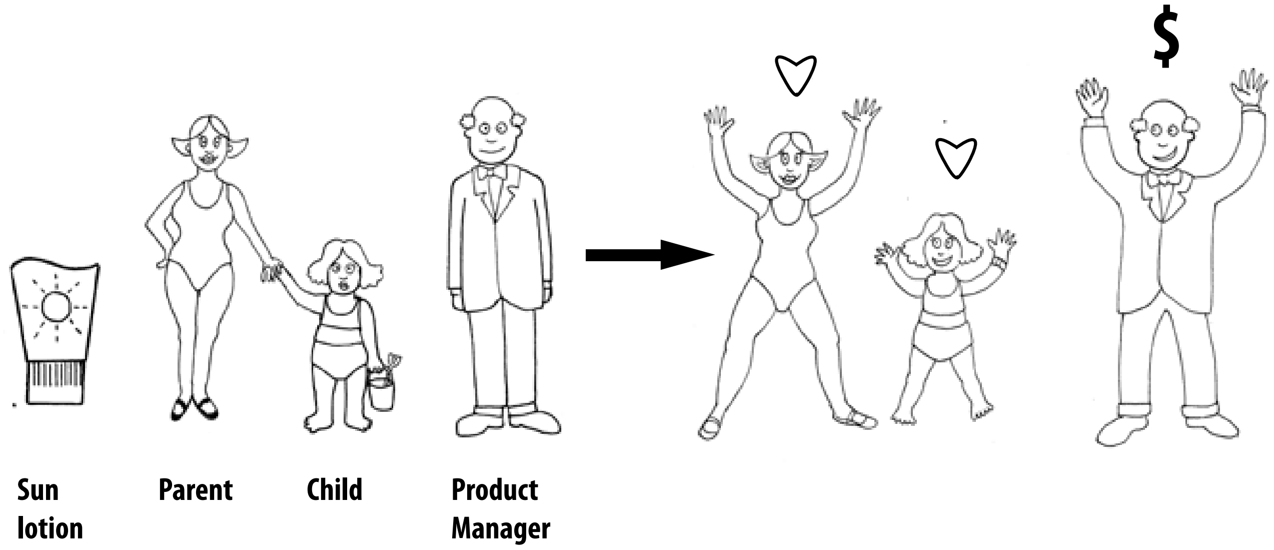


Figure 13. The Wheels of Value model focuses on balancing needs.

This example is inspired by Nivea's digital marketing campaign with the Nivea Protégé app and a sun protector strip. A video of Nivea's solution can be found here: http://bit.ly/TheProtectionAd

Three Ways to Apply the Wheels of Value Models

There are many different applications for using the Wheels of Value Model.

Making strategic decisions and guiding tactical work

Very often, complex business environments are crowded with many entities such as devices, technical systems, services, connections, layers, information flows, control flows, and organisational units. Therefore, it is very difficult to identify the main actors and their needs. It is especially difficult to understand how to make a business and to see how the different actors are involved in the value chains. The Wheels of Value Model helps you to identify the key actors with their needs in complex business environments. It can be used to create a shared understanding of your own role and the role of other important actors. Strategic decisions can be based on a good understanding of the needs of the different actors. A clear understanding of the different needs helps to identify options for satisfying these needs and to decide on what capabilities to develop. It can also help to evaluate existing or suggested capabilities.

Validating business models

Important assumptions that underlie a business model might not be viable or even wrong. If, for instance, an important assumption about value capture should turn out to be wrong, the cost for value delivery might not be covered and in the long run the business model might not work. Therefore, important assumptions should be identified and tested in order to mitigate the risk that your business model is not working. The Wheels of Value Model is a great way to analyse complex business situations and to identify business-critical value assumptions. It visualises assumptions on how value is delivered, created, and captured. These assumptions should be validated.

Finding new business

Applying the Wheels of Value model is also a good way to uncover new business opportunities. Based on the identification of the needs of different actors, you can look for new solutions to satisfy these needs. The Wheels of Value Model also helps to design new ecosystems and to transform your business to new areas.

Canvas Tools for Linking Experiments to Product and Business Development

To succeed, a software product needs acceptance from its stakeholders such as business owners and paying customers. It has to be made available for delivery, be different from its competitors, and it needs to support a scalable business model and a software architecture that meet the needs of the stakeholders also in the future. The success of the product concept is not based on a single stakeholder perspective, but on the value it provides for all key stakeholders (see Figure 14).

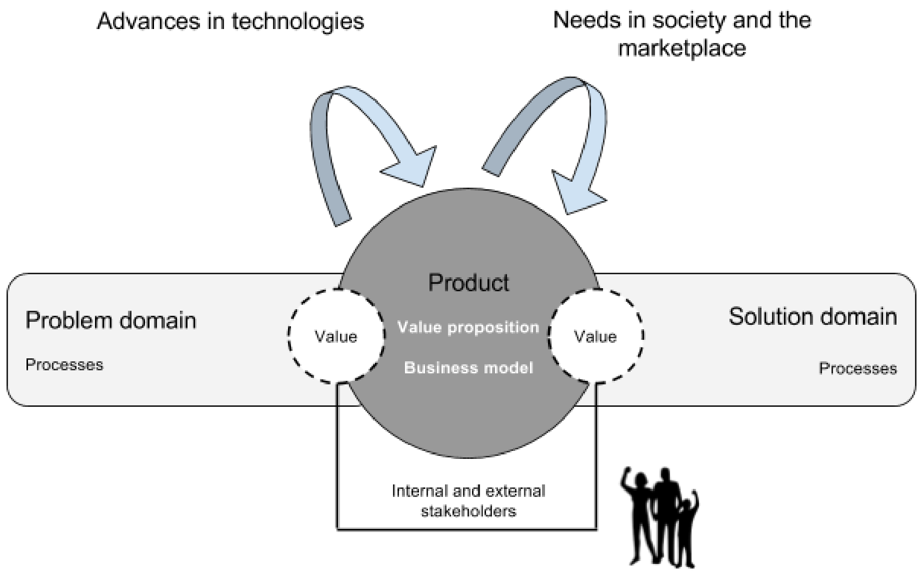


Figure 14. Product success is based on the value provided for all stakeholders.

The experimentation process can be seen as a structured voice-of-customer validation cycle that centres around users' reactions towards design elements – that reflect the product designers' expectations – crafted as hypotheses. The motivation for experimentation can be used to validate ideas or to discover new, actionable knowledge that can be used in designing the future of the product development effort. Experimentation can take place at any stage of the innovation process and with versatile types of experiment objects. The type of these objects is dependent on what knowledge is available about stakeholders, their problems and motivations, along with the value the product concept provides for its users. Figure 15 displays a general innovation process with examples of goals for experimentation at each stage.

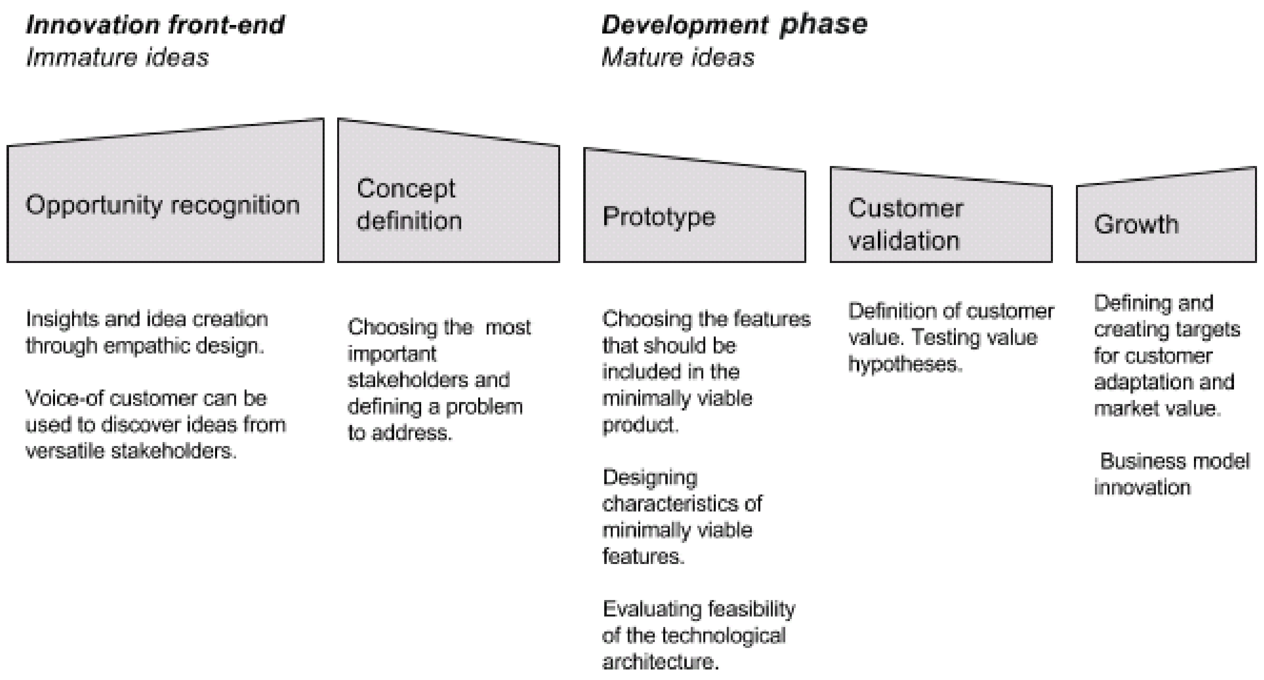

Innovation front-end
Immature ideas
Development phase
Mature ideas
Opportunity recognition
Concept definition
Prototype
Customer validation
Growth
Insights and idea creation through empathic design.
Voice-of customer can be used to discover ideas from versatile stakeholders.
Choosing the most important stakeholders and defining a problem to address.
Choosing the features that should be included in the minimally viable product.
Designing characteristics of minimally viable features.
Evaluating feasibility of the technological architecture.
Definition of customer value. Testing value hypotheses.
Defining and creating targets for customer adaptation and market value.
Business model innovation

Figure 15. A general innovation process with experiment goals for each stage.

While working in the realm of practical experimentation, it is easy to lose track of what the overarching goals of the product development effort were. Table 4 shows some characteristics of immature and mature product concepts. The product vision describes the "raison d'être" of the product development effort. Sufficient preparations at the project's front-end stage can help to define a precise enough vision and help to start the project faster and with less friction than when ambiguity of the project's goals prevail. Therefore, tracking the vision should be kept as an essential background process. This can be achieved by using the versatile set of canvas-based product development tools that can combine different perspectives for describing the software product. As each tool considers different knowledge in the envisioning process, the choice of canvas tool greatly affects the outcome.

Table 4. Characteristics of immature and mature product concepts. (Adapted from J. Kim and D. Wilemon, "Focusing the fuzzy front–end in new product development," R&D Management, vol. 32, no. 4, pp. 269–279, 2002.)

	Immature	Mature
Quality of ideas	Fuzzy and probable	Clear, fixed and specific
Quality of knowledge for decision-making	Informal and approximate	Formal and precise
Focus of the concept	Broad focus and thinly described	Narrow focus and detailed
Rejecting an idea	Easy	More difficult
Damage if project is abandoned	None or small	Substantial
Management methods	Unstructured, experimental and creative	Structured and systematic
Budget	Small or none	Designated and larger
Personnel involvement	Individual people or small teams	Full development team

Certainty about the product concept and its business model creates the boundaries within which experimentation can take place – and helps determine which factors require the most attention. Here, it is important to recognise maturity of the product concept: is it a minimally viable prototype? A commercially feasible pilot? A production quality software product that can be imagined to be scaled both as a technologically, and as a business asset? In each case, the choice of canvas tools should be made based on what knowledge is available and what knowledge the product development effort needs most urgently.

Since there is an immense selection of canvas tools available, the problem of choosing the right tool to fit the purpose becomes emphasised. However, there are many elements in each tool which relate to common concepts. Table 5 lists questions that can be used in identifying which factors of the product concept must be fortified by eliciting new knowledge

Table 5. Questions for evaluating maturity of the product concept.

Theme	Question
Stakeholders	Have we discovered who the stakeholders are? Who are the most important stakeholders? What are their goals? Who are the most potential for becoming paying customers?
Problem	What problem do the stakeholders experience? Which context/ processes does the problem relate to?
Solution	What is the name of the product? What kind of solution does the product offer? Which processes does it relate to or create? What value does it provide? What is the value proposition? Is the product designed at a coarse or a detailed level?
Market	How is the product positioned into the marketplace? Who are the competitors? What is the competitive advantage? How is the product delivered?
Project (development)	Which competencies are required? Are priorities of requirements well defined? What are the first steps for starting the project? Are quality requirements clear?
Business model	What is the mission statement of the company? Core identity and brand values? What is the product's cost structure and revenue stream? Which metrics define commercial success? How are customer relationships managed?

When choosing between canvas tools, consider whether they allow you, in your particular case, to evolve your product concept towards maturity (see Table 2) and whether they assist you to evaluate the questions in Table 3. Some examples of canvas tools are shown below.

Strategyzer Business Model Canvas and Value Proposition Canvas
https://strategyzer.com/canvas

IDEO – Human centered design toolkit: a collection of useful tools
http://www.designkit.org/

Futurice Lean Service Design Toolkit
https://www.leanservicecreation.com/

Lean Stack – The Lean canvas
https://leanstack.com/leancanvas

Roman Pichler – Go product roadmap
http://www.romanpichler.com/blog/goal-oriented-agile-product-roadmap/

VALUE Tool

An important part in the continuous experimentation cycle is making decisions based on experiment results. Decisions in software industries have largely been made in a value neutral setting, in which cost is the primary driver for every decision taken. However, better decisions can be made using a value-based approach, achieving cost-effective results and reliable construction and maintenance of products.

About the VALUE Tool

The VALUE Tool was created as part of the VALUE research project, a Tekes-funded Finland Distinguished Professor Programme (FiDiPro) at the University of Oulu. The tool was developed in cooperation with industry partners by FiDiPro professor Emilia Mendes and the M3S research group headed by prof. Markku Oivo.

Further information is available at:
VALUE project
http://valueproject.fi/

Value tool demo
https://www.youtube.com/watch?v=pi3yO5k-MZc

The Value Tool, which was co-created in collaboration with three software companies, is such a tool. It supports both individual and group-based decisions using several visualisation mechanisms. The tool's web-based interface (Figure 16) can be used by the key stakeholders participating in decision-making meetings. The tool provides means to represent value considerations by different stakeholders. Those value considerations are company-specific and they are elicited before taking the VALUE Tool into use. The result is a common vocabulary the key stakeholders explicitly apply in the decision-making.

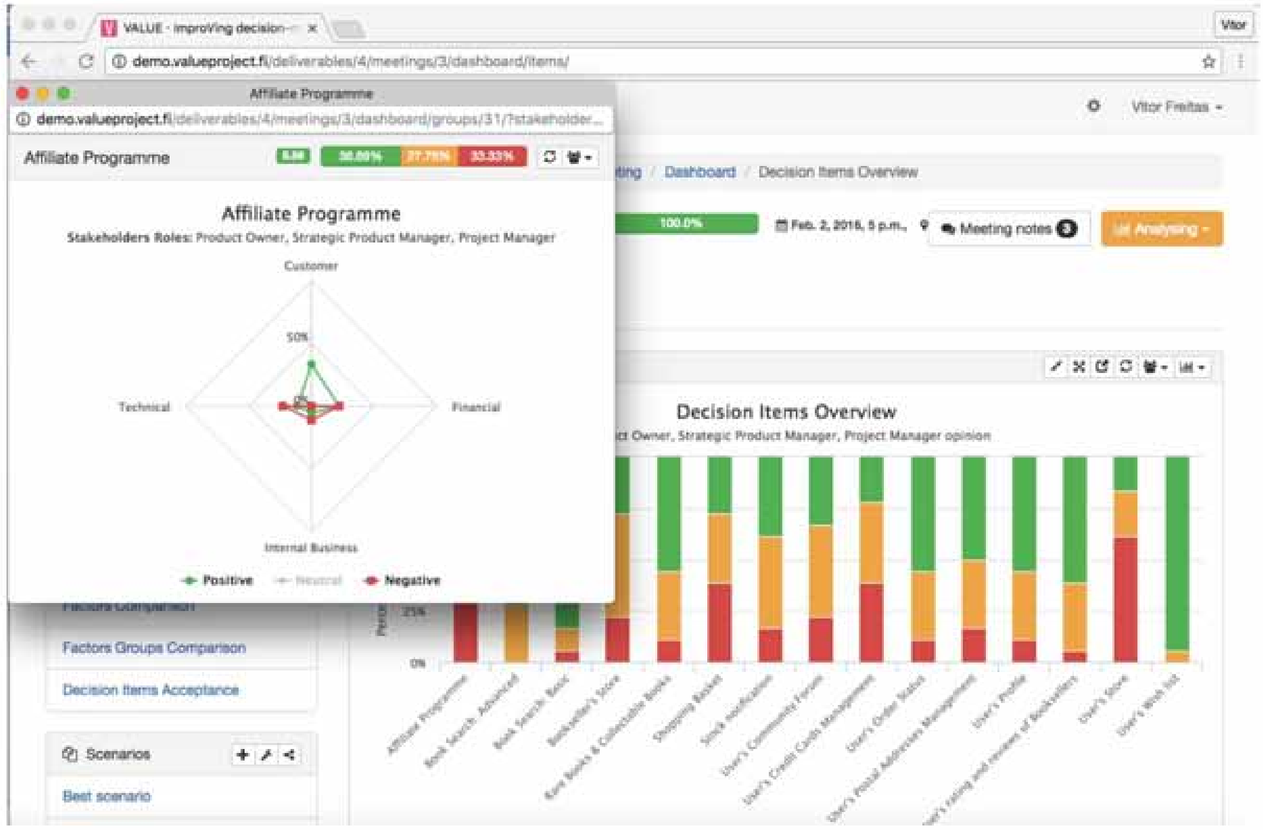
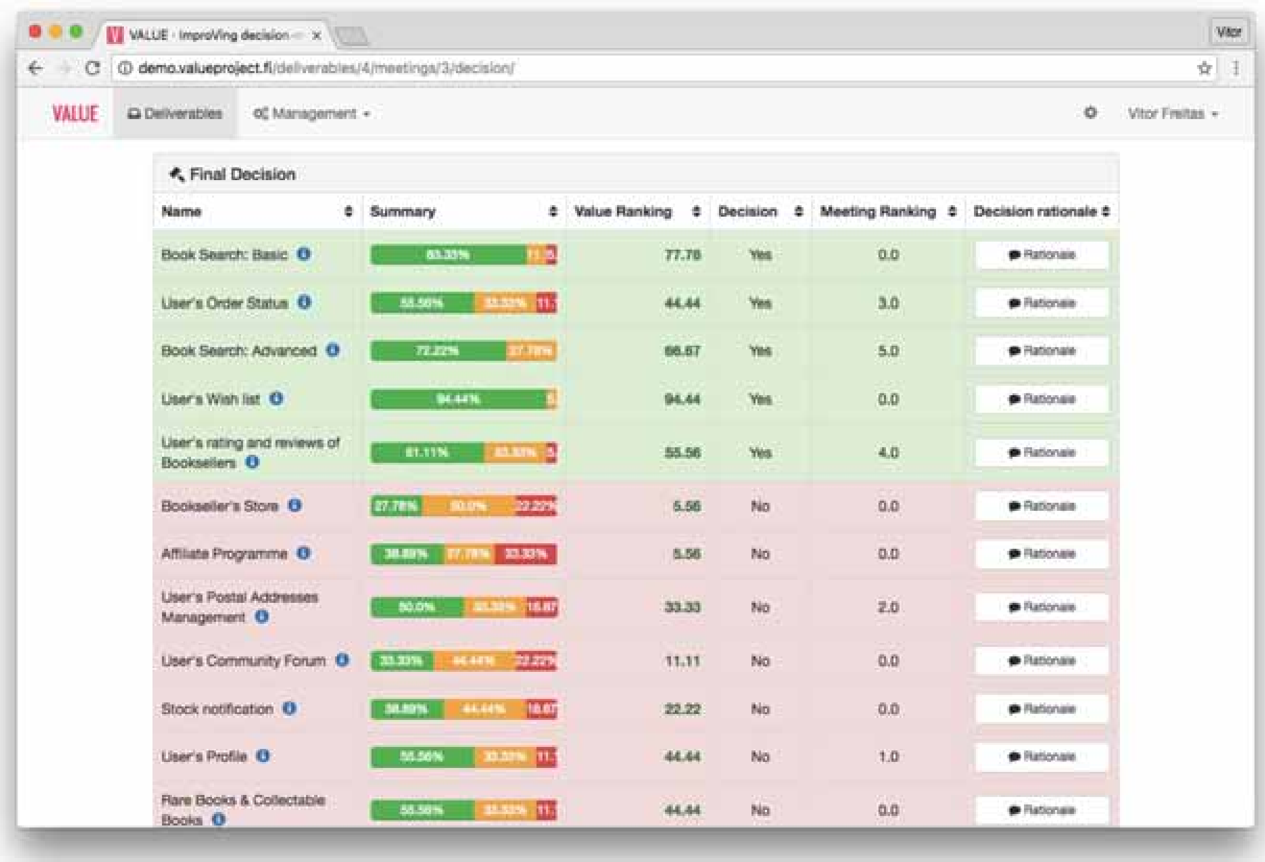

VALUE - improVing decision-
Vitor
demo.valueproject.fi/deliverables/4/meetings/3/dashboard/items/
Affiliate Programme
demo.valueproject.fi/deliverables/4/meetings/3/dashboard/groups/31/?stakeholder...
Affiliate Programme
5.56
38.89%
27.78%
33.33%
Affiliate Programme
Stakeholders Roles: Product Owner, Strategic Product Manager, Project Manager
Customer
50%
Technical
Financial
Internal Business
Positive
Neutral
Negative
Vitor Freitas
ting / Dashboard / Decision Items Overview
100.0%
Feb. 2, 2016, 5 p.m.
Meeting notes 3
Analysing
Decision Items Overview
ct Owner, Strategic Product Manager, Project Manager opinion
25%
0%
Factors Comparison
Factors Groups Comparison
Decision Items Acceptance
Scenarios
Best scenario
Affiliate Programme
Book Search: Advanced
Book Search: Basic
Bookseller's Store
Rare Books & Collectable Books
Shopping Basket
Stock notification
User's Community Forum
User's Credit Cards Management
User's Order Status
User's Postal Addresses Management
User's Profile
User's rating and reviews of Booksellers
User's Store
User's Wish list

Figure 16. Screenshot of Value Tool.



demo.valueproject.fi/deliverables/4/meetings/3/decision/

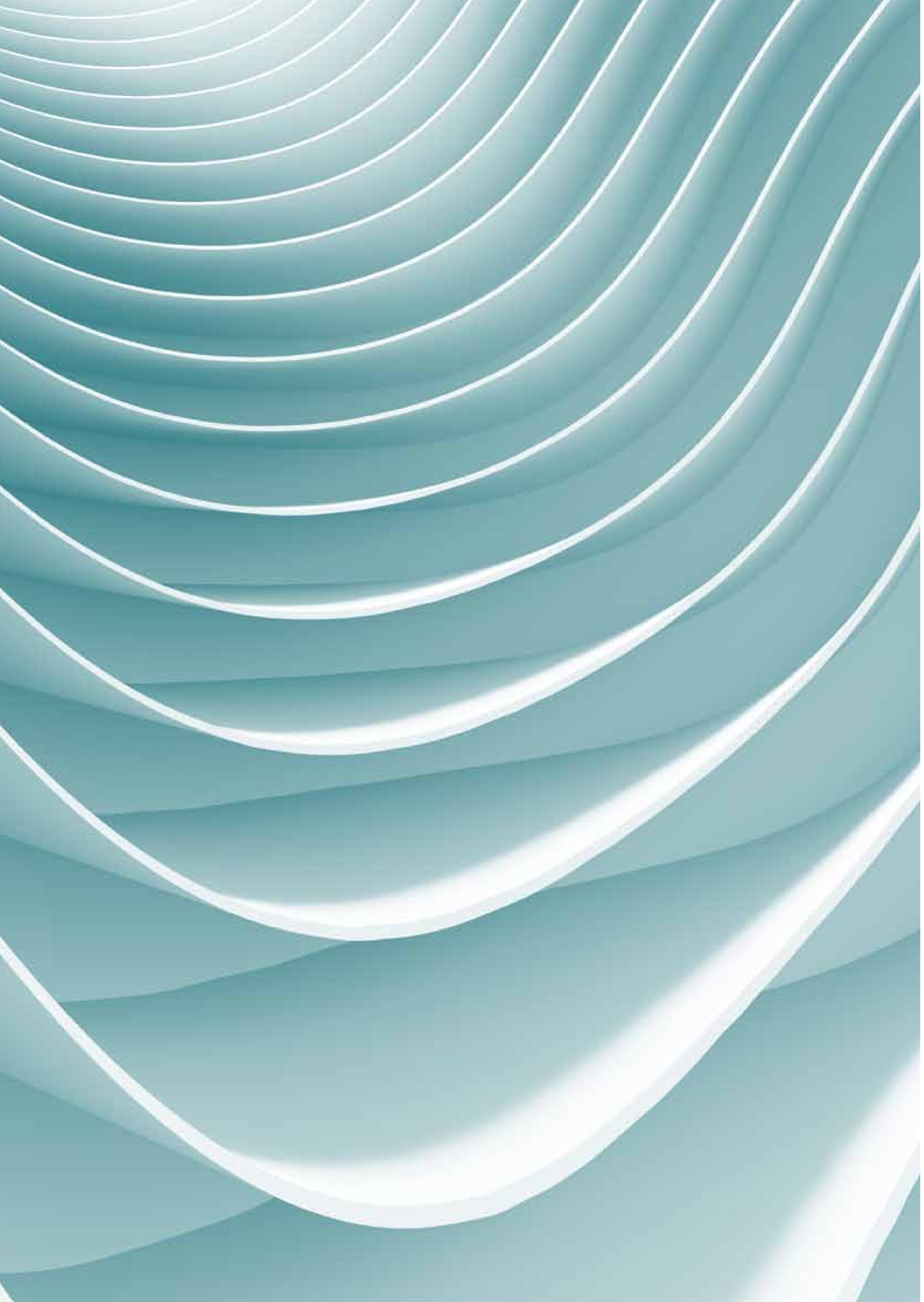
Final Decision

Name	Summary	Value Ranking	Decision	Meeting Ranking	Decision rationale
Book Search: Basic	83.33% 11.5	77.78	Yes	0.0	Rationale
User's Order Status	55.56% 33.33% 11.1	44.44	Yes	3.0	Rationale
Book Search: Advanced	72.22% 27.78%	66.67	Yes	5.0	Rationale
User's Wish list	94.44% 5	94.44	Yes	0.0	Rationale
User's rating and reviews of Booksellers	61.11% 33.33% 5	55.56	Yes	4.0	Rationale
Bookseller's Store	27.78% 50.0% 22.22%	5.56	No	0.0	Rationale
Affiliate Programme	38.89% 27.78% 33.33%	5.56	No	0.0	Rationale
User's Postal Addresses Management	50.0% 33.33% 16.67	33.33	No	2.0	Rationale
User's Community Forum	33.33% 44.44% 22.22%	11.11	No	0.0	Rationale
Stock notification	38.89% 44.44% 16.67	22.22	No	0.0	Rationale
User's Profile	55.56% 33.33% 11.1	44.44	No	1.0	Rationale
Rare Books & Collectable Books	55.56% 33.33% 11.1	44.44	No	0.0	Rationale

Figure 17. Final decision view.

After all the stakeholders input their opinion for each decision item being discussed, the VALUE Tool aggregates the results and display the group opinion in a rich dashboard. The data visualisation provided by the dashboard is the aggregated view by all the stakeholders. The dashboard is composed by six different reports, with many customisation options, such as chart types, ordering, sub-groups of stakeholders.

The final step of a decision-making meeting supported by the VALUE Tool is documenting the group's decision, which can be the prioritisation or selection of the decision items discussed (Figure 17). All past decisions are stored in the tool, and can be used to support future decisions, and to go over past decisions too.



Continuous Experimentation Infrastructure

Continuously running experiments can be made more efficient through automation and a proper technical architecture. Tasks such as data collection, management, and analysis are frequently occurring and the software supporting such tasks can be reused. The technical architecture of a software product or service can be designed to support experiments. For example, an architecture that allows flexible deployment and activation of experimental features allows experiments to be run in the same development pipeline as normal development.

The University of Helsinki has developed a reference infrastructure for continuous experimentation, which is available online (https://github.com/TheSoftwareFactory). The technology stack for continuous experimentation builds on well-known and existing continuous integration, delivery, and deployment pipelines. Three new blocks are needed in the system architecture: a set of tools for managing experiments and analysis experiment results, an experiment backend, and changes to the software product itself, the front-end. Figure 18 shows an overview of the system architecture. For simplicity, the figure omits the required security mechanisms.

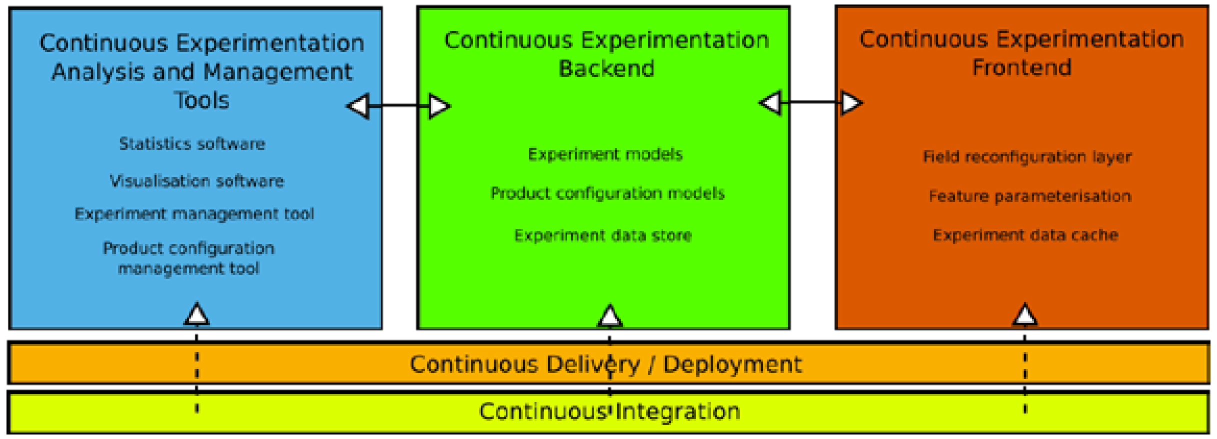
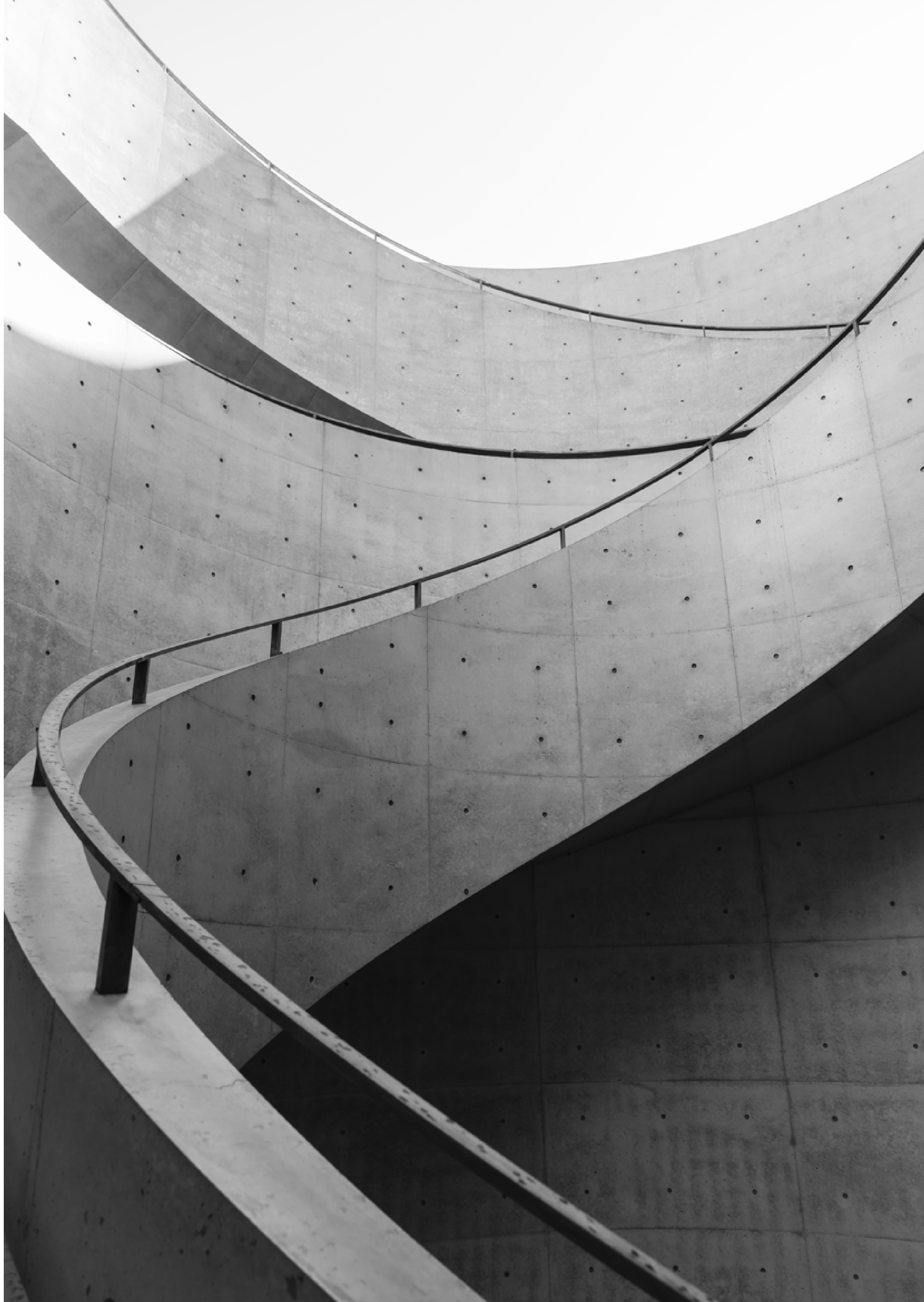

Continuous Experimentation Analysis and Management Tools
Statistics software
Visualisation software
Experiment management tool
Product configuration management tool
Continuous Experimentation Backend
Experiment models
Product configuration models
Experiment data store
Continuous Experimentation Frontend
Field reconfiguration layer
Feature parameterisation
Experiment data cache
Continuous Delivery / Deployment
Continuous Integration

Figure 18. Continuous Experimentation Technical Infrastructure.

The experiment management tool is used to define experiments: the conditions for the experiments such as start and end times, amount of participants, or the desired statistical power. Each experiment is associated with a product configuration, which is managed through a product configuration tool. These communicate with the continuous experimentation backend, which stores information on each experiment and the associated product configurations. The analysis and management tools also include software for analysing and visualising the experiment data. Various kinds of tools can be used depending on what is needed for the analysis.

The continuous experimentation backend stores all information regarding experiments and product configurations as well as the experiment data collected from front-end software. Experiments are defined in experiment models, which store all data required to keep track of running experiments, their present state, and conflicts between experiments. Each experiment is associated with a product configuration which defines how the front-end should configure itself when it is part of a particular experiment. Finally, the backend includes a high-performance data store which is capable of receiving and storing all experiment data arriving from frontends.

The continuous experimentation frontend is the software that is actually used by users. Apart from the main functionality that is visible to the user, the frontend can be enabled for continuous experiments with three crucial features that are not normally visible to the user. The field reconfiguration layer includes the logic required to contact the experimentation backend to advertise the frontend as being available for experiments, and to receive an up-to-date configuration for any experiments that the frontend is to participate in. The layer includes fallback mechanisms in case the backend cannot be contacted, and handles the reconfiguration of the software either at start time or continuously at run-time. To actually implement the reconfiguration, the frontend requires parameterised features, which can range from simple on-off feature toggles to complex parameter trees with runtime constraints. This may require some additional design of the other frontend components, but also allows shipping of inactive software that can be activated for experiments. Finally, the frontend has a cache for experiment data, which stores a small amount of data that can be sent to the experiment data store. The cache allows experiments to be run even in limited connectivity scenarios, and allows more fine-grained timing information to be collected locally.



Glossary of Terms

Category	Vocabulary	Description
Experiment	Experimentation	Refers to the actual process of rapidly and incrementally testing assumptions and uncertainties in your ideas.
	Target of experiment	The experiment target refers to what drives the experimentation e.g., concepts, insights, assumptions, uncertainties, features. This can be ideas or problems that need solving, uncertainties related to feature usage, assumptions, or concepts.
	Assumption	A thing, aspect of your idea that is accepted as true or as certain to happen, without proof.
	Hypothesis	A hypothesis is a proposed, testable explanation for a phenomenon. A hypothesis usually drives the experiment and can be derived from business strategies, innovation initiatives, qualitative and quantitative customer feedback, or results from on-going customer validation cycles.
	Experiment object	The experiment object is a MVF or MVP that represents critical aspects of the product or the feature that will be experimented on. In other words, hypotheses are tested with the experiment objects.
	Experiment plan	The plan describes how to test the set hypothesis. It includes the method (e.g., A/B testing, survey), experiment target, duration, data collection and analysis descriptions.
	Metrics	Metrics capture values pertaining to the product or feature at a specific time during experiment data collection.
	Success criteria	A measure or metric that can allow you to clearly assess whether a hypothesis has been validated or falsified.

Product	Value	Value here is the importance, worth, or usefulness of a feature, product or service. Creating, delivering, and capturing value from users or customers is a central motivation for conducting experiments (e.g., increase customer satisfaction or save R&D costs).
	Minimum viable product (MVP)	Refers to the smallest possible set of features of a product that adds value to a customer.
	Minimum viable feature (MVF)	The smallest composition of a feature that provides the essential value for both the users and owners from the product.
Process	Data collection	Refers to the process of acquiring quantitative (e.g., usage data) and/or qualitative data (e.g., interviews, observations, surveys). The method of collection depends on the hypothesis and metrics needed to validate the hypothesis. Qualitative methods are more frequent at the beginning until product/solution fit has been achieved or at the end of the experimentation to find out the reasons behind quantitative data.
	Analysis	Refers to the process of examining the collected data in order to validate the hypothesis (e.g., data analysis or gap analysis).
Company	Findings, learnings and decisions	The final element is to act on the findings and learnings gained from the analysis. This can be to kill the experiment idea, to rethink the idea or to continue with the next assumption.
	User	The person who uses your product or feature.
	Customer	The person or organisation who is paying for the products or services. A customer can be a user as well.

DIMECC Program
NEED FOR SPEED

Continuous Experimentation Cookbook

An introduction to systematic experimentation for software-intensive businesses

What do your users and customers actually need? How do you know? How can you find out?

The business landscape is changing radically because of software. Companies in all industry sectors are continuously finding new flexibilities in this programmable world. They are able to deliver new functionalities even after the product is already in the customer's hands. But success is far from guaranteed if they cannot validate their assumptions about what their customers actually need. A competitor with better knowledge of customer needs can disrupt the market in an instant.

This book introduces continuous experimentation, an approach to continuously and systematically test assumptions about the company's product or service strategy and verify customers' needs through experiments. By observing how customers actually use the product or early versions of it, companies can make better development decisions and avoid potentially expensive and wasteful activities. The book explains the cycle of continuous experimentation, demonstrates its use through industry cases, provides advice on how to conduct experiments with recipes, tools, and models, and lists some common pitfalls to avoid. Use it to get started with continuous experimentation and make better product and service development decisions that are in-line with your customers' needs.

DIMECC RESULT PUBLICATIONS 3/2017