Patterns for Continuous Experimentation

ALEXANDRE BITTENCOURT FARIA, Centro Paula Souza - FATEC Zona Sul, Brazil
DANILO ALVES DE JESUS, National Institute for Space Research, Brazil
FERNANDO PEREIRA, National Center for Monitoring and Early Warnings of Natural Disasters, Brazil
JOELMA CHOMA, National Institute for Space Research, Brazil
LUIS RICARDO ARANTES FILHO, National Institute for Space Research, Brazil
VANESSA GOMES ALBUQUERQUE, Group Being Educational - University Guarulhos - UNIVERITAS, Brazil
EDUARDO GUERRA, National Institute for Space Research, Brazil

The demand for software application development has grown exponentially in recent years and will certainly continue growing as new programming techniques and patterns are adopted. The development of applications that better adjust to the real needs of the customers is a challenge. Understanding the behavior of users to improve certain points of the software is also not an easy task. Often, this assessment is done by assumptions rather than being based on real data. One of the approaches to understanding the user behavior for software application development is through continuous experimentation. Continuous experimentation aims to obtain information about the behavior and preferences of the users' directly or indirectly through the analysis of the data generated by multiple experiments, in a continuous process and over time. Thus, such process can generate a large amount of data. In this article, we present a set of patterns related to the use continuous experimentation for the continuous improvement of software with a focus on the correct interpretation of data, as well as the most appropriate way of systematizing the process of experimentation of software products and services.

Categories and Subject Descriptors: H.5.2 [**Continuous Experimentation**]: Controlled Experiments—*Evaluation/ methodology*; H.1.2 [**Models and Principles**]: User/Machine Systems—*Human Information Processing*; I.5.1 [**Pattern Recognition**]: Data Processing—*Build Measure*

General Terms: Human Factors

Additional Key Words and Phrases: Patterns, Continuous Experimentation, Software Applications

ACM Reference Format:

Faria, A., Jesus, D., Pereira, F., Choma, J., Arantes Filho, L. R., Albuquerque, V., Guerra E. 2018. Patterns for Continuous Experimentation. HILLSIDE Proc. of Conf. on Pattern Lang. of Prog. 22 (October 2015), 16 pages.

1. INTRODUCTION

According to [Kuhn 1981], an experiment is a type of scientific research in which the researcher manipulates and interpret independent variables and observes the variation in the dependent variables concomitantly with the manipulation of the independent variables.

This work is supported by the Widget Corporation Grant #312-001.
Author's address: Faria, A., Jesus, D., Pereira, F., Choma, J., Arantes Filho, L. R., Albuquerque, V.; Av. dos Astronautas, 1.758 - Jardim da Granja, São José dos Campos - SP - 12227-010; email:[abittencourtfaria@gmail.com, daniloa47@gmail.com, fernando.opc@gmail.com, jh.choma@hotmail.com, luisricardoengcomp@gmail.com, vanessa.ga@gmail.com]; Guerra, E.; email: guerraem@gmail.com

The majority of the information about new software technologies (processes, methods, techniques, and tools) is still based on advertisements or self-opinions. However, scientific research cannot be based on commercial interest or opinions [Juristo and Moreno 2010]. In this sense, experimentation provides a systematic, disciplined and controlled path to evaluate human processes and activities [Wohlin et al. 2012].

Experiment-driven development approaches contribute to providing justifications for the use or not of technologies, based on indications about their effectiveness for software quality improvement. Thus, the results of experimental studies performed in different research scenarios can be used as starting points to define a set of criteria that support the decision-making process regarding the use of software technologies. One way to reduce the risks of software development is through continuous experimentation [Yaman et al. 2017].

Continuous experimentation (CE) is a development approach driven by experiments in products and services, which by iterative testing their assumptions can achieve success with less risk in an organization. The data collected through continuous experimentation is the requirement that must support the development and continuous improvement cycles of these products and services, also allowing a fast decision-making process [Schermann 2017].

Recently, studies in the field of continuous experimentation have pointed out that companies are increasingly interested in adopting this approach into their development process [Kevic et al. 2017] [Yaman et al. 2017] [Lindgren and Münch 2016]. By using continuous experimentation, some of these companies succeeded in reducing development efforts and to better understand the needs of their customers. Introducing continuous experimentation required organizational commitment and development of skills for experimentation. However, [Yaman et al. 2017] claimed that there are still few studies about how introducing continuous experimentation into an organization with an already established development process.

To help fill this gap, we sought to identify best practices within this research segment based on existing case studies in the literature. The objective of our study was a better understanding of the process of introducing and applying the stages of continuous experimentation already established in an organization.

In this paper, we present a set of patterns to promote continuous experimentation to improve software products and services. We have elaborated our patterns from related studies that indicate criteria for success and the construction of fundamental experimentation skills to systematize and support patterns based-on experiments. These patterns are against the adoption of empiricism for the construction of experimentation. These patterns present processes for possible solutions that fit into several situations, such as: the correct definition of hypotheses; the choice of the right people for experimentation; how to train and guide people through the process of experimentation; how to optimize the experimentation process and, how to interpret correctly the results of the experiments, avoiding unsubstantiated assumptions.

It is worth noting that the proposed patterns can be applied together, seeking the continuous improvement of the experimentation process. The integration of patterns based on continuous experimentation aims a better understanding, perception, motivation, and identification with their respective tasks of software engineers, data scientists, systems analysts, programmers, testers and all professionals involved in the software development process. Experimentation is a concept that must capture what people feel about the products, systems, and services to improve the user experience and consequently the success of projects and collaboration in organizations.

This paper is arranged as follows. Section 2 describes the concepts and examples of continuous experimentation. Section 3 describes the methodology in the development of the patterns. Section 4 introduces the patterns for continuous experimentation, an overview of each one and the relationships between them. Sections 5,6,7,8 and 9 explain in detail the proposed patterns indicating their motivations, positive and negative points, forces and main features. Finally, Section 10 refers to the discussions and concluding remarks.

2. BACKGROUND

In this section, we present some studies in continuous experimentation, as well as patterns of experiments delimit processes for their efficiency between the process of experimentation in organizations and their patterns.

Continuous experimentation has been suggested by [Fagerholm et al. 2014] as an approach for promoting continuous improvement of software products and services in line with customers' needs. In this study, the authors emphasize that in order to understand the main points of product improvement, it is necessary to obtain direct or indirect feedback from the clients and, from this, to implement each point observed by the customers. However, obtaining an accurate feedback may not be easy and its response cycle can be very slow due to the lack of efficient mechanisms of data collection and analysis [Olsson and Bosch 2014a]. There are some terms in continuous experimentation that are important to understand the patterns here presented:

—Experiment: refers to the process of testing assumptions [Munezero et al. 2017];

—Assumptions: Aspect of your idea that's accepted as true or as certain to happen, without proof [Munezero et al. 2017];

—Hypothesis: Is a proposed testable explanation for a phenomenon [Munezero et al. 2017];

—A/B test: is a way of testing that compares two versions (version A and version B), to decide what is better [Bakshy et al. 2014].

—Canary test: is a technique to reduce the risk of introducing a new software version in production by slowly rolling out the change to a small subset of users before rolling it out to the entire infrastructure and making it available to everybody [Shahin et al. 2017].

[Fagerholm et al. 2014] proposed a continuous experimentation system that seeks to evaluate the experiments data so that the decisions about changes in the software are taken and executed in the best way, avoiding assumptions and the development of applications based solely on users' opinions.

In the article "Model of Evolution of Experimentation" [Fabijan et al. 2017], three phases of evolution are detailed: technical, organizational and business evolution. The model aims to provide guidance to practitioners on how to develop and scale continuous experimentation in software organizations with the aim of becoming scale oriented.

For [Issa Mattos et al. 2018], the challenges and strategies in conducting continuous experimentation in organizations have shown in their studies of embedded systems, twelve different challenges in areas such as technicians, business and strategies, grouped into three categories architecture, data manipulation and development. [Bosch and Eklund 2012] present the architectural challenges for continuous experimentation in long-life embedded systems such as the ability to evolve and conduct experiments on the product already in use in a secure and controlled manner. They also conclude that not all embedded systems are suitable for continuous experimentation. For [Giaimo and Berger 2017], also related to the application of continuous experimentation in embedded systems, present criteria for experimentation in autonomous vehicles due to the critical aspects of security, real time response and limited resources present in this type of system. Some of these criteria are: testability, security, scalability, separation of interests, ease of integrating new developers, short development cycle, access to vehicle control, logging, transmission and data collection among others. For [Olsson and Bosch 2014b], software-intensive systems companies need to continually evolve their practices. Using conceptual model presented as the "Ladder to Heaven". This paper presents a transition process from traditional development to continuous software deployment.

Studies of web-controlled experiments were also addressed [Kohavi et al. 2009], establishing a relationship between changes and influence of user behavior. These experiments indicated return on investment (ROI), technical limitations, and organizations involving experiences such as Amazon, Microsoft, Dupont, and NASA that allowed rapid and effective innovation in various systems of controlled experiments, unifying them to return on investment and the construction of infrastructure for the experiment. The paper by [Rissanen and Münch 2015] analyzes the challenges, benefits and organizational aspects of continuous experimentation in the B2B domain. The results suggest that technical challenges are only part of the challenges a company faces in this transition. The company also needs to address customer and organizational culture challenges. Unique properties in each client's business play an important role and need to be considered when designing experiments. In addition, the

speed at which experiments can be conducted is relative to the speed at which production deployments can be made.

Web-oriented companies use online control and experiments to guide product development and accelerate innovation. Microsoft's Bing grows exponentially over time and in the running large-scale experiments require addressing multiple challenges in three areas: cultural / organizational, engineering and reliability. This article [Kohavi et al. 2013] aims to highlight the experimentation of the Bing quest and show that the system has accelerated innovation and increased annual revenues by hundreds of millions of dollars, enabling us to find and focus on evaluated ideas thousands of controlled experiments. Site owners always try to improve their sites by optimizing criteria using experiments. Still for [Kohavi et al. 2014], shares seven practical rules for experimenters, which we generalize from these experiments and their results, but the goal is to guide experimenters with basic rules that can help them optimize their sites and provide new research challenges on applicability, exceptions, and extensions in this subject.

According to [Tang et al. 2010] at Google, experimentation is practically a mantra. These changes such as modifications to a user interface or more subtle changes such as machine learning algorithms can affect the sorting or selection of content. The article specifically describes experimental processes that they have in place in Google, but believe that they can be generalized and applied by any entity interested in using experimentation to improve search engines and other websites. The creation of a system of experimentation mentioned by [Fagerholm et al. 2017], examines the preconditions for the creation of an experimentation system for continuous experiments with clients. He describes the RIGHT model for "Rapid Iterative Value Creation" through high-frequency testing, illustrating the building blocks required for such a system. The model is compared with the results of empirical case studies of two start-ups and more developed companies.

[Kevic et al. 2017] follows the same line as [Fagerholm et al. 2014] for continuous improvement in the development of software products and services. At this point, the authors propose that the software developer should improve its implementation process through continuous large-scale experimentation by evaluating small versions of software in a short time for end users. They proposed a framework for continuous experimentation to support the process of hypothesis definition and the creation of a set of metrics to see how software modifications impact the client. This process was applied in the Bing search tool where about 21220 experiments were used from 2014 to 2017. In this sense, to apply the continuous experimentation in large scale is something complex and for that reason, it is necessary to establish correctly the main elements that the process of experimentation should have.

3. RESEARCH METHOD

The process of identifying patterns of continuous experimentation has been done in three parts: literature review, brainstorming meetings and application of the pattern "Fly on the wall" [Lucrédio et al. 2004].

At the first phase, we carried out a literature review where the focus was mainly on works describing concrete examples of the use of the methodology different contexts, especially the results and learning achieved involving developers and system users.

In the next step, all authors participated in brainstorming sessions to discuss the results of relevant studies identified in the first phase. The main goal was to look for and recognize directives and behaviors transverse to all the works. From this discussion, it was possible to notice the main problems faced and the practices to be adopted. In this way the patterns described in this article were pre-identified.

Finally, to mature the identification of the patterns, it was applied the "Fly on the wall pattern". In this phase, the authors divided the writing process of the patterns and met to perfect their writing. In these meetings, for each pattern, the author of a specific pattern made a brief summary and then, only listened to the other authors to discuss positive points and points of improvement. The author then collected explanations, thus perfecting the writing of the pattern.

4. PATTERNS FOR CONTINUOUS EXPERIMENTATION

The patterns presented in this paper are intended to assist in the most correct configuration for the steps of creating continuous experiments on software products and services, as well as to demonstrate ways for developers to facilitate their process of establishing hypotheses and to test them before sending the final software product. All patterns have a common relationship among them, that is, they are patterns that when united allow a secure development of the continuous experimentation process obeying its main requirements.

These patterns function as recipes that fit the needs of a developer at a time of planning, execute and analyze the continuous experimentation process. Figure 1 indicates the relationship among patterns when the continuous experimentation process starts. The actions involved in the patterns embrace all the needs in the build of continuous experimentation process.

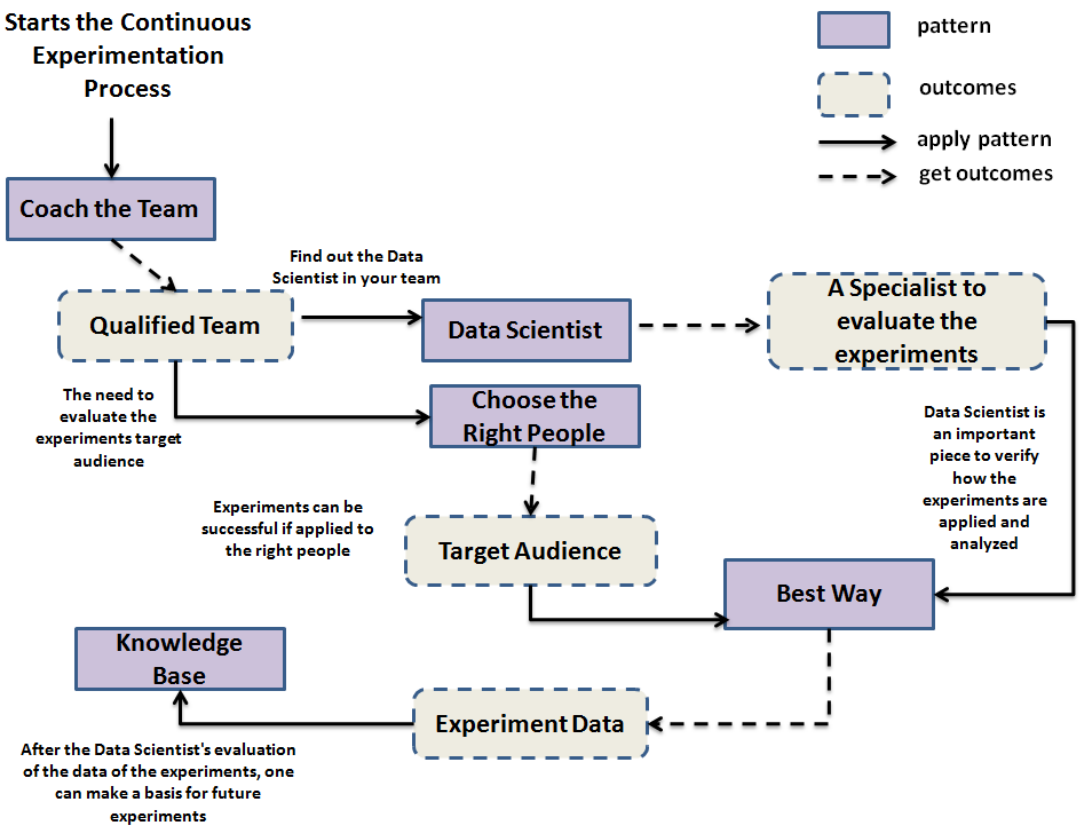

Fig. 1. Overview of the patterns and their relationships.

As shown in the Figure 1, the trained team, with the Data Scientist presence, is able to check the most appropriate target audience to apply the experiments. After the careful choice of the target audience, we do the approach of the best possible ways to configure the software improvement action by the data scientist assessment. At the end of the process, the data are carefully organized in a knowledge base that keeps the experiences obtained in the process. In future experiments we can be use the obtained experience, generating the optimization of the continuous experimentation process.

The pattern **Choose the Right People** addresses the need to select experts to work on projects that contemplate continuous experimentation. It is not trivial, but it is necessary to generate good results. The wrong choice of these experts can lead the project to failure. So choose the right people. Train teams for the ongoing experimentation process is not simple. It requires the team to be guided by data. From simple experiments we can disseminate the experimentation culture and then evolve them in this process, for that the pattern is **Coach the Team**. Experiencing the best possible way will help in the decisions to be taken. The results of the experiments allow the verification and analysis of related criteria, allowing process specialists a better analysis for judicious decision making, for that the pattern is **Best Possible Way**. The **Skills for Experimentation Data Scientists** seek help in the extraction of knowledge is essential the figure of the Data Scientist in this problematic "Continuous Experimentation". With this we highlight the importance of the Data Scientist and his abilities to exercise his skills in the process of experimentation. In **Knowledge Base for Continuous Experimentation**, we generate a knowledge base of the types of experiments performed. The target audience and their results is not trivial but extremely necessary to get results in new experiments. In this way, similar experiments can be implemented in other applications with resource use and reduced time.

5. COACHING THE TEAM

5.1 Motivational Example

Many professionals face difficulties in introducing new concepts and methodologies in their daily lives. Most of the time, these difficulties arise from a high learning curve, by the lack of interest of the team or by misaligned expectations in relation to the improvements that are intended to achieve. This is not different in the continuous experimentation area, and deploying it in the work environment can generate frustrations if it is not seen as a change of behavior. Experimentation requires that the team be guided by data rather than by assumptions and opinions. It is a cyclical process and companies that have been successful in its use make them evolve from simple experiments in order to spread the culture of experimentation and then evolve them.

5.2 Context

For teams unfamiliar with the practice of continuous experimentation, it may be difficult at the beginning of the methodology to formulate good hypotheses, build experiments, choose testers, and efficiently collect and analyze data, which makes it difficult to make assertive decisions.

5.3 Problem

How to start using the methodology of continuous experimentation in a new team in order to motivate it?

5.4 Forces

—Attempting to initiate the use of continuous experimentation in a team can be difficult because of the lack of knowledge in the new methodology.

—If the team cannot visualize a basic example of an experiment in a practical way in their working context, the experimentation methodology is less likely to be implemented.

5.5 Solution

Choose small teams, create simple experiments that aim to test small parts of the product. These experiments should serve as trials to disseminate the experimentation culture serving as training, generating interest and better adaptation to the new methodology.

Choosing teams, for example, up to 5 members will favor training, once the experimenter will have a greater control. Create simple experiments, as for example, logs generators or messages customization of the applications,

it will favor the experimentation once the experimenter can focus in the diffusion of methodology concepts and not in the in business rules. The same idea holds true for testing small parts of the product, which simplifies the initiation in the use of the continuous experimentation.

5.6 Consequences

(+) Training the team with simple experiments from your work context will stimulate it and it will be easier to evolve into sophisticated experiments.

(+) Teams trained from simple experiments come to better understand how continuous experimentation works and can see in methodology a way of being guided by data rather than by assumptions.

(-) There may be resistance on the part of managers to allocate resources for the adoption of the methodology of continuous experimentation, since the methodology may not bring immediate results.

5.7 Known uses

As an example can be mentioned the Ericson company that when starting the use of continuous experimentation sought to work from simplified experiments, with small teams and smaller modules of the product, training the team to then evolve their experiments in order to generate data and subsidize the decision making [Munezero et al. 2017].

6. CONTINUOUS EXPERIMENTATION BEST POSSIBLE WAY

6.1 Motivational Example

After planning for some change or development in a particular software application, you need to check and evaluate how best to execute this planning and make the best decisions. The decisions made through the results of the experiments allows the analysis in a criteria list related to how the decision should be made in the best way. This continuous experimentation pattern can be applied to the developer and customer in order to generate feedback based on the data from experiments.

6.2 Context

Develop experiments to verify how to apply in the best way certain premise (decision/hypothesis). Plan an intervention or treatment for a certain software feature and check in a criteria list that validates this intervention what is the best to follow. The created experiments aim to generate data to verify which criteria have the best acceptance.

6.3 Problem

In order to do software changes for performance improvement or customer acceptance, it is necessary to define a series of possible ways to carry out the proposed changes. In this sense, how to find what is the best way or the best criterion to confirm the proposed change?

6.4 Forces

—Create the experiment can decide what is best for the application.

—This pattern is integrated with the role of the data scientist

—Without experiments, the decision on what is the best for the application can be taken in the wrong way.

—Decisions will be based on data, not on people's opinions.

—There is an optimization in decision making.

—This pattern works as a heuristic.

6.5 Solution

After creating the hypotheses, plan a series of ways that can confirm these hypotheses. First, a list of criteria/ ways is developed indicating which methods can execute the initial idea. After the definition of the possible ways is carried out the experimentation.

Experimentation will generate data that, when interpreted by the data scientist, will indicate the suitability of the ways. Must be on the list, a default way, which allows you to check if all previous ways that have low acceptance. If the default way has the highest percentage of acceptance then the initial ways should be remodeled.

6.6 Use Examples - Costumer Vision and Developer Vision

Costumer Vision. Hypothesis: Modify the purchasing system in the e-commerce site to better serve customers.

—First, Develop the list of Criteria/Ways:
  —Way 1 : Modify the sales panel;
  —Way 2 : Modify the "Buy" button;
  —Way 3 : Modify sales options (card, ticket, Internet Bank);
  —Way 4 : Default, None of the Previous Functions.

—Develop the experiment to collect feedback data from the users.

—What is the best way? (Data Scientist)

—Make the decision for the best way. If the default gets the highest score then the other ways should be remodeled.

Developer vision. Hypothesis: Improve speed and reduce application start time.

—First, Develop the list of Criteria/Ways:
  —Way 1 : Modify the "execute" function;
  —Way 2 : Reduce load resources at start;
  —Way 3 : Transferring resources to a server - The app must have an internet connection;
  —Way 4 : Default, None of the Previous Functions.

—Develop the experiment to collect feedback data from the developers.

—What is the best way? (Data Scientist)

—Make the decision for the best way. If the default gets the highest score then the other ways should be remodeled.

Figure 2 shows the construction of the criteria/ways list, indicating the possible ways to test the defined hypothesis. In the end, is chosen the way with the best percentage, or acceptance rate.

6.7 Consequences

(+) The developer can check for minor changes to the software.

(+) The answers of the experiments allow to identify behavior.

(+) Deciding on a list of criteria reduces the possibility of performing software modification in the wrong way.

(+) Even if you reach the initial goals of the decision to be made, with the best possible feedback evaluation can be chosen. Or new hypotheses can be revealed.

(-) Creating the criteria list can be very complex.

(-) The wrong interpretation of the experiment can make the decision be taken the wrong way.

(-) It can be hard to see what are the pros and cons related to the way path.

(-) There may be abrupt changes in development when none of the experiments can observe suitability to the list.

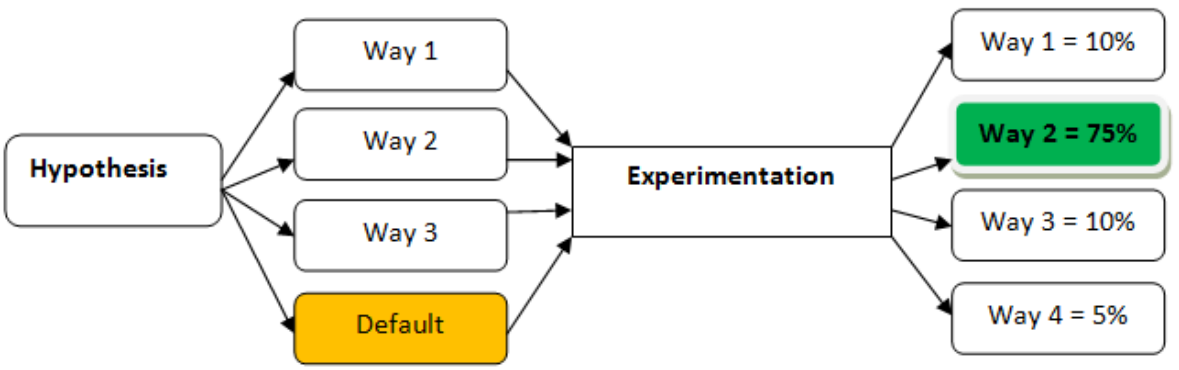

Fig. 2. Best Possible Way Pattern Scheme.

6.8 Known uses

Example of the Amazon website [Kohavi et al. 2014], which starts a series of experiments to verify how best to serve customers, i.e, is observed a search for solutions to execute the decision in the best way.

7. SKILLS FOR EXPERIMENTATION DATA SCIENTISTS

7.1 Motivational Example

Extracting knowledge from results generated from a continuous experimentation process is not trivial. To help in the extraction of knowledge, the figure of the Data Scientist in this problem is indispensable. Thus, we can highlight the importance of the data scientist and his abilities to play this role within the process of continuous experimentation.

7.2 Context

When a company applies the process of continuous experimentation, it is in search of results to make a decision. Within these processes, there are several forms, such as a/b tests, canaries, and others. However, in order to get a better interpretation of these results, it is necessary to have in the process a member of the experiment team who plays the role of the Data Scientist who can with his abilities, join those results with data from several sources and then treat him/the and then explore their skills in Mathematics, Statistics to generate accurate information, knowledge, and intelligence. Therefore incorporating this figure in this method is extremely relevant in the process.

7.3 Problem

How to introduce the data scientist and their skills into the process of continuous experimentation?

7.4 Forces

—Team member with key skills:
- —The communication
- —Structured Data Management
- —Mathematics
- —Project Management
- —Data Mining and Visualization

—the Design of Experiments
—Data Management
—Product Design and Development
—Statistical modeling
—Business development
—Knowledge of techniques in Artificial Intelligence and Machine Learning

—Obtain results and interpret them to transform them into metrics to better measure the experiment.

—Increase the possibility of the relationship with other results to obtain new qualitative and quantitative metrics.

—Store the results in time-space and generate histories to obtain in the future a temporal analysis of each experiment and increase the possibility of relating other information to obtain knowledge;

—Apply the KDD (Knowledge Discovery in Databases) from the results of each experiment to create relationships of interest that are not observed by the subject specialist, thus assisting in the validation of the extracted knowledge. KDD is a general concept of a knowledge extraction process from databases, created in 1989. According to [Fayyad et al. 1996], "KDD is a multi-step, non-trivial, interactive and iterative process for identifying comprehensible, valid, new and potentially useful patterns from large data sets".

7.5 Solution

To train a professional specifically in the above-mentioned skills in order to obtain knowledge of the results obtained in the experiments and to help in the extraction of patterns and generate knowledge to aid in the decision making.

7.6 Consequences

(+) Historical data to obtain better results in the analysis of information;
(+) Temporary storage to obtain better results in the analysis of the information;
(+) Extraction of patterns using artificial intelligence algorithms;
(-) If there is no Data Scientist who is a specialist in the business, you will need to hire or train;
(-) Learning in skills takes time and increases the cost of the project;
(-) Learning to know how to apply artificial intelligence algorithms;
(-) For better results and better know it is necessary to train some team member or to know the KDD process.
(-) Database (with data redundancy) causes problems in the KDD process;

7.7 Known uses

The use of tools such as Sipina[1] or Weka[2] can aid in decision making. These tools contain implementations of artificial intelligence algorithms to obtain standards and knowledge of the data. Currently, there are several tools that implement integrated environments that allow the execution of the operational steps of KDD, here follows a table that presents some of these tools. Table I presents some operational tools for KDD, according to [Passos and Goldschmidt 2005].

8. CONTINUOUS EXPERIMENTATION KNOWLEDGE BASE

8.1 Motivational Example

Consider a company that uses continuous experimentation and performs numerous experiments on the product/service during their development life cycle. These experiments need to be documented to have a knowledge

[1] http://eric.univ-lyon2.fr/ ricco/sipina.html
[2] https://www.cs.waikato.ac.nz/ml/weka/

Table I. Operational tools for KDD. Adapted from: [Passos and Goldschmidt 2005]

Name	KDD Tasks	Source
SPSS/Clementine	Classification, Association Rules, Clustering, Sequential Pattern Analysis, and Desviation Detection.	SPSS Inc. (www.spss.com)
Poly/Analyst	Classification, Regression, Association Rules, Clustering, Summarization, and Desviation Detection.	Megaputer Intelligence (www.megaputer.com)
Weka	Classification, Regression, Association Rules, and Clustering.	University of Waikato (www.cs.waikato.ac.nz/ml/weka)
Intelligent Miner	Classification, Association Rules, Clustering, Summarization, and Sequential Pattern Analysis.	IBM Corp. (www.ibm.com)
WizRule	Classification, Summarization, and Desviation Detection	WizSoft Inc. (www.wizsoft.com)
Bramining	Classification, Regression, Association Rules, and Summarization.	Graal Corp. (www.graal-corp.com.br)
SAS Enterprise Miner	Classification, Regression, Association Rules, and Summarization.	SAS Inc. (www.sas.com)
Oracle Data Mining	Classification, Regression, Association Rules, Clustering, and Text Mining.	Oracle (www.oracle.com)

base of all experiments, types of experiments, target audience and the results. Otherwise, similar experiments can be implemented in other applications with reduced resources and time.

8.2 Context

On the continuous experimentation, there's a diversity of tests types, as for example, A/B test, canary, and others which are often performed. Not all experiments are successful, but the importance of their results and how they were made is equally important to the life cycle of a product/service and needs to be documented.

This systematization of the experiments allows improvements in a product/service to be made in a systematic way and not through unsupported assumptions and opinions. This is done from data and feedback from the client/user, avoiding unnecessary risks in the decision making, thus creating a KNOWLEDGE BASE, i.e., a history of experimentation to assist future experiments.

8.3 Problem

How to document the tests made in continuous experimentation, in order to make them easy to understand, retrieve and also detailed?

8.4 Forces

—Not every experiment has positive results or is well executed, but its documentation is important;

—If an experiment fails, it is good to know of its occurrence, how it occurred and the user feedback;

—Not every experiment has direct user feedback, part is obtained through usage data;

—The register of the experiments must not only have the results, but also the modifications and/or addition of functionalities and the stipulated metrics as an expected result;

—In addition to the results of the experiment, the costs for its realization and the material and work resources used must be recorded;

—Decisions based on the experiments should be recorded and whether future results were satisfactory;

8.5 Solution

Document the experiments and their results, in order to create a KNOWLEDGE BASE for the Continuous Experimentation (CE) of a system, which allows analyzing failed experiments (that it was not possible to

validate its results), misused (who did not get the expected results) or successfully, their results and the decisions taken. The pattern KNOWLEDGE BASE store all instances of the continuous experimentation process.

The following information should be stored in the KNOWLEDGE BASE:

—Experiment Name: Brief title that describes the experiment;

—Assumption: represents an idea that is accepted as truth without proof;

—Hypothesis: The hypothesis explains a phenomenon;

—Experiment Plan: is the planning of how to test a hypothesis, including how the test was done, experiment object (represents critical aspects of the product that will be tested), collect (results of the collected experiments as qualitative data as use data or quantitative as interviews, observations and surveys) and data analysis (verification for validation of the stipulated hypothesis);

—Experiment Object: represents critical aspects of the product that will be tested;

—Testers: are the audience that will participate in the experiments;

—Metrics: represent the capture of the values of the product or resource at a specific time of data collection;

—Success criteria: analyze the metric that allows verifying if the hypothesis is correct;

—Decision Taken: the decision to be made based on the data. Can apply the idea, to rethink because it needs to be better worked or cancel because there is no point to continue.

The documentation of the experiment should be recorded as follows in this example:

—Experiment Name: Choice of related products in the shopping cart;

—Assumption: Recommending different products in relation to the shopping cart generates more purchases than similar products;

—Hypothesis: We believe that offering related products from a different category will have a 20% increase in sales with more than one item;

—Experiment Plan: An experiment will be carried out in two weeks in which products of a different category will be offered and will measure the sales in that period;

—Experiment Object: Section of related products on the sales site;

—Testers: Customers with a registration with more than two years in the site;

—Metrics: Sales with products offered in the same category compared to sales with products of different categories carried out in the period of two weeks;

—Success Criteria: Sales increase of 20%;

—Decision Taken: based on the 23% increase, the idea seems to be good and should be applied to a larger audience in a new experiment;

8.6 Consequences

(-) More time is needed to document the experiments.

(+) It is possible to analyze the history of experiments performed on a system and to know the decisions taken during the lifetime of the system.

(+) Check for successful experiments and make new ones based on them;

(+) Avoid repeating failed experiments;

(-) Discourage repeating experiments that have not been successful in the past, but which could be interesting at the present time.

8.7 Known uses

Tools such as VALUE ([Munezero et al. 2017]), use visualization mechanisms that aid in decision making, as shown in the Figure 3.

Fig. 3. Value [Munezero et al. 2017]

9. CHOOSE THE RIGHT PEOPLE

9.1 Motivational Example

Continuous experimentation (CE) in a business is primordial for growth and continuous improvement aspects such as Amazon and Microsoft. For this purpose, the selection of people who dominate specific subjects in each stage of the project is necessary so that the obtained results are in agreement with the target public. The wrong choice of people to participate in the experimentation project, causing an unnecessary expenditure of resources and increased risk in decision making.

9.2 Context

Present in the stages of continuous experimentation the need for qualified people at each stage of the experiment. To perform experiments, it takes people with different abilities to assist in the experiment in a positive way. Research within the process of continuous experimentation in organizations depends not only on structured tools and programs, but also on people capable of performing tasks and analyzing their processes in a qualitative way.

9.3 Problem

Among the stages of continuous experimentation, it is common to segment the processes to achieve the proposed objective effectively. For this, it is necessary to choose the right people for the whole process of experimentation. How to identify and select people with sufficient skills to assist in the experiment?

9.4 Forces

—Professionals who define the experiments and choose the people participating in them must also have a high knowledge of the project in all stages and methodologies of experimentation for assertive choice;

—Not all companies have internal teams with competence to participate in the experimentation process;

—Choosing the right people for the experiment is key to aggregating and achieving the objectives of the experiment;

—The measurement of the results obtained more positive and constructive for the data collection, analysis and treatment in the experiment;

—Comparisons with new teams and people outside the organization can assist in the process of measuring results.

—People who do not have expertise in the experimental area may not contribute to the success of experimentation.

9.5 Solution

Choose groups of people internal and external to the organization, based on the specific domain, leveling of knowledge and considering skills that can contribute to each phase of the experiment.

9.6 Consequences

(+) The experiment passes by competent people to perform specific tasks of their knowledge, making the results more efficient and objective.

(+) It will be possible to analyze qualitative data, making the process of decision making agiler.

(+) The interpretation of the experiment as well as the research data becomes more appropriate and focused on the experiment;

(+) It may contribute to greater internal competition among professionals for the participation of experiments according to their individual abilities.

(-) People who are not selected to participate in the experiment team may feel harmed or belittled.

9.7 Known uses

As an example the Multi-case study on the "Open-loop" closure, which uses the basis of data collection with interviews and workshops to identify specific groups in each process of the experiment in software analysis.

10. CONCLUSION

Pattern identification occurs by not finding new solutions to a problem but by abstracting existing solutions from the problem so that it can then be used in various contexts. By analyzing three common factors in the solutions, we identify common features about these three perspectives and develop the patterns from there:

—Context: It corresponds to the situations in which the pattern can be applied;

—Problem: What the pattern tries to solve;

—Solution: The steps to solve the problem;

In Continuous Experimentation, we identify common processes that occur in most situations and are important for the correct application of the experimentation process. Such processes should be considered in the deployment of a company/organization and have been described here by the patterns described in the article. Are they:

—Choice of team that conducted the experiment (Choice of Right People);

—Team training in a step-by-step manner (Choose the Right People)

—Analysis of the decision to be made based on the results of the experiment (Continuous Experimentation Best Possible Way)

—Need for a Data Scientist to extract knowledge from the results of experiments (Skills for Experimentation Data Scientists)

—Creation of Knowledge Base with the history of the experiments and their results (Continuous Experimentation Knowledge Base)

These patterns encompass all Continuous Experimentation and can be applied both together, having their solutions aiding the other standards, or as independently, where only the standard required for a particular stage of experimentation will be used.

REFERENCES

Eytan Bakshy, Dean Eckles, and Michael S. Bernstein. 2014. Designing and Deploying Online Field Experiments. *CoRR* abs/1409.3174 (2014). http://arxiv.org/abs/1409.3174

Jan Bosch and Ulrik Eklund. 2012. Eternal Embedded Software: Towards Innovation Experiment Systems. In *Leveraging Applications of Formal Methods, Verification and Validation. Technologies for Mastering Change*, Tiziana Margaria and Bernhard Steffen (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 19–31.

Aleksander Fabijan, Pavel Dmitriev, Helena Holmström Olsson, and Jan Bosch. 2017. The Evolution of Continuous Experimentation in Software Product Development: From Data to a Data-driven Organization at Scale. In *Proceedings of the 39th International Conference on Software Engineering (ICSE '17)*. IEEE Press, Piscataway, NJ, USA, 770–780. DOI:http://dx.doi.org/10.1109/ICSE.2017.76

Fabian Fagerholm, Alejandro Sanchez Guinea, Hanna Mäenpää, and Jürgen Münch. 2014. Building blocks for continuous experimentation. In *Proceedings of the 1st international workshop on rapid continuous software engineering*. ACM, 26–35.

Fabian Fagerholm, Alejandro Sanchez Guinea, Hanna Mäenpää, and Jürgen Münch. 2017. The RIGHT model for Continuous Experimentation. *Journal of Systems and Software* 123 (2017), 292 – 305. DOI:http://dx.doi.org/https://doi.org/10.1016/j.jss.2016.03.034

Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. 1996. From data mining to knowledge discovery in databases. *AI magazine* 17, 3 (1996), 37.

Federico Giaimo and Christian Berger. 2017. Design Criteria to Architect Continuous Experimentation for Self-Driving Vehicles. *2017 IEEE International Conference on Software Architecture (ICSA)* (2017), 203–210.

David Issa Mattos, Jan Bosch, and Helena Olsson. 2018. Challenges and Strategies for Undertaking Continuous Experimentation to Embedded Systems: Industry and Research Perspectives. (05 2018).

Natalia Juristo and Ana M. Moreno. 2010. *Basics of Software Engineering Experimentation* (1st ed.). Springer Publishing Company, Incorporated.

Katja Kevic, Brendan Murphy, Laurie Williams, and Jennifer Beckmann. 2017. Characterizing experimentation in continuous deployment: a case study on bing. In *Proceedings of the 39th International Conference on Software Engineering: Software Engineering in Practice Track*. IEEE Press, 123–132.

Ron Kohavi, Alex Deng, Brian Frasca, Toby Walker, Ya Xu, and Nils Pohlmann. 2013. Online Controlled Experiments at Large Scale. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '13)*. ACM, New York, NY, USA, 1168–1176. DOI:http://dx.doi.org/10.1145/2487575.2488217

Ron Kohavi, Alex Deng, Roger Longbotham, and Ya Xu. 2014. Seven Rules of Thumb for Web Site Experimenters. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, New York, NY, USA, 1857–1866. DOI:http://dx.doi.org/10.1145/2623330.2623341

Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M. Henne. 2009. Controlled Experiments on the Web: Survey and Practical Guide. *Data Min. Knowl. Discov.* 18, 1 (Feb. 2009), 140–181. DOI:http://dx.doi.org/10.1007/s10618-008-0114-1

Thomas Kuhn. 1981. *What are scientific revolutions*. Boston: MIT.

Eveliina Lindgren and Jürgen Münch. 2016. Raising the odds of success: the current state of experimentation in product development. *Information and Software Technology* 77 (2016), 80–91.

Daniel Lucrédio, Eduardo Santana De Almeida, Alexandre Alvaro, Vinicius Cardoso, and Eduardo Kessler Piveta Garcia. 2004. Student's PLoP Guide: A Pattern Family to Guide Computer Science Students during PLoP Conferences. In *Proceedings of the Latin American Conference on Pattern Languages of Programs (SugarLoafPLoP)*.

Myriam Munezero, Sezin Yaman, Fabian Fagerholm, Petri Kettunen, Hanna Mäenpää, Simo Mäkinen, Juha Tiihonen, Leah Riungu-Kalliosaari, Antti-Pekka Tuovinen, Markku Oivo, and others. 2017. Continuous Experimentation Cookbook. (2017).

Helena Olsson and Jan Bosch. 2014a. The HYPEX Model: From Opinions to Data-Driven Software Development. (08 2014), 155–164.

Helena Holmström Olsson and Jan Bosch. 2014b. Climbing the "Stairway to Heaven": Evolving From Agile Development to Continuous Deployment of Software. In *Continuous Software Engineering*. Springer, 15–27.

Emmanuel Passos and Ronaldo Goldschmidt. 2005. Data Mining: a practical guide. *Editora Campus, Rio de Janeiro (in Portuguese)* (2005).

O. Rissanen and J. Münch. 2015. Continuous Experimentation in the B2B Domain: A Case Study. In *2015 IEEE/ACM 2nd International Workshop on Rapid Continuous Software Engineering*. 12–18. DOI:http://dx.doi.org/10.1109/RCoSE.2015.10

Gerald Schermann. 2017. Continuous experimentation for software developers. In *Proceedings of the 18th Doctoral Symposium of the 18th International Middleware Conference*. ACM, 5–8.

Mojtaba Shahin, Muhammad Ali Babar, and Liming Zhu. 2017. Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices. *CoRR* abs/1703.07019 (2017). http://arxiv.org/abs/1703.07019

Diane Tang, Ashish Agarwal, Deirdre O'Brien, and Mike Meyer. 2010. Overlapping Experiment Infrastructure: More, Better, Faster Experimentation. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '10)*. ACM, New York, NY, USA, 17–26. DOI:http://dx.doi.org/10.1145/1835804.1835810

Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in software engineering*. Springer Science & Business Media.

Sezin Gizem Yaman, Myriam Munezero, Jürgen Münch, Fabian Fagerholm, Ossi Syd, Mika Aaltola, Christina Palmu, and Tomi Männistö. 2017. Introducing continuous experimentation in large software-intensive product and service organisations. *Journal of Systems and Software* 133 (2017), 195–211.