The Benefits of Controlled Experimentation at Scale

Aleksander Fabijan
Malmö University
Dep. of Computer Science
Malmö, Sweden
aleksander.fabijan
@mah.se

Pavel Dmitriev
Microsoft, Analysis &
Experimentation
Redmond, USA
padmitri
@microsoft.com

Helena Holmström Olsson
Malmö University
Dep. of Computer Science
Malmö, Sweden
helena.holmstrom.olsson
@mah.se

Jan Bosch
Chalmers University of Tech.
Dep. of Computer Science
Göteborg, Sweden
jan.bosch
@chalmers.se

***Abstract*— Online controlled experiments (for example A/B tests) are increasingly being performed to guide product development and accelerate innovation in online software product companies. The benefits of controlled experiments have been shown in many cases with incremental product improvement as the objective. In this paper, we demonstrate that the value of controlled experimentation at scale extends beyond this recognized scenario. Based on an exhaustive and collaborative case study in a large software-intensive company with highly developed experimentation culture, we inductively derive the benefits of controlled experimentation. The contribution of our paper is twofold. First, we present a comprehensive list of benefits and illustrate our findings with five case examples of controlled experiments conducted at Microsoft. Second, we provide guidance on how to achieve each of the benefits. With our work, we aim to provide practitioners in the online domain with knowledge on how to use controlled experimentation to maximize the benefits on the portfolio, product and team level.**

Keywords— 'controlled experimentation', A/B testing, 'leading metrics', 'lagging metrics', 'data-driven development'.

I. Introduction

Software development organizations and their product development teams are increasingly using customer and product data to support their decisions throughout the product lifecycle [1], [2]. Data-driven companies acquire, process, and leverage data in order to create efficiencies, iterate on and develop new products, and navigate the competitive landscape [1]. This is reflected in the recent software engineering research with many publications on how to change and efficiently conduct controlled experiments (A/B tests) in product management to drive incremental and continuous product improvements [3]–[10]. Although there are several ways in which software companies can use customer and product data [2], correctly conducted controlled experiments create a more accurate understanding of what customers value and provide sufficient evidence for concluding inference (for example, drawing a conclusion that a key metric has improved due to a particular change in a product) [3].

The benefit of Controlled Experimentation (CE) on continuous product improvement has been demonstrated a number of times both by research in the academia [2], [6], [8], and in the industry by Microsoft [11], [12], Google [13], Facebook [14] and other software-intensive companies [12], [15]. At Microsoft, and on top of several other benefits, hundreds of millions of dollars of additional revenue annually is generated as a result of controlled experimentation [3]. The impact of experimentation, however, extends beyond the direct revenue gain. CE fundamentally changes the way how the R&D work in product teams is planned, prioritized and measured.

In this paper, we provide a case study conducted at Microsoft in which we observed that the benefits of controlled experimentation at scale are in fact threefold. First, on the product portfolio level, CE aids in accurately identifying what constitutes customer and business value, and how the work of product teams contribute to it. Second, CE impacts development on the product level where decisions on product functionality, complexity, quality and infrastructure needs are made. Third, and what can fundamentally be improved by CE is at the team level where R&D effort is planned, executed and measured. We illustrate our findings with five example experiments conducted at Microsoft. With our contribution, we aim to motivate practitioners in software companies in the online domain to extend their data-driven practices and provide them with guidance on (1) where and (2) how to benefit from CE.

The paper is organized as follows. In section II we present the background and the motivation for this study. In section III, we describe our research method. We show illustrative examples of our empirical data in section IV. In section V, we present our main contribution – the benefits of controlled experimentation on the portfolio, product and team level. In section VI, we list a few challenges with CE. Finally, we conclude the paper in section VII.

II. Background

Software companies have always been evolving their product development practices. At first, they typically inherit the Agile principles on an individual development team level [16] and expand these practices across the product and other parts of the organization [17]. Next, companies focus on various lean concepts such as eliminating waste [18], removing constraints in the development pipeline [19] and advancing towards continuous integration and continuous deployment of software functionality [15]. Continuous deployment is characterized by a bidirectional channel that enables companies not only to deliver new updates to their customers in order to rapidly prototype with them [20], but also to collect feedback on how these products are used. By evaluating ideas with customers, companies learn about their preferences. In this process, the actual product instrumentation data (for example, features used in a session) has the potential to identify improvements and make the prioritization process of valuable features more accurate [21]. As it focuses on what customers do rather than what they say, it complements other types of feedback data [22] and improves the understanding of the value that the product provides [23], [24]. For example, knowing which features are used in a certain context helps companies in identifying customer preferences and possible bottlenecks. The intuition of software development companies on customer preferences, can be as much as 90% of the time inaccurate [25]–[27]. To identify the 10% and reliably evaluate ideas with customers, CE is becoming the norm in progressive and large software companies [3]–[8].

A. *Online Controlled Experimentation*

In a standard online controlled experiment (also known as A/B test or randomized trial), users of a software product are randomly distributed between several variants (for example, the two different designs of a product interface A and B) in a persistent manner (a user receives the same experience multiple times). User interactions with the product (e.g. clicks, dwell times, etc.) are instrumented and key metrics (e.g. engagement, task success, etc.) are computed [28]. The differences between the experiment variants (A/B/n) are evaluated on a metrics level to identify which of the versions yields improvements to key metrics that the product and portfolio team strive to impact [3], [29]. In case all the variants are identical, the experiment is known as an A/A test. Standard controlled experiments are typically used in situations where features are added or altered. Special cases of an A/B experiment are experiments where the variants are identical, known as A/A tests, and experiments where instead of adding a new feature to the product, an old feature is removed, known as 'reverse experiments'. In both cases, and to evaluate the results of the experiments, relevant product metrics are computed. We briefly describe their role in the evaluation of experiments next.

B. *Metrics and Value*

The importance of metrics as a mechanism to measure behavior has been recognized and studied in the management domain more than fifty years [30]. Metrics are a powerful tool used in organizations to set team goals, decide which new products and features should be released to customers, and how resources should be allocated [31], [32].

Ideally, a product team should identify and formalize a set of metrics for every product that describes long-term goals. These metrics are commonly labelled as 'Overall Evaluation Criteria' (OEC) [33]. An OEC should represent both the *business value* (i.e. a measure of changes that result in an increase of the revenue), and *customer value* (i.e. a measure evaluating customer satisfaction with the product). Dmitriev and Wu [31] have recently demonstrated how to use past "experimentation corpus", i.e. learnings from past experiments, to evaluate the sensitivity and other relevant parameters of metrics, to help practitioners in selecting suitable OEC metrics.

To identify metrics that are key to the business can be challenging [31], [34]. It can be intuitive to measure and consider short-term effect in OEC metrics [35]. For example, providing more weight to advertisement metrics increases short-term business value and creates business more profitable momentarily. On the long term, however, showing more advertisement may annoy users and increase churn to a competitor with less advertisement, leading to lower customer value. Since the short-term effect is not always predictive of the long-term effect, it should not be the sole component of an OEC [36]. Therefore, and to construct quality metrics, product and portfolio teams in software companies should differentiate between leading and lagging indicators and understanding the mapping between them.

C. *Leading and Lagging Indicators*

Motivated by the unpredictability of the market, researchers in the economics disciplines introduced a distinction between two types of indicators [37], [38]. A '*leading indicator*' is a measurable metric that changes within a short period after a change of conditions occurred, typically following a pattern or a trend. This type of indicators are predictive and lead to a goal that we can directly influence [39]. A 'lagging indicator' is an indicator that changes with a delay, typically after several other 'leading' indicators have already changed. In economic terms, it is a goal with consequence and value [39]. It consists of a performance gap (e.g. from X to Y) and a time (e.g. by 2020).

Companies should focus in setting product team goals on lead measures and avoid lagging indicators for performance goals. Studies indicate that committing to an achievable goal help improve individual performance and that setting challenging and specific goals can further enhance employee engagement in attaining those goals [40]–[42]. In practice, however, it is very difficult to identify which lead indicators are predictable of which lag indicators [3], [31], [35]. In our work, and as one of the benefits identified in this paper, we demonstrate that CE enables R&D teams in creating a mapping between 'leading' and 'lagging' indicators. This has implications for product development. It enables agile teams to focus their work on development activities that improve low-level 'leading indicators' which will in turn result in a definitive improvement of a high-level, and typically business value focused, 'lagging indicator'.

III. Research Method

In this paper, we report on an inductive case study, which was conducted in collaboration with the Analysis and Experimentation (A&E) team at Microsoft. The study is based on historical data points that were collected over a period of two years and complemented with a series of semi-structured interviews, observations, and meeting participations. In principle, it is an in-depth and single case study [43], however, our participants are from different organizational units and product teams with fundamentally different product and service offerings. Several of the participants worked in other data-driven organizations before joining A&E team. The A&E team provides a platform and service for running tens of thousands of controlled experiments to customers across almost all major Microsoft products. Its data scientists, engineers and program managers are involved with partner teams and departments across Microsoft daily. The participants involved in this research work are primarily collaborating with Bing, OneNote, Office, MSN.com, Xbox and Skype. The five experiments that we present in section IV are illustrative and representative examples of the work that our participants are involved with at Microsoft.

A. *Data Collection*

The data collection for this research was conducted in two streams. First, we collected archival data on past controlled experiments conducted at Microsoft. The first author of this paper worked with Microsoft Analysis & Experimentation team for a period of 10 weeks. During this time, he collected documents, presentations, meeting minutes and other notes available to Microsoft employees about the controlled experiments, the development of the experimentation platform and organizational developments conducted at Microsoft A&E over the last two years. In total, we collected approximately 130 pages of qualitative data (including several figures and illustrations).

The second stream consisted of three parts. The first author (1) participated on weekly experimentation meetings, (2) attended internal trainings on controlled experimentation and other related topics, and (3) conducted several semi-structured interviews with Microsoft employees. In all the three data collection activities, the first author was accompanied by one of the other three authors (as schedules permitted).

The second author of this paper has been working with the Analysis & Experimentation team at Microsoft for a period of six years and he has extensive experience with controlled experimentation. He was the main contact person for the other three researchers throughout the data collection and analysis period and advised the diverse selection of data scientists, managers and software engineers that we interviewed. In total, we conducted 14 semi-structured interviews (1 woman, 13 men) using a questionnaire with 11 open-ended questions. The participants that work with different product teams were invited for a half an hour interview. The interview format started with an introduction and a short explanation of the research being conducted. Participants were then asked on their experience with CE, benefits, pitfalls, and the pain points that they experience while conducting controlled experiments.

B. *Data Analysis*

To obtain our learnings, we first organized data that belongs to individual experiments together. In total, we analyzed and grouped the data that relates to over 300 controlled experiments. These data were both qualitative descriptions of individual experiments as well as quantitative data related to individual metrics or metric groups. Next, we read the information about every experiment word by word and extracted commonalities and the differences between the experiments, focusing on the experiment learnings, expected outcomes and unexpected benefits. By extracting these, we constructed codes and emerged with individual benefits. This technique is similar to thematic coding [44] and can be used to identify a theory where existing knowledge is limited. All of our interpretations were continuously discussed with the A&E team to clarify any misunderstandings.

C. *Validity Considerations*

To improve the study's *construct validity*, we complemented the archival data collection activities with individual semi-structured interviews, meetings and trainings. This enabled us to ask clarifying questions, prevent misinterpretations, and study the phenomena from different angles. Meeting minutes and interview transcriptions were independently assessed by three researchers. The first author conducted this research while collaborating with the second author who is permanently employed at the case company as a Principal Data Scientist. This set-up enabled continuous access and insight with the relative stakeholders. However, and since this approach differs from a traditional case study [43], the contributions of this paper risk of being biased from this extensive inside view. Considering *external validity*, our study involved sub-cases that focus their work on products close to the online domain. And although Microsoft is one company, we emphasize that our sub-cases (for example Bing, Skype, Office etc.) within the longitudinal study are diverse and could itself be considered their cases. Each of them has from several hundred to several thousands of employees. At the same time, several of the engineers, managers and data scientists that provided feedback on our work have experience with CE from other large software companies (specifically from Amazon, Google and Facebook). Taking this into consideration, we believe that the results of our study can apply to other large and software-intense companies that adopt the controlled experimentation methodology at scale.

In the next section, we present five illustrative examples of controlled experiments from our collection of empirical data.

IV. Empirical Data and Findings

Microsoft conducts tens of thousands of controlled experiments every year across dozens of products. In this section, we present five of these experiments in greater detail to give a flavor of the wide variety of experiments and their benefits. The space limitations and proprietary concerns make it difficult to show all the depth and breadth of our empirical data. Due to these limitations, we select the illustrative experiments that are diverse and yield several benefits, and present them in detail below.

The first experiment with the Office Contextual Bar highlights the use of CE for discovery of instrumentation issues and the need for focusing the experimentation on leading metrics to identify the exact amount of impact on the value of the product. The second experiment illustrates how CE ensures software quality. Next, we show an example experiment with a birthday animation where CE enabled the product team to increase the customer and business value of Bing search page. Next, we present a design experiment with the Bing website where the effect of changing a background color was identified as negative. We conclude the section with an experiment from Skype where CE predicted infrastructure capacity.

A. *Office Contextual Command Bar Experiment*

Description: Microsoft Office for mobile is a well-known suite of products designed to increase work productivity. The product team responsible for edit interface in Office mobile apps conducted a design experiment with their Word mobile app. They believed that introducing a Contextual Command Bar (see Fig. 1 below) would increase the engagement (e.g. the number of times the users of the app edit their documents). **Hypothesis:** The hypothesis was that mobile phone users will do more editing on the phone because the contextual command bar will improve editing efficiency and will result in increased commonality and frequency of edits and 2-week retention. **Outcome:** During the set-up of the experiment, the product team experienced issues with measuring the number of edits. After a few rounds of troubleshooting, they identified that the telemetry implementation did not accurately log the edit events in the instrumentation pipeline, which resulted in inaccurate interpretation of the initial results. They resolved the instrumentation issues and confirmed this with an A/A test (exposing the control and the treatment group with the same version of the product). The results of a following two-week controlled experiment indicated a substantial increase in engagement (counts of edits), but no statistically significant change in 2-week retention metric.

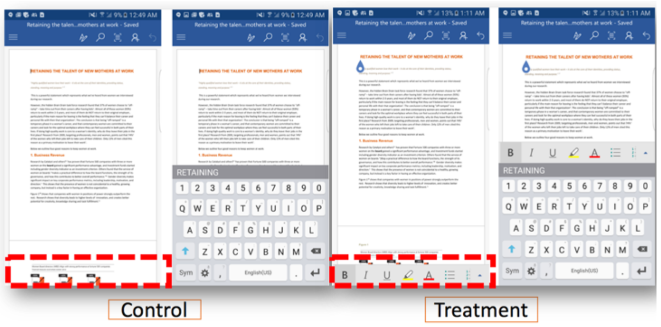

Fig. 1. Office Contextual Bar Experiment.

Learning: The experiment provided the team with two key learnings: (1) Proper instrumentation of existing features and the new feature is essential for computing experiment metrics, and (2) defining metrics that are good leading indicators and that they can expect to see a change in a reasonable timeframe is important. **Benefits:** Because of this experiment, the product team (1) resolved an instrumentation issue and (2) identified the exact impact of introducing this type of design change to one of their products on a leading metric. These learnings were strong evidence to generalize the knowledge and prioritize a similar change to the other products in their portfolio.

B. *OneNote Publishing Pane Experiment*

Description: Microsoft OneNote is an information gathering and multi-user collaboration application. In one of the experiments with this application, the product team experimented with a modification to the 'Share Pane'. **Hypothesis:** The OneNote team suspected many people wanted to share only a page of a notebook, and that the new feature and user interface would increase overall sharing. With the new experimental feature, the user is given the option to publish a page, rather than the entire notebook. (see Fig. 2).

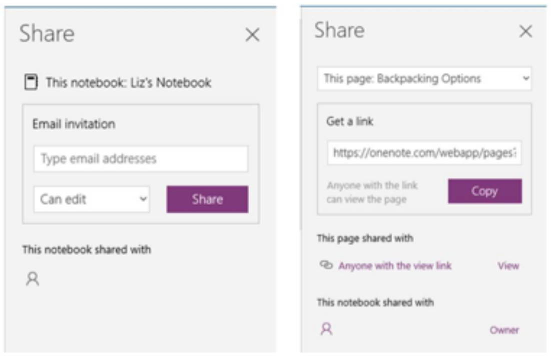
Share
This notebook: Liz's Notebook
Email invitation
Type email addresses
Can edit
Share
This notebook shared with
Share
This page: Backpacking Options
Get a link
https://onenote.com/webapp/pages
Anyone with the link can view the page
Copy
This page shared with
Anyone with the view link
View
This notebook shared with
Owner

Fig. 2. OneNote Share Pane experiment.

Outcome: The controlled experiment indicated positive results on the key leading metrics. The fraction of users that succeeded in progressing throughout the sharing funnel—defined as creating a share link or successfully sending a share e-mail – increased. Users in the treatment group also created fewer new notebooks. This suggested that some users that previously wanted to share a subset of a notebook's content were going through the trouble of creating a brand-new notebook with that subset of content. The new feature clearly frees these users from going through that trouble. While overall the results were positive, the team saw a mismatch in the number of users in control and treatment for a small subgroup of users using a specific older OS version. **Learning:** By examining the experiment in detail, the team learned that instrumentation data was lost when the application crashed during the experiment on the older OS version. This caused unusual movements in metrics and the user mismatch. By investigating this experimental alert, the team discovered that a software bug in the new feature caused a crash in the code path related to the new feature. **Benefits:** With this experiment, the product team obtained an important learning. CE can help detect critical bugs that only appear with heavy usage in production, and that experimentation can help validate that a new feature or user interface helps customers. Although products under development traverse rigorous testing steps, many issues become visible only in a production environment, and controlled experiments are a great way to catch them before they hit all users of the product. As a benefit of detecting and mitigating such issues, product teams at Microsoft ensure better quality of their software.

C. Bing Search Results Birthday Greeting Experiment

Description: In an experiment on Bing results page, a product team evaluated the impact of a feature that displays an animated birthday greeting for registered users on the day of their birthday anniversary (see Fig. 3). **Hypothesis:** With this experiment, the team aimed to promote the feature on Bing search result page to users whom Bing knows they have birthday on the current day. The Bing team hypothesized that this feature would increase usage of Bing and will not introduce any harm. **Outcome:** After analyzing the results of the experiment, the team learned that several percent of users who saw the promotion, engaged with it. Since this was a good engagement rate, they decided to deploy the features. The analysis of the offline data, however, showed that there was an abnormal distribution of birthdays on the first of January, likely most of which are not true birthdays.

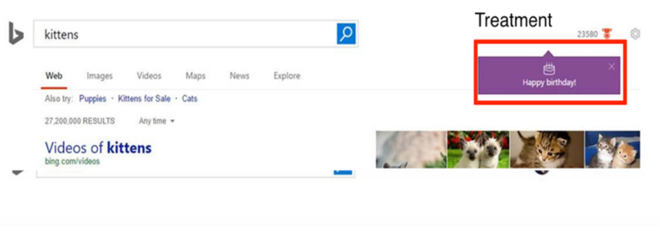

Fig. 3. Bing Search Results Birthday Greeting Experiment.

Learning: The product team considered not showing the birthday experience for users with birthday anniversaries on the first of January. However, the solution they selected was to show it anyway, but increase the prominence of a link stating "Not your birthday?". This link directs the users on a page to correct their birthday. The results of a follow-up experiment indicated that about 1/5 of users who receive the birthday wish on the first of January initiate the update of their birthday. **Benefits**: The team (1) improved their product and (2) increased its customer and business value. And even though the experiment didn't yield the exact expected result, valuable learnings were obtained leading to a new improved feature eventually deployed. We illustrate this with a quote:

"This is a general pattern with experimentation; finding a surprise that is still valuable."

– Principal Data Scientist

D. Bing Background Color Experiments

Description: Bing team conducted controlled experiments with two different background color variations: light grey and dark grey, and compared them to the regular white background. Their goal was to identify the effect of changing the background color on the key metrics. **Hypothesis:** Bing team hypothesized that experiments of this type will be successful and that the team should further invest into optimizing the background color of the product. **Outcome:** The experiments indicated a strong degradation in key metrics for both variants. **Learning:** In this experiment, the product team learned that (1) product changes of this type should not be deployed to the overall user base, and (2) further investment into optimizing background colors should not be prioritized. **Benefits**: The teams benefited from this by identifying that (1) changing background color in relation to the regular white version yields negative impact on the key business value metrics, ad that they should (2) prioritize other design ideas in the future. In general, there are many possible user interface treatments that could be attempted, nearly infinite in fact, and this method of deciding which to pursue (and avoiding ones that might cause harm) is valuable across multiple products in the portfolio.

E. Skype Infrastructure Migration

Description: Skype is a well-known product for video and audio communication. Recently, the team completed migration of the product to a new call-routing algorithm. **Hypothesis:** Among other effects, the team hypothesized that enough infrastructure capacity will be available to support the new approach to call routing. **Outcome:** To evaluate their hypothesis, the Skype team conducted a controlled experiment with only a few percent of Skype users. The difference in load on the infrastructure between the new and the old algorithm was estimated, and projected to 100% of user population, confirming that the capacity will be sufficient. **Learning:** The Skype team learned that based on the effect of the limited treatment group, enough capacity is available within the new infrastructure to continue with the transition. **Benefits:** By identifying the exact effect on the infrastructure, the team was confident that the infrastructure capacity is within the range needed for the complete product migration.

V. The Benefits of Controlled Experimentation

In the previous section, we presented five illustrative examples of controlled experiments conducted at Microsoft. In this section, and based on the analysis of several hundreds of experiments, interview and workshop notes, we provide (1) a comprehensive list of benefits of controlled experimentation, and (2) guidelines on how to achieve them. We discuss these benefits in relation to example experiments from previous section IV.

In our work, we differentiate between three distinct levels of benefits. First, we describe how CE benefit the company portfolio level, which stretches across the organization and typically across several products. Second, we describe the benefits of CE on the product level. There, we show how CE controls product complexity, quality, instrumentation assurance, and how it can be used to predict infrastructure needs. Finally, and on the team level, we show that controlled experimentation impacts the planning of team activities and how it can be used to define measurable and realistic performance goals for teams. As illustrated with our examples from section IV, a single experiment can yield more than one benefit. We present each of the identified benefits and how to achieve them in the remainder of this section.

A. Portfolio Level

Value discovery and validation: Effective portfolio development and identification of what delivers value to customers is vital to successful product innovation [3], [29]. By conducting a controlled experiment, software companies can measure the exact amount of impact on the value from different features and products. In this context, controlled experimentation forces product development teams across multiple products in the portfolio to (1) define what the value is, and to (2) identify what changes across the product portfolio contribute to it. By conducting many controlled experiments and analyzing their outcomes, companies can learn what is valuable for the business, and what is valuable for the customers. The procedure to discover and define business and customer value is the following: (1) The portfolio team hypothesizes customer or/and business value, (2) value measurements are formalized in terms of leading metrics, (3) hypotheses are evaluated in multiple experiments across multiple products, and finally, (4) hypotheses that were accepted define customer or/and business value. Step (3) is crucial and typically consists of several learning experiments (e.g. intentional degradations) where leading metrics that indicated changes to lagging metrics in (2) are discovered (e.g. the effect of a performance slowdown on user satisfaction).
The Office Contextual Bar Experiment and the Birthday Greeting Experiment that we presented in section IV, for example, provided the Microsoft with knowledge of portfolio benefits about what customers of these products value. The learnings were validated and applied across the Microsoft portfolio. A team of data scientists is continuously working on improving the leading metrics, by analyzing past experiments and conducting new learning experiments to gain insights into metrics behavior and learn the mappings between 'lagging' and 'leading' indicators. Without CE, changes to metrics and correlations between metric movements are much harder to detect due to large variance, seasonality effects, and other factors, making establishing a clear relationship between 'leading' and 'lagging' indicators nearly impossible.

B. Product Level

Incremental product improvements: This is the most well known benefit of A/B testing in software product development and there are several publications available that discuss it in detail [11], [13], [29], [36]. Causality with regular and incremental product improvements (e.g. what was the effect of introducing a new feature on the key metrics) is proved by running a controlled experiment. And by deploying the changes that increase the key metrics, the product is incrementally improved. The way how incremental product improvements are done is the following: First, (1) instrumentation data (e.g. clicks per user session, dwell time, etc.) are collected during an experiment, (2) key metrics are calculated based on the instrumentation data, (3) Statistical tests are conducated to detect the difference between the variants (e.g. the version of the product without a new feature and the one with the new feature) is measured. Finally, (4) variants with improvements to the key metrics are deployed. This benefit is illustrated in several experiments in section IV.

Optimizing and predicting product infrastructure and capacity needs: When a product is incrementally updated (e.g. an algorithm is optimized) companies risk introducing an effect on the infrastructure. By using controlled experimentation, practitioners can be more proactive and guarantee that (1) necessary capacity will be available for their changes, and (2) estimate the capacity cost/savings of the change before it is fully deployed. This can be achieved by measuring the change in infrastructure load via an experiment on a small population, e.g. 10% vs 10%, and then project what will happen at 100% deployment. A gradual ramp-up of the experiment can be used to estimate non-linear effects.
Skype infrastructure migration experiment described in section IV, for example, enabled the product team to predict infrastructure needs based on a smaller treatment and continue the deployment.

Ensuring product quality: Although the changes to products are typically well tested using end-to-end and unit tests, certain scenarios in product changes require validation with production customers. By experimenting with customers in a gradual manner (e.g. starting with a low % in the treatment group and increasing exposure over time), the impact of a change that introduces product issues is limited to only a defined number of customers. As a side benefit, the investments into virtual testing infrastructure mimicking production environment are minimized. The risk of exposing product changes with defects that could not have been detected through unit and pre-production testing to large sets of customers are lower with controlled experimentation than they would be without it.
OneNote Share Pane experiment, for example, provided the evidence that product quality can be better assured using CE by, for example, alerting for instrumentation discrepancies during experiment execution.

Stabilizing and lowering product complexity: Features are developed because teams believe that they will be useful and provide value. Yet in many domains, most ideas fail to improve key metrics [3]. Only one third of the ideas tested on the Experimentation Platform at Microsoft improved the metrics they were designed to improve [3], [11]. CE enables product development teams to (1) deploy only the changes (e.g. new features) that deliver an increase in business or customer value, and (2) to remove those that do not.
The way in which this product benefit is achieved is twofold: (1) features that have no impact on the business or customer value (e.g. do not indicate a statistical significance in key metrics) are not deployed to customers. Second (2) by running

reverse experiments (e.g. a feature is hidden from the treatment group), product teams get an indication on how much value the removed feature adds to the product.

Product instrumentation quality assurance: As companies increasingly rely on instrumentation to measure their successes and failures, controlled experiments should regularly be performed to identify whether the metrics are up to date and react as it is expected from them. This benefit can be achieved in two different ways: (1) A/A tests should continuously be conducted (e.g. every time before starting an A/B experiment). Second (2) by conducting experiments with known outcomes, the instrumentation quality and logic can be verified by comparing the expected and actual outcome. At Microsoft, CE is used regularly to evaluate instrumentation [31].

C. *Team Level*

Team activity planning: Experiments provide learnings that consist of treatment and causal effect pairs. These learnings can be used to identify the development activities that should be prioritized (for example, the changes teams made that caused the most impact). By (1) knowing what changes improved the key metrics in the past and by how much, product development teams can (2) generalize and organize their schedule to focus their work on the type of changes that were shown to have the most beneficial effect on the product.

The Successful Office Contextual Bar Experiment enabled sibling teams working on related products to prioritize their work and capture customer and business value earlier than they would otherwise.

On the other hand, the Bing Background Color experiments yield the negative learning and was used to deprioritize similar development activities.

Defining performance goals for teams: With CE, product management can express team goals in terms of changes to 'leading metrics', and their progress can be monitored over time. For example, Bing's ranking team has their goal defined by a single metric, a leading indicator for lagging metrics such as long-term query share and revenue. The team's progress towards that goal is measured as a sum of the statistically significant changes to this metric from all the controlled experiments performed by the team over a period. Without CE, it is very difficult to accurately evaluate team progress since measuring the exact magnitude of impact from new features is hard, and the mappings between 'leading' and 'lagging' indicators are unknown.

We summarize the benefits that we described in this section and guidelines to achieve them in Table 1.

TABLE I. BENEFITS OF CONTROLLED EXPERIMENTATION AND GUIDELINES ON HOW TO ACHIEVE THEM.

	Benefits	Guidelines to achieve the benefit
Portfolio	Value discovery and validation	(1) Customer and Business value are hypothesized on a portfolio level. (2) Measurement of the value is formalized in terms of leading metrics. (3) Hypotheses are evaluated in multiple experiments across multiple products. (4) Hypotheses that were confirmed indicate value on a portfolio level.
Product	Incremental product improvements	(1) Instrumentation data of a single experiment are collected, (2) Metrics are calculated based on the collected data, (3) Statistical difference between variants is measured, (4) Variants with improvements to key metrics are deployed.
	Optimizing and predicting product infrastructure and capacity needs	(1) Change is deployed on a low % of treatment, (2) Changes in infrastructure are monitored, (3) Treatment group is gradually increased if resources allow.
	Ensuring product quality	(1) Product changes that degrade key metrics are not deployed.
	Stabilizing and lowering product complexity	(1) Product increments with no impact on the key metrics are not deployed. (2) Product features with no impact on the key metrics are removed with reverse experiments.
	Product Instrumentation quality assurance	(1) A/A experiments are conducted to identify noisy instrumentation. (2) Experiments with known outcomes validate instrumentation quality.

Team	Team activity planning	(1) Changes/features that improve key metrics are shared among teams. (2) Teams generalize learnings to identify feature areas that should be prioritized.
	Defining performance goals for teams	(1) By measuring the exact amount of impact that changes of one team had on the leading metrics over a period, a realistic goal can be set for the next period.

VI. CHALLENGES WITH CE

In the previous section, we demonstrated that the benefits of controlled experimentation at scale extend beyond the recognized scenario of incremental product improvements. To implement CE in a software company and experience the benefits, however, considerable effort and domain knowledge are required. In this section, and to contrast the positive implications of the benefits above, we present and discuss several challenges with CE.

Gradual realization of benefits: Several product teams at Microsoft experience the benefits that we present in Section V in their development process. The realization of them, however, is not instant, but a gradual process of evolvement over time [9]. Companies typically start to conduct controlled experiments on significant changes (for example, a new interface) to gain traction and support for investments into A/B testing infrastructure. As they scale, and as we show in [9], every bug fix and minor change is a priori released through an A/B test. Optimizing the planning of team activities by using CE, for example, can only be achieved at scale [9], [12].

Evolvement of skills and technology: To scale CE, several new skills such as Data Scientists and Data Engineers [45] are needed. At the same time, investments in an experimentation platform and instrumentation of existing products are needed [46]. And although these are initially a cost for the company, a significant learning and ROI are seen over time at Microsoft and other companies [9], [11].

Mitigating pitfalls: Related literature [47]–[50] identified several pitfalls of using CE. As the usage of controlled experiments grows, it is becoming more important to understand the opportunities and pitfalls one might face when using CE in practice. For example, *survival bias detection* in long-lasting experiments (i.e. experiments that last over 10 weeks), *underpowered experiments* (i.e. example experiments with insufficient number of users), *hypothesizing after results are known, incorrectly computing confidence intervals for percent change*, etc. These are pitfalls that make CE less trustworthy and need to be controlled and mitigated.

Prioritization of changes: Experimenting in production environment is becoming the norm for product changes in experimentally evolved online companies that operate in non-safety critical domains [11]–[15]. Investments in existing parts of the development pipeline that are resource demanding today (for example in pre-production test systems) are becoming deprioritized due to CE. As a consequence, software companies that evolve CE will increasingly prioritize experimenting in the production. This introduces two types of challenges: (1) deciding which product changes to prioritize for CE vs. regular pre-production testing, and (2) timing the experiments. With CE, it is common to reveal features under development to a large set of users. Companies that study their competitors can exploit this situation in order to earlier predict the direction of the software company.

VII. CONCLUSIONS AND FUTURE WORK

The benefit of Controlled Experimentation (for example A/B testing) in incremental product development has been demonstrated a number of times both by research in the academia [2], [6], [8], and in the industry by Microsoft [11], [12], Google [13], Facebook [14] and other software-intensive companies [12], [15]. In this paper, and based on a case study at Microsoft, we demonstrate that benefits of controlled experimentation extend beyond this recognized scenario. First, we show the benefits on the product portfolio level, where CE aids in accurately identifying what constitutes customer and business value, and how much exactly the work of product teams contribute to it. Next, we show that CE impacts development on the product level where decisions on product functionality, complexity, quality and infrastructure needs are made. Finally, we show that CE benefits the team level where R&D effort is planned, executed and measured. We illustrate our findings with five illustrative experiments conducted at Microsoft at several product teams.

Initial guidance on how to achieve certain benefits from continuous experimentation has been presented in the literature both by us [5], [11] and other researchers [7], [51], [52]. In this paper, we provide guidance also for achieving the benefits related to the portfolio and team level. In our opinion, however, more work is needed in this area. Specifically, we believe that the research and industry would benefit from (1) a deeper understanding of the experiment lifecycle, (2) guidance on detection of patterns between leading and lagging metrics, and (3) providing automation and support for product teams in this process. Also, further studies are needed to identify in what contexts to allow CE, and where not to use this methodology. For example, CE is positively perceived by the customers in the context of introduction of new features [7], [51], [52], However, in situations where certain users loose a feature due to CE (for example in a reverse experiments where we study the effect of a feature removal on customer and business value) customer response is not sufficiently understood.

ACKNOWLEDGMENT

We wish to thank Brian Frasca, Greg Linden, Ronny Kohavi and others across Microsoft Corporation that provided valuable input and feedback on this research.

REFERENCES

[1] D. J. Patil, "Building Data Science Teams," *Oreilly Radar*, pp. 1–25, 2011.

[2] A. Fabijan, H. H. Olsson, and J. Bosch, "Customer Feedback and Data Collection Techniques in Software R&D: A Literature Review," in *Software Business, ICSOB 2015*, 2015, vol. 210, pp. 139–153.

[3] R. Kohavi and R. Longbotham, "Online Controlled Experiments and A/B Tests," in *Encyclopedia of Machine Learning and Data Mining*, no. Ries 2011, 2015, pp. 1–11.

[4] H. H. Olsson and J. Bosch, *The HYPEX model: From opinions to data-driven software development*. 2014.

[5] H. H. Olsson and J. Bosch, "Towards continuous customer validation: A conceptual model for combining qualitative customer feedback with

quantitative customer observation," in *Lecture Notes in Business Information Processing*, 2015, vol. 210, pp. 154–166.

[6] F. Fagerholm, A. S. Guinea, H. Mäenpää, and J. Münch, "Building Blocks for Continuous Experimentation," *Proc. 1st Int. Work. Rapid Contin. Softw. Eng.*, pp. 26–35, 2014.

[7] F. Fagerholm, A. S. Guinea, H. Mäenpää, and J. Münch, "The RIGHT model for Continuous Experimentation," *J. Syst. Softw.*, vol. 0, pp. 1–14, 2015.

[8] L. Madeyski and M. Kawalerowicz, "Software Engineering Needs Agile Experimentation: A New Practice and Supporting Tool," in *Advances in Intelligent Systems and Computing*, vol. 504, L. Madeyski, M. Śmiałek, B. Hnatkowska, and Z. Huzar, Eds. Cham: Springer International Publishing, 2017, pp. 149–162.

[9] A. Fabijan, P. Dmitriev, H. H. Olsson, and J. Bosch, "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*, 2017, pp. 770–780.

[10] F. Girardin and N. Lathia, "When User Experience Designers Partner with Data Scientists," in *AAAI 2017 SPRING SYMPOSIA*, 2017, pp. 376–381.

[11] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne, "Controlled experiments on the web: survey and practical guide," *Data Min. Knowl. Discov.*, vol. 18, no. 1, pp. 140–181, Feb. 2009.

[12] R. Kohavi, A. Deng, B. Frasca, T. Walker, Y. Xu, and N. Pohlmann, "Online controlled experiments at large scale," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, pp. 1168–1176.

[13] D. Tang, A. Agarwal, D. O'Brien, and M. Meyer, "Overlapping experiment infrastructure," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '10*, 2010, p. 17.

[14] E. Bakshy, D. Eckles, and M. S. Bernstein, "Designing and deploying online field experiments," in *Proceedings of the 23rd international conference on World wide web - WWW '14*, 2014, pp. 283–292.

[15] P. Rodríguez *et al.*, "Continuous Deployment of Software Intensive Products and Services: A Systematic Mapping Study," *J. Syst. Softw.*, 2015.

[16] R. C. Martin, *Agile Software Development, Principles, Patterns, and Practices*. 2002.

[17] H. H. Olsson, H. Alahyari, and J. Bosch, "Climbing the 'Stairway to heaven' - A mulitiple-case study exploring barriers in the transition from agile development towards continuous deployment of software," in *Proceedings - 38th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2012*, 2012, pp. 392–399.

[18] S. Mujtaba, R. Feldt, and K. Petersen, "Waste and lead time reduction in a software product customization process with value stream maps," in *Proceedings of the Australian Software Engineering Conference, ASWEC*, 2010, pp. 139–148.

[19] E. M. Goldratt and J. Cox, *The Goal: A Process of Ongoing Improvement*, vol. 2nd rev. e, no. 337 p. 2004.

[20] E. Ries, *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. 2011.

[21] A. Fabijan, "Developing the right features: the role and impact of customer and product data in software product development," Malmö University, 2016.

[22] L. Cao and B. Ramesh, "Agile requirements engineering practices: An empirical study," *IEEE Softw.*, vol. 25, no. 1, pp. 60–67, 2008.

[23] P. Bosch-Sijtsema and J. Bosch, "User Involvement throughout the Innovation Process in High-Tech Industries," *J. Prod. Innov. Manag.*, vol. 32, no. 5, pp. 1–36, 2014.

[24] H. H. H. Olsson and J. Bosch, "From opinions to data-driven software R&D: A multi-case study on how to close the 'open loop' problem," in *Proceedings - 40th Euromicro Conference Series on Software Engineering and Advanced Applications, SEAA 2014*, 2014, pp. 9–16.

[25] G. Castellion, "Do It Wrong Quickly: How the Web Changes the Old Marketing Rules by Mike Moran.," *J. Prod. Innov. Manag.*, vol. 25, no. 6, pp. 633–635, 2008.

[26] The Standish Group, "The standish group report," *Chaos*, vol. 49, pp. 1–8, 1995.

[27] J. Manzi, *Uncontrolled : the surprising payoff of trial-and-error for business, politics, and society*. Basic Books, 2012.

[28] K. Rodden, H. Hutchinson, and X. Fu, "Measuring the User Experience on a Large Scale : User-Centered Metrics for Web Applications," *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, pp. 2395–2398, 2010.

[29] M. L. T. Cossio *et al.*, *A/B Testing - The most powerful way to turn clicks into customers*, vol. XXXIII, no. 2. 2012.

[30] V. F. Ridgway, "Dysfunctional consequences of performance measurements," *Adm. Sci. Q.*, vol. 1, no. 2, pp. 240–247, 1956.

[31] P. Dmitriev and X. Wu, "Measuring Metrics," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management - CIKM '16*, 2016, pp. 429–437.

[32] A. Deng and X. Shi, "Data-Driven Metric Development for Online Controlled Experiments : Seven Lessons Learned," *Kdd*, 2016.

[33] R. C. Van Nostrand, "Design of Experiments Using the Taguchi Approach: 16 Steps to Product and Process Improvement," *Technometrics*, vol. 44, no. 3, pp. 289–289, Aug. 2002.

[34] A. Fabijan, H. H. Olsson, and J. Bosch, "Time to Say 'Good Bye': Feature Lifecycle," in *42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Limassol, Cyprus. 31 Aug.-2 Sept. 2016*, 2016, pp. 9–16.

[35] H. Hohnhold, D. O'Brien, and D. Tang, "Focusing on the Long-term," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '15*, 2015, pp. 1849–1858.

[36] R. Kohavi, A. Deng, and R. Longbotham, "Seven Rules of Thumb for Web Site Experimenters," *Kdd*, pp. 1–11, 2014.

[37] A. M. Ghalayini and J. S. Noble, "The changing basis of performance measurement," *Int. J. Oper. Prod. Manag.*, vol. 16, no. 8, pp. 63–80, Aug. 1996.

[38] F. A. Manuele, "Leading & Lagging Indicators," *Prof. Saf.*, vol. 54, no. 12, pp. 28–33, 2009.

[39] C. McChesney, S. Covey, and J. Huling, "The 4 disciplines of execution : achieving your wildly important goals," *Four Discip. Exec.*, pp. 1–8, 2012.

[40] E. A. Locke, K. N. Shaw, L. M. Saari, and G. P. Latham, "Goal setting and task performance: 1969-1980.," *Psychol. Bull.*, vol. 90, no. 1, pp. 125–152, 1981.

[41] X. M. Bezuijen, K. van Dam, P. T. van den Berg, and H. Thierry, "How leaders stimulate employee learning: A leader-member exchange approach.," *J. Occup. Organ. Psychol.*, vol. 83, no. 3, pp. 673–693, 2010.

[42] E. a Locke and G. P. Latham, "Building a practically useful theory of goal setting and task motivation: A 35-year odyssey," *Am. Psychol.*, vol. 57, no. 9, pp. 705–717, 2002.

[43] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empir. Softw. Eng.*, vol. 14, no. 2, pp. 131–164, 2008.

[44] C. Robson, *Real World Research: A Resource for Social Scientists and Practitioner-Researchers*. 2002.

[45] M. Kim, T. Zimmermann, R. DeLine, and A. Begel, "The emerging role of data scientists on software development teams," in *Proceedings of the 38th International Conference on Software Engineering - ICSE '16*, 2016, no. MSR-TR-2015-30, pp. 96–107.

[46] T. Barik, R. Deline, S. Drucker, and D. Fisher, "The Bones of the System: A Case Study of Logging and Telemetry at Microsoft," 2016.

[47] P. Dmitriev, B. Frasca, S. Gupta, R. Kohavi, and G. Vaz, "Pitfalls of long-term online controlled experiments," in *2016 IEEE International Conference on Big Data (Big Data)*, 2016, pp. 1367–1376.

[48] R. Kohavi, A. Deng, B. Frasca, R. Longbotham, T. Walker, and Y. Xu, "Trustworthy online controlled experiments," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '12*, 2012, p. 786.

[49] T. Crook, B. Frasca, R. Kohavi, and R. Longbotham, "Seven pitfalls to avoid when running controlled experiments on the web," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09*, 2009, p. 1105.

[50] P. Dmitriev, S. Gupta, K. Dong Woo, and G. Vaz, "A Dirty Dozen: Twelve Common Metric Interpretation Pitfalls in Online Controlled Experiments," in *Proceedings of the 23rd ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '17*, 2017.

[51] E. Lindgren and J. Münch, "Raising the odds of success: The current state of experimentation in product development," *Inf. Softw. Technol.*, vol. 77,

pp. 80–91, 2015.

[52] O. Rissanen and J. Munch, “Continuous Experimentation in the B2B Domain: A Case Study,” *Proc. - 2nd Int. Work. Rapid Contin. Softw. Eng. RCoSE 2015*, pp. 12–18, 2015.