# Predicting and Analyzing Feature Value when R&D is an Experiment System

Master’s thesis in Software Engineering

Markus Ekström

Ívar Daði Þorvaldsson

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2016

Predicting and Analyzing Feature Value when R&D is an Experiment System
Markus Ekström & Ívar Daði Þorvaldsson



Supervisor: Jan Bosch, Department of Computer Science and Engineering
Supervisor: Helena Holmström Olsson, Faculty of Technology and Society, Malmö University.
Supervisor: Aleksander Fabijan, Faculty of Technology and Society, Malmö University.

Examiner: Eric Knauss, Department of Computer Science and Engineering

Master's Thesis 2016
Department of Computer Science and Engineering
Division of Software Engineering
Chalmers University of Technology
SE-412 96 Gothenburg

# Abstract

Software development has experienced major changes in the last decades, from traditional waterfall development to the agile way of working. In the last years, companies have been moving beyond agile practices, towards both continuous integration (CI) and continuous deployment (CD), before advancing to continuous experimentation. When continuously experimenting, R&D can be viewed as an experiment system, with the aim of rapidly testing and validating software based on customer and user feedback. However, it has proven problematic to gather the relevant data from customers, resulting in an 'open loop' between customers and product management. As a consequence, decisions are often based on 'gut feeling' rather than actual data. Several process models have been created to analyze feature value throughout the software development process. Nevertheless, no concrete procedure for predicting and analyzing feature value has been developed. This study presents the DVOCE model to fill that gap. DVOCE is a detailed and extended version of the previously published high-level HYPEX model and covers the pre-development and development phases. DVOCE provides a detailed procedure in how to model the feature to enable the prediction. In addition, it includes a sub-process for selecting the appropriate customer feedback and data collection techniques so the value can be tracked before analyzing whether a feature lives up to its expectations. In the study, design science research (DSR) is used as a research methodology. To help with validating the process model, a prototype was created based on the core aspects of the model. The process model was then validated in eight validation sessions at four companies, with a total of ten participants. The results suggest that the DVOCE process model can be used to model, predict and analyze feature value, as well as to track the realized feature value using the appropriate customer feedback and data collection techniques. Further work is needed to validate the model with a larger audience.

**Keywords:** continuous experimentation, R&D as an experiment system, feature value, modeling, prediction, tracking, analyzing, deploying, HYPEX.

# Acknowledgements

We would like to thank our supervisors Jan Bosch, Helena Holmström Olsson and Aleksander Fabijan for their support during the thesis work. In addition, we would like to thank all of the interviewees and participants in the validation sessions for their prompt responses, welcoming us to their workplace and providing us with valuable information.

Markus Ekström & Ívar Daði Þorvaldsson, Gothenburg, September 2016.

# Table of Contents

# 1

# Introduction

There have been increasing demands on software development in the last decades to deliver higher-quality products with reduced time-to-market [1]. At the same time it has also been necessary for companies to more quickly adapt to rapidly changing customer requirements and evolving technologies. For these reasons agile practices have become increasingly attractive, partly due to their emphasis on short feedback loops between developers, customers and management [2]. Companies have also been working towards shorter and shorter feedback loops during this time, which today typically range from a few days to a few weeks.

However, the agile way of working within research and development (R&D) can also be viewed as an intermediate step towards continuously experimenting with the aim of maximizing customer value [3]. The 'Stairway to Heaven' model shows a typical evolution of a company from traditional development to agile practices and beyond. To advance beyond agile practices, companies need to adopt both *continuous integration* (CI) and *continuous deployment* (CD). In CI, software is integrated multiple times a day using, e.g., automated tests and builds [4]. In CD, software is continuously deployed to customers, opening up the possibility of gathering customer feedback more rapidly. The final step is *R&D as an experiment system* which focuses on continuous experimentation to ensure customer value.

Continuous experimentation has been used in areas such as software as a service and in cloud computing [5]. It has been extensively used in the web domain [6, 7, 8], especially through A/B testing where an original version (A) of the web application is tested against an alternative version (B) [9]. According to recent research [3, 10, 11], few companies have adopted continuous experimentation as the transition from CD to R&D as an experiment system has proven difficult [3], especially when it comes to gathering the right data from the customers [12]. In order to help companies in continuously experimenting with the customer, several models (HYPEX [12], QCD [13], EVAP [14], RIGHT [15] and ESSSDM [16]) have been created. Nevertheless, as is evident by the low number of companies that have managed to climb the entire 'Stairway to Heaven', there exists room for improvement.

In this study, a process model for continuous experimentation is created. For this purpose, the 'Hypothesis Experiment Data-Driven Development' (HYPEX) model is used as a basis. The HYPEX model was created to close the 'open loop' between customers and product management as defined in [12]. This loop results in a situation where customer feedback is not easily accessible for decision-making. Therefore, decisions tend to be based on opinions and 'gut feeling' rather than relevant data [5, 12]. The HYPEX model describes how companies can run feature experiments to close the 'open loop' in order to confirm that a feature that was selected for development has the expected value. However, neither the HYPEX model nor other related models ([13], [14], [15] and [16]) provide details on how a feature can be modeled to enable value prediction in a concrete way. Furthermore, they do not cover the details on how to use appropriate customer feedback and data collection techniques to collect relevant data and determine the realized value. This study seeks to fill that gap. While several methods are used in order to investigate the problem area, interviews at two Business-to-Business (B2B) companies play a central role, as well as workshops that included two additional B2B companies.

## 1.1 Problem Statement

In an ideal world, companies would be able to estimate and track the creation of value for a feature with good precision and accuracy. They would then also be able to compare the tracked realized value with their predictions and use the result to make informed decisions about the future of the feature. In reality, this has proven difficult for companies to achieve [3, 10, 11, 12]. Thus, the following two problems are still largely unsolved in the realm of continuous experimentation:

- *Problem 1:* How to model and predict a value of a feature in a concrete way?
- *Problem 2:* How to track feature value throughout the development using the appropriate customer feedback and/or data collection techniques?

While several models exist in the area [12, 13, 14, 15, 16], they are on a high conceptual level and lack the amount of detail required to fully solve Problem 1 and 2.

## 1.2 Research Objective and Research Questions

The objective of the study is to create a process model that can help companies in running feature experiments to close the 'open loop'. The idea is that companies can use the process model to model a feature and predict its value in a concrete and directly applicable way. Then the model should also allow for the tracking of realized value during development of the feature as well as comparisons between the realized value and the predicted one in order to conduct an analysis on whether the feature is living up to its expectations.

Based on the problem statement and the objective, the study has two research questions:

- *RQ1:* How to model, predict and analyze feature value in a way that is useful and directly applicable by companies?
- *RQ2:* How to track the realized feature value using appropriate customer feedback and data collection techniques?

## 1.3 Software Center Project

This study is an offshoot of a larger ongoing project carried out in the Software Center which is a collaboration between universities and companies [17]. The main goal of the Software Center is to increase productivity in industry, as well as providing knowledge transfer between industry and academia. This larger project is named 'Fast Customer Feedback in Large Scale SE' and will hereafter be referred to as 'the Software Center project'. The main idea behind the Software Center project is that by speeding up the feedback loop with customers and making use of the available customer and product data, companies can optimize resources by only working on features that add value [18]. The goal is to refrain from having to make decisions based on 'gut-feeling' and rather make informed decisions based on available data.

## 1.4 Outline of the Thesis

The remainder of this thesis is structured as follows. In chapter 2, the necessary background concepts which are the foundation for this study are covered. Chapter 3 covers the related work of the study. In chapter 4, the research methodology is described. Chapter 5 presents findings from the interviews and the workshops, as well as the process model and the prototype. Chapter 6 discusses the outcome of the validation. In chapter 7, the findings are compared to other findings in the literature, the results are discussed with regards to the research questions, the contribution to knowledge is presented, as well as validity threats and future work. Finally, the thesis is concluded in chapter 8.

# 2

# Background

This chapter provides the background concepts that are the foundation for this study. In section 2.1, continuous experimentation and related concepts are discussed, including the Stairway to Heaven and its last step, R&D as an experiment system, the 'open loop' problem and challenges that companies face when working with continuous experimentation. Section 2.2 describes the HYPEX model, its limitations and how this study aims to improve and extend the HYPEX model. Finally, section 2.3, describes how relevant data can be collected for the experimentation process.

## 2.1 From Traditional Development to Continuous Experimentation

Traditional software development was conducted in stages; starting with identification of the requirements to the design, implementation, testing and maintenance of the software. Each subsequent stage starts after the previous one has been finished, in what is often named 'the Waterfall process'. However, the business environments of today's software development are rapidly changing and contain a lot of uncertainty. This is true for the markets companies operate in, the requirements from their customers and users as well as for the ever-advancing technology their products are utilizing [1]. In response, agile practices favoring flexibility and shorter time-to-market through faster cycles have been widely adopted among software development companies. However, the adoption of agile methods within R&D can be viewed as only a step in the evolution of software development practices of companies [3].

This evolution path, named 'Stairway to Heaven' [3], depicts the typical development of a company from traditional development to agile practices and beyond. It consists of five steps:

- Step A, *Traditional development* is the starting point where companies use traditional processes, e.g. 'the Waterfall process'.

- Step B, *Agile R&D organization*, introduces agile practices to product development.
- Step C, *continuous integration*, introduces agile practices to system validation. In addition, both product development and system validation adopt very short cycles – down to once a day or even less. Continuous integration can be defined as integrating software multiple times a day using, e.g., automated tests and builds [4].
- Step D, *Continuous deployment*, brings agile practices to product managers and customers. After reaching step D, the entire organization is involved in an agile development cycle. Continuous deployment can be defined as continuously deploying software to customers, opening up the possibility of gathering customer feedback more rapidly [3].
- Step E, *R&D as an experiment system*, is about introducing short feedback cycles to product management and customers. This is done by changing the development to focus on hypotheses that are rapidly tested and validated, or discarded, based on customer and user feedback [5]. The core idea is that companies should invest as little as possible until deploying to customers. Successfully adopted it allows the entire organization to rapidly respond to and act on instant feedback from customers and users.

R&D as an experiment system tries to utilize the possibilities enabled by continuous deployment. When products are deployed continuously a company has the possibility of being data-driven to a new extent. This kind of value-based software engineering is about continuously collecting relevant data, e.g. expenses and profits, and using that data to make decisions on what future steps should be taken [19]. In other words, to allocate resources with the aim of minimizing waste and maximizing the overall value of the company in question. Products can then in theory be developed based on requirements that are updated in real time based on actual customer and user data from current versions of the product [5]. This is in big contrast to the traditional way of opinion-based decision-making about how future requirements will be like when the product is released a few years down the line. Many companies also have large amounts of data already collected from the three phases of development: pre-development, development (also known as non-commercial deployment) and post-deployment (also known as commercial deployment) [5, 20]. R&D as an experiment system provides a golden opportunity to leverage that data in the decision making as well [20]. This step can be described as a continuous experimentation of customer value utilizing both customer feedback and product data. In [10], several practices are introduced that can help companies in reaching the last step of the 'Stairway to Heaven'.

The act of continuously experimenting made its debut in areas such as software as a service (SaaS) and in cloud computing [5]. Furthermore, it has been widely adopted in the web domain. Companies such as Google [6], Microsoft [7] and eBay [8] all embrace the experiment aspect of software engineering. An often used practice in the web domain is A/B testing, where an original version (A) of the web application is tested against an alternative version (B) in order to draw a statistical conclusion on which version is better [9]. Bosch [5] provides three distinguishing factors for continuous experimentation over traditional development approaches:

1. The software is continuously evolved by deploying the software rapidly.

2. Gathering relevant data, both directly from customers and through their usage of the product in the field, throughout the development process.

3. Many ideas are tested and, therefore, requirements are not set in stone. This is done in order to maximize both customer satisfaction and revenue.

In [21], Bosch and Eklund add a fourth factor:

4. Customers can get software updates throughout the lifecycle of the product, allowing the product to be usable and of value for a longer time.

This is in line with the last step of the 'Stairway to Heaven' [3].

There are few companies that have already succeeded in climbing the entire stairway and thus can perform R&D in a rapid, experiment-focused way [3, 10]. The ones that succeed in this climb are then able to handle the need to continuously evolve their product and stay competitive through flexibility, efficiency and speed [3, 10, 21]. However, continuous experimentation can be used wherever sufficient data collection can be carried out [5]. There are many challenges that companies need to overcome in order to reach this final stage of the stairway [3]. In particular, for taking the step to R&D as an experiment system, it has been noted [12] that collecting data from customers is not always straightforward. Traditionally, agile practices [2] make use of a product owner as a customer representative. In large-scale software development, the product owner can talk to a subset of the customers but that does not always represent the whole [22]. Furthermore, customers often do not know what they actually want [11]. This can lead to a situation where requirements are prioritized in an opinion-based manner instead of being data-driven. Thus, R&D resources are at risk for focusing on non-value adding features [23], especially since they are not continuously validated throughout the development [12]. In [12] the term 'open loop problem' is coined. It refers to an 'open loop' between customers and product management, i.e. that customer feedback is not readily available when it comes to making decisions. Therefore, decisions continue to be based on opinions and 'gut feeling' rather than the relevant data [5, 12].

As can be seen, continuous experimentation has been an active research area in the past years. Within continuous experimentation, several challenges have been identified. Lindgren and Münch [11] found that a key challenge in using continuous experimentation is getting the organization as a whole on board. A finding that Olsson, Alahyari and Bosch [3] mention as well. In addition, Lindgren and Münch [11] state that although the companies collect large amount of data, both customer feedback and product data, the product data is not focused enough towards the data-driven aspect. In other words, the focus is more on using product data for troubleshooting and other related activities, not for data-driven development. Furthermore, they discuss potential future work in including the experimentation aspect in the development process itself, as well as developing a tool for the experimentation. In [3], Olsson, Alahyari and Bosch emphasize the need for instrumentation when moving to the last step in the 'Stairway to Heaven', which Bosch [5] also recognizes and adds that the instrumentation is crucial in the development phase, as well as in post-deployment. In [24], Sauvola et al. recognize the difficulties of collecting data, analyzing it and using it in the development process. In [25], Fagerholm et al. define prerequisites for an experiment system. Such a system must, e.g., be able to release a minimum viable feature with the required instrumentation.

Furthermore, several models (HYPEX [12], QCD [13], EVAP [14], RIGHT [15] and ESSDM [16]) have been created, aiming to contribute to a solution that would allow continuous experimentation with the customer in order to steer R&D in the right direction. The HYPEX model will be discussed in section 2.2, while the other models will be presented in chapter 3: Related Work.

## 2.2 HYPEX

In [12] four problems, related to the 'open loop' between customers and product management, are discussed:

- There is not enough confirmation from the customers that the features being developed will be valuable to them.
- The prioritization process is driven by opinions, especially of senior staff.
- It is not fully clear what a feature should contain.
- There is a risk of developing a product that does not fulfill the wishes of the customers.

In order to solve or mitigate these problems and closing the 'open loop' the 'Hypothesis Experiment Data-Driven Development' (HYPEX) model was developed. The model is illustrated by Fig. 1 [12].

It is a development process model based on six practices and describes how R&D as an experiment system can be done on a team process level. In the first practice, *Feature backlog generation*, a set of features that might be valuable for the customer and thus possibly implemented are selected. When a new feature can be selected from the backlog, the second practice of *Feature selection and specification* dictates that a feature should be selected with regards to its priority by the company or the customer. The potential value that the feature can add is then described, as well as how it connects to the overall business goals of the company and its expected behavior. When a feature has been selected and specified it is time for the third practice, *Implementation and instrumentation*. There, a minimal viable feature (MVF) is implemented and the feature

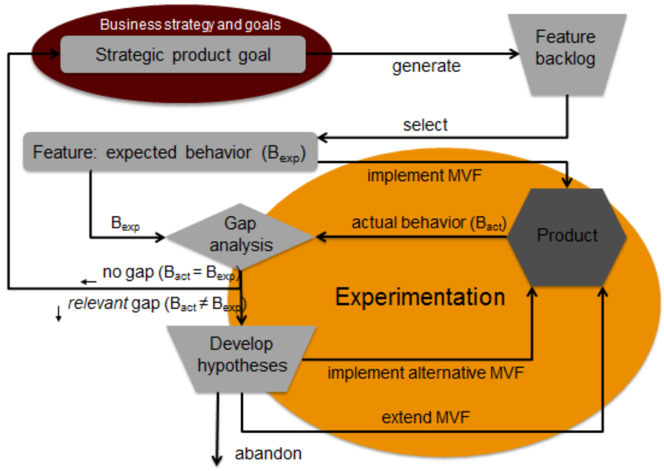


**Fig. 1:** The HYPEX model as depicted in [12].

is instrumented so relevant data can be collected about the actual behavior of the feature. In the fourth practice, *Gap analysis*, an analysis is carried out to determine whether the actual behavior of the feature is close to its expected behavior. This is a core step in closing the 'open loop', as the company makes use of the available data from the instrumentation instead of assuming that the feature lives up to its expectations. If the gap is small enough, the feature can be widely-deployed. However, if the gap is too large, either hypotheses that explain the difference are generated or a decision is made to abandon the feature. The fifth practice, *Hypothesis generation and selection*, has to do with the actual generation of hypotheses to explain the gap between the actual behavior of the feature and its expected behavior. Two possible scenarios arise once hypotheses have been generated: 1) If the customer does not see the benefits of the feature, e.g. if the customer does not consider the feature to be an MVF as it lacks functionality (and it is not suitable to abandon it), it needs to be extended, following through with a similar process as has been described. 2) If the quality of the initial MVF is questioned, an implementation of an alternative version must be carried out. Then the sixth practice, *Alternative implementation*, describes how the original version (A) is tested against the alternative version (B). If working with embedded systems, randomly assigning a customer to either use the A version or B version is not as straightforward as in the web domain [6, 7, 8]. Therefore, the testing must be done sequentially in order to find out which version is superior.

The HYPEX model can be viewed as a way of conducting feature experiments with the aim of improving the prioritization process, i.e. not having it opinion-based but data-driven, and closing the 'open loop' between customers and product management [12]. Therefore, the HYPEX model fits well with the notion of R&D as an experiment system, and as a final step in the 'Stairway to

Heaven'.

The aim of this study is to create a process model that is an extended and more concrete version of HYPEX. HYPEX does not go into detail on how a feature and its value is modeled and defined. Because of this, it is not possible to give a concrete answer on the actual difference between the expected behavior and the actual behavior of the feature. In addition, HYPEX only mentions that a feature needs to be instrumented in order to gather the relevant data, but no guidelines or methods are discussed on how the instrumentation can be carried out. Therefore, this study has the following contributions regarding the extension of HYPEX:

- A detailed procedure for modeling a feature.
- Once modeled, how the value of a feature can be predicted.
- How to select the appropriate customer feedback and/or data collection techniques to track the value of the feature, as well as instrumenting the system if needed.
- An extended description of comparing the tracked realized value to the predicted one in order to make informed decisions on the future of the feature.

As can be seen, this study seeks to both extend the HYPEX model, as well as making it more concrete by providing a way to calculate the actual value of the feature.

## 2.3 Metrics, Instrumentation and Data Collection

Measurements are an important aspect in all engineering fields, and software engineering is no exception [26]. Measurement programs [27] provide a way to monitor the quality of a software artifact, as well as making predictions and estimations. Although, the importance of measurements and measurement programs are evident, implementing a measurement program within an organization is challenging [26]. An organization that takes decisions based on measurements needs to know if they are actually measuring the attribute that they think they are measuring [28]. Otherwise, the decisions are not based on a solid ground.

According to [26], most measurement planning models and tools adhere to goal based approaches. A well-known goal based approach is the Goal/Question/Metric (GQM) paradigm [29]. GQM uses a top-down approach where goals are identified, questions are formulated about how the goals can be evaluated, and metrics are used to answer the questions. The mapping between goals, questions and metrics is not one-to-one as a single metric can be used to answer more than one question, and a question can be connected to more than one goal [30]. It should be noted that there is a difference between a measurement and a metric. A metric is a measurement function or "the function that assigns a value to the attribute," and a measurement "is the empirical, objective assignment of numbers, according to a rule derived from a model or theory, to attributes of objects or events with the intent of describing them" [28].

Instrumenting a system is the practice of adding log statements to a code with the aim of monitoring the system [31, 32, 33]. Previously, logging was mainly intended for troubleshooting and debugging. However, recently it has gathered more interest due to data-driven practices being on the rise [31, 32, 34]. It is important to find a balance between logging too little and logging too much. If too little is logged then necessary information might not be gathered while logging too much increases the need for resources, especially storage space [35]. In [36], it is argued that "bigger is not better" when it comes to big data and care must be taken when dealing with such a large amount of data. This is in line with [13]. An implication from too much logging is discussed in [32], where a developer mentions how much is being logged and that at a feature level the amount of data can be hard to work with. In [35], the need for an automatic logging tool is recognized. The authors of [35] propose an automatic classification approach for logging. Their results suggest that automatic logging can be feasible but recognize that further work is needed. That is in line with a recent paper [37], stating that there is a need to help developers know when and where to log.

However, instrumentation is not the only way to gather information about the system. In [31], two main categories are presented for gathering data: 1) user action and 2) user attitude. Log statements in the code can be used to gather information about the first category, while e.g. surveys are useful for the second category. For the logs, a question of whether the relevant aspects are being logged or not should be considered. For the surveys, it must be possible to send out a survey regularly to gather the necessary data.

Similarly, a literature review on customer feedback and data collection techniques in software R&D [20] categorizes the techniques, based on [9], to qualitative ones and quantitative ones. Qualitative feedback techniques require the customer or the user to be an active participant. In quantitative data collection techniques, the customer or the user is usually not an active participant and unaware of his or her involvement. Usually, the latter case produces significantly more data than the former case [9, 20]. It should be noted that continuous experimentation opens up the possibility of gathering a large amount of quantitative data with a relatively low cost [21]. In addition to categorizing the techniques, the literature review covers what techniques are applicable in the different stages of the development process. Qualitative feedback techniques are prevalent in the early phases, while quantitative ones are more dominant in the later phases of the software development.

It should be noted that in the Business-to-Business (B2B) domain there is a need to distinguish between a customer and a user [11]. Generally, a customer buys a product while the user uses it. The previously mentioned literature review on customer feedback and data collection techniques in software R&D [20] focuses on customers rather than users, i.e. on the B2B domain. As the four investigated companies are in the B2B domain, the literature review is considered a good fit. Nevertheless, most of the techniques can be used with either customers or users. For example, as has been recognized in the web domain [6, 7, 8], A/B testing is a widely used technique in order to find out which version delivers more long-term value in relation to business goals. Operational data and incident reports can give information on how a user uses a product and if any problems occur. Surveys and customers can be sent to users as well as customers and so forth.

# 3

# Related Work

In this chapter the related work of this study is presented. In order to find relevant literature a snowballing approach is used [38]. The approach can be used to carry out a systematic literature review. Although the aim of this study is not to do a systematic literature review on models that cover continuous experimentation in software engineering, doing it in a systematic way increases the chance that all relevant studies are found. The snowballing approach will be discussed in section 4.1, within the Research Methodology chapter.

In [16], Bosch et al. discuss the challenges that software startup companies face and the uncertainty that they must endure. The authors developed the Early Stage Software Startup Development Model (ESSSDM) in order to help software startup companies in 1) investigating multiple products ideas in parallel, 2) move product ideas forwards, 3) abandon a product idea when needed, and 4) using suitable techniques to validate product ideas. The model consists of three steps. The first one is to generate an idea. The second is to prioritize the ideas while in the third step the ideas are validated using the Build-Measure-Learn loop. The Build-Measure-Learn loop [39] turns ideas into products (Build), then the usage of the product is measured (Measure) before analysis of the data is performed and improvements can be made (Learn). The third step has four stages, where the ideas are validated at different stages: problem validation, solution validation, minimum viable product validation at a small-scale and a minimum viable product validation at a large-scale. After each Build-Measure-Learn loop, a decision must be taken on the future of the idea.

In [13], Olsson and Bosch discuss the challenges that large-scale software intensive companies face when prioritizing their work. They recognize that there is an 'open loop' between customers and product management, i.e. that customer feedback is not readily available when it comes to making decisions. Although the companies gather a large amount of data, it is not used in a systematic way and the qualitative data that is gathered in the early stages of development is not validated later on, giving rise to commercially deployed features that have not been fully validated. There is then an increased risk that these features will not see much, if any, use. In order to solve these problems, Olsson and Bosch present the 'Qualitative/quantitative Customer-driven Development' (QCD) model. It is a conceptual model where requirements are not considered to be set in stone, i.e. they are treated as hypotheses. These hypotheses are not only specified early in the development process, but throughout it. Furthermore, the hypotheses need to be validated with the customer, in order to confirm that they are in fact valuable to the customer. By doing this, QCD can help companies in closing the 'open loop'. By continuously validating features, QCD can also help companies in reducing the number of unused features as well as reducing the number of incorrectly implemented features. QCD emphasizes the usage of many different customer feedback techniques, which can aid companies in better understanding the needs of their customers. In addition, collecting data about feature usage can help in knowing what can be improved within the feature. Finally, by combining different kinds of data it is easier to validate the collected feedback from the customer and makes it clearer what data is important and what is not. Continual validation also helps with this. In both [22] and [40], Olsson and Bosch continue with their validation of the QCD model.

In [14], Fabijan, Olsson and Bosch develop a technique to help validate hypotheses in the QCD model, named Early Value Argumentation and Prediction (EVAP). The EVAP technique aids companies in dynamic prioritization of features, in developing a MVF (Minimum Viable Feature) and in stopping development if the expected value of the feature does not meet expectations. All this helps product management in redirecting the efforts of R&D to features that have more value [14], i.e. to develop features that add value to the product and discard those that do not.

In [15], Fagerholm et al. present the RIGHT model for continuous experimentation, which is based on previous work [25] by Fagerholm et al. They recognize that the problems that companies face today are moving from being of technical nature to identifying what the customer actually wants. This model is in the context of R&D as an Experiment System [3]. As with [16], it makes use of the Build-Measure-Learn loop. In the RIGHT model [15], assumptions are made on what needs to be carried out in order for a product or service to be marketable. The assumptions are tested with experiments and formulated as hypotheses. Based on these hypotheses, a minimum viable product or a minimum viable feature can be created to use in the experiment, given that the proper instrumentation is available. Data is then collected and used to make informed decisions on the future of the product or the feature.

The ESSSDM model can be useful in a startup environment where continuous validation of customer value is necessary. However, it is more focused on the idea behind the minimum viable product than the actual product. For example there is no concrete definition of the value that the product brings or how the value can be determined. In addition, as it is intended for use in a startup environment, it might not be suitable in large-scale R&D. In the QCD model, the requirements are not set in stone, thus welcoming the opportunity of continuously experimenting with their feasibility. Though QCD does not cover the actual development of a feature (or a product) and how the customer feedback can be used to know if a feature is living up to its expectations or not. The EVAP technique, which builds upon QCD, goes further and discusses the development of a minimum viable feature and the expected value that the feature is supposed to bring. Nevertheless, the details on the actual expected value and how companies can know whether it has been achieved are not discussed. The RIGHT model has the most similarities to this study. It covers the whole development cycle, discusses how a hypothesis is selected as well as how it can be tested and used for decision-making. However, no concrete values are created through usage of the model which would help companies in determining whether a feature is a suitable candidate. Finally, the details of gathering the customer feedback data and the actual instrumentation are not covered.

In addition to the models that were found during the systematic literature review there is also release planning. It did not show up in the snowballing procedure as release planning is not directly connected to continuous experimentation. However, the concept is well known and does provide an interesting view on the prioritization of features. The idea is that software is developed in an incremental and iterative way and that features are prioritized in order to divide them into different releases of the software [41]. Release planning has been widely studied and several process or methods have been created, e.g. [42], [43], [44], [45] and [46]. However, release planning focuses mainly on selecting the right features for implementation and not on tracking the value of the feature throughout development in order to find out if it meets expectations or if other alternatives are better suited. Therefore, release planning can very well be used alongside continuous experimentation, e.g. to select features for experimentation, but it does not replace it or solve the underlying problems that this study seeks to address.

The discussion of the related work presented in this section and the previously discussed challenges in section 2.1 clearly show that there is a need within the research area for the contributions of this study. In particular, the main contributions of 1) creating a detailed procedure for modeling a feature, 2) predicting feature value, and 3) selecting the appropriate customer feedback and data collection techniques to track the value of the feature and instrumenting the system if needed have not been successfully applied.

# 4

# Research Methodology

The overall goal of the research is to create a process model that helps decision-makers to estimate and track the creation of value for a feature during development. In order to facilitate the use of the model in practice, an example implementation of it is created in the form of a prototype. Design science research was selected as a research methodology. Design science research [47] is concerned with the iterative process of designing an artifact and investigating it in context. An artifact can be considered anything from a prototype to methods and techniques. There is some variation in literature when it comes to defining the design science research process [47, 48, 49, 50]. However, they all have an underlying theme of identifying a problem, designing and creating an artifact and evaluating the artifact.

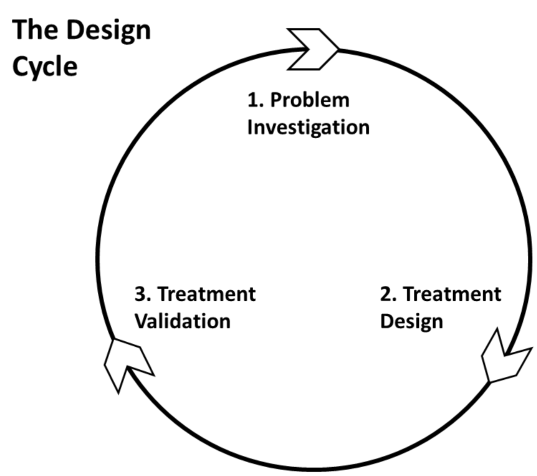


**Fig. 2:** The design science research process model that is adopted in this study. It is called the design cycle and is described in [47].

The process model that is adopted in this study is described in [47]. The process model, referred to as the design cycle (Fig. 2), consists of three steps or phases: 1) problem investigation, 2) treatment design, and 3) treatment validation. The author uses treatment instead of a solution as a treatment fits well with how an artifact can be used in a specific context to treat a problem [47]. The design cycle is a part of a larger cycle, named the engineering cycle (Fig. 3), which consists of two extra steps: 4) treatment implementation, and 5) implementation evaluation.

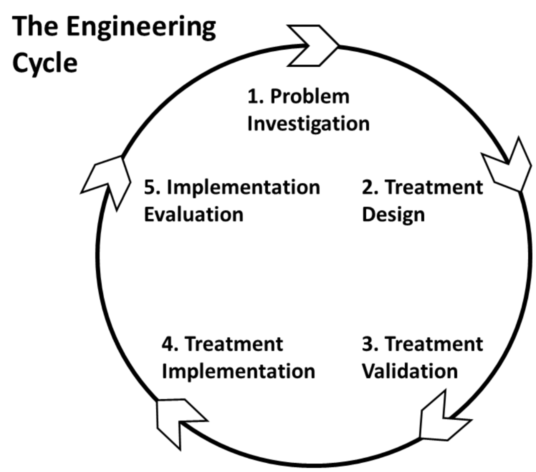


**Fig. 3:** The design cycle is a part of the engineering cycle. The latter has two extra steps that are usually not covered in a design science research project [47].

In [47] the author explains how the actual implementation is viewed differently for a researcher and a stakeholder. From the researcher's point of view, several implementations and evaluations are made in the research project. From the stakeholder's view, only possible treatments are designed and validated. The author concludes that the problem should be viewed from a stakeholder's perspective and therefore the design cycle is used in research projects as the actual implementation and evaluation is done following the completion of the project. In other words, implementing the actual artifact in a real-world setting is not part of the research project.

Research problems in design science research can be divided into design problems and knowledge questions [47]. There can be many solutions to a design problem and the solution is judged based on the goals of a stakeholder. The aim is to change an aspect that concerns the stakeholder. Knowledge questions assume that there is only one answer, although it might not be fully known. The aim is to learn about a certain aspect, but not change it. This study addresses a design problem as there is no universally correct answer to the research questions of this study. However, knowledge questions can be used to study the problem, its context and how an artifact interacts with the context that it is applied in. Another iteration through the design cycle can then be made with the gained knowledge. Therefore, the design cycle is not just about solving a design problem as it also creates knowledge along the way [47].

In section 4.1, the design cycle is discussed and its activities are presented. Section 4.2 presents a protocol for the data collection, analysis and validation. In section 4.3, the procedures for the data analysis are described. Section 4.4 covers the validation procedures. In section 4.5, guidelines for applying design science research are presented. Finally, section 4.6 discusses potential alternatives to design science research.

## 4.1 The Design Cycle

The design cycle consists of three steps: problem investigation, treatment design and treatment validation. The overall goal of the study was to create a process model. In addition, a prototype was created based on the model in order to facilitate the use of the model in practice and help with validating the model. Five iterations were made through the design cycle with regard to the process model, thus giving five versions of the model, each one building on gathered knowledge from the previous one. Table 1 provides an overview of the three steps of the design cycle and its main activities.

### 4.1.1 Problem Investigation

In the first step of the design cycle, the problem itself is investigated. This is done in order to understand the problem at hand and learn what the stakeholders are interested in and how they want to improve. When applicable, the validation of the artifact from a previous iteration is also considered in this step.

A problem can be identified in many ways and through multiple sources [47, 48]. What follows is a description of the activities.

#### Literature

In order for the researchers to have a basic understanding of the core problem and of the research area, it was important to review relevant literature. Chapter 3 is an output of that work. In order to identify relevant literature, a snowballing approach was used. When snowballing [38] a start set of papers needs to be identified, e.g. by using a database search. Potential papers are than either included in the start set or excluded based on a pre-determined criteria. Once a start set has been identified, backwards snowballing is applied. For each paper in the start set, the title of the papers in the reference list is read. If the paper is considered a candidate the next step is then to look at the place of the reference in the paper. If it is still considered for inclusion, the paper is located and the abstract and other parts of the paper is read as needed to see if it should be included or excluded. Once backwards snowballing is done,

**Table 1:** The three steps of the design cycle and their main activities.

| Step in the Design Cycle | Iteration | Activities |
|---|---|---|
| Problem investigation | Iterations I-III | • Literature.<br>• Discussions with researchers.<br>• Interviews.<br>• Workshops.<br>• Previously collected data.<br>• Input from treatment validation (not in Iteration I) |
| | Iteration IV-V | • Literature.<br>• Discussions with researchers.<br>• Input from treatment validation |
| Treatment design | Iterations I-III | Process model. |
| | Iteration IV-V | • Process model.<br>• Prototype. |
| Treatment validation | Iterations I-III | Presentation to the researchers in the Software Center project. |
| | Iteration IV | • Presentation in a Software Center project workshop.<br>• Validation sessions at companies. |
| | Iteration V | • Validation sessions at companies. |

forward snowballing is performed. Then citations to the paper being examined are identified and those papers are viewed in a similar way as before in order to determine whether they should be included or excluded. For every new paper that is added to the start set, both backwards snowballing and forwards snowballing need to be applied until no further papers are found.

For this study, the abstract and citation database Scopus is used. The following search string was used:

*TITLE-ABS-KEY((("feature experiments" OR "continuous experimentation" OR "customer validation" OR "Stairway to Heaven" OR "open-loop" OR "R&D as an experiment system") AND ("software development" OR "data-driven development" OR "data-driven software development" OR "value-based development" OR "value-based software development" )) OR ((logging OR telemetry OR instrumentation OR "event logging") AND ("data collection" OR "customer feedback" OR "end-user feedback") AND development)) AND ( LIMIT-TO(SUBJAREA,"COMP"))*

It gave 97 results and, after going through every title, 13 papers were considered for inclusion. The final inclusion criterion was:

*The paper presents a model that covers requirements, features, products or systems in the context of continuous experimentation.*

After reading the abstract, and other parts of the paper if necessary, five papers (P1-P5) were included in the start set. After applying both backwards snowballing and forward snowballing, two additional papers were added to the start set. One paper (P6) while performing backwards snowballing and one paper (P7) for forward snowballing. The papers, the number of references, references to the start set and any potential new papers that were found are listed in Table 2.

### Discussions with Researchers

There are three researchers in the Software Center project. All of them were a valuable source of information for this study. The communication was in the form of two week sprints that started with a meeting with all of them. These two week sprints became the foundation of the workflow for the project. In the second week of each sprint a meeting with one of the researcher was held. Through discussions in these scheduled meetings the problem area became clearer.

### Interviews

Semi-structured interviews [51, 52, 53], were a central part of the problem investigation step in the design cycle. As they allow both for planning of questions in advance and for exploring interesting topics that might arise during the interview, they seemed a good fit for this research. The same interview questions

**Table 2:** Papers identified during the snowballing procedure.

| Paper No. | Paper Title | No. of References | References to Start Set | New Papers |
|---|---|---|---|---|
| P1 | Olsson, H.H., and Bosch, J. 2015. Towards continuous validation of customer value | 21 | P5 | None |
| P2 | Fagerholm, F., et al. 2014. The RIGHT model for Continuous Experimentation | 32 | P6 | None |
| P3 | Olsson, H.H., and Bosch, J. 2015. Towards continuous customer validation: A conceptual model for combining qualitative customer feedback with quantitative customer observation | 30 | P5 | P6, P7 |
| P4 | Fabijan, A., Olsson, H.H., and Bosch, J. 2015. Early value argumentation and prediction: An iterative approach to quantifying feature value | 17 | P3, P5 | None |
| P5 | Fagerholm, F., et al. 2014. Building blocks for continuous experimentation | 24 | P6 | None |
| P6 | Bosch, J. et al. 2013. The Early Stage Software Startup Development Model: A Framework for Operationalizing Lean Principles in Software Startups | 22 | None | None |
| P7 | Olsson, H.H., and Bosch, J. 2016. From Requirements to Continuous Re-prioritization of Hypotheses | 21 | P3, P5 | None |

(Appendix A) were used throughout the interviews. In addition, an interview guide [52] was created which included the interview questions, along with possible further questions, as the nature of semi-structured interviews allows for some diversity depending on how a specific interview plays out. Furthermore, some possible additional questions were added to the interview guide due to new insights from analysis of previous data collection. The original interview guide was reviewed by two researchers in the Software Center project. The interviews were divided into three parts, influenced by [51]:

- *Introduction:* A short introduction was provided on the thesis work, the purpose of the interview and possible relevance of the interviewee to the work.
- *Context:* General questions followed, with focus on day-to-day tasks of the interviewee and his or her immediate colleagues.
- *Hypothesized feature value realization and validation:* The main part of the interview consisted of a series of questions addressing hypothesized feature value realization and validation.

The Software Center project provides the context for this research and therefore it was appropriate to select companies that were already part of the project. Interviews were carried out at two companies, Company A and Company B. For Company A, two units participated, hereafter referred to as Company A-I and Company A-II:

- *Company A* is one of the world's leading telecommunications providers with a major presence in the mobile network infrastructure market.
  - *Company A-I* has a stronger focus on the embedded side of the company.
  - *Company A-II* has a stronger focus on the software side of the company.
- *Company B* develops software specialized for navigational information, operations management and crew and fleet management solutions.

Both researchers carried out the interviews, where one researcher was in charge of asking questions while the other one took notes. In the beginning of the interview, the interviewee was reminded that the researchers had signed a non-disclosure agreement (NDA) and that they wished to audio record the interview. All interviewees agreed to having the audio recorded. After the first part of the interview, where a short introduction of the thesis work was provided, the interviewee was asked if they had any questions at that point. At the end of the interview the interviewee was offered to share additional thoughts and encouraged to contact the interviewers if any questions came to mind or if some aspects of the interview needed to be clarified.

A total of six interviews were conducted. Table 3 shows how they were divided into the first three iterations of the design cycle. It also shows the company

**Table 3:** The interviews were divided into the first three iterations of the design cycle.

| Iteration | Interview | Company | Role of interviewee(s) |
|---|---|---|---|
| I | #1 | A-I | Area Product Owner |
| II | #2 | A-I | Product Manager |
| | #3 | B | System Architect |
| III | #4 | B | Product Owner |
| | #5 | A-I | Developer |
| | #6 | A-II | *Two interviewees:*<br>Program Manager<br>Technical Coordinator/Developer |

of the interviewee as well as the interviewees role. The first five interviews took place at the respective company, lasted for approximately one hour and had one interviewee. The sixth one was conducted over phone, lasted for approximately half an hour and had two interviewees. Three of the interviewees are employees at Company A-I, two are employees at Company A-II and two are employees at Company B. All of the interviews were transcribed and the interviewee was offered the transcript. This was done to ask for feedback and allow the interviewee to clarify or even change their answer in case of misunderstandings [51].

### Workshops

During the thesis work, three workshops were held in the Software Center project. The first workshop was a part of the first iteration in the design cycle, the second one was part of the second iteration and the third workshop was used for validating the process model and the prototype in the fourth iteration of the design cycle. The third workshop will be discussed in section 4.4. One researcher was an observer in the first workshop and both researchers were observers in the second one. The first and the second workshop lasted 2 hours and 30 minutes. Company A-I and Company B participated in the workshops along with two other companies: Company X and Company Y. The interviews played a central role in the problem investigation step and Company A-I, Company A-II and Company B are thus the major sources of information, while Company X and Company Y provide additional information.

- *Company X* is a multinational manufacturing company mainly focused on trucks, buses and construction equipment.
- *Company Y* is a network video company and sells, e.g., network cameras, video encoders and video management software.

The first workshop was held in February 2016 and was the first data collection opportunity in the thesis work. All three researchers from the Software Center project participated, along with two employees at Company A-I and one employee at Company B. It provided concrete information about the Software Center project and an insight into how the companies deal with e.g. customer feedback and feature value prediction. This study was briefly introduced which laid the ground for future interviews with some of the other participants in the workshop.

The second workshop was in March 2016. All three researchers from the Software Center project participated, along with one employee from Company A-I, one employee from Company B, one employee from Company X and one employee from Company Y. Both the employees from Company A-I and Company B participated in the first workshop as well. This workshop lead to a deeper understanding of the problem area and an update of the progress in the Software Center project.

#### Previously Collected Data

Although direct methods in the form of semi-structured interviews and workshops was the largest part of the data collection, an independent analysis was also carried out on previously collected data. This data was from interviews that had already been carried out by a researcher in the Software Center project. The main contribution of the data was providing the researchers with context and background information about the specific features that had been used before which aided the design of the interview guide.

#### Input from Treatment Validation

The gathered knowledge from the third step in the design cycle, Treatment Validation, was used as an input in this step. Therefore, in Iteration II, the treatment validation from Iteration I was used as an input. In Iteration III, the treatment validation from Iteration II was used as an input and so on.

### 4.1.2 Treatment Design

The design of the process model was based on the outcome of the problem investigation. The interviews and the workshops were the main focus of the problem investigation. The findings and derived hypotheses from the findings are described in section 5.1. For Iteration V, the treatment validation from Iteration IV also played a key role. The process model is described in section 5.2.

For Iteration IV and V, when the prototype was created, a set of requirements was specified. Due to time constraints, the prototype could not cover all aspects of the process model. Therefore, it was important that only the core aspects needed to realize the model were selected. With that in mind, the prototype is only an example implementation of the process model. The prototype is described in detail in section 5.3.

### 4.1.3 Treatment Validation

The third and final step of the design cycle is validating the artifact. In [47], several methods are described for validating an artifact. One of those methods is expert opinion, where the goal is not to gather statistical data about whether the artifact fulfills its goals, but to ask the experts to provide feedback based on how they think it will work in a real-world setting. Several aspects can be considered:

- *Effect questions:* What happens when the artifact interacts with the context?
- *Trade-off questions:* How does the artifact perform in comparison with similar artifacts?
- *Sensitivity questions:* Will the artifact be useful in different contexts? How does it scale?
- *Requirements satisfaction questions:* Does the artifact fulfil the requirements of the stakeholder?

In addition to expert opinion, simulations were carried out [50]. It is an experimental method where the prototype is used with test data in order to show how it performs.

Due to time and resource constraints, the process model could not be validated with the companies during the first three iterations. Instead, each iteration it was first presented to one of the researchers in the Software Center project before it was refined further and presented to all of the researchers in the beginning of the next sprint. At the end of Iteration III, the process model was deemed ready for validation at the companies and, therefore, it was time for Iteration IV where the prototype was created. In Iteration IV and Iteration V a validation was carried out in collaboration with the companies, as will be described in section 4.4.

## 4.2 Protocol

Collecting data through interviews and workshops is of flexible nature. Nevertheless, it is important to plan these activities from the beginning [51]. Therefore, a decision was made to keep a protocol, influenced by the practice of case study protocols. The protocol consists of the following sections, as proposed in [54] and summarized in [51] but with minor modifications from the researchers:

- *Preamble:* Describes the purpose of the protocol.
- *General:* An overview of the project and the companies.
- *Procedures:* A detailed description for conducting the interviews.
- *Research Instruments:* Instruments used to collect data, e.g. interview guides.
- *Data Analysis Guidelines:* Procedures for data analysis.
- *Miscellaneous:* Additional information that does not fit well within the above sections, e.g. validity threats and validation procedures.

In [51] it is mentioned that having a protocol will make the research concrete early on, while providing a good way of keeping track of information that can be useful during reporting of the research. Finally, as the data collection and analysis is an iterative process, the protocol will serve as a good source of how the final conclusions are drawn. The protocol is kept under version control. Therefore, older versions of it are available along with a brief change history that describes the main changes between versions. By doing that, it is possible to see at a glance how the data collection evolved and changed throughout the research.

## 4.3 Data Analysis Procedures

When it comes to analyzing qualitative data, it is important to keep a clear chain of evidence from the first data collection to the final conclusion [51]. The protocol plays a central role in this aspect.

As the research is of exploratory nature, a hypotheses generation method was suitable. That is, to find hypotheses based on the data to explain the phenomena under study [51, 52]. The qualitative analysis can then be divided into three steps, influenced by [51]:

- Data is coded and categorized.
- The data is combined into comments or notes which form the findings. For reporting of the findings, quotes from the interviewees are provided in order to keep a clear chain of evidence [51].
- A set of hypotheses is generated.

As has been noted, the data collection and analysis is an iterative process. Therefore, more data needs to be collected once a first set of hypotheses has been generated. Then the analysis is carried out again, forming a new or updated set of hypotheses. This is done as often as needed [51, 52].

When it comes to coding and categorizing the data, an editing approach is followed. The approach advocates that the researchers should have a handful of preliminary codes. Those codes are then revised and augmented during the analysis [51, 52]. This was the case for the semi-structured interviews as they were audio recorded and transcribed. The workshops were not audio recorded and therefore the analysis was not as thorough. Nevertheless, written notes were coded where possible.

The output of the analysis can be seen in section 5.1.

## 4.4 Validation Procedures

In Iteration IV and V of the design cycle, the process model was validated in collaboration with the companies.

### Iteration IV

In Iteration IV, the validation was carried out in a Software Center project workshop as well as with validation sessions at the companies. Both of the validations are categorized as expert opinion. Furthermore, the usage of the prototype was demonstrated with example data in the validation sessions at the companies which acts as a simulation.

#### Validation in a Software Center project workshop

When the development of the prototype was in progress, the process model and the half-finished prototype were presented to participants in the third workshop in the Software Center project which took place in April. Two of the three researchers from the Software Center project participated, along with nine other participants: two from Company A-I, four from Company B, two from Company X and one from Company Y. The presentation was 10 minutes, where the process model was described, the prototype shown and unimplemented features of the prototype were discussed. A 10 minute discussion of the model and the prototype followed. Verbal feedback from the participants was audio recorded.

#### Validation sessions at the companies

The process model was validated with help from the prototype in two different companies. A total of 3 sessions were held at the companies, with a total of 4 employees participating (validation session #3 had two participants). Table 4 shows the three validation sessions, at which company it was held and the role of the participant(s). All the sessions lasted for one hour.

In the workshop, 1) the findings were presented, 2) the process model was shown, 3) the prototype was described and what aspects of the model it covers, 4) the researchers showed how the prototype works with example data (simulation), 5) the participants used the prototype with a small feature and 6) the participants filled out a questionnaire about the process model.

The output of the validation sessions was threefold:

- Recorded audio of verbal feedback from the participants.

**Table 4:** Three validation sessions were held at the companies in Iteration IV

| Validation Session | Company | Role of participant(s) |
|---|---|---|
| #1 | A-I | Developer |
| #2 | A-I | Area Product Owner |
| #3 | B | System Architect<br>Product Owner |

- A questionnaire that each participant filled out at the end of the session. The questionnaire had effect questions, trade-off questions, sensitivity questions and requirements satisfaction questions [47], as well as other general questions and was reviewed by one researcher in the Software Center project. The questionnaire that was used in the fifth and last iteration, which is similar to the one used in this iteration, can be seen in Appendix B.
- Observations made by the researchers.

The results of these validations are presented only shortly in chapter 6 as this validation was not carried out on the final version of the process model.

## Iteration V

In Iteration V, the final version of the process model was validated in validation sessions at the companies. Thus the validation is categorized as expert opinion. In addition, a video was shown of the usage of the prototype which acts as simulation.

### Validation sessions at the companies

The process model was validated with help from the prototype at the four companies. In this validation, eight sessions were held with a total of ten employees participating (validation sessions #3 and #4 had two participants each). Table 5 shows the eight validation sessions, at which company it was held and the role of the participant(s). All the sessions lasted for one hour.

**Table 5:** Eight validation sessions were held at the companies in Iteration V

| Validation Session | Company | Role of participant(s) |
|---|---|---|
| #1 | A-I | Developer |
| #2 | B | System Architect |
| #3 | Y | Manager<br>System Administrator |
| #4 | A-II | Program Manager<br>Technical Coordinator/Developer |
| #5 | A-I | Area Product Owner |
| #6 | X | Innovation and Partner Manager |
| #7 | A-I | Line Manager |
| #8 | A-I | System Manager |

In the workshop, 1) the background of the model was presented, 2) a running example was introduced (the same as is used in describing the process model in section 5.2), 3) the process model was shown, 4) a video was shown of how the prototype works with the running example (simulation), 5) the participants used the prototype with a small feature and 6) the participants filled out a questionnaire about the process model.

The output of the validation sessions is the same as before: 1) recorded audio of verbal feedback from the participants, 2) a questionnaire (Appendix B) and 3) observations made by the researchers.

The results of these validations are presented in chapter 6.

## 4.5 Guidelines

In addition to the design cycle, seven guidelines from [50] are followed throughout the research. The guidelines, along with how they are followed, are:

- *Guideline 1 - Design as an artifact:* The result from a design science research needs to be an innovative artifact that has a clear aim or purpose. However, the artifact is seldom complete.
  - *How is it fulfilled?* The aim of the process model is made explicit through the research objective, which is then refined into two research questions. The model has a clear purpose of helping decision-makers to model, predict, track and analyze feature value. The second artifact, the prototype, has the purpose of facilitating the use of the model in practice and aid in the validation.

    The process model is innovative as it is addressing a problem area that is unsolved to a large extent.
- *Guideline 2 - Problem relevance:* The problem that the artifact is to solve needs to be relevant and of importance.
  - *How is it fulfilled?* The first step in the design cycle is problem investigation. The researchers in the Software Center project are a driving force in the problem area, as can be seen in chapter 2 and chapter 3. Their work is done in close collaboration with industry, thus ensuring that they are addressing a real-world problem. In order for the researchers of this study to better understand the problem at hand and make sure that the problem slice that is addressed is of interest to the stakeholders, interviews were conducted. In addition, the researchers took part in workshops in the Software Center project. The findings from the interviews and workshops (section 5.1) support that. Finally, working in close collaboration with the researchers in the Software Center project kept this study focused on the core problem.

- *Guideline 3 - Design evaluation:* Evaluation methods are diverse and focus on different aspects. No matter the method, the evaluation must be systematic, well thought through and show that the artifact is useful.
  - *How is it fulfilled?* The third step in the design cycle is treatment validation where the artifacts are validated. In Iteration I-III, the process model was validated by the researchers in the Software Center project. In Iteration IV, the model was validated with the companies through a presentation in a Software Center project workshop and in three validation sessions with four participants. In Iteration V, the model was validated in eight validation sessions at the companies with a total of ten participants.

    The validation will be discussed in chapter 6.
- *Guideline 4 - Research contributions:* The contribution is often in the form of the artifact itself. Nevertheless, it must be clear what the contribution is.
  - *How is it fulfilled?* The contribution is in the form of a process model, while the prototype is about realizing the model and helping with the validation of it. The contribution is made explicit through solving a real problem as is discussed in Guideline 2.
- *Guideline 5 - Research rigor:* The research itself must be done in a proper way.
  - *How is it fulfilled?* Several steps are taken to make the research rigorous:
    * Following the design cycle.
    * Following the guidelines.
    * Documenting iterations and steps taken in the design cycle.
    * Keeping a detailed protocol for collecting the data, analyzing it and validating.
- *Guideline 6 - Design as a search process:* As design science research is iterative in nature, it can be viewed as a search process for discovering a useful solution.
  - *How is it fulfilled?* Going through five iterations of the design cycle ensures that early knowledge influences later decisions. Furthermore, the thesis work consisted of two week sprints which started with a meeting with the three researchers in the Software Center project. On these meetings the work of the previous sprint was reviewed, current problems were discussed and future activities were planned. These meetings made the thesis work even more iterative.
- *Guideline 7 - Communication of research:* The research must be communicated to different audiences, e.g. those who are on the technical side and those on the management side, in different ways.

– *How is it fulfilled?* The outcome of the research is a master thesis. In Iteration IV the findings, the process model and the prototype were presented in validation sessions to four employees at companies that had previously been interviewed. In Iteration V, the process model and the prototype were presented to ten employees at the four companies. Furthermore, if asked for by interested parties, the research can be communicated in other ways.

It is noted [50] that all of the guidelines should be addressed, in one way or another, so it can be justified that the research was carried out in a proper manner. Though they need not be followed to the letter, hence the term 'guideline'.

## 4.6 Alternatives to Design Science Research

#### Action Research

Action research is a research methodology where the researcher is considered to be an active participant in solving the problem at hand [55, 56]. Design science research and action research share some similarities as has been noted in [57]. The aim of design science research is the design of an artifact and the investigation of it in context [47]. This is done to make sure that the artifact is of value. However, in action research the researcher is focused more on solving a problem in the particular organizational context that it originates from [55]. Although the companies provide useful information for this study in the problem investigation phase of the design cycle, the goal of the study is not to solve a problem within a particular organizational context which makes action research a worse fit.

#### Case Study

A case study is a research method that can be used when objects of the study cannot be isolated and controlled [51]. Which is the case when it comes to predicting and analyzing feature value, as the features and their value co-exist in a bigger context. That makes it difficult to study the value of the features individually without other, unknown, aspects interfering. A case study does not discuss the use of an artifact, e.g. a model or a prototype, to achieve a goal as design science research does [47]. Nevertheless, a case study is suitable to investigate a phenomena and even to validate a solution [51] but a case study does not explicitly cover the phase between the problem and the validation; i.e. creating the actual solution. This is an essential part of this study and thus design science research is better suited than a case study.

# 5

# Results

In this chapter, the results of the study are presented. In section 5.1, the findings from the interviews and workshops will be discussed. Section 5.2 presents the process model. Finally, section 5.3 describes the prototype.

## 5.1 Findings

Interviews and workshops are an essential part of the first step, problem investigation, in the design cycle. Five iterations were made through the design cycle but only three iterations of data collection and analysis were carried out.

In this section, the findings from each of the three iterations will be presented along with quotes from the interviews to keep a clear chain of evidence. A set of hypotheses are then generated from the findings. In subsection 5.1.1, the findings from Iteration I along with generated hypotheses are presented. Subsection 5.1.2 covers Iteration II in the same way but previous findings will also be revisited, and potentially refined, with the data gathered from Iteration II. This is done as the data collection and analysis is an iterative process. Finally, subsection 5.1.3, covers Iteration III and gives a final set of generated hypotheses.

Table 6 shows all of the findings and generated hypotheses from the findings, divided down into the iterations where the findings were realized. As has been pointed out, previous findings were revisited in Iterations II and III. However, only the hypothesis for Finding 3 was adjusted in subsequent iterations. The table reflects the final version of the findings and the hypotheses.

### 5.1.1 Iteration I

The first iteration consisted of:

- The first workshop in the Software Center project. Participants were from Company A-I and Company B.

**Table 6:** The findings and generated hypotheses from the findings, divided down into the iterations where the findings were realized.

| Iteration | Finding | Hypothesis |
|---|---|---|
| I | #1<br>Low usage of collected data | The large amount of data that companies collect is not being used in a systematic way to predict and track feature value. |
| | #2<br>Value categories | A feature is categorized based on its type of value to customers. |
| | #3<br>Mainly feature value modeling and prediction | Companies mainly do feature value modeling and prediction. Some realization tracking and comparison is done, but it is in many cases done ad-hoc and is not systematic. |
| II | #4<br>Value is different between customers | It depends on the customer, what the value of a feature is. |
| III | #5<br>A need for a simple process to spread the value thinking | A simple process to spread the value thinking within the companies is needed. |
| | #6<br>An automatic process | An automated process to improve the prediction and tracking of feature value is needed. |

- An interview with an area product owner at Company A-I.

From the workshop it was evident that the companies have a large amount of data, mostly quantifiable. This data is only used in a limited way but not in a systematic way for predicting and tracking feature value as an area product owner at Company A-I notes: *"[T]here are GBs of KPIs and we have those but we are not using them. The only time we are using them today is, more or less, when we do troubleshooting."* The main reason for not making more use of the data is the amount of it: *"[I]t's not easy to just extract it and make sense of it. It's really complex and expensive as well."* (Area product owner, Company A-I). However, some experiments are being carried out on a feature level in order to prove the value of a single feature. By doing this, a small process could be built that would allow for generalization later on instead of starting with the entire collection of data. This gives rise to the first finding:

**Finding 1:** Low usage of collected data.

In order to make the value of a feature more concrete and easier to market, the features are categorized. This categorization is in the form of value areas or value packages: *"We have pre-defined [value] packages that we are charging customers for. It's like a cable TV subscription, you buy a base package, a sport package, these kind of things. Here you can buy different things ... for instance, high availability and redundancy."* (Area product owner, Company A-I). Therefore, the second finding is:

**Finding 2:** Value categories.

Before a value of a feature can be predicted, the value must be modeled, i.e. define what the value is to a stakeholder. The value categories play a key role in modeling the value that a feature has. It was noted that the modeling and the prediction of the value is resource demanding, e.g. as it is expensive to analyze the collected data. This might hinder the progress of feature value modeling and prediction within that specific company.

Once a value has been realized it is rarely compared to the predicted value, as is noted by an area product owner at Company A-I: *"The team does this feature analysis and every third week we have a sprint review where ... the team demonstrates the feature. Here, it would be great if the team could demonstrate the value in a measurable term, but we are far from there."* Despite that, the area product owner is happy that they are discussing value as that is something that they did not do until recently. The third and final finding for Iteration I is:

**Finding 3:** Mainly feature value modeling and prediction.

**Hypotheses**

Based on the aforementioned findings the following hypotheses were generated:

- *Hypothesis 1:* The large amount of data that companies collect is not being used in a systematic way to predict and track feature value.
- *Hypothesis 2:* A feature is categorized based on its type of value to customers.
- *Hypothesis 3:* Companies begin with feature value modeling and prediction, without tracking the realization of the feature value later on or comparing it to the predicted one.

### 5.1.2 Iteration II

The second iteration consisted of:

- The second workshop in the Software Center project. Participants were from Company A-I, Company B, Company X and Company Y.
- Two interviews:
  - Product manager at Company A-I.
  - System architect at Company B.

The companies have many customers and these customers have different goals. Therefore, a feature that is valuable for one customer is not necessarily valuable for another customer: *"That means that a feature we add can add very different value to different customers or no value at all."* (Area product owner, Company A-I).

However, by using the value categories the companies can generalize the value of a feature but only to a certain degree: *"They differ [the KPIs]. We have actually done some work together with the customers, looking at common KPIs, so they can get a set of KPIs from us if they like but they will probably have a bit different."* (System architect, Company B). In hindsight, the aspect of different value between different customers was even brought up during the first iteration. Thus, the fourth finding is:

| |
|---|
| **Finding 4:** Value is different between customers. |

This was the only new finding from Iteration II. As the data collection and analysis is an iterative process, it is necessary to look at the findings from Iteration I, with the data collected in Iteration II. Finding 1, *Low usage of collected data*, is supported in Iteration II as it was a recurrent theme throughout the iteration, e.g. a system architect at Company B mentions the following: *"[W]e have quite a lot of data but it would be interesting to start to structure it, so you could query this database."*

The aspect of different value categories, Finding 2, was not as evident in Iteration II as in the first one. However, it was noted that value can be different between customers. In addition, the companies have at most tens of KPIs or factors that are derived from hundreds of smaller factors. This resonates well with the categories discussed in Iteration I as it is easier to market a feature if it is aimed at a specific aspect.

Finding 3, *Mainly feature value modeling and prediction*, is still clear from Iteration II. However, more feature value realization tracking and comparison between the realized value and the predicted one is being done than was evident from Iteration I. This is noted by a system architect at Company B: *"When we launch the feature to the market, etc., at some point we do that [compare the realized value to the predicted one] but we are not learning from how good our predictions are."* Nevertheless, the realization and the comparison is in many cases done ad-hoc and is not as structured as the feature value modeling and the prediction

**Updated Set of Hypotheses**

Based on the first two iterations, the following is an updated list of generated hypotheses:

- *Hypothesis 1:* The large amount of data that companies collect is not being used in a systematic way to predict and track feature value.
- *Hypothesis 2:* A feature is categorized based on its type of value to customers.
- *Hypothesis 3:* Companies mainly do feature value modeling and prediction. Some realization tracking and comparison is done, but it is in many cases done ad-hoc and is not systematic.
- *Hypothesis 4:* It depends on the customer, what the value of a feature is.

### 5.1.3 Iteration III

The third iteration consisted of three interviews:

1. Product owner at Company B.
2. Developer at Company A-I.
3. Two interviewees: program manager and technical coordinator/developer at Company A-II.

The value thinking is not widespread within the companies. It depends on who you talk to and their role in the company whether the aspect of feature value is clear. It was mentioned, even in earlier iterations, that it was necessary to have a simple process to spread the value thinking. A technical coordinator/developer at Company A-II mentioned that people need to see the logic in the model and need to be working with it as *"[i]t can't be just a few people that try to drive it because every engineer that works on that project must have that mindset."*

This is especially important in order to gather support from management for this new way of working as is noted by an area product owner at Company A-I: *"One [of the challenges in value modeling and prediction] is to have an interest in the organization because I'm interested in it, but it's impossible for only one person ... so there needs to be an interest in the company."* Thus, the fifth finding is:

| |
|---|
| **Finding 5:** A need for a simple process to spread the value thinking. |

During the third iteration it became clear that an automatic process, or as automatic as possible, was needed to improve the prediction and tracking of feature value. This was mentioned in earlier iterations as well. The companies have several manual data analysis tools that are being used to some extent. However, it can be time-consuming and expensive to manually enter measurements and other necessary data as is mentioned by an area product owner at Company A-I: *"[Y]ou need to be able to get the value measurements cheap, that is really the key, you can't have hard manual labor for all features as soon as you want a value proof. It needs to be cheap."* The sixth and final finding is:

| |
|---|
| **Finding 6:** An automatic process. |

When looking at the data from Iteration III with respect to Finding 1, *Low usage of collected data*, it was still evident that the collected data is not used in a systematic way at the companies. Company A-II being an exception that stands out in this aspect: *"I think one of our problems are ... that we have a lot of data, but it feels that we need to expand, that is looking on other data."* (Technical coordinator/developer at Company A-II).

Finding 2, *Value categories*, was only brought up in one of the three interviews. It supports previous findings that the companies do focus on certain aspects when it comes to marketing towards customers as a technical coordinator/developer at Company A-II notes: *"Right now, we have a lot of different KPIs defined and we keep track of them, and customers are interested in it."*

For Finding 3, *Mainly feature value modeling and prediction*, it was clear that Company A-II has a more advanced process of predicting and tracking feature value. In addition to feature value modeling and prediction, both comparisons between the realized value and the predicted one was done as well as tracking of the feature value: *"[I]f we discover that the gain is different from the prediction, then it would be noted down what the new gain is. So, to keep track of, we have the constant thinking of what the feature will gain. ... Product line does [a comparison between the final realized value and the predicted values]. They need to set a pricing on the feature, so they will definitely look at how well it performed in the end. And, there is also some learning, if you didn't gain anything, or very little, then that's fed back and it's a basis for the coming features."* (Program manager, Company A-II). For the other two interviews, the situation was described similarly to before. However, it was evident that it greatly depends on the role of the employee to what extent they come in contact with feature value modeling and prediction and that the value mind set is not well spread within the companies.

Iteration III reinforced Finding 4, *Value is different between customers*. However, it was mainly discussed within the aspect of different levels of complexity or maturity as a product owner at Company B mentions: *"So we have customers that really need something that is simple and that is robust, that always works. And for some other customers, maybe the performance is the key thing."*

**Final Set of Hypotheses**

Based on the three iterations, the following is a final list of generated hypotheses:

- *Hypothesis 1:* The large amount of data that companies collect is, in most cases, not being used in a systematic way to predict and track feature value.
- *Hypothesis 2:* A feature is categorized based on its type of value to customers.
- *Hypothesis 3:* Companies mainly do feature value modeling and prediction. Some realization tracking and comparison is done, but it is in many cases done ad-hoc and is not systematic.
- *Hypothesis 4:* It depends on the customer, what the value of a feature is.
- *Hypothesis 5:* A simple process to spread the value thinking within the companies is needed.
- *Hypothesis 6:* An automated process to improve the prediction and tracking of feature value is needed.

## 5.2 The DVOCE Process Model

This section presents the main result of the study: the Data-Driven and Value-Oriented Continuous Experiment (DVOCE) process model. An overview of DVOCE can be seen in Fig. 4 and all eight steps are described in detail within this section. As can be seen from Fig. 4, the first three steps: *Select*, *Model* and *Predict* are part of pre-development during the first iteration. Subsequent iterations do not include the first step, *Select*, but the second and third steps, *Model* and *Predict*, are then part of the development phase.

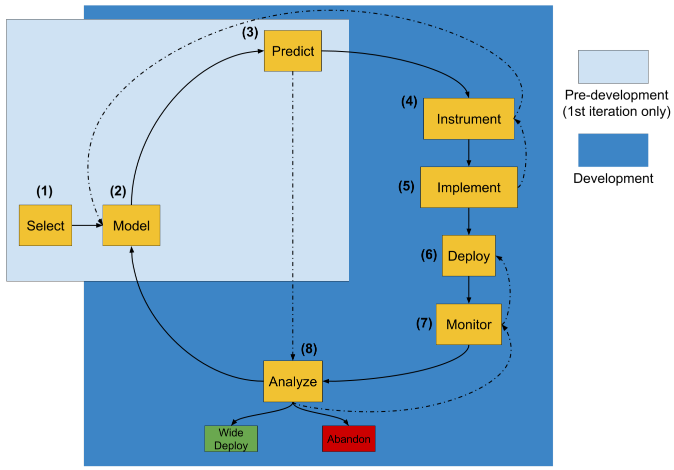


**Fig. 4:** An overview of the DVOCE process model.

The DVOCE process model is a detailed and extended version of the HYPEX model. It is based on the outcome of the activities in the first step of the design cycle, *Problem investigation*, where interviews and workshops played a central role. The interviews and workshops yielded six findings, as can be seen in Table 6. The process model is based on four out of the six findings:

- *Finding 1: Low usage of collected data.*
- *Finding 3: Mainly feature value modeling and prediction.*
- *Finding 5: A need for a simple process to spread the value thinking.*
- *Finding 6: An automatic process.*

The other two findings, *Finding 2: Value categories* and *Finding 4: Value is different between customers*, were deemed outside of the scope of the DVOCE process model. However, all of the findings will be discussed in section 7.1 with regards to previous findings in the literature.

The process model is presented in this section using a running example focusing on the fictive company Distributed Car Networks (DCN) in order to exemplify how DVOCE can be utilized and make the process model easier to follow. Text related to this running example will be completely in italics. The following paragraph introduces the fictive situation.

*DCN has developed a system for peer-to-peer connections between cars and this system is used as an extra level of safety through cars communicating their intentions in advance. Because of the safety aspect involved, reliability is extra important and DCN has noticed that reconnecting broken connections is a problem area during city traffic. A new feature is considered which would theoretically reduce both the number of reconnects required and the average reconnection time. DCN has good knowledge about the average number of reconnections but* *not much is known about the average reconnection time during city traffic. DCN uses DVOCE during the development to track the value created and decide on when to deploy the feature.*

It should be noted that when using the process model, it is assumed that the organization has a system with existing customers that are capable of being a part of continuous experiments. Any situation where this is not the case is outside of the scope of this study.

### Step 1: Select

The selection of the feature is not a focus of this model but HYPEX states that the feature with the highest priority should be selected and that there are many criteria which can affect this priority [12]. To follow the value-based logic would be to select the feature that is believed to bring the highest return on investment (ROI) as that is the feature which will bring the most value to the company.

*In the example, DCN noticed a problem with their reconnect algorithm when cars were driven in city traffic. As the reliability of their product is considered a key aspect, this algorithm problem is causing considerable negative value. Therefore, the development of an improved version of the algorithm is selected as the next feature.*

### Step 2: Model

The *Model* step identifies and details the feature's factors and how they relate to measuring points of the system. A factor is essentially a measuring point of the system that can bring either positive or negative value. This step also defines how the factors connect to the strategic business goals of the organization. The *Model* step as whole is further divided into three substeps:

1. Identify the expected functionality and affected factors.

2. Identify the starting state of the selected factors.

3. Define the value constants.

The first substep is to identify the expected behavior of the feature and affected factors. The expected functionality of the feature is defined before using the Goal/Question/Metric method to find the affected factors. The affected factors are measuring points that are relevant for this feature. Relevant in this context often means that they are expected to contribute positive or negative value, but can also mean that it is a critical factor that the organization wants to keep under observation. The GQM method is a simple method grounded in identifying the goal of the feature and tying this goal to a metric based on a measuring point [29]. In this process model, this measuring point is called a factor as stated earlier. The GQM process is carried out by answering the following three questions:

- *Goal*: What are the areas that bring value to the feature?
- *Question*: What questions need to be asked to see if the goal has been fulfilled?
- *Metric*: How can we answer the questions in a quantitative way?

There is a need for including this simple process for defining the factors and the goal-factor mapping. All the companies collect a large amount of data (as described by Finding 1) about their systems which shows that their systems have many probes that can potentially be used. However, for Company A-I and Company B, this data is not being used in a systematic way to predict and track feature value and doing so is described as difficult because of the vast amount of data. It is thus shown that there is a gap between collecting the data and using the data for understanding the value that a certain feature brings to the organization. By starting with the goals of the organization and defining the relevant measuring points from those goals, they can increase the possibility that the resources spent on collecting and analyzing data is not wasted because of the data being irrelevant [28]. It also reduces the task of interacting with the collected data they have available from *finding what is useful* to *seeing if this useful data is already there* which is more manageable. In addition, most measurement planning models and tools adhere to goal based approaches [26] which makes the case for a goal based approach stronger. Many goal based approaches are also derived from GQM [26] which indicates that it is an accepted and proven method. Finally, Finding 5 describes the need for an overall simple process which further strengthens the support for the GQM method.

*Table 7 shows how DCN has determined their factors. They had two goals that they were concerned about: the reliability of the connection and the user satisfaction. For the goals they then created the questions shown in the second column of the table and the metrics to answer these questions which can be seen in the third column. The relevant measuring points are noted down as factors in the fourth column.*

In the second substep of *Model*, it is determined if the starting states of the factors are known or unknown to the organization. To determine the value of any changes to the factors, it is necessary to know the starting state for them. Without a starting state, both determining a change and the impact of it becomes more difficult. However, in the *Model* step the starting state can be unknown as it will be amended later on in the *Instrumentation* step.

*As can be seen in Table 8, DCN knows the average number of required reconnects per hour of city traffic but the average reconnection time during city traffic has not been tested previously so it is unknown. The perceived user safety is an important factor they keep track of for every feature and also has a known starting state.*

The third substep of *Model* is to define the value constants. The idea of the value constant is to capture the value created or lost in a simple and intuitive way. A value constant is just how much the value of the feature is changed when the metric tied to the factor is modified by a certain amount. This amount is

**Table 7:** DCN uses the GQM method to define appropriate factors.

| Goal | Question | Metric | Factor |
|---|---|---|---|
| Reliable Connection | How often is there a need to reconnect for an average car in city traffic? | Average number of reconnects per hour = total number of reconnects / hours in city traffic | Average number of reconnects per hour |
| | How long is the average reconnection time for an average car in city traffic? | Avg reconnection time [ms] = total reconnection time / number of reconnects | Average reconnection time |
| User Satisfaction | Do the users perceive that DCN software makes them safe? | Perceived safety [-5 to 5] = quantified (coded) prototype testing interviews | Perceived user safety |

**Table 8:** The starting state for the factors that DCN has identified.

| Factor | Starting State |
|---|---|
| Average number of reconnects per hour | 10 times per hour in city traffic |
| Average reconnection time | Unknown |
| Perceived user safety | 3 |

the defined base unit for the factor that decides the granularity of the scale the factor is being measured by. In this process model, that amount is called the *unit of change* for that factor. Although not stated as a finding, it can be challenging for the companies to know what the actual value is. However, they might have an indication that something could be improved. Therefore, the unit of change can be either absolute if the starting state is known or relative if it is unknown.

The value constant is determined by the user of the process model based on relevant data if it is available or their domain expertise if it is not. A value constant is relative to value constants of other factors in the feature model, i.e. it is on a ratio scale, which can make it difficult to accurately determine if it cannot be directly tied to a concrete value. The difficulty of tying the value to a concrete value should not be underestimated. A product manager a Company A-I stated that they have tried to measure value in US dollars but that it was not that easy as the functionality of the feature is just one part of a larger area. This makes it hard to say that this specific functionality saves this much money or increases revenue by this much. In practice, the value constant for a new factor $F_{new}$ is thus mainly going to be determined by looking at the value constants for previously defined factors $F_{prev}$ and relating how valuable a change in $F_{new}$ is with the value of a change in one or more of $F_{prev}$.

While the value constant may be inaccurate in the beginning it has the opportunity to be tuned throughout development as users of the process model develop a better understanding of the feature and its context. In addition, as the process model is being used for more and more features within a system the experience and data acquired will aid the accuracy of the value constants of all features.

The relationships between feature, factors, value constants and data can be modeled as a mathematical sum. In the sum, the value of the factors are terms with the value constants being coefficients ($VC_m$) to the change in the variable data ($X_m$). With $m$ factors, this sum becomes the *total feature value sum* seen below.

$$X_1VC_1 + X_2VC_2 + ... + X_mVC_m = Total\ Feature\ Value\ (Equation\ 1)$$

*In the example, DCN creates the unit of change and value constants for all their factors which can be seen in Table 9. They first decide that the average number of reconnects per hour is going to be integer based with a unit of change of -1. It is a negative number as the value increases when the number of reconnects is reduced. They do not have a real value to concretely tie it to at this point and, since it is the first factor they are defining, they just decide to set the value per unit of change to 300. They then continue with defining the average reconnection time factor. This factor is given a unit of change of -1 %. When deciding what value to assign to it, they compare it with the value they gave the first factor. Reducing the number of reconnects required by 1 is deemed more valuable than reducing the average reconnection time by 1 %. At the same time DCN knows that there will always be reconnects due to the nature of moving vehicles and* *reducing the average reconnection time reduces the impact of all reconnects. In the end, they settle for 3 % time reduction being equal value to 1 reconnect. The final factor, perceived user safety, is not expected to change but if it does it is a great increase in value as it increases the desirability of the product so DCN decides it is worth four times the reduction of a reconnect per hour.*

**Table 9:** DCN's factors along with their values, including the predicted value. UoC in the table stands for the unit of change.

| Factor | UoC | Value/UoC |
|---|---|---|
| Average number of reconnects per hour | -1 reconnects / hour | 300 |
| Average reconnection time | -1 % (relative) | 100 |
| Perceived user safety | 1 | 1200 |

### Step 3: Predict

In the *Predict* step, estimations of the change to the factors are made. These estimations of change are then made into predictions of value using the information added in the *Model* step. Finally, a *tolerated gap threshold* is decided upon. Thus, there are three substeps to *Predict*:

1. Estimate the change the feature will put on the factors.
2. Use the information from the *Model* step to predict the value.
3. Decide on a tolerated gap threshold.

The first substep of *Predict* is to estimate the change the selected feature will put on the factors identified in the *Model* step. These estimations must adhere to the scale defined by the factor's unit of change. While ideally the estimations should be data-driven and based on already collected data, the interviews showed that they will often be made based on the expertise and experience of the user and their colleagues within the organization. While the companies potentially have the data, they do not have the capability currently to analyze that data as a whole and use it for data-driven estimations. At least not cheaply enough to do them regularly. This is clear from Finding 3. The estimations also have the opportunity to be adjusted as development progresses and the *Predict* step is revisited.

When the change to the factors is estimated, a prediction is created using the information determined in the *Model* step. This is the second substep of

*Predict*. The estimated change is multiplied by the value constant and summed together as shown in Equation 1.

*Table 10 shows the factors and their related information that DCN has at the end of the Predict step. The first factor of average number of reconnects per hour had a starting state of 10 and DCN estimates that they will be able to lower it by 2, giving an estimated final state of 8. Since the unit of change is -1 and the value constant is set to 300, the predicted value for the first factor is 600. For the second factor, average reconnection time, the starting state was unknown. This means that the final state cannot be estimated at this point but DCN believes that they can lower it by a relative percentage of 15 % which would mean a positive value of 1500. The perceived user safety is not expected to change at all, but it is such an important factor that DCN wants to keep a close eye on it. The predicted value for the factors are summed to acquire the total predicted value which in DCN's case is 2100.*

**Table 10:** DCN's factors along with their values, including the predicted value.

| Factor | Starting state | Estimated final state | Change | UoC | Value/UOC | Predicted value |
|---|---|---|---|---|---|---|
| Average number of reconnects per hour | 10 | 8 | -2 | -1 | 300 | 600 |
| Average reconnection time | ??? | ??? | -15 % (relative) | -1 % | 100 | 1500 |
| Perceived user safety | 3 | 3 | 0 | 1 | 1200 | 0 |

In the third substep the tolerated gap threshold is determined. This threshold describes in a concrete way when the gap between the realized value and the predicted is deemed what HYPEX calls "sufficiently small" [12]. When this threshold is passed the feature is considered ready for commercial deployment. In DVOCE, the threshold can either be set as a percentage of the predicted value or as an absolute value not directly related to the prediction. The choice depends on the current context. If the experience of working in a value-based way is low then choosing a percentage of the predicted value is the easier approach. If the value constants are changing often due to inexperience, an absolute threshold that is not connected to the prediction could lead to a premature deployment. On the other hand, an experienced organization could set an absolute value that is the same for all features in order to promote the idea of small features and iterative development.

*DCN had a total predicted value of 2100. They do not have a lot of experience in setting value constants yet so they decide to go for the safer approach of setting a tolerated gap threshold as a percentage of the predicted value. They deem that they will be happy if they are within 10 % of the predicted value, giving a tolerated gap threshold of 1890.*

### Step 4: Instrument

After the *Predict* step has been completed, the fourth step of this process model is *Instrument*. In HYPEX, the instrumentation of the system is briefly mentioned together with the implementation of a Minimal Viable Feature [12]. In this process model, the instrumentation and implementation of an MVF is split into two distinct parts with a clear order as can be seen in Fig. 5. The *Instrument* step where decisions about data collection are taken and the system is instrumented and the *Implement* step where the development of the feature occurs. The *Implement* follows the *Instrument* step but there is a loop back from *Implement* as a need for additional instrumentation might be discovered during the development in a single *Implement* step.

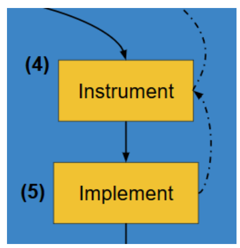


**Fig. 5:** In the DVOCE model, instrumentation and implementation are two distinct steps whose relation is illustrated in this figure.

In this step guidelines will be presented that aim to increase relevancy of gathered data and provide consistently useful instrumentation. The guidelines are based on pre-existing research and the findings presented in section 5.1. The step also includes a sub-process to aid in the selection of customer feedback and data collection techniques.

The guidelines are enumerated and then explained in detail below:

1. Technique selection guidelines:
    (a) If possible, use both qualitative and quantitative techniques for the metrics.
    (b) If possible, select a technique for each of the three phases of development.
    (c) Consider how often feedback or data needs to be collected.
2. System instrumentation guidelines, i.e. adding logs to the code:
    (a) Better to instrument at a system level, rather than on a feature level.
    (b) Quality over quantity.
    (c) Try to have as automatic as possible.

The guidelines can be divided into two categories where the first one is *technique selection guidelines*. There are many customer feedback and data collection techniques which can potentially be used. For this model, the literature review on customer feedback and data collection techniques in software R&D [20] is used as a reference. Thus, the model covers the currently recognized customer feedback and data collection techniques in the field.

Guideline 1.a) emphasizes the importance of combining qualitative and quantitative techniques which is recognized in [11] and [13]. Both discuss how qualitative techniques can be used in order to make sense of quantitative data and [13] suggests that quantitative techniques can be used in the later stages of development to validate already gathered qualitative data with a larger audience. With that in mind, it is recommended to use both qualitative and quantitative techniques to gather feedback and collect data.

Both qualitative and quantitative data can be gathered in all the three phases of development and guideline 1.b) is about the importance of selecting a technique for each of the three phases so data can be gathered throughout the development cycle. This is needed in order to enable data-driven decision-making regarding assignment of R&D resources throughout the entire development cycle. In pre-development, the collected data can be gathered before any significant contribution has been made by R&D. In development, the instrumentation process plays a key role in determining whether a feature is living up to its expectations or if other alternatives are better suited. In post-deployment, the gathered data can be helpful in maximizing the revenue of a particular feature. However, it is important to note that in pre-development only active feedback can be collected as an MVF has not yet been developed [5]. Therefore, instrumenting a system cannot be done until in the development phase. Although all three development phases are important, finding out sooner rather than later that a feature is not a feasible candidate saves R&D resources.

Guideline 1.c) recommends to consider how often feedback or data needs to be collected as the possible frequency depends on the technique. For techniques that use logging, data can be collected continuously which is not possible for the qualitative techniques. For example, a survey is sent out periodically or possibly only once. How often a qualitative technique is used depends on the available resources of a company.

Table 11 shows the different customer feedback and data collection techniques as depicted in the literature review on customer feedback and data collection techniques in software R&D [20]. They are divided according to the development phase they are most suitable for and whether they provide qualitative or quantitative data. It should be noted that all of the qualitative techniques are categorized as customer feedback techniques, as well as *Beta testing* and *Developers as customers*. All other quantitative techniques are considered data collection techniques.

As R&D is an experiment system, [3] emphasizes short iterations and continuous experimenting. This leads to some of the techniques depicted in Table 11 not being suitable. In [20], challenges and limitations for each of the techniques are presented. 6 out of the 18 techniques (*Interviews*, *Observations*,

**Table 11:** Customer feedback and data collection techniques in software R&D as presented in [20].

| | Pre-development | Development | Post-deployment |
|---|---|---|---|
| **Qualitative** | • BASES testing<br>• Interviews<br>• Observations<br>• Theater sessions<br>• Surveys<br>• Questionnaires | • Prototype testing | • Walkthroughs<br>• Customer pairing |
| **Quantitative** | • Crowd-funding success<br>• Online ads<br>• In-product surveys | • Operational data<br>• Beta testing<br>• Developers as customers | • Incident reports<br>• A/B testing<br>• Social network data |

*Surveys*, *Questionnaires*, *Walkthroughs* and *Developers as customers*) are considered time-consuming and, therefore, are not suitable for continuous experimentation. In addition, 3 out of the other 12 techniques (*Crowd-funding success*, *Operational data* and *Social network data*) can potentially give large amounts of data. Therefore, organizations must be able to make use of large amounts of data, which has been recognized as a challenge by Company A-I.

Fig. 6 presents a simple sub-process aiding in the selection of suitable customer feedback and data collection techniques which has taken these guidelines into consideration. After the factors have been derived using the Goal/Question/Metric approach the first question is if the factors are already being measured? If not, the relevant data has to be gathered. Table 11 is accompanied by two decision questions:

1. What type of data is desired? Qualitative or quantitative?
2. What is the current development phase?

Answering these two questions points to the relevant part of the table. Techniques which are time-consuming are then colored red in the table. Finally, some techniques are in italics which indicates data collection techniques. They can provide data continuously, while data from the other techniques can be gathered periodically or ad-hoc.

If the selected technique will make use of instrumentation then there are several things to consider which is the area of the second category of the guidelines. 2.a) is about keeping in mind that the instrumentation needs to be at the appropriate level, as is discussed in [32]. There it is presented that feature-level logging gives a large amount of information that is often hard to work with. Therefore, when possible the instrumentation should be done at a higher-level as doing so gives the opportunity to use it for more than only one feature. Guideline 2.b) comes from that the balance of logging the right amount and not too much or too little has been described as important [35]. By following the GQM paradigm [29], the specific instrumentation that is needed is clearer. This

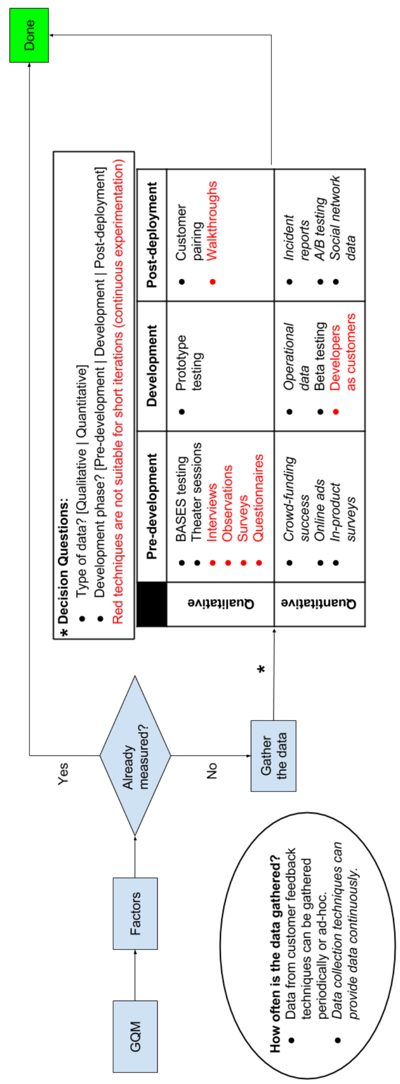


**Fig. 6:** The overall process of using GQM to identify factors and then collecting relevant data.

means that the gathering of data has a clear purpose which makes the question of what to log easier. 2.c) stems from that acquiring the measurements cheap is of vital importance according to an area product owner at Company A-I. An aspect that several participants in the Software Center workshops also agreed on. Here, an automatic logging tool would be beneficial [35]. Finding 6 also directly supports the need for a highly automated process.

Today, the companies mainly use their large amount of gathered data for troubleshooting (as described by Finding 1) but, as is recognized in research [31, 32, 34], the instrumentation and data collection can be used in a data-driven way; e.g. for predicting and measuring feature value which is what this model aims for. However, the finer details of instrumenting the system and carrying out the customer feedback or data collection techniques are outside the scope of this study.

**Table 12:** The customer feedback and data collection techniques that DCN used for their first two factors.

| | **Pre-development** | **Development** | **Post-deployment** |
|---|---|---|---|
| **Qualitative** | • BASES testing | • Prototype testing | • Walkthroughs |
| **Quantitative** | *No technique selected* | • Operational data | • Incident reports |

*As an example, DCN uses the presented sub-process (see Fig. 6) to select customer feedback and data collection techniques for their first two factors: average number of reconnects per hour and average reconnection time. They use the same techniques for both factors as they focus on similar aspects. For pre-development, they use BASES testing as a qualitative customer feedback technique because it can be good at determining if a new feature should be implemented [5]. Unfortunately, they could not select a quantitative technique for pre-development as the techniques available are deemed unsuitable with their embedded system in this situation. For development, they used prototype testing as a qualitative customer feedback technique and operational data as a quantitative data collection technique. Thus, they get both qualitative data on how the customers perceive the outcome and quantitative data on the actual usage [14]. For post-deployment, they have walkthroughs as a qualitative customer feedback technique and incident reports as a quantitative data collection technique. Walkthroughs can help DCN in being sure that everything works as intended and incident reports provide DCN with information about any possible problems that* *the customer might experience [14]. With their techniques selected, they are ready to start implementing the feature. The selected techniques can be seen in Table 12*

**Step 5: Implement**

In the *Implement* step of the first iteration, a minimal viable feature is implemented. An MVF contains the minimum amount of functionality needed in order to add value to the customer and be a viable candidate for deployment. This means that the most critical functionality has to be developed first and that development stops when around 10-20 % of the functionality is implemented [12]. In following iterations, the MVF is extended by the minimum viable amount in order to meet the new requirements specified during the *Model* step and thus be of additional value to the customer. When the development of an MVF is close to an end, it needs to be ensured that the existing instrumentation is sufficiently capable of gathering the desired data as defined during the *Model* step. If, at any point during the *Implement* step, it is discovered that this is not the case then the loop back to the *Instrumentation* step is used and the instrumentation is extended or modified until it is able to meet the expectations put on it.

**Step 6: Deploy**

When an MVF has been successfully implemented and the instrumentation is deemed adequate, the feature is deployed to a small set of key customers. There are a number of challenges involved in continuously deploying an MVF with every iteration [58]. In [58] customer reluctance towards new versions, customer confusion and lack of customer awareness with regards to the new changes are discussed. To handle this, the authors recommend *pilot customers*. Pilot customers are customers with high engagement that are willing and capable of testing out the new feature. It is mainly these customers that are of consideration during the *Deploy* step.

The deployment can potentially be done as a staggered deployment where the number of customers is increased in stages. This might be a good idea if a lot of data is needed as any severe bugs or issues that surface early will be contained to fewer customers. It is a simple case of reducing risk, as the deployment schedule can be aborted if early results are negative while still allowing for the collection of the needed amount of data in a controlled manner.

**Step 7: Monitor**

When the MVF has been deployed its value is monitored as new measurements are done and data is gathered. If a staggered deployment is chosen, a loop back to the *Deploy* step is done to extend the deployment when it is deemed suitable. Following the guideline described in [12], this step is finished when the collected data is considered to be statistically relevant. If any issues arise during the *Monitor* step that prevents further data collection, one continues to the *Analyze* step to analyze the cause and decide on how to proceed using the data available.

There are different ways of monitoring and there are aspects such as data complexity, the amount of data and data security that can hinder monitoring, but that is not the focus of this study.

*In DCN's case, they might need to do some processing within the embedded system instead of sending all the data back as bandwidth can be expensive. In addition, they might not want to monitor all aspects because of integrity reasons and potential liabilities.*

### Step 8: Analyze

In the *Analyze* step, the current state of the feature is analyzed utilizing the collected data from the *Monitor* step and decisions are made about the future of the feature. The main activity of the *Analyze* step is the gap analysis. But before performing the gap analysis, the collected data must be evaluated. After the gap analysis, a decision is taken regarding how to proceed next. Thus, the three substeps of *Analyze* are:

1. Evaluate the collected data.
2. Perform gap analysis.
3. Decide on how to proceed.

During substep 1), the evaluation of the data, value can either be expected or unexpected. Expected value is value that was included in the prediction while unexpected value was not included. Due to the low usage of collected data (as described by Finding 1), it seems reasonable to assume that additional factors that were not expected to contribute to the value of the feature might surface and either give a positive or negative value to the feature. The unexpected value is discussed in [14], stating the need to realize that early so R&D resources can be redirected to capture that value. However, if the unexpected value is negative there is also a need to consider and minimize the negative effects which is not discussed in [14].

In order to perform a gap analysis, there can only be expected values as the model, predicted value and tolerated gap threshold are likely to change if there are any unexpected values. If any unexpected values are identified, one should immediately proceed to the *Model* and *Predict* steps and incorporate the unexpected value into the prediction. After the unexpected value is incorporated, it is possible that no further implementation needs to be done and the data already acquired is sufficient. In that case, one would use the shortcut in the model to go directly from the *Predict* step to the *Analyze* step.

If the *Monitor* step had to be ended prematurely because of a bug or other issues then the issue needs to be analyzed before moving on to the *Model* step for a new and hopefully quick iteration. If the issue is very extensive in a negative way for future iterations then the feature can potentially be abandoned.

*After DCN implemented and monitored their new feature, they evaluated their collected data. The MVF improved their two main factors, average number of reconnects per hour and average reconnection time, but not to the extent of the prediction. In addition, they found unexpected positive value. It turns out that the feature caused reduced packet loss, which is a factor that had been previously instrumented in the system. Due to there being unexpected value, DCN cannot perform the gap analysis yet but must continue to the Model and Predict steps to incorporate this new value into the prediction. They add their new packet loss factor to their feature model and create a prediction for it. They also decide to keep their old tolerated gap threshold of 1890 as they figure this new value does not raise their requirements for a commercial deployment. Since the packet loss factor had already been instrumented and DCN decides that they have enough data from the last Monitor step, they use the shortcut to go directly from Predict to Analyze. In Fig. 7 the final result of the evaluation is shown, including DCN's new factor.*

| Factor | Starting State | Measured Final State | Change | UoC | Value/UOC | Realized / Predicted Value |
|---|---|---|---|---|---|---|
| Average number of reconnects per hour | 10 | 9 | -1 | -1 | 300 | 300/600 |
| Average reconnection time | 10 ms | 9 ms | -10 % | -1 % | 100 | 1000/1500 |
| Perceived user safety | 3 | 3 | 0 | 1 | 1200 | 0/0 |
| Packet loss | 0.4% | 0.1% | -0.3% | -0.1 % | 100 | 300/300 |

**Fig. 7:** DCN's final factors with relevant information after adding the packet loss factor.

The second substep is the gap analysis which is described as a critical practice in HYPEX [12]. In DVOCE, the gap analysis has been made concrete in that the realized total value is compared to the predicted total value of the feature. The gap itself is also specified by introducing the tolerated gap threshold to the analysis.

*When DCN completes their gap analysis, they notice that they have not exceeded the tolerated gap threshold – even with the extra value from the new packet loss factor. Since the 1600 value they realized is not enough, they now have the task of deciding how to proceed. The result of DCN's gap analysis is shown in Fig. 8.*

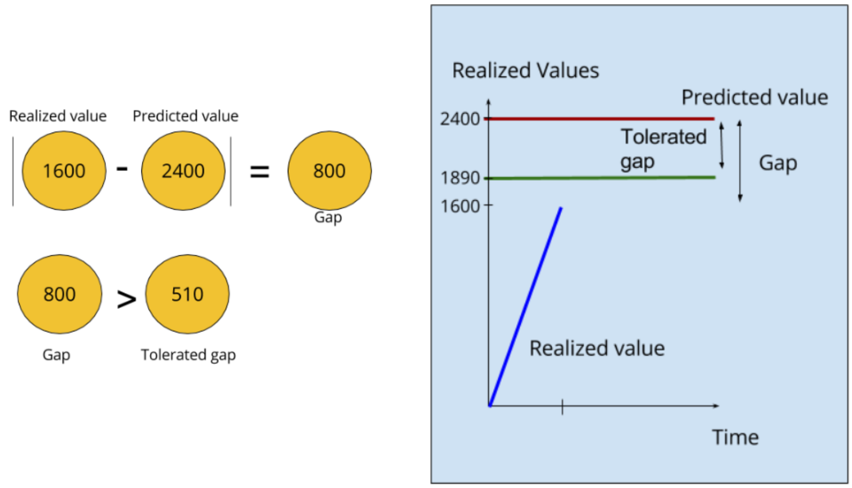


**Fig. 8:** DCN's gap analysis result.

In substep 3) there are four main options regarding how to continue after a gap analysis ended. 1) Go back to *Monitor* and gather more data. 2) Commercially deploy the feature. 3) Abandon the feature. 4) Start a new iteration. The first option should be selected if the data is deemed not conclusive enough. The second option is for if the gap is small enough, that is if the realized value exceeds the tolerated gap threshold. [12] describes that the feature is then ready for commercial deployment. The authors of [12] points out that there might be a certain amount of work involved in finalizing the feature, e.g. a clean-up of instrumentation which was useful during development but negatively impacts the customer experience. The final two options are relevant if the gap does not exceed the threshold. The third option should be selected if the progress is deemed not good enough and it is unlikely that it will be much better in the coming iteration(s). If none of the first three options is selected then a new development iteration will be performed.

In the case of a new iteration, a more thorough analysis should be performed with the goal of explaining the gap between the realized and the expected value [12]. Once such an analysis is complete, [12] describes that there are two possible situations in general. Either the developed MVF is not complete enough and needs to be extended. If so, a new iteration begins by continuing to the *Model* step. The other situation is that the developed MVF is not good enough and that an alternative version needs to be developed. Then the instance of the process should be forked into a new one where the alternative version is developed. Once the alternative version has reached the *Analysis* step again and another gap analysis has been carried out, the result is compared before entering substep 3) again and deciding on how to proceed with the version that had the highest amount of realized value.

*DCN decide that the data is conclusive and they did not exceed the threshold in the gap analysis, so the first two options of going back to Monitor or commercially deploy the feature cannot be selected. The team at DCN feels that they made good progress however and are confident that they will be able to improve the feature by extending it. Abandoning the feature is thus out of the question and they decide to do another iteration where they extend the MVF and make improvements. In their second full iteration, they manage to exceed the tolerated gap threshold by increasing the value of the feature to 1900. In the next Analyze step they take the decision to go ahead with a commercial deployment.*

## 5.3 The Prototype

The prototype is an example implementation of the DVOCE process model and is used as an aid when validating the model. Subsection 5.3.1 lists the requirements for the prototype as only part of the process model is realized. In subsection 5.3.2, the actual implementation of the prototype is discussed. An example usage of the prototype with screenshots can be seen in Appendix C.

### 5.3.1 Scope and Requirements

The purpose of the prototype was to help with the validation of DVOCE by providing increased clarity for the participants through usage visualization. Therefore, a user must be able to go through a full cycle of the model to understand how DVOCE can be realized. What follows is a list of requirements for each of the eight steps of the model.

1. *Select*
    - R1: The prototype shall allow the user to create, delete and select already created features.
2. *Model*
    - R2: The prototype shall allow the user to create one or more factors for the selected feature.
    - R3: Each factor shall have a name, a starting state, a unit of change, a value constant and a unit.
    - R4: The prototype shall allow the user to update all information about a factor.
3. *Predict*
    - R5: The prototype shall allow the user to add or update a predicted value for each of the factors.
    - R6: Once a predicted value has been added or updated for a factor, the prototype shall calculate a predicted value for the selected feature.

- R7: The prototype shall allow the user to specify or update a tolerated gap for the value of the feature. The tolerated gap shall be specified in either a percentage from the predicted value or as an absolute value.
- R8: Once a tolerated gap has been specified or updated, the prototype shall calculate the gap threshold for the selected feature.

4. *Instrument* [No requirements were specified]
5. *Implement* [No requirements were specified]
6. *Deploy* [No requirements were specified]
7. *Monitor* [No requirements were specified]
8. *Analyze*
   - R9: The prototype shall allow the user to add a measurement to a factor.
   - R10: Once a measurement has been added, the prototype shall calculate a realized value for the selected feature.
   - R11: Once a realized value has been calculated, the prototype shall determine if the selected feature is ready for wide-deployment.
   - R12: The prototype shall allow the user to view the predicted value, the tolerated gap and the realized value of the selected feature on a graph.

Several aspects are not included in the prototype due to the time limitations of the study. In the *Model* step, a user cannot apply the Goal/Question/Metric approach. The GQM approach helps the user to find the affected factors for the feature. However, it was not included in the prototype as it can be well-explained verbally and only the output of it, i.e. the identified factors, is used in the prototype. The *Instrument*, *Implement*, *Deploy* and *Monitor* steps were not included in the prototype, as a verbal description of these steps was deemed sufficient. In addition, for carrying out these steps in a concrete way there is a need for connecting the prototype to the feature's system which was not possible in the context of this study. Currently in the *Analyze* step, any unexpected value needs to be added manually and the actual decision based on the outcome of the gap analysis, i.e. the comparison between the predicted value and the realized value, needs to be done outside of the prototype. In an improved version, unexpected values would be added automatically and the decision process of the analyze step would be integrated into the prototype. However, the prototype does indicate if a feature is ready for wide-deployment. Apart from these exclusions the prototype is a complete representation of the DVOCE model and can thus be used as visualization tool for the validation.

### 5.3.2 Implementation

The prototype was written in JavaScript using the MEAN stack[1]. The MEAN stack uses MongoDB, Express.js, Angular.js and Node.js and is a fullstack JavaScript framework for web development. As the goal with the prototype was to show an example implementation of the process model to aid with validating the model, there were no compatibility issues that needed to be addressed. However, a decision was made to have the prototype as an online application so it could easily be accessed by the companies. When it came to choosing a web development stack or framework there were many possibilities. As the researchers of this study had limited experience in web development, they went with a recommendation to use the MEAN stack by one of the researchers in the Software Center project who has extensive experience in web development. Going with that recommendation also enabled the researchers of this study to easily seek advice when necessary.

An example usage of the prototype with screenshots can be seen in Appendix C. Furthermore, a video of the usage of the prototype can be seen on YouTube[2].

[1] http://mean.io/. [Accessed: 5th of September, 2016.]

[2] https://www.youtube.com/watch?v=e-V5sAMjS2g [Accessed: 5th of September, 2016.]

# 6

# Validation

Several methods can be used to validate an artifact in design science research. For the purpose of this research, two methods were selected: expert opinion and simulation. Section 6.1 covers the main outcome of the validation that was carried out in Iteration IV of the design cycle, which was used as an input for Iteration V. In section 6.2 the final validation of the DVOCE process model will be covered.

## 6.1 Iteration IV

The DVOCE process model was validated with help from the prototype in both a Software Center project workshop and in three validation sessions at the companies. These are categorized as expert opinion. In addition, a simulation was carried out which showed how the prototype works with example data.

From the validation in the Software Center project workshop, it was evident that it was challenging for the participants to map their work to the process model and the prototype. In addition, the need to have the data collection into the prototype more automatic was stressed. As a response to that, a simulation was carried out in the validation sessions at the companies to show how an automatic solution might work. It is not possible to draw any definite conclusions from the validation sessions in Iteration IV due to the low number of participants. Nevertheless, it was clear that DVOCE needed to be amended. There was a need to simplify the model and make it more concrete and coherent.

## 6.2 Iteration V

In Iteration V of the design cycle, eight validation sessions were held at the companies with a total of ten participants. The goal of the validation sessions was to gather feedback from experts about the process model with help from the prototype. In subsection 6.2.1, the results from the questionnaire will be presented. Subsection 6.2.2 covers an analysis of the multiple-choice questions of the questionnaire. In subsection 6.2.3, the open-ended questions are analyzed. Finally, subsection 6.2.4 discusses observations made by the researchers, as well as verbal feedback from the participants.

### 6.2.1 Questionnaire Results

At the end of each validation session the participant(s) filled out a questionnaire (see Appendix B) consisting of a question about the participant's role, 10 multiple-choice questions, and one open-ended question at the end. A statistical analysis cannot be carried out on the results from the questionnaire due to the low number of participants. However, the results can provide an indication of how a software implementation based on the process model would fare in real-world setting. The results from the multiple-choice questions are displayed in Fig. 9 to 18.

2. I understand the main aspects of the process model.

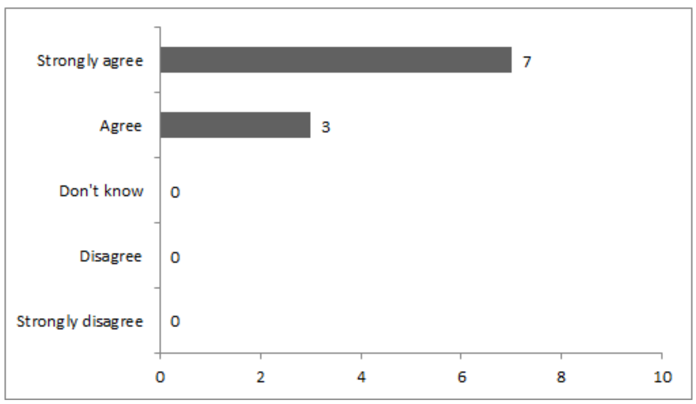


**Fig. 9**

3. I believe that a software implementation based on the process model could be helpful when *modeling and predicting* feature value.

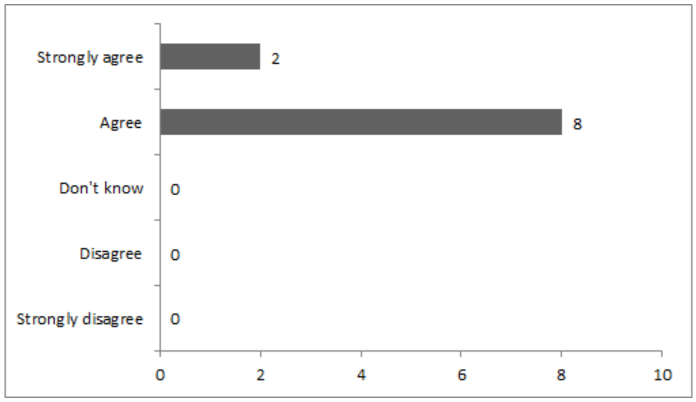


**Fig. 10**

4. I believe that a software implementation based on the process model could be helpful when *tracking* feature value.

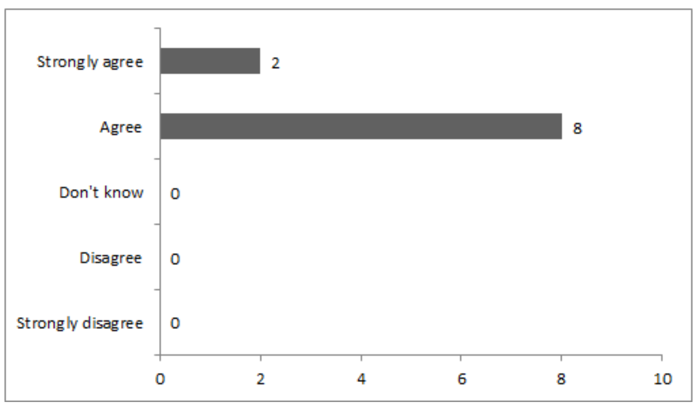


**Fig. 11**

5. I believe that a software implementation based on the process model could be helpful when *deploying* features.

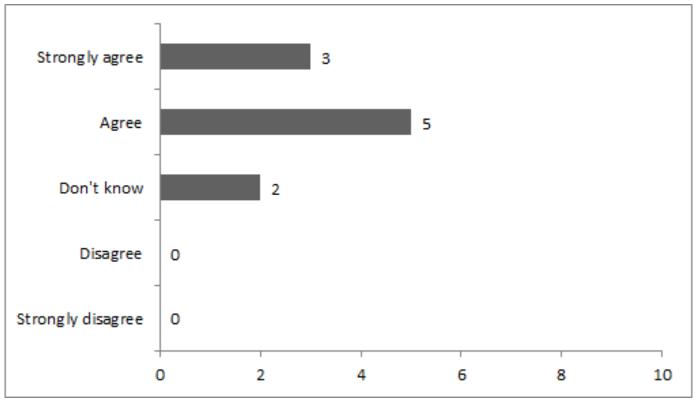


**Fig. 12**

6. I know of one or more projects that I have been part of where a software implementation based on the process model could have been useful.

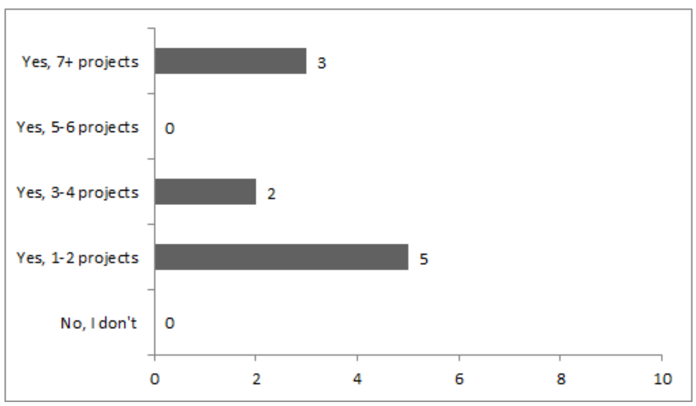


**Fig. 13**

7. How do you feel the process model compares to other similar process models that you have encountered?

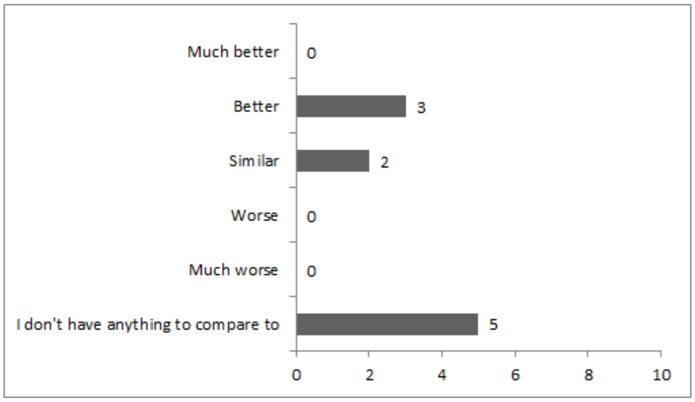


**Fig. 14**

8. I believe that a software implementation based on the process model could be useful for similar sized companies in a different field.

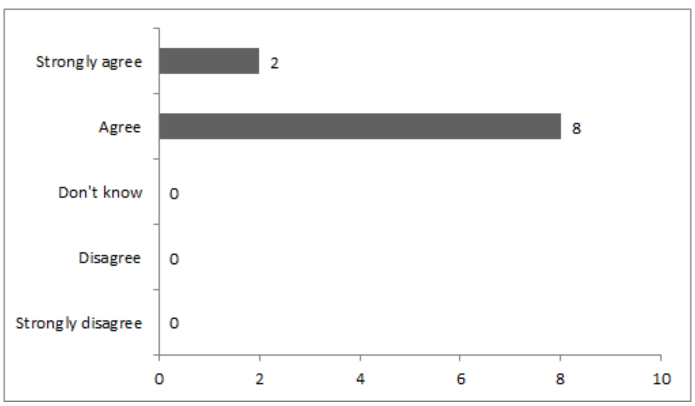


**Fig. 15**

9. I believe that a software implementation based on the process model could be useful for larger companies.

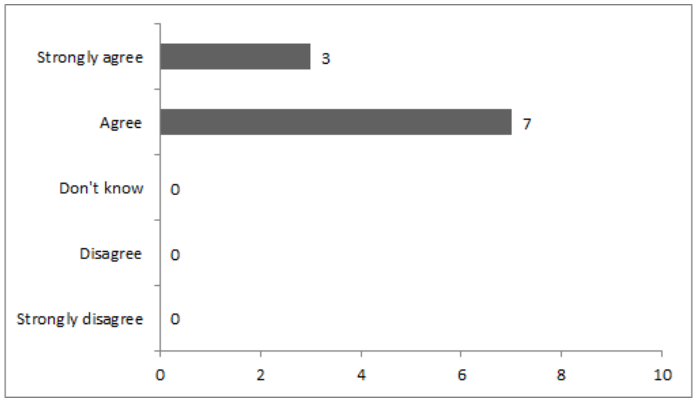


**Fig. 16**

10. I believe that a software implementation based on the process model could be useful for smaller companies.

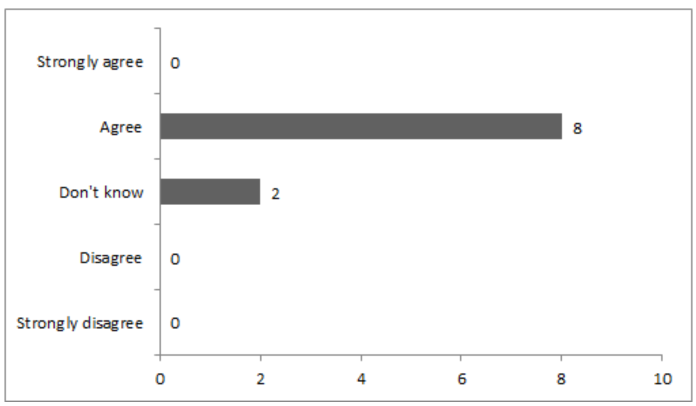


**Fig. 17**

11. I believe my company would be willing to invest in a software implementation based on the process model.

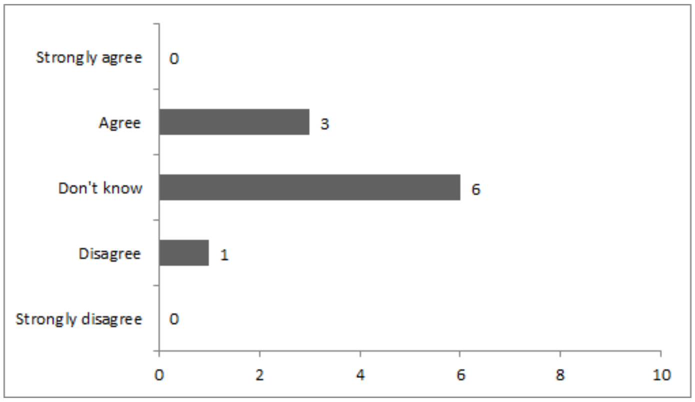


**Fig. 18**

At the end of the questionnaire, two open-ended questions are provided:

12. If you *agree* with statement 11, what was the deciding factor?

13. If you *disagree* with statement 11 or selected *don't know*, what would need to be changed or added to the *process model* for you to agree?

Depending on their answer in statement 11, participants answered either 12 or 13. The participants' answers can be seen in Table 13.

### 6.2.2 Analysis of Multiple-Choice Questions

The multiple-choice questions or statements can be divided into four categories: effect questions, trade-off questions, sensitivity questions and requirements satisfaction questions [47]. However, the second question serves as a general question of whether the participants understand DVOCE. If the participants do not understand the process model, then any follow-up questions about it are not considered valid. As can be seen from Fig. 9, all participants agreed or strongly agreed to understanding the process model.

The categories [47], their description and the results from the multiple-choice questions can be seen in Table 14. What follows is a discussion of each of the categories, its statements and results.

#### Effect Questions

Effect questions focus on what happens when a software implementation based on the process model interacts with the context that it is being used in [47]. The following statements fall into this category:

**Table 13:** The participants' answers to the open-ended questions of the questionnaire.

| Participant | Answer to Statement 11 | Answer |
|---|---|---|
| Developer at Company A-I | Agree | (Q12) I think [the company would] at least be interested in giving the idea a try as an internship. |
| Manager at Company Y | Agree | (Q12) How flexible the tool is and how easy it is to integrate/combine with other tools. I might prefer a Excel sheet with some macros and graphs. |
| Innovation and Partner Manager at Company X | Agree | (Q12) Based on the fact that I personally think it would be worth it. It is a matter of selling it to other parts of the company. However the tool needs to be offered by a professional company, probably together with education before [the company] can buy it. A long journey. |
| System Architect at Company B | Don't know | (Q13) I think we really need to sell the idea with stakeholders, with actual examples from our business, and not just deploy the software without this context. Then I think there is a good chance. |
| System Administrator at Company Y | Don't know | (Q13) Higher focus on using data. |
| Program Manager at Company A-II | Don't know | (Q13) Clarify the cost of trying it out. |
| Technical coordinator/ developer at Company A-II | Don't know | (Q13) The impression I got was that – compared to what we do today – the model unifies the way we express the value of a feature (almost similar as what we do with resource points in the resource domain). Today we quantify feature value by putting expectations the explicit KPIs/use-cases and we are not trying to unify the value. There could be a value in doing this (so that we can prioritize according to value better), but then we also need to find a way to map the expectations in such value points and I don't know how complicated such a thing is to create/maintain. |
| Line Manager at Company A-I | Don't know | (Q13) *No answer to this question.* |
| System Manager at Company A-I | Don't know | (Q13) Current uncertainties. |
| Area Product Owner at Company A-I | Disagree | (Q13) Proof how to integrate it in our environment, with our logs, etc. |

**Table 14:** The four categories that the multiple-choice questions or statements are divided into, a description of what each category answers and the results.

| Category | Description | Result |
|---|---|---|
| Effect questions | What happens when a software implementation based on the process model interacts with the context that it is being used in? | A software implementation of the process model has the potential to be helpful when 1) modeling and predicting feature value, 2) tracking feature value and 3) deploying features. |
| Trade-off questions | How does the process model perform in comparison with similar process models? | No definite conclusion can be drawn. |
| Sensitivity questions | What happens when a software implementation based on the process model is used in a different context and how does it scale? | A software implementation of the process model has the potential to be useful for similar sized companies in a different field and for both larger and smaller companies. |
| Requirements satisfaction questions | How well does a software implementation of the process model fulfil the requirements of the stakeholder. | No definite conclusion can be drawn. |

3. I believe that a software implementation based on the process model could be helpful when *modeling and predicting* feature value.

4. I believe that a software implementation based on the process model could be helpful when *tracking* feature value.

5. I believe that a software implementation based on the process model could be helpful when *deploying* features.

6. I know of one or more projects that I have been part of where a software implementation based on the process model could have been useful.

Statements 3 to 5 (Fig. 10-12) focus on the main tasks of DVOCE: 1) modeling and predicting feature value, 2) tracking feature value and 3) knowing when to deploy features. As can be seen from Fig. 10-12, all participants either agree or strongly agree that a software implementation of the process model could be helpful when modeling, predicting and tracking feature value. For deploying features, 8 agree or strongly agree while 2 are not sure. Furthermore, all participants have been part of a project or projects where a software implementation based on DVOCE could have been useful.

Based on the answers to the effect questions, a software implementation of DVOCE has the potential to be helpful when 1) modeling and predicting feature value, 2) tracking feature value and 3) deploying features.

#### Trade-Off Questions

Trade-off questions focus on how the process model performs in comparison with similar process models [47]. The following statement falls into this category:

7. How do you feel the process model compares to other similar process models that you have encountered?

Statement 7 (Fig. 14) shows that 7 out of the 10 participants either feel that it is similar to other process models that they have used or have nothing to compare to. Three participants feel that it is better than other process models that they have encountered. Based on that, no definite conclusion can be drawn.

#### Sensitivity Questions

Sensitivity questions focus on what happens when a software implementation based on the process model is used in a different context and how it scales [47]. The following statements fall into this category:

8. I believe that a software implementation based on the process model could be useful for similar sized companies in a different field.

9. I believe that a software implementation based on the process model could be useful for larger companies.

10. I believe that a software implementation based on the process model could be useful for smaller companies.

All participants agree or strongly agree to statements 8 and 9 (Fig. 15 and 16). For statement 10 (Fig. 17), 2 participants are unsure whether it would work for smaller companies while 8 believe that it would be useful. Based on the answers to the sensitivity questions, a software implementation of DVOCE has the potential to be useful for similar sized companies in a different field and for both larger and smaller companies.

**Requirements Satisfaction Questions**

Requirements satisfaction questions focus on how well a software implementation of the process model fulfills the requirements of the stakeholder [47]. The following statement falls into this category:

11. I believe my company would be willing to invest in a software implementation based on the process model.

Statement 11 (Fig. 18) is answered in a neutral way by 6 participants, while 3 agree to the statement and 1 disagrees. Based on that, no definite conclusion can be drawn.

### 6.2.3 Analysis of Open-Ended Questions

The participants' answers to the open-ended questions can be seen in Table 13. Their answers will be discussed in this subsection.

3 out of the 10 participants agreed to statement 11 (*I believe my company would be willing to invest in a software implementation based on the process model*). A developer at Company A-I thinks that the company would be interested in trying it out on a trial basis. An innovation and partner manager at Company X has a similar answer but emphasizes that it would be a long journey until the company would be willing to invest in it, as proper training would be needed as well as selling the solution to other parts of the company. Finally, a manager at Company Y mentions how flexible the tool is and how easy it is to integrate with other tools although he believes Excel would be able to solve many of the aspects that the prototype currently focuses on.

6 out of the 10 participants were not sure if their company would be willing to invest in a software implementation based on the process model. A system architect at Company B emphasizes the need to have real examples from their business so it can be deployed in the right context and in that way sell the idea to the stakeholders. A system administrator at Company Y wants to see more focus on using the data. A program manager at Company A-II says that it needs to be clarified what the cost of trying it out would be. A technical coordinator/developer at Company A-II, states that it maps well to what they do at the moment and that although it can be beneficial to unify the value it can be challenging to do so. Finally, as one participant did not write a response to

this question, the final answer was from a system manager at Company A-I who mentioned current uncertainties at the company as a motive for not knowing whether the company would be willing to invest in a possible solution based on the process model.

1 out of the 10 participants disagreed to statement 11, stating that his company would not be willing to invest in a software implementation based on the process model. However, it should be noted that while answering the question the area product owner at Company A-I said that the main reason was that the company had already started working on a solution for the problems that the process model seeks to address. The given answer was similar to the one that was given by a system architect at Company B, that it needs to be clear how it can be integrated to their environment, using their logs, etc.

Based on the answers to statement 11, no definite conclusion can be drawn on whether the companies would be willing to invest in a software implementation based on the process model. However, when looking through the answers to the open-ended questions, one can see that the main barrier has to do with mapping the process model and/or prototype to the individual companies. Therefore, it would be interesting to have an example from each of the companies' domain and show how it can be used within their environment, as well as giving an estimate of how much time it would take to integrate the solution.

### 6.2.4 Observations and Feedback

During the validation sessions a few observations were made. It was noted that most of the participants were already familiar with the research area and specifically with the HYPEX model. A few were not and in those cases care was taken to introduce the background concepts. In addition, a substantial number of the interviewees described working in an environment where they feel the customer lead times are often too long to effectively use DVOCE. Even in those cases, they could still recall a few projects where the process model could have been useful. As the companies manage to climb further up the 'Stairway to Heaven' [3], the applicability of DVOCE is likely to increase further.

The most common aspect to be discussed during the sessions were the value constants and how to create them. An interesting connection featured in sessions with three of the participants: that deciding on value constants is a similar process to deciding on story points in Scrum development. These discussions brought up ideas of using the model in collaboration with a Scrum process and using value instead of story points in planning, burn down graphs, etc.

# 7
# Discussion

In this chapter, the outcome of the study is discussed. In section 7.1, the findings from the interviews and workshops are discussed in relation to other findings in the literature. Section 7.2 focuses on the research questions and how they were answered. Then section 7.3 presents the contribution to knowledge of this study. Validity threats are discussed in section 7.4. Finally, possible future work is described in section 7.5.

## 7.1 Findings

The findings from the interviews and workshops and generated hypotheses from the findings can be seen in Table 6.

Finding 1 states that there is a low usage of collected data and that it is not being used in a systematic way. Often the data is only used for troubleshooting. This finding reinforces similar findings in the literature and shows the need for a simple, concrete and systematic approach like DVOCE. In both [11] and [13], it is recognized that companies collect a significant amount of data. In [11], it is discussed that although the companies collect large amount of data, both customer feedback and product data, the product data is not focused enough towards the data-driven aspect and as the experimentation process is usually not systematic, the collected data is under-utilized. In [13], the collected data is mainly used for troubleshooting, as is the case in [59] and [60]. In [61], it was found that companies only benefit from a small amount of the data that they gather. In [24], it is discussed that there does not exist a systematic way for collecting and making use of the data.

Finding 2 states that a feature is a part of a value category, i.e. a feature is categorized based on its type of value to customers. This aspect has not been discussed in other research in the field. It could potentially be an entry point into making the value definition more concrete. One interviewee mentioned that part of the difficulty was to figure out how much value a feature has for an entire system as it is not easy to pinpoint its exact effect. By using a divide and conquer approach, value categories could be a middle step between the total value and the feature value and function as a stepping stone to link the feature value to a concrete selling point value.

Finding 3 states that mainly feature value modeling and prediction is conducted at the companies while tracking value and analyzing it is rarely done. Furthermore, it is often done ad-hoc and thus not in a systematic way. Many of the aspects that are discussed in relation to Finding 1 also apply to this finding as they are closely intertwined. The main point is that there is a need for a systematic way for collecting data and carrying out continuous experimentation in a data-driven way [11, 13, 24].

Finding 4 states that the value is different between customers. In [62], it is discussed that while a single experiment can provide value for one customer, it might not deliver the same value for another customer. An aspect that is important to keep in mind for companies that have many customers. DVOCE does not consider this aspect currently due to time-constraints and Finding 5, which states that there is a need for a simple process to spread the value thinking. Once the companies have embraced the value-oriented approach, this is worth looking into. If it is only a small part of the customer base that values a new feature, perhaps there is another feature that is more important to focus on.

Finding 5 states that there is a need for a simple process to spread the value thinking, especially to get support from management for this new way of working. That is related to a finding in [11], where it is discussed how important it is that the organizational culture supports the experimentation process. A similar finding is presented in [62], where the interviewees voice their concerns if only part of the organization has adopted continuous experimentation as there is a need for everyone to understand it. DVOCE tries to be this simple process and in the validation sessions everyone was able to understand the model.

Finding 6 states that there is a need for an automatic process to improve the prediction and tracking of feature value as the companies want to get measurements cheap. This need has not been explicitly described in previous research in the field. However, when it comes to collecting data and logging, the need for an automatic tool has been recognized [35, 37]. Such a logging tool would help developers make the right decisions and minimize the cost of gathering data. The prototype of DVOCE included an import function for data from .csv files to showcase how adding measurements and tracking value could be made automatic.

## 7.2 Research Questions

This study had two research questions:

- *RQ1:* How to model, predict and analyze feature value in a way that is useful and directly applicable by companies?

- *RQ2:* How to track the realized feature value using appropriate customer feedback and data collection techniques?

Determining the success of answering these research questions was the objective of the validation sessions, described in section 6.2. The multiple-choice questions were divided into four categories and two of those can be used to answer the two research questions: effect questions and requirements satisfaction questions.

The former category, effect questions, looks into what happens when a software implementation based on the process model interacts with the context that it is being used in. Based on the analysis of the multiple-choice questions (subsection 6.2.2), a software implementation of the process model has the potential to be helpful when 1) modeling and predicting feature, 2) tracking feature value, and 3) deploying features. Here, deploying features is the actual analysis of whether a feature is living up to its expectations or not; this answers the question: should the feature be commercially deployed or not? Therefore, the effect questions show that RQ1 and RQ2 have been answered.

The latter category, requirements satisfaction questions, looks into how well a software implementation based on the process model fulfills the requirements of the stakeholder. Based on the analysis of the multiple-choice questions (subsection 6.2.2), no definite conclusion can be drawn and it is not known whether it can be directly applicable to the companies. However, as all participants could think of one or more projects where a software implementation based on the process model could have been useful, the process model is directly applicable to the companies to a certain degree. Furthermore, based on the answers from the effect questions and the open-ended questions (subsection 6.2.3), the participants see a potential for using a software implementation of DVOCE to model, predict, track and analyze feature value, although they are reluctant to answer whether their company is willing to invest in such a solution. The main barriers are mapping it to their domain and clarifying how expensive it would be to invest in the solution and integrate it into their system.

Therefore, the process model could be used to model, predict and analyze feature value as well as to track the realized feature value using the appropriate customer feedback and data collection techniques.

## 7.3 Contribution to Knowledge

The aim of this study was to create a process model that is an extended and more concrete version of HYPEX [12]. The outcome was the DVOCE process model which extends HYPEX in the following way:

- A detailed procedure for modeling a feature.
- Once modeled, how the value of a feature can be predicted.
- How to select the appropriate customer feedback and data collection techniques to track the value of the feature and instrument the system if needed.

- An extended description of comparing the tracked realized value to the predicted one in order to make informed decisions on the future of the feature.

These aspects have not been addressed in other process models [13, 14, 15, 16] and this study fills that gap.

In addition to the process model, interviews were carried out at two companies. The findings describe the challenges that the companies are facing when it comes to continuous experimentation, especially how to model, predict, track and analyze feature value. As described in section 7.1, some of the findings have been identified in previous literature, thus reinforcing the notion that these challenges still exist, while some have not been identified in a concrete way before.

## 7.4 Validity Threats

Validity [51, 56] is an important aspect of research. A study must be well designed and executed in such a way that the findings can be trusted. In addition, the findings should not be biased. Therefore, procedures for increasing validity must be taken during the early steps of the research. Validity threats are usually categorized as they focus on different aspects of the research. In this study, the categorization shown in [51] and [56] was used as the research is of qualitative nature and most of the validity threats apply to the interviews, the workshops and the validations. There, validity threats are divided into four categories:

- *Construct validity:* Is the outcome actually a contribution of the phenomena under study? That is, does it represent what the researchers had in mind?

- *Internal validity:* Is there a third, unknown, factor that affects the relationship between the two investigated factors?

- *External validity:* Can the findings be generalized? Are the findings of interest for other individuals or companies?

- *Reliability:* Is the data and the analysis dependent on the researchers? Would another researcher arrive at the same conclusion?

For quantitative analysis, conclusion validity is used instead of reliability. Conclusion validity is about the relationship between the treatment and the outcome, i.e. that a statistical significance exists. That does not apply in this case, as the data is qualitative.

### 7.4.1 Construct Validity

The first construct validity threat was that the researchers might have affected the outcome of the research, as it only dealt with qualitative data and the researchers' interpretation might have been biased by their expectations.

The second threat was that the researchers and either the interviewees or participants in the validation sessions might not interpret the content in the same way.

Triangulation [51] was used to mitigate these threats, as it increases the validity of research by providing different perspectives towards the studied object. It is especially important when dealing with qualitative data as the information gathered is more diverse than for quantitative data. Four types of triangulation are described in [51]: data, observer, methodological and theory triangulation. This research provides both data and observer triangulation.

Data triangulation was achieved as three units were selected for the interviews at two different companies. In addition, data from workshops, including two additional companies, and previously conducted interviews were used as a source of information. Furthermore, the validation sessions were carried out at all the four companies.

Observer triangulation was achieved by having two researchers conducting interviews, participating in workshops and carrying out the validation sessions. Furthermore, two researchers in the Software Center project reviewed the interview guide which was the basis for the interview questions and one researcher reviewed the validation questionnaire.

### 7.4.2 Internal Validity

There is always a possibility that conclusions which are drawn from data are not correct due to an unknown factor being the actual reason for the found relationship. However, this study does not seek to find causal relationship [51] and it is thus not a concern.

Nevertheless, the participants in the validation sessions 1) might be inclined to give more positive answers as they filled out the questionnaire while in the same room as the researchers, 2) know that the researchers can see their answers afterwards, and 3) know that by answering the questionnaire in a positive way they will help the researchers achieve their goals. In order to make the best use of the gathered data, the role of the participant is important but as most of the participants had different roles, anonymity was not achievable. Although it is not possible to directly compare the validations of Iteration IV and Iteration V, the results from Iteration V suggest that this was not a major problem as the participants were more positive in Iteration V than in Iteration IV when answering the questionnaire and in discussions.

### 7.4.3 External Validity

Working mainly with two companies can be a threat to external validity, i.e. when it comes to generalizing the findings of the research. However, there was some diversity in the companies as one is in the embedded systems area (Company A) while the other one is a pure software company (Company B). Furthermore, for Company A, two units were investigated, where the first one had a stronger focus on the embedded side and the second one had a stronger focus on the software side of the company. Company X and Company Y mitigate the threat to some extent, especially when it came to validating the process model as validation sessions were held at all the four companies.

Only a small scale validation of the process model could be conducted within the time frame of this study. A larger and more thorough validation would be needed to be able to generalize the findings to another context. However, to mitigate this threat the validation questionnaire included sensitivity questions [47] which focused on what would happen when a software implementation based on the process model would be used in a different context and how it would scale.

### 7.4.4 Reliability

With regard to reliability [51], different researchers may come to different conclusions. To mitigate this threat, three steps were taken.

One of the researchers transcribed the interviews and coded them. Afterwards, the other researcher listened to the recordings to verify the transcription and reviewed the coded transcriptions. This can decrease the likelihood of one researcher affecting the outcome too much [51] and it might therefore increase the likelihood of replicating the study with similar results.

Each iteration of the design cycle was documented. This included a detailed protocol for the data collection, analysis and validation [51]. These documents were used to report the findings of the research which increases the likelihood of being able to follow the exact steps of the research.

To minimize the risk of having the interview questions or the validation questionnaire ambiguous, two researchers in the Software Center project reviewed the interview guide which was the basis for the interview questions and one researcher reviewed the validation questionnaire.

## 7.5 Future Work

As only a small scale validation of the process model was carried out, a larger and more thorough validation would be welcome.

Additionally, defining the value constants is a critical point of the DVOCE model. As mentioned in the *Model* step in section 5.2, defining the value constant so that it is tied to a concrete value is very difficult for most features. This study went with a simple approach due to Finding 5, that a simple process is needed in the industry to spread the value thinking. This finding also meant that the simple approach is necessitated by RQ1, that the model should be directly applicable by companies. A future study could be done to develop a more intricate approach to defining the value constant that would make it possible to tie the constant to a concrete value, e.g. profit, in more cases.

The connection to Scrum and similar frameworks is interesting and a study could be performed how to best integrate DVOCE and its value-oriented approach with these frameworks.

Finding 2, *Value categories*, and Finding 4, *Value is different between customers*, were decided to be out of the scope of this study. How features are categorized and what impact this has on value would be interesting to explore further. In addition, that the value of a feature depends on which customer one is asking can potentially have big ramifications. For example, a large amount of resources can be spent on developing a feature that only a very small part of your customer base actually values highly. A study on how to incorporate this aspect into the DVOCE model would also be welcome.

# 8

# Conclusion

According to recent research [3, 10, 11], few companies have been able to climb the entire 'Stairway to Heaven'. A contributing factor for why it has been problematic to move from CD to R&D as an experiment system is the gathering of relevant data from customers resulting in an 'open loop' between customers and product management [12]. Therefore, decisions are often based on 'gut feeling' rather than relevant data [5, 12]. Several models ([12], [13], [14], [15] and [16]) have been created to help companies reach continuous experimentation. The aim of this study was to create a process model that is an extended and more detailed version of the HYPEX model [12]. By using the HYPEX model, companies can run feature experiments to close the 'open loop' and confirm that a feature that was selected for development has the expected value that was predicted. However, neither the HYPEX model nor other models ([13], [14], [15] and [16]) provide a concrete way to model a feature, predict its value, track the realized value with appropriate customer feedback and data collection techniques and compare the realized value to the predicted one in order to make informed decisions based on actual data rather than 'gut feeling'. In this study, design science research was used as a research methodology while interviews and workshops played a central role in investigating the problems that companies face. The DVOCE process model, a detailed and extended version of the HYPEX model, was presented. The model takes into consideration the identified problems found during the study as well as other research within the field. It provides a detailed way of modeling, predicting, tracking and analyzing feature value. In a small-scale validation, the DVOCE process model showed potential to be useful for companies in their efforts to model, predict and analyze feature value. It also showed potential to enable tracking of the realized feature value using appropriate customer feedback and data collection techniques.

# Bibliography


[1] Fogelström, N.D., et al. 2010. The Impact of Agile Principles on Market-Driven Software Product Development. Journal of Software Maintenance and Evolution: Research and Practice, 22, pp. 53-80.

[2] Highsmith, J., and Cockburn, A. 2001. Agile Software Development: The business of innovation. Software Management, 34, pp. 120-122.

[3] Olsson, H.H., Alahyari, H., and Bosch, J. 2012. Climbing the "Stairway to Heaven": A multiple-case study exploring barriers in the transition from agile development towards continuous deployment of software, In Proceedings of the 38th Euromicro Conference on Software Engineering and Advanced Applications, September 5-7, Cesme, Izmir, Turkey.

[4] Humble, J. and Farley, D. 2011. Continuous Delivery: Reliable Software Releases through Build, Test and Deployment Automation, Addison-Wesley: Boston.

[5] Bosch, J. 2012. Building Products as Innovations Experiment Systems. In Proceedings of the 3rd International Conference on Software Business, June 18-20, Cambridge, Massachusetts.

[6] Eisenberg, B., and Quarto-vonTivadar, J. 2008. Always Be Testing: The Complete Guide to Google Website Optimizer. Indiana: Wiley publishing.

[7] Kohavi, R. et al. 2009. Controlled experiments on the web: survey and practical guide. Data Mining and Knowledge Discovery, 18, pp. 140-181.

[8] Davenport, T.H. 2009. How to Design Smart Business Experiments. Harvard Business Review, February.

[9] Bosch-Sijtsema, P., and Bosch, J. 2014. User involvement throughout the innovation process in high-tech industries. Journal of Product Innovation Management, 32, pp. 793-807.

[10] Karvonen, T. et al. 2015. Hitting the target: practices for moving toward innovation experiment systems. In Proceedings of the 6th International Conference on Software Business, ICSOB, June 10-12, Braga, Portugal.

[11] Lindgren E., and Münch, J. 2016. Raising the odds of success: the current state of experimentation in product development. Information and Software Technology, 77, pp. 80-91.

[12] Olsson, H.H., and Bosch, J. 2014. From Opinions to Data-Driven Software R&D: A Multi-Case Study On How To Close The 'Open Loop' Problem. In Proceedings of the 40th Euromicro, Software Engineering and Advanced Applications, August 27-29, Verona, Italy.

[13] Olsson, H.H., and Bosch, J. 2015. Towards Continuous Customer Validation: A Conceptual Model for Combining Qualitative Customer Feedback with Quantitative Customer Observation. In Proceedings of the 6th International Conference on Software Business, June 10-12, Braga, Portugal.

[14] Fabijan, A., Olsson, H.H., and Bosch, J. 2015. Early Value Argumentation and Prediction: An Iterative Approach to Quantifying Feature Value. In Proceedings of the 16th International Conference on Product-Focused Software Process Improvement (PROFES 2015), December 2-4, Bolzano, Italy.

[15] Fagerholm, F., et al. 2016. The RIGHT model for Continuous Experimentation. Journal of Systems and Software. In Press.

[16] Bosch, J. et al. 2013. The Early Stage Software Startup Development Model: A Framework for Operationalizing Lean Principles in Software Startups. In Proceedings of the 4th International Conference on Lean Enterprise Software and Systems, LESS, December 1-4, Galway, Ireland.

[17] Software-center.se. 2015. Software Center - Software Center, University of Gothenburg, Sweden. [Online]. Available: http://www.software-center.se/. [Accessed: 29-Sep-2015].

[18] Software-center.se. 2015. Fast Customer Feedback in Large-Scale Software Engineering - Software Center, University of Gothenburg, Sweden. [Online]. Available: http://www.software-center.se/research-themes/customer-data-ecosystem-driven-development/project–5. [Accessed: 29-Sep-2015].

[19] Biffl, S., et al. (eds.). 2006. Value-Based Software Engineering. Berlin/Heidelberg, Germany. Springer-Verlag Berlin Heidelberg.

[20] Fabijan, A., Olsson, H.H., and Bosch, J. 2015. Customer Feedback and Data Collection Techniques in Software R&D: A Literature Review. In Proceedings of the 6th International Conference on Software Business, June 10-12, Braga, Portugal.

[21] Bosch, J., and Eklund, U. 2012. Eternal Embedded Software: Towards Innovation Experiment Systems. In Proceedings of the 5th International Symposium On Leveraging Applications of Formal Methods, Verification and Validation, October 15-18, Heraklion, Crete.

[22] Olsson, H.H., and Bosch, J. 2015. Towards Continuous Validation of Customer Value. In Proceedings of the 3rd International Workshop on Large-Scale Agile Development, May 25-29, Helsinki, Finland.

[23] Kohavi, R. et al. 2013. Online controlled experiments at large scale. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 11-14, Chicago, IL, USA.

[24] Sauvola, T. et al. 2015. Towards Customer-Centric Software Development: A Multiple-Case Study. In Proceedings of the 41st Euromicro Conference on Software Engineering and Advanced Applications, SEAA, August 26-28, Funchal, Madeira, Portugal.

[25] Fagerholm, F., et al. 2014. Building Blocks for Continuous Experimentation. In Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering, RCoSE, May 31 - June 7, Hyderabad, India.

[26] Tahir, T., Rasool, G. and Gencel, C. 2016. A systematic literature review on software measurement programs. Information and Software Technology, 73, pp. 101-121.

[27] Tahir, T., and Jafar, A. 2011. A Systematic Review on Software measurement programs. In Proceedings of the 9th International Conference on Frontiers of Information Technology, FIT, December 19-21, Islamabad, Pakistan.

[28] Kaner, C., and Bond, W.P. 2004. Software Engineering Metrics: What Do They Measure and How We Do Know? In Proceedings of the 10th International Software Metrics Symposium, September 11-17, Chicago (IL), USA.

[29] Basili, V.R. 1992. Software modeling and measurement: the Goal/Question/Metric paradigm, College Park, MD, USA.

[30] Botella, X., et al. (2004) ISO/IEC 9126 in practice: what do we need to know? Software Measurement European Forum 2004.

[31] Rodden, K., Hutchinson, H., and Fu, X. 2010. Measuring the User Experience on a Large Scale: User-Centered Metrics for Web Applications. In Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI, April 10-15, Atlanta, GA, USA.

[32] Barik, T., et al. 2016. The Bones of the System: A Case Study of Logging and Telemetry at Microsoft. In Proceedings of the 38th IEEE International Conferences on Software Engineering Companion, May 14-22, Austin, TX, USA.

[33] Kempf, T., Karuri, K. and Gao, L. 2008. Software Instrumentation. Wiley Encyclopedia of Computer Science and Engineering, 1-11.

[34] Kim, M., et al. 2016. The Emerging Role of Data Scientists on Software Development Teams. In Proceedings of the 38th International Conference on Software Engineering, ICSE, May 14-22, Austin, TX, USA.

[35] Fu, Q., et al. 2014. Where Do Developers Log? An Empirical Study on Logging Practices in Industry. In Proceedings of the 36th International Conference on Software Engineering, ICSE, May 31 - June 7, Hyderabad, India.

[36] Fisher, D., et al. 2012. Interactions with Big Data Analytics. Interactions, 19, pp. 50-59.

[37] Lal, S., and Sureka, A. 2016. LogOpt: Static Feature Extraction from Source Code for Automated Catch Block Logging Prediction. In Proceedings of the 9th Indian Software Engineering Conference, ISEC, February 18-20, Goa, India.

[38] Wohlin, C. 2014. Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering. In Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, EASE, May 13-14, London, England.

[39] Ries, E. 2011. The Lean Startup: How Constant Innovation Creates Radically Successful Businesses. London: Penguin Group.

[40] Olsson, H.H., and Bosch, J. 2016. From Requirements to Continuous Reprioritization of Hypotheses. In Proceedings of the 38th International Conference on Software Engineering, ICSE, May 14-22, Austin, TX, USA.

[41] Ruhe, G. 2010. Product Release Planning: Methods, Tools and Applications. Auerbach Publications.

[42] Jung, H.W. 1998. Optimizing value and cost in requirements analysis, IEEE Software, 15, pp. 74-78.

[43] Bagnall, A.J., Rayward-Smith, V.J., and Whittley, I.M. 2001. The next release problem, Information and Software Technology, 43, pp. 883-890.

[44] Denne, M. and Cleland-Huang, J. 2004. The incremental funding method: Data-driven software development, IEEE Software, 21, pp. 39-47.

[45] Greer, D., and Ruhe, G. 2004. Software release planning: An evolutionary and iterative approach, Information and Software Technology, 46, pp. 243-253.

[46] van den Akker, M., et al. 2008. Software product release planning through optimization and what-if analysis, Information and Software Technology, 50, pp. 101-111.

[47] Wieringa, R.J. 2014. Design Science Methodology for Information Systems and Software Engineering. Berlin/Heidelberg, Germany. Springer-Verlag Berlin Heidelberg.

[48] Vaishnavi, V., and Kuechler, W. 2004. Design Science Research in Information Systems. Last updated: November 15, 2015. [Online]. Available: http://www.desrist.org/design-research-in-information-systems/. [Accessed: 7-Mar-2016].

[49] Peffers, K., et al. 2006. The Design Science Research Process: A Model for Producing and Presenting Information Systems Research. In Proceedings of the 1st International Conference on Design Science Research in Information Systems and Technology, DESRIST, February 24-25, Claremonte (CA), USA.

[50] Hevner, A., et al. 2004. Design Science in Information Systems Research. MIS Quarterly, 28, pp. 75-105.

[51] Runesson, P., and Höst, M. 2009. Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering, 14, pp.131-164.

[52] Seaman, C. 1999. Qualitative Methods in Empirical Studies of Software Engineering. IEEE Transactions on Software Engineering, 25, pp. 557–572.

[53] Hove, S.E, and Anda, B. 2005. Experiences from Conducting Semi-Structured Interviews in Empirical Software Engineering Research. In Proceedings of the 11th IEEE International Software Metrics Symposium, September 19-22, Como, Italy.

[54] Pervan, G., and Maimbo, H. 2005. Designing a case study protocol for application in IS research. In Proceedings of the 9th Pacific Conference on Information Systems, PACIS, July 7-10, Bangkok, Thailand.

[55] Peffers, K., et al. 2007. A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems, 24, pp. 45-77.

[56] Wohlin, C., et al. 2012. Experimentation in Software Engineering. Berlin/Heidelberg, Germany. Springer-Verlag Berlin Heidelberg.

[57] Järvinen, P. 2007. Action research is similar to design science. Quality & Quantity, 41, pp. 37–54.

[58] Rissanen, O., and Münch, J. 2015. Transitioning Towards Continuous Delivery in the B2B Domain: A Case Study. In Proceedings of the 16th International XP Conference, May 25-29, Helsinki, Finland.

[59] Olsson, H.H., and Bosch, J. 2014. Towards Agile and Beyond: An Empirical Account on the Challenges Involved When Advancing Software Development Practices. In Proceedings of the 15th International XP Conference, May 26-30, Rome, Italy.

[60] Olsson, H.H., and Bosch, J. 2013. Towards Data-Driven Product Development: A Multiple Case Study on Post-deployment Data Usage in Software-Intensive Embedded Systems. In Proceedings of the 4th International Conference on Lean Enterprise Software and Systems, LESS, December 1-4, Galway, Ireland.

[61] Fabijan, A., Olsson, H.H., and Bosch, J. 2016. The Lack of Sharing of Customer Data in Large Software Organizations: Challenges and Implications. In Proceedings of the 17th International Conference on Agile Processes in Software Engineering and Extreme Programming, XP, May 24-27, Edinburgh, UK.

[62] Rissanen, O., and Münch, J. 2015. Continuous Experimentation in the B2B Domain: A Case Study. In Proceedings of the 2nd International Workshop on Rapid Continuous Software Engineering, RCoSE, May 16-24, Florence/Firenze, Italy.

# Appendix A

# Interview Questions

**Part I: Introduction to our project and the purpose of the interview**

**Part II: General questions (context)**

1. Could you describe what your role in the company is? What are the typical tasks of a typical day?
2. How does the typical workflow of a feature look like for you and your colleagues?
   (a) How are the people and teams involved organized?

**Part III: Hypothesized feature value realization and validation**

3. How do you model the value of a feature, i.e. what are the techniques you use to learn if the feature is valuable?
   (a) When do you model this value?
   (b) What data do you use to accomplish this?
      i. How is this data stored and accessed?
      ii. For whom is this data available?
   (c) Does the value you model change over time? If it changes, how does it change and how do you document these changes (how is a change in value visualized)?
4. How do you predict the future value of a feature?
   (a) When do you create your first prediction?
   (b) What data do you use to do this?
      i. How is this data stored and accessed?
      ii. For whom is this data available?
   (c) How is this prediction updated and maintained?
5. How do you find out that this new predicted value is likely to be realized?

6. How do you measure the realized value of a feature?
    (a) When is the first time you do this measurement?
    (b) What data do you use to do this?
        i. How is this data stored and accessed?
        ii. For whom is this data available?
    (c) How often do you measure the realized value?
    (d) How do you validate this measurement?
7. How do you compare the realized value to the predicted value?
    (a) At what times during the process do you do this?
    (b) How often during the lifecycle do you perform the comparison?
8. How do you make use of the information gained from performing the comparison?
9. What are the main challenges in value modeling and prediction? E.g. what makes predicting, measuring or validating the hardest?
10. How do you think the process could be improved, i.e. what actions could be taken to make value modeling and prediction more accurate?

# Appendix B

# Validation Questionnaire

1. What is your role (e.g. developer, product owner, etc.) in the company?

   ________________________________________________________

2. I understand the main aspects of the process model.

   ☐ Strongly agree.
   ☐ Agree.
   ☐ Don't know.
   ☐ Disagree.
   ☐ Strongly disagree.

3. I believe that a software implementation based on the process model could be helpful when *modeling and predicting* feature value.

   ☐ Strongly agree.
   ☐ Agree.
   ☐ Don't know.
   ☐ Disagree.
   ☐ Strongly disagree.

4. I believe that a software implementation based on the process model could be helpful when *tracking* feature value.

   ☐ Strongly agree.
   ☐ Agree.
   ☐ Don't know.
   ☐ Disagree.
   ☐ Strongly disagree.

5. I believe that a software implementation based on the process model could be helpful when *deploying* features.

   ☐ Strongly agree.

   ☐ Agree.

   ☐ Don't know.

   ☐ Disagree.

   ☐ Strongly disagree.

6. I know of one or more projects that I have been part of where a software implementation based on the process model could have been useful.

   ☐ No, I don't.

   ☐ Yes, 1-2 projects.

   ☐ Yes, 3-4 projects.

   ☐ Yes, 5-6 projects.

   ☐ Yes, 7+ projects.

7. How do you feel the process model compares to other similar process models that you have encountered?

   ☐ Much better.

   ☐ Better.

   ☐ Similar.

   ☐ Worse.

   ☐ Much worse.

   ☐ I don't have anything to compare to.

8. I believe that a software implementation based on the process model could be useful for similar sized companies in a different field.
   - ☐ Strongly agree.
   - ☐ Agree.
   - ☐ Don't know.
   - ☐ Disagree.
   - ☐ Strongly disagree.
9. I believe that a software implementation based on the process model could be useful for larger companies.
   - ☐ Strongly agree.
   - ☐ Agree.
   - ☐ Don't know.
   - ☐ Disagree.
   - ☐ Strongly disagree.
10. I believe that a software implementation based on the process model could be useful for smaller companies.
    - ☐ Strongly agree.
    - ☐ Agree.
    - ☐ Don't know.
    - ☐ Disagree.
    - ☐ Strongly disagree.
11. I believe my company would be willing to invest in a software implementation based on the process model.
    - ☐ Strongly agree.
    - ☐ Agree.
    - ☐ Don't know.
    - ☐ Disagree.
    - ☐ Strongly disagree.

12. If you *agree* with statement 11, what was the deciding factor?

13. If you *disagree* with statement 11 or selected *don't know*, what would need to be changed or added to the *process model* for you to agree?

# Appendix C

# Prototype Example

The prototype is described in section 5.3. What follows is an example usage of the prototype with screenshots.

Fig. C.1 shows the landing page. On the landing page, the name of the process model is presented and there is a video showing the usage of the prototype.

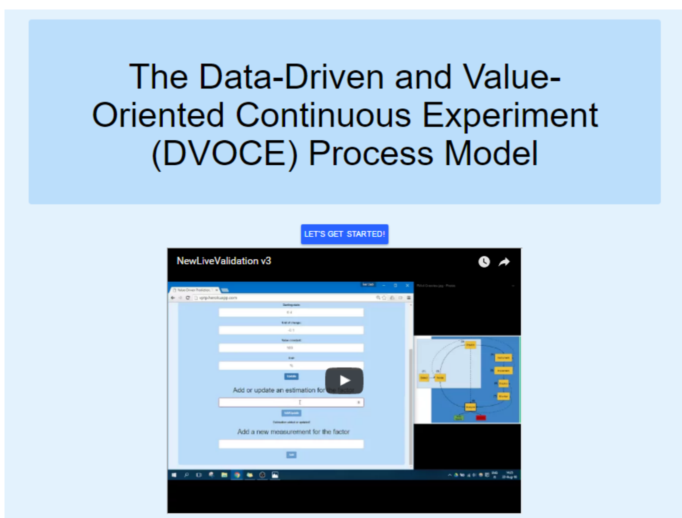


**Fig. C.1:** The landing page showing the name of the process model and a video of using the prototype.

Once the user is past the landing page (Fig. C.2), the first step is to select a feature and for this example, a Peer-to-Peer Reconnection is selected. A new feature can also be added and selected.

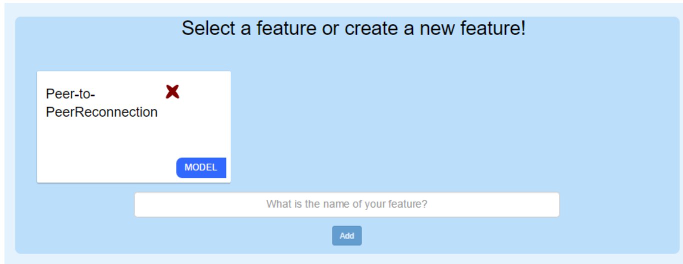


**Fig. C.2:** The first step is to select a feature.

The second step is then to model the feature. As can be seen from Fig. C.3, the feature has four factors. In addition, at the top of the figure, the predicted value of the feature can be seen, along with its gap threshold, its realized value and if the feature is ready for wide-deployment (realized value > gap threshold). It should be noted that these values are created once a prediction has been made, a gap threshold has been set and measurements have been added.

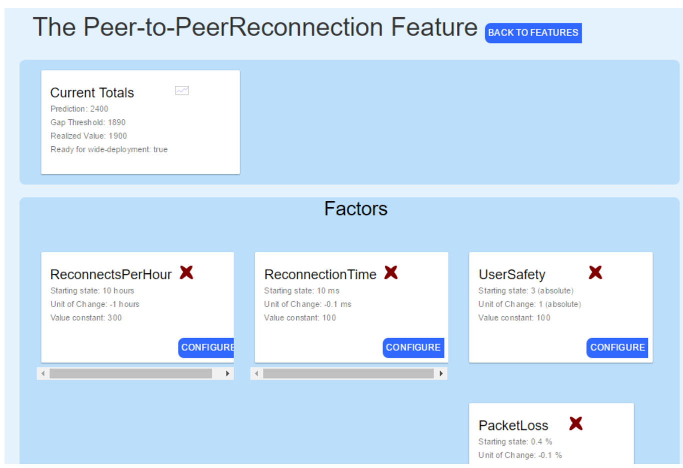


**Fig. C.3:** The second step is to model the feature.

Fig. C.4 then shows the bottom of the page where one can add factors. Each factor has a name, a starting state, a unit of change, a value constant and a unit. Then the tolerated gap can be set and estimations can be added through an import function.

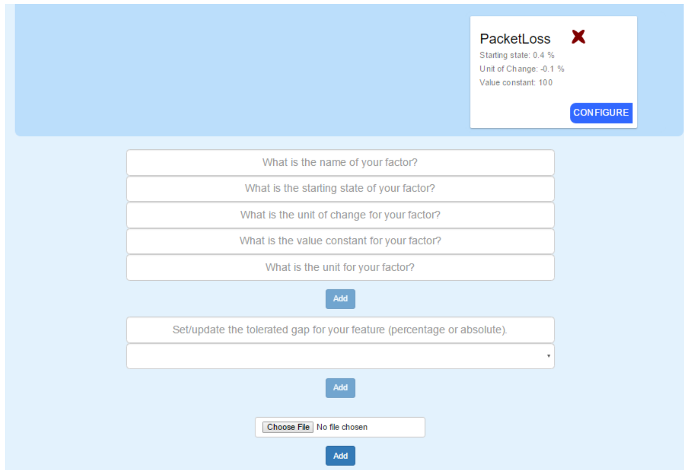


**Fig. C.4:** Adding factors, set a tolerated gap and import estimations.

When a factor is selected, it can be updated and a prediction (Step 3) can be made (Fig. C.5).

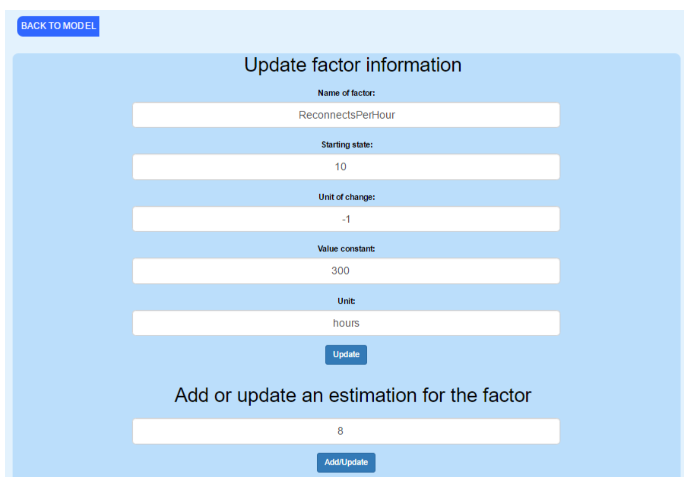


**Fig. C.5:** Updating factors and adding an estimation.

Once a feature has been instrumented (Step 4), a minimum viable feature has been implemented (Step 5), it deployed (Step 6) and monitored (Step 7), one can add the measured value for the factors (Fig. C.6), which can also be added through the import function depicted in Fig. C.4.

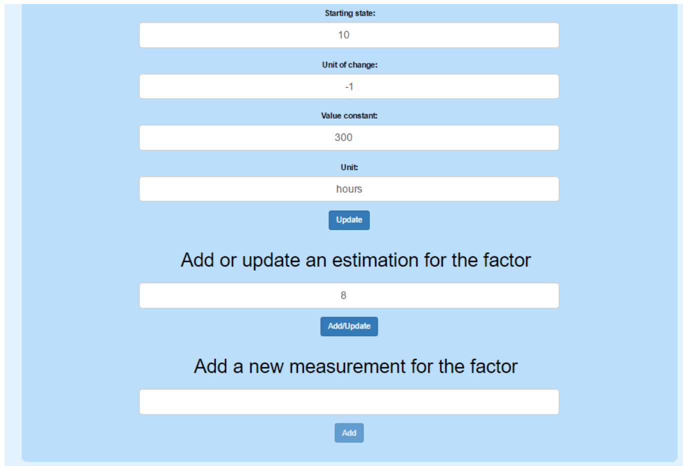


**Fig. C.6:** Adding a measurement for the factor.

Finally, when measurements have been added, the predicted value, the gap threshold and the realized value can be viewed on a graph (Fig. C.7), and an analysis can be carried out in the eighth and last step of the process model. The current values, and whether the feature is ready for wide-deployment, can be seen in Fig. C.3.

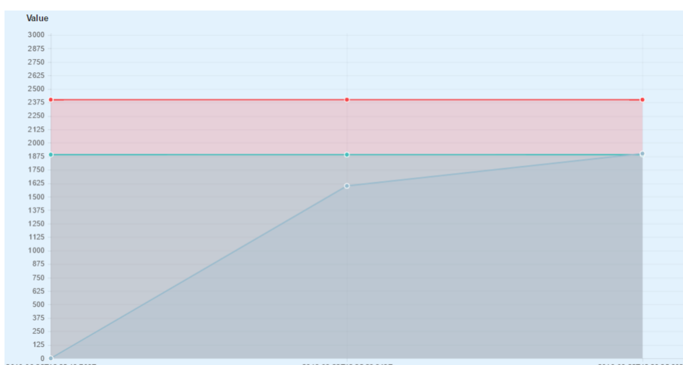


**Fig. C.7:** A graph showing how the realized value measures up to the predicted one.