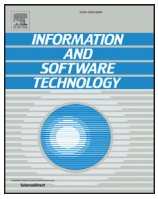
Information and Software Technology 120 (2020) 106244

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof



Patterns of user involvement in experiment-driven software development

Sezin Yaman [a,*], Fabian Fagerholm [b], Myriam Munezero [b], Tomi Männistö [b], Tommi Mikkonen [b]

[a] KPMG Finland, Töölönlahdenkatu 3B 00101, Helsinki, Finland
[b] Department of Computer Science, University of Helsinki, Finland

A B S T R A C T

Background: Experiments are often used as a means to continuously validate user needs and to aid in making software development decisions. Involving users in the development of software products benefits both the users and companies. How software companies efficiently involve users in both general development and in experiments remains unclear; however, it is especially important to determine the perceptions and attitudes held by practitioners in different roles in these companies.

Objective: We seek to: 1) explore how software companies involve users in software development and experimentation; 2) understand how developer, manager and UX designer roles perceive and involve users in experimentation; and 3) uncover systematic patterns in practitioners' views on user involvement in experimentation. The study aims to reveal behaviors and perceptions that could support or undermine experiment-driven development, point out what skills could enhance experiment-driven development, and raise awareness of such issues for companies that wish to adopt experiment-driven development.

Methods: We conducted a survey within four Nordic software companies, inviting practitioners in three major roles: developers, managers, and UX designers. We asked the respondents to indicate how they involve users in their job function, as well as their perspectives regarding software experiments and ethics.

Results and Conclusion: We identified six patterns describing experimentation and user involvement. For instance, managers were associated with a cautious user notification policy, that is, to always let users know of an experiment they are subject to, and they also believe that users have to be convinced before taking part in experiments. We discovered that, due to lack of clear processes for involving users and the lack of a common understanding of ethics in experimentation, practitioners tend to rationalize their perceptions based on their own experiences. Our patterns were based on empirical evidence and they can be evaluated in different populations and contexts.

1. Introduction

Software development companies are increasingly utilizing user and product data to guide their development decision-making [13,15]. Experimentation with users is often used to understand user behavior and needs as they interact with products and services. The benefits of experimentation involving users is recognized by both academia and industrial research conducted by well-known software companies such as Google [34] and Microsoft [12]. However, how and where to involve users and how to run experiments with them is often shaped by company contexts, existing practices and the varying perspectives and attitudes of practitioners. Currently, not many studies have investigated the reality of these issues in software development companies.

Thus, this paper presents a survey study that was conducted with software companies to investigate the status of existing software development and user involvement practices, the perspective of the practitioners in involving users in their product or service development, and how software experiments are currently understood. The views of practitioners in different roles can help companies better understand and enhance the practice of experiment-driven development. The need for the study emerged in the context of a large Finnish research program[1] that aimed to enhance Finnish ICT companies' ability to deliver value in real time. Besides providing insights at the company level and at the functional role level of their staff, this study identified six patterns of perceptions and attitudes with respect to software experiments and user involvement, based on the data analysis of different practitioner groups. The patterns contribute to theory development in understanding how practitioners reason during the experiment design process.

The survey included four general sections: background, current status of software development, experimentation, and ethics. The preliminary results of the ethics section, in particular the ethics of notifying and involving users in experiments, have been reported in the author's prior work [38]. The full analysis of survey responses with respect to all survey sections is reported in this publication.

The results show that there are no mutually exclusive groups of practitioners who think fundamentally differently about user involvement and software experiments, but rather several patterns that reflect differences in perceiving and practising experiment-driven development with user involvement. For instance, wide data collection, i.e., collecting rich and detailed data from users even without having an up-front assumption or hypothesis, was preferred by a large group of developers, whereas UX designers and managers tended to opt for focused data

* Corresponding author.
E-mail addresses: sezin.yaman@helsinki.fi, sezin.yaman@cs.helsinki.fi (S. Yaman), fabian.fagerholm@helsinki.fi (F. Fagerholm), myriam.munezero@helsinki.fi (M. Munezero), tomi.mannisto@helsinki.fi (T. Männistö), tommi.mikkonen@helsinki.fi (T. Mikkonen).

[1] http://www.n4s.fi/en/

https://doi.org/10.1016/j.infsof.2019.106244
Received 24 March 2018; Received in revised form 29 November 2019; Accepted 12 December 2019
Available online 17 December 2019

collection. Allowing exceptions in user notification in experiments, e.g., that it is acceptable not to disclose some experiment details to users, was associated with the perception that users would like to take part in experiments. On the other hand, cautious attitudes about user notification, i.e., that users should always be notified, was associated with the perception that users should be convinced of the benefits of an experiment before taking part in it.

The rest of the paper is structured as follows: Section 2 presents the background and related work relevant to this study. The study's research method, including the research questions and research design, are presented in Section 3. Results of the study are described in Section 4, which presents the results of the survey responses at both company and role levels, as well as patterns that emerged from the entire set of survey responses. Section 5 discusses the results in relation to the research questions, and Section 6 expands on the implications and limitations of the study. The conclusions and potential future work are in Section 7.

2. Background and related work

Software companies seek methods to assess and evaluate the user value and success of their products. Collecting product and user data to continually guide decision-making processes is a practice that has received increasing attention, especially in the last decade [13]. Big companies like Microsoft are reported to run tens of thousands of experiments every year across dozens of products [12]. Approaches in which product decisions are guided based on experiments involving users, while varying in detail and implementation, can be termed *experiment-driven software development*. Despite the increasing attention, there are not many studies that offer an understanding of organizational contexts and practitioners' standpoints with respect to involving users as participants in experiments.

Involving users in software development has a long history and has emerged in various research sub-disciplines of software engineering, such as participatory design, user-centric design, usability engineering, human computer interaction, requirements engineering, and information systems [2,13]. For instance, participatory design has been an important field of research in which software users make effective contributions to reflect their needs and perspectives [27]. Likewise, while the emphasis in participatory design is on democratic participation and skill enhancement, usability engineering is the process of defining, measuring and improving the usability of products, and overlaps with user-centric design principles [24]. Gould and Clayton proposed three main principles of user-centric design in the 1980s: 1) early focus on users and tasks; 2) empirical measurement; and 3) iterative design [17]. Similarly, human computer interaction seeks to improve the usability of human-computer interfaces [18]. These principles are widely followed and they have inspired other development approaches. They can be used as instantiations or parts of an experiment-driven software development approach.

However, the means of involving users in software development processes have also been transforming in attempts to adapt to new development approaches. For instance, in their systematic literature review on customer feedback and data collection techniques, Fabijan et al. [13] report that experiments, such as A/B tests, are a common data collection technique, especially in the domain of Web 2.0 and software-as-a-service (SaaS). Yaman et al. [41] also report in their literature review that experiments and tests are one of the most frequently used methods to collect user data, especially since 2009. With experiments, users are involved in shaping the software product and services by being subject to software experiments.

It is important to note that experimentation in software engineering is not a new topic of interest; research sub-disciplines such as evidence-based software engineering and empirical software engineering have been emphasizing the need for empirical studies to develop or improve processes, methods, and tools for software development and maintenance since the1970′s [5,22]. However, experimentation as a term has been used inconsistently by software engineering researchers and practitioners. Sjoberg et al. [33] in their survey study on experimentation in software engineering, reveal that the term *experiment* has often been used to refer to any empirical studies in general. In recent years, controlled experiments, mostly in the form of A/B tests, have been used by web-facing companies to continually improve their systems [26], as they help companies to establish a causal relationship between a variation, such as a new feature or a change, in their system and observed user behavior [23]. While these experiments often take place at the post-deployment stage of a product, there have also been attempts to enable experimentation at early stages of software development, especially to test business ideas, models and concepts early on. For instance, the continuous experimentation approach emphasizes the need to support research and development (R&D) decisions through iteratively conducting experiments, in which hypotheses are closely linked to business goals.

Continuous experimentation has been receiving increasing attention as a software development approach in which R&D activities are driven by iteratively conducting experiments and collecting user feedback [15,31]. Fagerholm et al. [14] emphasize the need to observe user behavior continuouslythrough field experiments that are derived from business strategies and finding out what the user wants. A recent mapping study by Ros and Runeson [32] provides a general definition for continuous experimentation: "conducting experiments in iterations". They add that it is "a general term for a wide variety of experiments and the implications of experiments on the whole software engineering process". In continuous experimentation, users are consequently involved in the decision-making process as experimental subjects, providing data by interacting with the experimental materials, such as the software features being developed or related design artifacts. The product value is tested by observing actual user behavior rather than relying on secondary sources, opinions, or assumptions. This leads to a transformation from agile software development to continuous business experiments and business model evaluations [21,28]. In addition, R&D as an experiment system is then driven by real-time customer feedback [28].

Several experimentation models have been proposed in the software development literature. Fagerholm et al. introduce the RIGHT model, which builds on the build-measure-learn (BML) loop as a means to test product assumptions iteratively [15]. In each BML iteration, experiments derived from business vision are conducted, and thus business value is evaluated through each iteration. Olsson et al. introduce a qualitative/quantitative customer-driven development model and view user requirements as hypotheses that need to be validated with experiments during the development cycle [29]. These hypotheses can emerge from various channels such as business strategies or customer feedback. Similarly, other models such as the innovation experiment systems model [7] and early stage software startup development model [6] also follow the BML principle introduced by the Lean Startup approach [30] – testing hypotheses in BML loops. The loop starts with identifying a hypothesis to be tested, and continues with building minimum viable products (MVP) for data collection and learning from what is measured [30].

Furthermore, Yaman et al. [40] identified core elements of experimentation arising from experimentation models including feedback loop, hypotheses, MVPs, data collection methods and analysis. Short feedback loops enable data collection in order to evaluate the assumptions rapidly. Assumptions are formed as testable hypotheses, and they can be derived from business strategies or an ongoing validation cycle. MVPs refer to the smallest possible set of functionalities of a product that is to be tested against pre-defined hypotheses. In order to perform the experiment, different data collection methods can be followed, such as qualitative, qualitative or mixed methods. The data collected should be analyzed with respect to hypotheses to see if they are validated or falsified.

On the other hand, continuous experimentation is a software development approach that involves experiments targeted at users of software

products, who are human subjects, and it involves collecting data from them. Therefore, ethical issues and principles have to be taken into account in the design, execution and analysis of experiments and involving users in experiments, as well as the methods and tools used for data collection. While there is a significant amount of literature on ethics in scientific research, and an abundance in business ethics research (see, e.g., [35]), little has been published on ethics in continuous experimentation specifically (our prior research being one exception [38]). What ethical aspects play the largest role in continuous experimentation in the field, and how practitioners reason about those aspects, is unclear. In this paper, we do not seek to define a new normative ethic for continuous experimentation, but rather understand empirically what ethical aspects are considered by practitioners and how. There can be many ethical concerns in principle, but a few key questions appear to be important when a company is starting their first experiments. Data collection in general is subject to several considerations, such as informed consent and protection of privacy. Whether the user is notified of an experiment up-front, or whether personal information is collected through experiments, are important issues to clarify. Other first considerations include the potential harm done to participants and the risk to the company if it is involved in ethical breaches.

Practitioners could turn to the scientific literature or tradition for guidance on ethical concerns. Several guidelines have been proposed in the existing literature for involving users in experiments. For example, Vinson and Singer [36] identified four key principles for conducting empirical studies involving human subjects: 1) Subjects must give their informed consent for their participation; this implicitly includes the requirement to notify users in order to allow them to give consent. 2) Before conducting experiments, it is important to assess whether the benefits outweigh the harm, risks, and efforts, and whether the obtained user data will really be trustworthy, whether the experiment results can be used for decision making, and whether the time spent on experiments is worthwhile. 3) Experimenters must take all possible measures to maintain confidentiality. 4) The experiment should have value in order to motivate subjects to expose themselves to the risks.

In light of Vinson and Singers' first two principles, Yaman et al. performed prior work on a part of this study on ethics, and reported on how practitioners in software companies understand the ethical aspects of involving users [38]. The results revealed that employees in different roles in software companies perceive ethical issues and attitudes differently. For instance, while managers are more cautious about customer-company relationships, UX designers were found to be more familiar with experimentation practices. With the full data analysis in this paper, we examine how previous findings match up with the new findings.

In order to better understand the continuous experimentation approach and user involvement, it is important to investigate existing software development methods and tools. Arriving at this understanding requires an examination of organizational context and existing ways of working, as well as practitioners' standpoints, since organizational changes often begin at the individual level [10]. In this paper we report on a survey aimed at addressing these issues.

3. Research method

To gain an understanding of ongoing software development and user involvement activities, as well as experimentation involving users at software development organizations, we designed and conducted a survey. In this section, we describe the research questions, research design, data collection, and data analysis.

3.1. Research questions

We sought to answer the following research questions:

1. How do software companies involve users in software development and experimentation with software products and services?
2. How do developer, manager and UX designer roles perceive and involve users in experimentation?
3. What patterns emerge from practitioners' views on user involvement in experimentation?

The first research question aims to address companies' ongoing user involvement practices in general software development and particularly in experimentation. The second question focuses on understanding experimentation with users from the points of view of different job functions. For instance, we seek to understand what a UX designer considers common user data collection methods. Finally, the last question examines the dataset of the survey responses as a whole, to uncover important associations between different questions of the survey. Such associations can take the form of patterns that relate different aspects of working with experiment-driven software development. For instance, *Given that a respondent is a developer and given that (s)he has more than five years of experience, how likely is it that they agree with the following statement: "Users do not need to know that they are involved in an experiment"?* The purpose is to uncover patterns that can be used to construct theory that allows reasoning regarding the choices made in designing experiments and working with an experiment-driven approach.

3.2. Survey design

An online survey was used to collect data from industry respondents. The survey included a background section to collect demographic data, and sections that addressed user involvement and experimentation practices, and attitudes and ethical concerns towards involving users in experimentation. To accommodate differences in the companies, the survey was constructed as a template so that certain items were tailored to each company (e.g., using company-specific job position titles). The survey template is provided in Appendix A.

The survey was iteratively designed by three researchers, taking existing survey and questionnaire research into consideration (e.g., [8,42]). We developed a conceptual framework based on prior research and our own observations from prior studies with companies. In particular, we used insights developed and questions raised by practitioners during our research with two companies that were introducing continuous experimentation [39,40]. The framework consisted of three main areas: i) the current state of involving users in software development, ii) views on experimentation with users, and iii) views on notifying and involving users.

These areas are reflected in the survey structure, with three sections following a background section.

The first area is intended to probe the current state of user involvement in the company. Based on prior research by us and others, we assumed that involvement may differ in terms of software life-cycle stages but also in terms of how information from involvement is accessed, disseminated, and used inside the company, and in the means of obtaining information from users – ranging from direct observation and interaction during or after use to automatic recording or logging of user actions. Furthermore, we assumed that there may be differences in the closeness between persons in the company and the users.

The second area is intended to probe perceptions of what constitutes an experiment, the kind of data collected, and the reasons for collecting it. As there are many ways to carry out an experiment operationally, there may be differences in how practitioners think about them. For example, A/B tests can be carried out by delivering two software versions to two groups of users and inviting them to a structured interview to collect data on how they were received, but they can also be carried out by tracking a specific outcome variable without further interaction with the users. A/B tests can also be more and less rigorous, with varied attention to sample randomization and statistical power. There are trade-offs between the alternatives that practitioners must resolve when carrying out an experiment, for example, the effort and cost to obtain and analyze data, and the richness of the data obtained.

Based on this reasoning, and our prior observations, we identified three conceptual dimensions to investigate within this area. The first concerns whether to always gather as much data as possible or to collect data only when there is a specific question to test. We observe that it might be easy to add logging of user actions to applications, yielding large amounts of data that could be mined for interesting patterns. The conceptual opposite is to carefully design the data collection with respect to specific questions of interest and collect only information relevant to those questions. The second dimension concerns the balance between qualitative, rich data and quantitative, specific data. Rich, qualitative data, such as recordings of usage sessions, could be used to answer complex questions and retain contextual information. Specific, quantitative data, such as key metrics on certain user actions, lends itself well to statistical analysis. Finally, the third dimension concerns the degree of active involvement on the part of users. On the one hand, users can be invited to collaborate closely with developers, creating a dialogue. On the other hand, users can be observed while they carry out their regular tasks without actively involving them.

The third area is intended to investigate the perceptions of the ethics, trustworthiness, and required resources of experiments. This area was raised as an important concern by participants in our prior studies who were starting to design and carry out their first experiments. We constructed a scale that captures different aspects of experimentation, including ethical and operational concerns. For example, in some psychological experiments, users' awareness of being experimented on can bias the results. It may be defensible to carry out such experiments provided that the harm done is negligible. In scientific experimentation, it is considered obligatory to disclose this to participants afterwards, and to allow them to withhold consent to use the data. We constructed another scale that captures different aspects of trustworthiness and required resources. We assumed that there may be differences in how experimentation is perceived when practical limitations are taken into account – such as the time available and the expectations that may be raised among users when they are exposed to tentative software versions.

The conceptual framework described here was a guide to designing the survey, and while it stems from empirical observations in the related literature and by ourselves, it was not intended as a result in itself. In operationalizing the conceptual areas and dimensions, we did not assume that they would be polar opposites and instead allowed respondents to indicate their agreement with each end of the dimensions separately. In other words, we allowed respondents to indicate both agreement and disagreement with both ends of a dimension simultaneously. Thus we hoped to capture patterns that we could not have foreseen.

While operationalizing our general framework into the final survey, we formed questions using various design techniques and elements, including check-box, radio button, drop-down selections, open text fields and Likert-type scales [16]. Likert-type scale questions were partly organized hierarchically and intentionally overlapping: general statements (e.g. "Users do not need to know they are involved") were followed by more specific statements (e.g. "Users can be involved in an experiment without their knowledge if we let them know afterwards"). This was done to allow respondents to express general attitudes as well as exceptions under special conditions. Respondents rated the statements on a five-point scale, ranging from "strongly agree" to "strongly disagree". The option "I don't know" was also provided and counted as a missing answer. Furthermore, open text field questions were designed to collect rich and free-form data from the respondents. They included questions such as: "Please describe a typical experiment you have seen or been involved with in your company, including the roles."

Before deploying the survey, the *background* section in particular, but also the other sections where applicable, were tailored so that terms and concepts matched the contexts of the companies. For instance, practitioner roles were tailored to match the actual titles or job definitions in each company. Furthermore, we used the terms "customer" and "user" involvement interchangeably in the survey in order to refer to the primary user – someone who uses the relevant software in each company context. For instance, for the employees of Company C, "user" is the relevant term as the company has direct access to users, whereas for Company D, which is a consultancy company, "customer" is the party for whom the products are developed. We maintained a mapping between company-specific job titles and the role categories we considered most prominent in software companies: "developers" (persons performing technical duties, such as programmers, architects, and testers), "managers" (e.g., team, product, or line managers), "UX designers" (e.g., persons involved in planning user interfaces, usability, and visual design of user interfaces), and "other" (e.g., office administrators and sales). At this stage, we considered such a coarse-grained division to be appropriate, given the lack of prior work on the subject. The mappings between company terminology and concepts in our framework were created by interviewing a company representative to ensure accuracy.

3.3. Data collection and analysis

The survey was administered in four different software companies based in Finland, though respondents were distributed over the companies' offices in Europe and the United States. Table 1 gives a brief description of the companies and the size of the target group. Data was collected for two weeks in each company from November 2016 to April 2017.

In each company, a division or team deemed relevant by a company contact person was selected as the target group. The relevance was determined through discussion, especially according to whether the members of the target group were engaged in work activities related to software used outside the company and whether their mode of work included obtaining information about users. We deployed the survey as a web-based, company-specific on-line form. The contact person in each company distributed a link to the form directly to the target group in the respective company. A reminder was sent after roughly one week. A total of 130 practitioners from the four companies responded to the survey. The respondents remained anonymous to the researchers at all stages. Company-specific response numbers are shown in Table 2.

As the survey included different types of questions, e.g., Likert-scale as well as open questions, there was both quantitative and qualitative data to analyze. Therefore, we used both quantitative and qualitative data analysis methods. First, we pre-processed the data from each of the four companies, so that it could be merged into a single dataset. As the survey was designed so that the background section (e.g. job functions) and terminology (e.g. user vs. customer) was different for each of the four companies, we transformed the raw data into a consistent form with new categories for further operations such as comparisons or aggrega-

Table 1
Description of surveyed companies and target groups.

Company	Company type and domain	Target group
A	A division of a very large telecommunications network company	231 people
B	A large information security company	25 people
C	A medium-sized company providing a user interface development toolkit	135 people
D	A large digital consultancy providing software development services	397 people

Survey companies were mostly located in the Nordic countries but operated globally.
The target group represents the population size the survey was sent to in each company.

Table 2
Demographics of the survey respondent sample.

	Company A	Company B	Company C	Company D	Total
Developers	20 (57%)	1 (13%)	6 (29%)	44 (66%)	**71 (55%)**
Managers	10 (29%)	6 (75%)	4 (19%)	3 (0.5%)	**23 (18%)**
UX	2 (0.6%)	-	4 (19%)	16 (24%)	**22 (17%)**
Others	3 (0.9%)	1 (13%)	7 (33%)	3 (0,5%)	**14 (10%)**
Total	**35**	**8**	**21**	**66**	**130**
Women	5 (14%)	1 (13%)	0 (0%)	6 (9%)	12 (9%)
Men	27 (77%)	6 (75%)	21 (100%)	57 (86%)	111 (85%)
Not specified	3 (9%)	1 (13%)	0 (0%)	3 (5%)	7 (5%)
Team size (mean)	6–10	>20	6–10	6–10	6–10 people
Age range (mean)	41–50	40	31–40	31–40	31–40 years old
Years in current position (mean)	2–3	2–3	2–3	4–5	3–5 years

The percentages under the company names represent the ratios within each company.

tions. The transformation was performed in accordance with the mapping we had created in collaboration with each company representative.

We employed descriptive statistics and association rule learning (ARL) [20], depending on the need and purpose of the analysis. Descriptive statistics was used to understand the background of the respondents, and to summarize the responses to each survey question that was in quantitative form. We applied iterative thematic analysis [9] to the qualitative data. Furthermore, we applied ARL [20] on the full dataset to identify underlying patterns.

The decision to apply ARL to the entire dataset as the main analysis technique was based on the study design. As the research topic is novel, our survey design and its underlying conceptual framework had to be created for the study. Our goal was to obtain empirical evidence for theory-building. ARL allowed us to extract patterns from the data. Furthermore, no prior knowledge existed about the respondent population with respect to our study focus. Due to lack of prior knowledge on the population, and the theory-building goal, statistical testing was not used. We instead followed an exploratory data analysis approach. ARL is a rule-based machine learning method, corresponding to Bayesian analysis, and is based on conditional probability. It is used to discover interesting relationships between different variables in large datasets, and was a good fit for discovering the patterns in our study. ARL finds direct relationships between different subsets of values of variables, and can thus yield results that can be further analyzed to construct tentative theories.

For ARL, the Apriori algorithm [3] is the best-known algorithm used in a transactional dataset to mine frequent itemsets and then generate association rules. Association rules are generated after crossing the threshold for parameters, including support and confidence. Fig. 1 shows the formulae used for these parameters, for the rule of $X \Longrightarrow Y$. *Support* is an indication of how frequently the X and Y appear in the database together, *confidence* indicates the number of times the if/then statements have been found to be true and *lift* of a rule is the ratio of the observed support to that expected if X and Y were independent. In other words, high support means the items co-occur frequently enough, high confidence indicates that the rule is true often and high lift indicates dependence and that the rule is not just a coincidence. To perform the ARL analysis, we cleaned and pre-processed the entire

$$Rule\ X \Longrightarrow Y \begin{cases} Support = \frac{Frequency\ (X,Y)}{N} \\ Confidence = \frac{Frequency\ (X,Y)}{Frequency(X)} \\ Lift = \frac{Support}{Support(X) * Support(Y)} \end{cases}$$

Fig. 1. Formulae for support, confidence and lift for the association rule $X \Longrightarrow Y$.

dataset of survey responses to prepare for the Apriori algorithm. We selected the categorical variables and regrouped them into three groups in order not to increase the complexity. For instance, all the Likert-type scales were regrouped into the categories disagreement, indecisiveness, agreement. After cleaning, we had a 130 * 80 dataset of variables of the survey responses and we ran the Apriori algorithm to generate relationships between different subsets of values of variables. Out of the thousands of rules generated by the Apiori algorithm, we experimented with different combinations of support, confidence and lift parameters, considering what each parameter indicates.

In order to answer RQ1, we segmented the entire dataset by companies and performed descriptive analysis for each segment separately. For RQ2, we segmented the entire dataset by participant roles, i.e., developers, managers, UX designers and other roles, and performed descriptive analysis on each segment and compared them to each other. The role category *other* was removed from the set as it included dissimilar roles that may not have related directly to software development, e.g., sales. We also examined the responses to each individual question and cleaned the data before further processing. All responses were usable although some had missing answers for some questions (indicated through the "I don't know" option). One question (question 8; see Appendix A) did not offer meaningful results and was excluded from the analysis. We believe the question may have been too complicated, as it required participants to make judgments regarding roles other than their own.

In order to confirm our analysis methods and the results, we employed additional data analysis, such as calculating the Pearson correlation coefficient [1] among variables, using hierarchical cluster analysis and variable importance analysis [4,37]. The findings were in line with our main analysis methods, descriptive statistics and ARL, and did not offer additional insights. Therefore, we excluded them in this paper. All analysis was done using the R programming language.

Finally, in order to confirm our conclusions and to allow our participants to comment on the results and conclusions and possibly use them in their own contexts, we employed member checking [11] in company-specific feedback sessions with the company representatives and selected participants. The member checking sessions resulted in some minor clarifications of, for example, company job roles, and we received feedback on how the questions had been understood. This strengthened our confidence in the decision to omit question 8 from the analysis.

4. Results

In this section, we first present the profiles of the survey respondents (4.1). Second, we look into company-specific results, in order to better understand their context and way of working with respect to user involvement and experimentation practices (4.2). Afterwards, we explore the overall survey responses from practitioner job functions' standpoint – among developers, managers, and UX designers (4.3). Finally, using

ARL analysis, we identify underlying relationships and emerging trends in the responses and report on the most important ones (4.4).

4.1. Sample

In total, we received 130 responses to the survey. Table 2 summarizes the demographics for each company. As shown in the table, Company D had the highest number of respondents (66), with the majority being developers (44), while Company B had the lowest number of respondents (8), with the majority being managers (6). The table further shows that the respondents from Company C and D were on average younger than those from Company A and B, where the ages ranged between 31–40 years and 40–50 years respectively. In particular, respondents from Company D had the longest experience in their respective current functional roles, i.e., an average of 4–5 years. Furthermore, the majority of the respondents worked in teams ranging from 6 to 10 people on average, except for those from Company B, who worked in teams of 20 on average. This is understandable, considering that the majority of respondents (75%) from Company B were managers, who are likely to oversee a larger number of employees.

4.2. The state of user involvement and experimentation in companies

To identify how software companies involve users in their development practices, we asked the respondents to answer several questions regarding their current development practices, and the methods and tools that they use in experiments. Furthermore, we asked respondents to tell us about situations where it was challenging to involve users in their work. We also asked them to describe a typical experiment in their opinion and inquired further into experimentation. Here, first we look into responses on general software development, then we present the responses on experimentation practices.

4.2.1. General software development

To begin with, respondents were asked which of their development activities most involved users. Three of the companies stated that users were more involved in *specifying requirements* – 71%, 75% and 79% agreement in Companies A, B, and D respectively. This is not surprising as requirements for a product or service often come from existing or potential users and customers. Company C, on the other hand, stated that *testing* was the activity in which they most commonly involved users (57% agreement). In total, the aggregated results show that while specifying requirements is the activity where the users are involved the most, with 72% agreement, software implementation is where the users are involved the least, with 33% agreement.

Fig. 2 shows the mean of the responses from each company on the statements regarding user involvement, which constitutes question 7 of the survey. Overall, practitioners stated that they knew who used the software they contributed to at work and that they had enough information about them. There are a few exceptions: for instance, respondents from Company A, a large telecommunications company, showed more

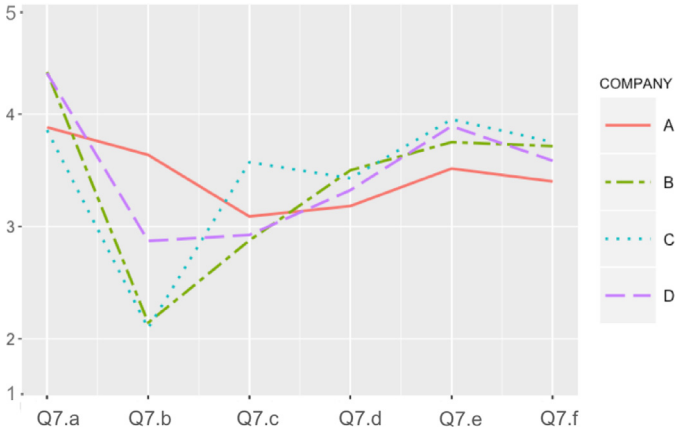


Fig. 2. User involvement statements (means) from question 7 of the survey. Q7.a: I know who uses the software I contribute to in my work, Q7.b: I need to ask for permission to contact users, Q7.c: I frequently have direct contact with users, Q7.d: I have sufficient information about users' needs, Q7.e: I have information about users that is relevant to my work, Q7.f: The information I have about users is up to date. (Response options: 1: completely disagree, 2: disagree, 3: neither agree nor disagree, 4: agree, 5: completely agree.).

agreement on needing permission to contact their users. Respondents from Company B, which constitutes the smallest sample with 8 respondents, with 6 of them being managers, indicated on the contrary that they did not need permission to contact users. Respondents from Company C, a company offering development toolkits to their users, agreed particularly strongly that they had frequent contact with users.

On the other hand, we received 40 responses to the open question where we asked respondents to describe a situation where involving users in development would be useful, but it was not possible to do so. Table 3 shows the major reasons for users being inaccessible for involvement in software development and experimentation, based on our thematic analysis. The most commonly mentioned reason was the challenge of accessing the end users of the software due to the multi-layered structure of users and customers. Practitioners often work for their companies, which deliver the software solution to their customer, who might then sell or deliver the product to their own users. Therefore, practitioners might not be allowed to contact the end users, and have to rely on the pre-determined user requirements delivered to them. A practitioner said: *"Sometimes the user requirements show a lack of understanding of the technical solution. But instead of discussing this with the user and exploring alternatives, it is done by the defined requirements. [.] The resulting solution is often correct by the book but not what the user needed"*. Another respondent elaborated on reasons why (s)he could not involve users: *"The customer feels that either they are so knowledgeable about their users that they do not need to gather further information from them, or they plan to do so themselves and are not*

Table 3
Major reasons why users could not be involved in the development activities, according to practitioners.

Theme	Description	# people
Multi-layered user/customer structure	Companies might often have *customers* who sell or deliver the software product to their own users; therefore, it might not be possible to access their users. There might also be financial conflicts between different layers. In addition, the customers might think that they are already knowledgeable about what the user wants.	11
Time and budget constraints	Even if users might be accessible, due to ongoing commitments and tight deadlines, it might be difficult to reach them. The customers might find it costly to allow practitioners involve users in the process.	7
Lack of process	There might be no clear process as to when and where to involve the users in development activities. Heavy bureaucracy, such as getting the right permits to contact users, might also slow down the development.	5
Consent and privacy	It might be difficult or impossible to obtain users' consent due to privacy reasons. Alternative solutions, such as test labs, might not be the same as monitoring users on-board an actual flight.	4
Pre-determined requirements	The user requirements might already be determined in advance, and practitioners are told to follow them.	5

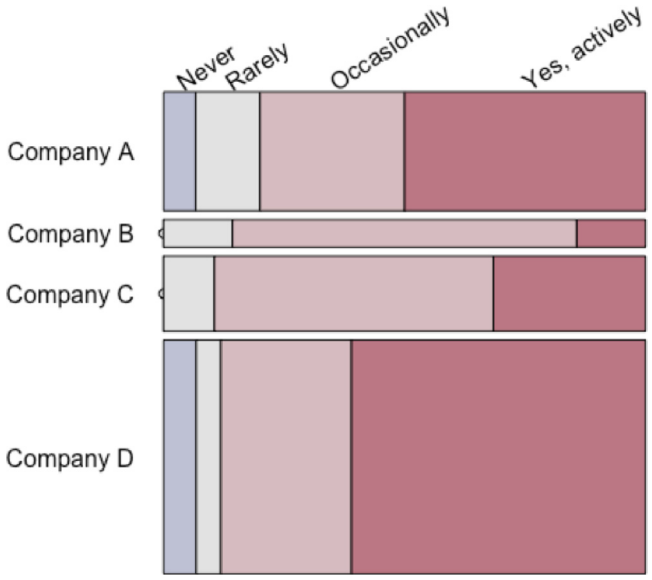


Fig. 3. The proportions of the responses from companies to the question 11 – "Does your company conduct experiments involving the users?".

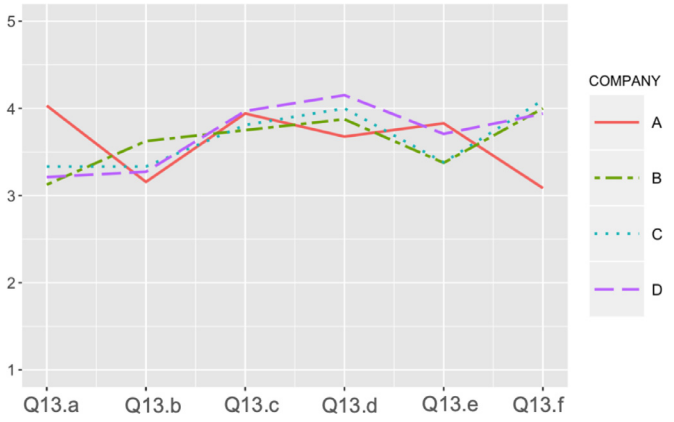


Fig. 4. Responses to the Question 13 (means) of the survey. - To understand users' needs better, ... Q13.a: data should always be collected because it might be needed later, Q13.b: data should only be collected when there is a known need or assumption, Q13.c: rich, detailed data about what users do is useful, Q13.d: focused data on a specific user action or behavior is useful, Q13.e: users themselves must be actively involved in shaping the software, Q13.f: we need to measure user behavior to decide what the software should be like. (Response options: 1: completely disagree, 2: disagree, 3: neither agree nor disagree, 4: agree, 5: completely agree.).

interested in involving us due to financial interests, or they may be concerned about the cost of involving us". According to these responses, when such multi-layered structure exists between the practitioners and the end users, practitioners are left isolated from end users' real needs.

Furthermore, a second common reason why users could not be involved in the development was time and budget constraints. A practitioner explained that due to time pressure, it sometimes might be easier to not involve users in the development but implement the user requirement right away: *"A huge time pressure: implementing a solution takes only a few hours, which are available right now, today. [Implementation with no user involvement] won't solve things but let's see what the feedback will be"*. We also learned that the lack of a process for involving users poses a challenge. One practitioner pointed out: *"[In one project] the business goals and the product idea did not match. Without user insight it was not possible to formulate any sensible next steps for development. [.] In the process someone had already decided how the users would be involved in the process, but it did not quite fit the current situation"*. Other practitioners also mentioned that they tended to avoid going through the bureaucracy of involving users or they simply did not know how to involve users in development activities. Lastly, we found out that practitioners might not involve users in the development due to issues of user consent and privacy. While some practitioners explained that they could not get the permits needed to involve users due to concerns about user privacy, one practitioner told us about the difference between observing user behavior with and without their knowledge: *"The best feedback would be available onboard a flight [the environment in which the software would be being used by the user], but it is hard to get consent [.]. Users can be invited to a test lab, but it is a different setting compared to being onboard an actual flight"*.

4.2.2. Experimentation

In addition to general software development and user involvement, practitioners from each company were asked about experimentation. Fig. 3 shows the distribution of the responses over companies, when the respondents were asked about how often their company conducted experiments involving users. The majority of the respondents from Company A and D stated that their companies actively conducted experiments (42% and 54% agreements), whereas those from Company B and C reported occasionally conducting experiments (62% and 54% agreements) more. It is important to emphasize that no definition of experiments or experimentation was provided to the respondents in the survey. Right after question 11, we asked the respondents to describe a typical experiment they had seen or been involved in within their company in an open question.

Experiments can be understood differently, depending on various factors such as companies' way of working and the different duties required by a job function. 46 respondents provided written comments on what a typical experiment was in their company, according to their experiences and beliefs. Table 4 summarizes the two types of descriptions of a typical experiment. 28 people described experiments as UX/UI activities and user studies organized by practitioners from UX/UI teams. Commonly used terms to describe these experiments included: *user studies, scenarios, surveys, interviews and walkthroughs*. One practitioner explained the experimentation process in his company as: *"Our UX designers typically present initial drafts of the UI to actual users, and test them out before any code is written. Similarly, users are often involved throughout the development to test ready software and give feedback about it"*. On the other hand, 22 respondents described the experiments using the following terms: *hypothesis-based, A/B tests, user analytics, building MVPs and releasing part of a feature or software to (a subset of) users to collect data*. Some respondents described the experimentation process in their company involving both types as: *"[When] a complexity is observed in a workflow, this results in a hypothesis on how it could be simplified, to improve the user's experience. The hypothesis is discussed, with drawn sketches in case of UI issues, with users. When a satisfactory design is identified, a simple implementation with pre-defined analytics hooks is created and deployed. Afterwards, deployment feedback from pilot users is gathered and compared with the analytics data. Based on both outcomes, the feature is either left permanently in, or removed or refactored"*. These findings indicate that experiments can be understood differently.

To gain a deeper understanding of the companies' practices in experiment-driven development, we further inquired about data collection practices in question 13 of the survey, as shown in Fig. 4. The statements of the question were constructed to better determine whether different strategies for data collection and user involvement, such as focused data collection, would be more preferable. From the responses, it was identified that practitioners from Company A prefer data to be collected not only when there is a need but in case it might be needed later. For companies B, C and D, both constant collection of data and focused data on a specific user action is welcomed, with a greater preference for focused data. Rich and detailed data about what users do was found to be useful by all the companies. Likewise, user behavior should be measured

Table 4
Type of experiments, as described by respondents.

Type	Description	# people
UX/UI activities and user studies	Practitioners referred to activities and user studies organized and conducted by job functions, such as UX/UI designers, to describe experiments. The activities are referred to are: BDD stories, user stories, scenarios, usability tests, surveys, interviews, shadowing sessions, workshops, walkthroughs, talkalouds, mockups.	28
Hypothesis-based experiments and analytics	Experiments that are driven by pre-defined hypotheses, measuring user behavior, collecting and using user data and analytics for experiments. Practitioners referred to the following terms when describing these experiments: A/B tests, prototypes, MVPs or MVFs, partial or limited release, piloting with proxy users.	22

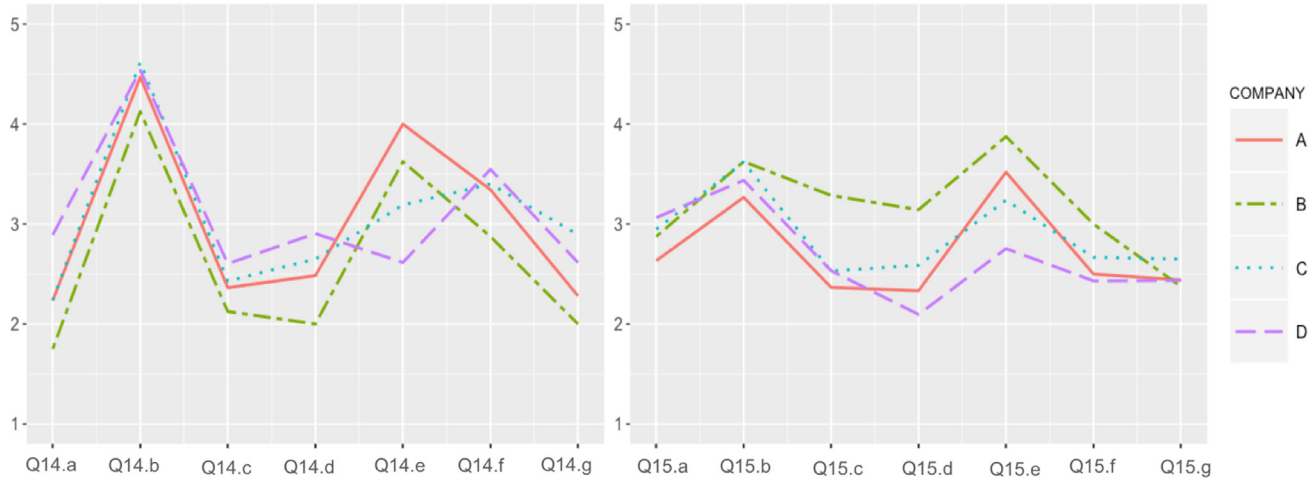


Fig. 5. Responses to ethics statements (means) from questions 14 and 15 of the survey. (Note: Selected labels mentioned in the text are included here, the full statement-set can be found in the appendix.) Q14.a: Users do not need to know they are involved, Q14.b: If we collect personal information, users need to be notified, Q14.c: If no laws are being broken, users do not need to be notified, Q14.d: Users can be involved in an experiment without their knowledge if we let them know afterwards, Q14.e: Users should always be notified when they are being involved in an experiment, Q14.f: It is ok not to disclose all the experiment details to users involved, Q14.g: It is ok to intentionally deceive or mislead the user if experiment results depend on it, Q15.a: I cannot trust that the results will be correct, Q15.b: Involving users in experiments is time-consuming, Q15.c: Our company does not have the needed technical infrastructure, Q15.d: Users would not like to be part of software experiments, Q15.e: Users have to be convinced of the benefits before taking part, Q15.f: Experiments give users false expectations, Q15.g: Experiments reveal secrets about the product strategy (Response options: 1: completely disagree, 2: disagree, 3: neither agree nor disagree, 4: agree, 5: completely agree.).

and users themselves also must be actively involved in the development. Even though practitioners from Company A showed more indecisiveness on measuring user behaviour to make software development decisions, the difference among companies were observed to be small. Overall, practitioners do not disagree with a particular data collection strategy.

The final set of questions in the survey, questions 14 and 15, explored the ethical perceptions of involving users in experiments, as well as attitudes toward experimentation. Fig. 5 shows the statements on the ethics of experimentation. From the results, we observe both common perceptions shared by all practitioners, and also exceptions for specific companies. For instance, all practitioners showed strong agreement on notifying the users about an experiment they are involved in, if their personal information is to be collected. While they also tend to agree on average that involving users in experiments is a time-consuming task, they disagree on average that experiments would reveal secrets about product strategy. Furthermore, we observed differences in the understanding of user notification. For instance, employees of Company A and B on average agreed that users should always be notified when they are involved in an experiment (Q14.e), while Company B neither disagreed or agreed clearly, and Company D disagreed. In addition, we see that respondents from Company B indicated more strongly that it is not acceptable to involve users in experiments without their knowledge, even if they let them know afterwards (Q14.d). Another exception specific to Company D is that all companies' respondents, except Company D, claimed that users have to be convinced of the benefits before taking part in an experiment (Q15.e). In other to understand the survey responses better, we investigate next whether any role differences exist among each of the four companies.

4.3. Software practitioner role analysis

Here, we look at the full dataset of survey responses with respect to developer, manager and UX designer roles. As Fig. 2 showed, of the 130 responses, 71 were developers, 23 were managers and 22 were UX designers. In addition, we see that the managers were all in the over-50 age group and worked with more than 20 people on a daily basis, whereas developers worked with 6–10 people on a daily basis and UX designers with 3–5 people.

Fig. 6 shows the average responses given by all three job functions to questions 7 and 9 of the survey, which inquire about user involvement activities and the tools and methods used to involve users. We observe that there are differences in the average responses. For instance, managers did not need feel the need to always ask for permission to contact users, while developers and UX designers might (Q7.b). While developers' responses suggested that on average they did not have frequent contact with users, managers and UX designers did (Q7.c). UX designers in general used all user involvement methods, while managers' responses indicate that they did not use any of the user involvement methods often, and developers' responses indicate that they sometimes used recorded usage data of a software product, such as log data.

With respect to experimentation, Fig. 7a shows that by proportion, UX designers reported on conducting active experimentation the most, followed by developers. In fact, none of the UX designers reported no or rare experimentation. On the other hand, a small proportion of developers and managers never conducted experiments. Furthermore, the differences in perception of different roles regarding ethics in experimentation can be seen in Fig. 7b. In general, UX designers expressed

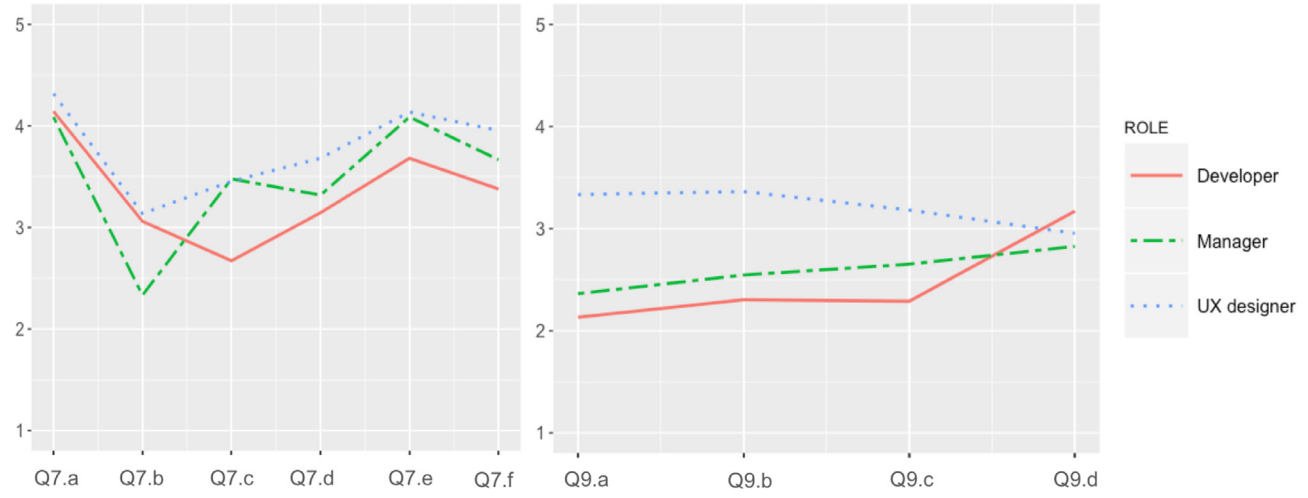


Fig. 6. Responses to questions 7 and 9 (means) over roles. Q7.a: I know who uses the software I contribute to in my work, Q7.b: I need to ask for permission to contact users, Q7.c: I frequently have direct contact with users, Q7.d: I have sufficient information about users' needs, Q7.e: I have information about users that is relevant for my work, Q7.f: The information I have about users is up to date. (Response options: 1: completely disagree, 2: disagree, 3: neither agree nor disagree, 4: agree, 5: completely agree.) Q9.a: I remotely observe or interact with users when they are using the software (e.g., screen sharing), Q9.b: I interact with the users before they use the software, Q9.c: I interact with users after they use the software (e.g., post-use interview, feedback), Q9.d: Through recorded usage data (e.g., log data)) (Response options: 1: never, 2: rarely, 3: sometimes, 4: often, 5: always.).

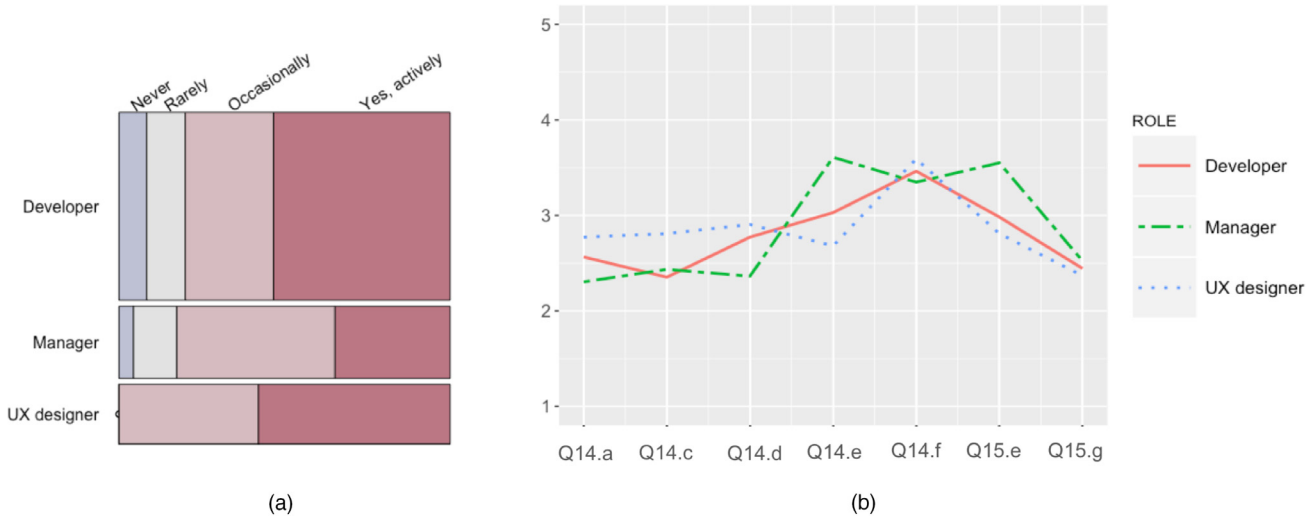


Fig. 7. (a) The proportions of the responses from three job functions to the question 11 – "Does your company conduct experiments involving the users?". (b) Responses to questions selected statements of 14 and 15 (means) over roles. Q14.a: Users do not need to know they are involved, Q14.c: If no laws are being broken, users do not need to be notified, Q14.d: Users can be involved in an experiment without their knowledge if we let them know afterwards, Q14.e: Users should always be notified when they are being involved in an experiment, Q14.f: It is ok not to disclose all the experiment details to users involved, Q15.e: Users have to be convinced of the benefits before taking part, Q15.g: Experiments reveal secrets about the product strategy. (Response options: 1: completely disagree, 2: disagree, 3: neither agree nor disagree, 4: agree, 5: completely agree).

more indecisiveness on notifying users of an experiment. In particular, the statement "Users should always be notified when they are being involved in an experiment" (Q14.e) separated all roles: managers agreed, UX designers disagreed and developers showed indecisiveness. Managers showed a different perception on needing to convince the users to take part in experiments, as well (Q15.e) – they thought that users should be convinced of the benefits of the experiments before taking part in it. With respect to execution of experiments, on average all job functions seemed to accept exceptions, such as that some details of the experiments could be kept away from the users. Even though we noticed differences in the average responses of different job functions, we keep in mind that the sample sizes are not equal, and company contexts may have influenced the way the equivalent roles worked. Next, we look at all the results from all points of views we have reported so far, in order to explore underlying relationships.

4.4. Association rule learning

Association rules are if/then statements that help uncover relationships between seemingly unrelated data in a database. We examined the full dataset with association rule learning (ARL) analysis [20] to explore what relationships exist among the variables of the dataset of

Table 5
Most important rules produced by the Apriori algorithm. (Note 1: The wording in the rules are adjusted for understandability. Note 2: Count represents the frequency of the rule occurring in the dataset.).

#	Antecedent		Consequent	Confidence	Support	Count
1	{Data does not need to always be collected in case it might be needed later AND we need to measure user behavior to decide what the software should be like}	⟹	{Data should only be collected when there is a known need or assumption}	1	0.169	22
2	{It is acceptable not to disclose all the experiment details to users involved AND I have information about users that is relevant to my work AND I often use log data}	⟹	{Focused data on a specific user action or behavior is useful}	1	0.2	26
3	{I am actively conducting experiments AND I have information about users that is relevant to my work AND I often use log data }	⟹	{Focused data on a specific user action or behavior is useful}	1	0.16	21
4	{I am a developer AND data does not need to only be collected when there is a known need or assumption}	⟹	{Rich, detailed data about what users do is useful}	1	0.215	28
5	{Data should always be collected because it might be needed later AND I often use log data}	⟹	{Rich, detailed data about what users do is useful}	1	0.2	26
6	{Users need to know they are involved AND even if no laws are being broken, users need to be notified AND Users cannot be involved in an experiment without their knowledge even if we let them know afterwards AND users have to be convinced of the benefits before taking part}	⟹	{Users should always be notified when they are being involved in an experiment}	1	0.13	17
7	{Users need to know they are involved AND users cannot be involved in an experiment without their knowledge even if we let them know afterwards AND rich, detailed data is useful AND users themselves must be actively involved in shaping the software}	⟹	{Users should always be notified when they are being involved in an experiment}	1	0.15	20
8	{Users need to know they are involved AND even if no laws are being broken, users need to be notified AND involving users in experiments is time-consuming AND users would like to be part of experiments}	⟹	{Users should always be notified when they are being involved in an experiment}	1	0.13	17
9	{Users need to know they are involved AND Even if no laws are being broken, users need to be notified AND users can not be involved in an experiment without their knowledge even if we let them know afterwards AND I cannot trust that experiment results will be correct}	⟹	{Involving users in experiments is time-consuming}	1	0.092	12
10	{I am a manager AND users need to know they are involved AND users should always be notified when they are being involved in an experiment}	⟹	{Users cannot be involved in an experiment without their knowledge even if we let them know afterwards}	1	0.107	14
11	{I work in Company D AND users do not need to know they are involved AND It is acceptable not to disclose all the experiment details to users involved}	⟹	{Users do not need to always be notified when they are being involved in an experiment}	1	0.115	15
12	{I work in Company D AND I actively conduct experiments AND users do not always need to be notified when they are being involved in an experiment AND users do not have to be convinced of the benefits before taking part}	⟹	{It is acceptable not to disclose all the experiment details to users involved}	1	0.107	14
13	{Users do not always need be notified when they are being involved in an experiment AND rich, detailed data about what users do is useful AND I often use log data}	⟹	{I work in Company D}	1	0.16	21
14	{I do not need permission to contact users AND users would like to be part of experiments AND I have sufficient information about users' needs AND I have information about users that is relevant to my work}	⟹	{It is easy for me to obtain user info}	1	1.805	21
15	{I frequently have direct contact with users AND it is easy for me to obtain user info AND we need to measure user behavior to decide what the software should be like}	⟹	{I have information about users that is relevant to my work}	1	0.23	30

survey responses. These variables correspond to each statement under the survey questions. As a result, we chose the most important rules, as presented in Table 5. For instance, Rule 1 appears as:

{Data does not need to be always be collected in case it might be needed later AND we need to measure user behavior to decide what the software should be like} ⟹ {Data should only be collected when there is a known need or assumption} (Confidence = 1, Support= 0.169, Count = 22)

This rule indicates that the respondents who ranked the statements on the left hand side (aka *antecedent*) as shown, also agreed with the statement in the right hand side (aka *consequent*). Confidence parameter equal to 1 means that the rule is a logical rule, meaning that the rule occurs 100% of the time. Count represents how many times the rule has been observed in the dataset. In this case, 22 practitioners responded to the three statements stated in Rule 1, as shown in the table. For all the rules, we filtered them by highest confidence and support, while simultaneously considering the highest lift parameter. In Table 5, we present confidence and support parameters, as well the number of occurrences (count). It is important to emphasize that all the rules listed in the table have the confidence parameter 1, which means that for all the responses containing the antecedent, the consequent was found to be true. Therefore, *count* represents the number of respondents who responded to the survey statements as described in each rule.

The most important rules found b y ARL analysis reveal several trends across the entire dataset with respect to various aspects of experiment-driven development and user involvement. By further analyzing the important rules, we established six categories (named as patterns) that describe the rules. These patterns are listed in Table 6 and constitute the basis for the theoretical contribution of this paper.

When we looked at the rules that indicate practitioners' preference for focused data collection (Rules 1–3), we observed that a large subset of these practitioners also reported that they weer confident about knowing the users and that they had enough information about them. Furthermore, these practitioners were also likely to report on active experimentation, as well as allowing exceptions such as not disclosing all the details of an experiment to the users. We formed the *focused*

Table 6
Patterns created based on association rules.

#	Pattern	Description	Related Rule
1	Focused data collection	A pattern indicating that data does not always need be collected in case it might be needed later. User behavior should be measured to know how the software should be like. People who follow this pattern report that they actively conduct experiments, often use log data and have relevant user information, and they opt for focused data collection.	Rule 1, 2 and 3
2	Wide data collection	A group of respondents is associated with the pattern that data should not only be collected when there is a known need or assumption, but instead, rich and detailed data about what users do is always useful. A large group of developers is associated with this pattern, which also includes respondents who agree that data should always be collected because it might be needed later and who often use log data as a data collection method.	Rule 4 and 5
3	Conservative ethical attitude	Regardless of any exception, users should always be notified of an experiment. This pattern also includes respondents who are likely to think that users have to be convinced of the benefits before taking part in an experiment, experiment results might not be trustworthy, and involving users in experimentation is time-consuming.	Rule 6, 7, 8, 9 and 10
4	Permissive ethical attitude	A group of respondents disagree that users should always be notified or need to know that they are involved in an experiment. They also feel that it is acceptable to not disclose some experimental details to users. Respondents in this pattern are likely to think that users do not need to be convinced to take part in experiments.	Rule 11, 12 and 13
5	Unrestrained experimentation	A pattern including a group of respondents who opted for wide data collection (Pattern 2) and who are associated with permissive ethical attitude (Pattern 4), such as not allowing the disclosure of all the experiment details to the users and disagreeing that the users have to be convinced to take part. A subset of these respondents also reported active experimentation practices. Company D was found to be highly associated with this pattern.	Rule 4, 5, 11, 12 and 13
6	Easy user access	Easy access to user information is associated with not needing permission to contact users, direct user access, and having relevant and sufficient user information. People who opted for these statements were also likely to think that users would want to be part of experiments.	Rule 14 and 15

data collection pattern based on these associations, which resides on the polar side of the *wide data collection* pattern. Wide data collection pattern was formed based on the responses of a group of developers (28 developers) who disagreed with collecting data only when there is a need for it, and simultaneously agreed that rich and detailed data about what users do is useful. Likewise, a subset of responses indicates that these practitioners were also likely to think that data should always be collected because it might be needed later and they reported using product usage data, such as log data, to involve users in development.

We observed several rules regarding ethical perceptions of experimentation and involving users that point to two different trends. On one side we observed a trend in which a group of practitioners adopted a *conservative ethical attitude* (Rules 6–10). These practitioners opted to notify users of an experiment, regardless of any exception such as notifying them after the experiment. A group of managers (14 managers) shared a stance: users cannot be involved in an experiment without their knowledge even if we let them know afterwards. We found that these practitioners were also likely to believe that users have to be convinced of the benefits of an experiment before taking part in it, and that involving users in experiments is a time-consuming task. Furthermore, these practitioners suspected the trustworthiness of experiment results.

On the other hand, we observed that another group of respondents thought otherwise: users should not always be notified of experiments and they do not always need to know that they are involved in an experiment (Rules 11–13). A subset of these respondents came from Company D and they agreed that not disclosing some experiment details to the users was acceptable. Moreover, we also observed an association between allowing exceptions in user notification and expecting users to take part in experiments willingly. The *permissive ethical attitude* pattern was formed to describe these associations.

Rules 4–5, 11–13 shared a common value for a variable for one group of practitioners: they were all from Company D. Upon further investigation, rule 13 revealed that 21 respondents from Company D opted for statements that are in-line with both permissive ethical attitude and wide data collection patterns. We had already seen in Rules 4–5 and 11–12 that people who opted for wide data collection are likely to be developers and using usage data, and they allow exceptions to user notification in an experiment. We therefore constructed the pattern *Unrestrained experimentation* to describe this inter-pattern, which is specific to Company D.

Lastly, another subset of practitioner responses indicated that not needing permission to contact users, and contacting users often, is associated with having relevant and sufficient information about users, which led us to construct the pattern called *easy user access*. Furthermore, this group of respondents tended to believe that they should measure user behavior and that users would like to take part in experiments. We further discuss these patterns in the next section.

5. Discussion

In the Results section, we first described the sample together with demographics. Next, we outlined the overall results regarding the current status of user involvement in companies, while reporting on interesting company-specific results. Then, we looked at the full dataset and examined it from developers', managers' and UX designers' perspectives. Lastly, we further analyzed the entire dataset to uncover underlying relationships that indicate different trends regarding involving users in experimentation. In this section, we discuss the study findings while answering each research question.

RQ1: How do software companies involve users in software development and experimentation with software products and services?

The state of user involvement in software development revealed both common and distinctive findings on companies. We learned that in all the companies, users were most involved in specifying the requirements, while implementation was reported as the software development activity in which the users were least involved. Furthermore, respondents indicated that even though, on average, they did not contact the users often, they were knowledgeable about user needs and had relevant, up-to-date information. However, when we asked them to describe situations in which they wanted to involve users in their job function but could not, we discovered interesting reasons.

A practitioner explained: *".in most projects (designing and building applications for customers) we have little or no access to actual end users [.] This lack of continuous direct contact regularly leads to bad design decisions simply due to insufficient information on user needs"*. Our thematic analysis

pointed out that multi-layered user structures pose a great challenge for practitioners in obtaining access to users. This finding is in line with the challenge pointed out by Lindgren and Munch [25] in experiment systems, which is limited user access due to business-to-business (B2B) company structures. Financial conflicts can occur in between B2B companies, and it might also be expensive to involve users in software development because of time constraints. Furthermore, issues regarding user consent and privacy, as well as a lack of process, hinder the user involvement process. Findings from Rissanen and Munch [31]'s case study on experimentation in B2B domains also confirm our findings: users can be customers' customers and legal agreements might be required to collect data. In addition, respondents mentioned the problem of having pre-determined user requirements that are not representative of what the user actually wants. One practitioner gave an example of the consequences of invalidated user requirements: *"If I could get a hold of [user] analytics, I could show that one of the buttons that took me a few days to build was not clicked by anyone in the past year. This would give me leverage to better optimize our development schedule"*. Testing the usage of the product through experiments would validate the user requirement mentioned in the practitioner's example. However, not being able to access users prevents the practitioner from conducting such experiments.

As we discussed in the background section, software experimentation as a term has been used inconsistently by the existing work [33]. Furthermore, we believe that the current state of experimentation in companies is affected by how individuals and different roles in an organization understand experiments and experimentation. Motivated by these concerns, we intentionally avoided offering a definition for the term, but rather asked the respondents to describe what an experiment is according to their experiences and beliefs. We found that there are two major perspectives on what experiments are. The majority of the respondents described typical experiments as UX/UI designers' activities conducted with users. For instance, one employee described a typical experiment as: *"The UX team arranges frequent observation workshops where new use cases are walked through with different customers"*. On the other hand, the descriptions provided by the rest of the respondents showed more awareness of hypothesis-based experimentation and other related methods and techniques such as A/B tests, user analytics and limited time release. For instance, one of them described the experimentation process as: *"We've done several feature experiments, published them to the production and then followed their take-up rate. We've also done many sorts of 'growth hacking' experiments within our service and we are also currently in the middle of a big A/B test that will have a big impact on our future roadmap"*. These descriptions are in-line with the core elements of experiment-driven development, which indicates that respondents are following the approach.

It is important to note that there are two different perspectives on describing the experiments, one as UX/UI activities and one as hypothesis-based experiments and analytics, and we do not claim that only one of them is acceptable. In the background and related work section we have reviewed the definition, main concepts and elements of experiment-driven development, such as evaluating software product features with testable hypothesis and short validation cycles. However, we also acknowledge that experiment-driven development can use instantiations or parts of other practices such as usability engineering and user-centric design, and can cover a broader scope. For instance, a broader description of experimentation was offered by Gutbrod et al. [19], where they report on their multi-case study with several startup companies, where experiments were run in various forms, including interviews, trade show testing, landing page, A/B testing and MVP testing, depending on the specific need for the experiment. However, when there are differences in perceiving the approach in an company, there might be risks stemming from mismatched understandings. For instance, some practitioners might believe that experiments are/should be conducted only by the UX designers. When they do not consider themselves to be designing or running experiments, they might be resistant to adapting to experiment-driven development. Therefore, it is advisable for the companies to address such differences and fix the objectives for adopting the approach accordingly.

We also looked into findings that stand out for the companies. For instance, Company D had the biggest proportion of respondents engaged in active experimentation, and respondents showed more flexibility on user notification and experiment execution strategies, such as not disclosing some of the details of an experiment. These findings might mean that, as employees of a large digital consultancy company which provides software solutions to its customers, the survey participants were familiar with experiment-driven development.

RQ2: How do developer, manager and UX designer roles perceive and involve users in experimentation?

Practitioner role directly affects individuals' working methods and the tools they use with respect to both general software development and experimentation. Furthermore, practitioner role affects not only how individuals perceive software experiments, but also the ethical issues concerning the experiments. In particular, practitioners tend to rationalize what is acceptable on ethical issues in accordance with their job function, especially when their organizations' regulations and policies concerning user data collection are not clearly defined. On the other hand, we see a clear boundary in that practitioners show a unified understanding on users always being notified if their personal information is collected in the experiments.

To begin with UX designers, we found that they have the most frequent and direct contact with the users, they are confident about the quality of the information they have about the users, and they conduct experiments actively. Most interestingly, they are the group who allows for exceptions in user notification the most: users do not need to be always notified of experiments and it might be acceptable not to disclose all the experiment details to them. Furthermore, they did not think that users have to be convinced of the benefits in advance nor that experiments would reveal product secrets. The UX designers' enthusiastic attitude towards experimentation can be due to the nature of their job function. They are in general familiar with user studies and different methods such as prototyping, mockups, user surveys and interviews. As we discussed in answering RQ1, the broader description of experimentation fits well with UX designers' way of working.

While UX designers allow for exceptions such as letting users know of the experiment they take part in after the experiment has concluded, managers stood out for their protective attitude and caution toward experimentation involving users. For instance, managers believe most strongly that users must be informed of an experiment and they forbid exceptions to this. Due to their job function and way of working, managers might not be familiar with experiment-driven development; however, they are protective about company-customer relationships.

In general, developers reported the least frequent contact with users, and ranked the lowest on having sufficient and up-to-date user information. We also identified in our ARL analysis that a significant number of developers favored wide data collection. In addition, the results showed that developers also consider exceptions in notifying users of experiments, such as users being informed after the experiments. It is important to consider that our sample included 71 developers, which constitutes 55% of the total population. Why do the majority of developers opt for wide data collection? Unfortunately, since we did not have any assumptions about the job functions and associations with experiment-driven development, it is difficult to interpret the results. To our best knowledge, there is no directly related work on experiment-driven development from the point of view of practitioners' roles. However, the role of practitioners and their point of view in software experimentation has been indirectly addressed by publications such as Mattos et al. [26], in terms of the roles required in experiments, such as data analysts and data scientists.

RQ3: What patterns emerge from practitioners' views on user involvement in experimentation?

Table 6 lists the six patterns that emerged in our ARL analysis. Overall we observed two patterns regarding user data collection

(focused data collection and wide data collection); two patterns concerning ethics of user notification and involvement (conservative ethical attitude and permissive ethical attitude); a pattern on practicing experimentation adapting to a company's way of working (unrestrained experimentation) and finally, a pattern on accessing the user (easy user access). It is important to keep in mind that the patterns were not formed to describe the survey responses dataset in a mutually exclusive fashion. Instead they have been constructed to explain subsets of trends that emerged from the dataset.

Focused data collection indicates that there should be up-front questions, assumptions, and hypotheses with which to start or guide the data collection. Otherwise, data collection with no guidance may result in vast amounts of data, which might become difficult to perform meaningful analysis on. Lindgren and Munch [25] explain that the core idea of experiment-driven development, data collection for the experiments, is carried out based on business-related questions and assumptions to be evaluated. Therefore, we treat focused data collection pattern as a part of the experiment-driven development mindset. As we observe that the respondents who conducted experiments actively and who measured user behavior also opted for focused data collection, we argue that they are already familiar with experiment-driven development.

On the other hand, we observed a large group of developer respondents, who believed that data does not need to be collected only when there is a need or assumption. Another statement that complements this thought is that data should be always collected because it might be needed later. Even though focused and wide data collection are not logical contraries, we can observe that respondents tend to rate these statements consistently, ending up in only one of the groups. The main exception is that the respondents who opted for wide data collection also reported that they used methods such as log data to observe user behavior, the same as the focused data collection pattern. This might mean that respondents involved in both focused data and wide data collection patterns acknowledged observing user behavior, but one group opted for more structured data collection that requires preparatory work, whereas the other opted for wide data collection. This could be due to protective reasons in case of emerging situations where certain data, such as usage data of a product, might become necessary. Furthermore, we know from the results that respondents emphasize the difficulty in involving end users in their work due to multi-layered user/customer structure. We also discussed that UX designers are close to the end users due to the nature of their work and they are familiar with experimentation and its core elements. Based on this, we might conclude that developers might not be as familiar with experiment-driven development as UX designers, since it might be harder for them to gain access to end users to plan and conduct experiments involving them.

Along the ethical line of inquiry, we were able to identify two opposing ways of perceiving the ethics of experimentation. Interestingly, respondents who believed that users should always be notified of experiments were likely to think that users have to be convinced to take part in an experiment, experiments can be time consuming and experimental results might not be correct. Remembering the findings on UX designers' enthusiastic attitude toward experimentation and managers' cautious attitude towards user notification, we may argue that respondents who think negatively about experimentation and user involvement might not be familiar with experiment-driven development. On the other hand, a group of respondents allowed for exceptions in user notification, such as withholding some experiment details from users when necessary. One important question to raise is: Can allowing such exceptions in user notification regarding ethics be an element of the experiment-driven mindset? Even though the permissive ethical attitude pattern is associated with a positive attitude on users wanting to take part in experiments and the trustworthiness of experimentation results, it is difficult to answer this question. Unfortunately, there have not been many studies on the ethics of experimentation in literature. Ethical issues have only been addressed in two related works to our knowledge: Mattos et al. [26] address issues such as how users involved in experiments should be assured that their data would not be used for other purposes, while in their mapping study, Ros et al. [32] draw attention to the ethics of experimentation and the need to investigate the topic further. We also identified from the open question of our survey that unclear policies and concerns about user consent and privacy pose a challenge for practitioners in involving users in their job function. As research that focuses on this subject is only just accumulating, we need more studies to evaluate the question we have raised.

Company D was found to be highly associated with wide data collection and permissive ethical attitude patterns. As we discussed earlier, Company D had different characteristics from the others: it had the highest number of respondents in total and also the highest ratio of developers with respect to all developers (86%). Therefore, we acknowledged that Company D was likely to be seen to follow various rules in our ARL analysis due to statistical power. However, at the same time we also have evidence to believe that Company D, being a large digital consultancy company that provides software solutions to its customers, who then deliver the solution to their users, follows active experimentation practices. Company D being highly associated with two patterns might mean that the company has its own way of practicing experiment-driven development. For instance, while being active in experimentation, it might be still difficult for them to plan experiments with end users due to multi-layered structures. As a result, they might opt to collect as much as data as possible from the end users when possible, as otherwise there might be no data available. Company D can benefit from a careful examination of their experimentation process, especially in terms of user involvement, in order to address difficulties in accessing end users.

The easy user access pattern described the group of respondents who did not need permission for user contact, who had direct access to the users and who were confident about the quality of the user information they had. These people were also likely to believe that users would want to take part in experiments. Even though we believe that needing permission to contact users is company-dependent, this pattern indicates that there is a link between easy user access and practitioners' attitudes toward experimentation involving users. Furthermore, our finding from the open question revealed that lack of process regarding user involvement in the companies might discourage practitioners. We can therefore interpret that organizations might need to define their user involvement process in order to enable experiment-driven development.

6. Implications and limitations

Due to the novelty of the research field of continuous experimentation, there has been no common understanding on different angles of experimentation, such as operational and organizational aspects, and where practitioners' views reside. In this study, the six patterns were constructed based on our ARL analysis to describe existing trends in practitioners' views on experiment-driven development and user involvement. We believe that our study findings, the patterns in particular, can be used by software practitioners to examine their positions with respect to experimentation and user involvement. We do not strictly claim which set of patterns constitutes the right mindset for an experiment-driven development approach. On the contrary, we found and discussed in this study that companies can adopt and practise the approach so that it fits their way of working, as well as their organizational goals. For instance, we pointed out that even though focused data collection is associated with experiment-driven development by existing literature, Company D was found to opt for wide data collection, while they reported conducting active experimentation and held a welcoming attitude toward experimentation practices in general. Questions such as: "Do practitioners want to gather as much data as possible because it is convenient in case it is needed?" can be further asked, and implications, such as the complexity of data analysis, can be investigated to improve development activities. Similarly, we may

seek answers to the question, "Does the conservative user notification policy emerge from lack of understanding of scientific experimentation and biases or from concerns about user consent and privacy?". We learned from our survey results that uncertainties about data collection, processing and user notification with respect to experimentation and user involvement posed a challenge, and can prevent organizations from transitioning to continuous experimentation.

In addition, the patterns on the ethical line of inquiry in particular showed that there are two opposite trends in perceiving ethics of experimentation and user involvement, and they are highly associated with familiarity with experiment-driven development and practitioner roles. In fact, we claim that practitioners tend to rationalize what is acceptable in ethical issues in accordance with their job function. We argue that, due to a lack of clear processes for involving users in development activities and experimentation in organizations, as well as the lack of research and regulations about ethics of experimentation in general, practitioners' views can be based on their own experiences and beliefs. Once future research and regulations such as European GDPR[2] offer more insights, the ethical issues involved in experiment-driven development can be better understood. However, organizations themselves should examine their practises and enable processes for user involvement, as well as creating an organizational culture for continuous experimentation. Our patterns can act as guidelines to initiate such examination.

Furthermore, our study might be subject to several validity threats and limitations. To begin with the survey, the initial survey was drafted by the first three authors of this paper – researchers who have experience in experiment-driven development. Following that, representatives from each company were contacted. Due to the different company domains and contexts, the survey was tailored together with each representative, in an iterative fashion, while maintaining the same goal and structure. For instance, options for the job functions were added, removed or modified depending on the actual roles in each company. For the data analysis, however, we grouped the roles into categories so as to enable comparison between the companies. The grouping was largely informed by the representatives' descriptions of the roles. For example, through discussions, we were able to group *data analytics and operations* people as *developers*.

In order to prevent bias, we purposefully avoided including an explanation or example of what an experiment is, the elements of experimentation or implications of ethical issues. We wanted to observe what the respondents considered to be an experiment and how they perceived ethical issues by themselves. This was also one of the objectives of the study – to identify how software practitioners perceive what experiments are. For this purpose, we asked respondents to describe a typical experiment that they had conducted or been involved in, as well as any challenges they had faced in involving users in these experiments. These descriptions helped us identify the similarities and differences in their perceptions of what an experiment or experimentation with users is. Furthermore, different forms of data helped us cross-validate the findings.

An important challenge was the fact that the number of respondents from each company and the distribution of the roles differed greatly. This introduced some risks. For example, due to their high number, developers had a greater influence in the identified patterns. However, we analyzed the whole dataset from multiple angles: we looked at each company in detail with the aim of understanding their way of working and culture with respect to user involvement in general development and practicing experimentation. Furthermore, we looked at the data from practitioner roles' point of view, in order to better capture similarities and differences in skills and practices with respect to experiment-driven development. Lastly, we looked at the general trends over the whole dataset using ARL analysis, and compared the results to the previous findings from the aforementioned analyses. Data triangulation helped us determine the consistency of the findings. By employing different analysis techniques, we were able to acknowledge the differences in the population. Furthermore, in terms of the conceptual framework we developed for the survey design, we have gained insights into our assumptions, such as the assumption that practitioners might understand the experimentation differently and they might prefer different data collection methods accordingly.

In general, it is difficult to fully predict how the company domains, structures and cultures influenced the practitioners' responses. Nowadays, middle and large-sized companies in particular have multiple simultaneous development projects that might require different sets of development methods and techniques. In our study, company contexts differed, ranging from telecommunications to digital consultancy. Furthermore, we discovered that the companies' multi-layered user and customer structures make a difference in how practitioners practice experiment-driven development. As we did not have any prior knowledge on the population or the companies' way of working, we focused on determining existing trends and patterns across the whole responses, while trying to utilize the information we gathered on the demographics of the respondents to better understand the existing results. In terms of generalizability, we are very interested in seeing how these patterns apply to different populations and in different company domains and cultures. As future work, we intend to conduct additional research in different populations and contexts, so that our findings can be evaluated.

7. Conclusion

The benefits of experiment-driven development and data-driven decision making are acknowledged by many successful software development companies and academia. Big technology pioneers are known to run up to hundreds of experiments at a time on their own experimentation platforms. However, in many other organizations, especially where the experimentation culture has not been fully established, lack of understanding of resources and capabilities can impede transitioning to continuous experimentation. In this study, we used a survey to examine four Nordic development companies to identify their current state of user involvement in software development and views on experimentation, as well as the ethics of involving users in experiments. We identified six patterns from the responses that describe the different perceptions and attitudes held by practitioner groups with respect to experiment-driven software development, and we discussed the influence of the practitioner role.

Experimentation can take place in different forms and at different stages of software development. Most importantly, it can be perceived and practiced differently in different organizations and by different practitioner roles. In this study, we discovered that personal beliefs and work experience have a strong influence on how software experiments are perceived. For instance, UX designers tend to recognize experiments as UX/UI-related studies and prefer methods of user data collection that are in line with their job function. Furthermore, we found that the practitioner role affects not only how individuals perceive software experiments, but also the ethical issues involved. Practitioners tend to rationalize what is acceptable in ethical issues in accordance with their job function. For instance, managers are cautious about the company-customer relationship, and they think that users should always be notified of experiments in advance, whereas UX designers allow for exceptions such as letting them know afterwards. Such differences in ethics indicate that organizations have to work on their regulations and policies concerning user data collection, especially in accordance with data protection regulations that their organizations are subject to.

The patterns we identified from the practitioners' responses revealed how different viewpoints and beliefs come together when practitioners were asked to describe experiment-driven development. For instance, we identified a group of respondents who were already familiar with the

[2] https://www.eugdpr.org/

approach and they valued the core elements of experimentation, such as having testable hypotheses. On the other hand, we found a group of people who were hesitant about involving users in experiments due to various reasons: experimentation can be time-consuming, users might not want to participate in experiments and experiment results might not be correct. These practitioners at the same time also stated concerns about unclear user involvement processes in their organizations, including multi-layered company structures and user consent. Due to these concerns, practitioners might not be ready or willing to plan and conduct experiments with users. Organizations need to address such concerns, especially when there is no clear process or common understanding on user access, user consent and experimentation.

In this study, we took the first steps toward theory-building. Our patterns can be used to detect existing trends and to describe and understand software organization stances on experiment-driven development. Therefore, the existing ways of working or processes that are undermining experiment-driven development could be determined, and we can discover what skills and tools could enhance experiment-driven development. Such examination can aid a better evaluation of organizational needs and goals in adopting continuous experimentation. In future work, replicating our study in different company contexts and populations would be important to gain additional data for evaluating and improving our findings.

Declaration of competing interest

The authors declare that they do not have any financial or nonfinancial conflict of interests.

CRediT authorship contribution statement

Sezin Yaman: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Visualization, Writing - original draft, Writing - review & editing. **Fabian Fagerholm:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Resources, Software, Writing - original draft, Writing - review & editing. **Myriam Munezero:** Conceptualization, Investigation, Methodology, Resources, Writing - original draft. **Tomi Männistö:** Project administration, Supervision, Funding acquisition, Methodology. **Tommi Mikkonen:** Project administration, Supervision.

Acknowledgements

We thank Dr. Anton Antonov for his suggestions on ARL analysis.

The research leading to this publication has received funding from the Extreme Continuous Experimentation in Software Engineering (xCESE) Project (Academy of Finland, project number 317657).

Appendix A. User involvement survey template

USER INVOLVEMENT SURVEY (Untailored)

BACKGROUND

1. **Which of the following most closely matches your primary job function?**
   - ☐ Developing software.
   - ☐ UX/UI design.
   - ☐ Management.
   - ☐ Other: _____
2. **How long have you been working in your current company role?**

   _____
3. **Which of the following best describes you?** ☐ Female ☐ Male ☐ Other / Prefer not to say
4. **What is your age range?**
   - ☐ 20 or less
   - ☐ 21 - 30
   - ☐ 31 - 40
   - ☐ 41 - 50
   - ☐ 51 or more
5. **How many people do you work with on a regular basis in the company?**
   - ☐ Less than 3
   - ☐ 3 - 5
   - ☐ 6 - 10
   - ☐ 11 - 20
   - ☐ More than 20

CURRENT STATE OF INVOLVING USERS

There are different ways of involving users in software development. Please answer based on your own experiences and beliefs.

6. **In which development activities are users involved in your company? (click all that apply)**
   - ☐ Specifying requirements.
   - ☐ Software design.
   - ☐ Implementation.
   - ☐ Testing.
   - ☐ The activities after release.
   - ☐ If other, please specify: _____
7. **How much do you agree with the following statements?**

	completely agree → ← completely disagree
(7.a) I know who uses the software I contribute to in my work.	☐ ☐ ☐ ☐ ☐
(7.b) I need to ask for permission to contact users.	☐ ☐ ☐ ☐ ☐
(7.c) I frequently have direct contact with users.	☐ ☐ ☐ ☐ ☐
(7.d) I have sufficient information about users' needs.	☐ ☐ ☐ ☐ ☐

(7.e) I have information about users that is relevant for my work. ☐ ☐ ☐ ☐ ☐

(7.f) The information I have about users is up to date. ☐ ☐ ☐ ☐ ☐

8. In your experience, how easy is it for the following roles to get information from users? Please consider the roles in your company context.

	← very easy				very difficult →
Developers.	☐	☐	☐	☐	☐
Managers.	☐	☐	☐	☐	☐
UX Designers.	☐	☐	☐	☐	☐
Myself.	☐	☐	☐	☐	☐

9. How often do you use the following ways to get information about users?

	← never				always →
(9.a) I remotely observe or interact with users when they are using the software (e.g., screen sharing).	☐	☐	☐	☐	☐
(9.b) I interact with the users before they use the software.	☐	☐	☐	☐	☐
(9.c) I interact with users after they use the software (e.g., post-use interview, feedback).	☐	☐	☐	☐	☐
(9.d) Through recorded usage data (e.g., log data).	☐	☐	☐	☐	☐

10. Try to remember a situation where you knew that involving users in development would be useful, but you could not involve them. Please describe the situation and what challenges you faced.

_____

_____

_____

EXPERIMENTATION WITH USERS

Experiments can be used in many different ways. Please answer based on your own experiences and beliefs.

11. Does your company conduct experiments involving the users? ☐ Never ☐ Rarely ☐ Occasionally ☐ Yes, actively ☐ I do not know

12. Please describe a typical experiment you have seen or been involved with in your company, including the roles. (Skip this if you have not seen or been involved in any experiments.)

_____

_____

_____

_____

13. Below are three pairs of statements about collecting data for understanding user needs. How much do you agree with each statement? There are no right or wrong answers.

For understanding user needs better...	← completely disagree				completely agree →
(13.a) data should always be collected because it might be needed later.	☐	☐	☐	☐	☐

(13.b) data should only be collected when there is a known need or assumption. ☐ ☐ ☐ ☐ ☐

(13.c) rich, detailed data about what users do is useful. ☐ ☐ ☐ ☐ ☐

(13.d) focused data on a specific user action or behavior is useful. ☐ ☐ ☐ ☐ ☐

(13.e) users themselves must be actively involved in shaping the software. ☐ ☐ ☐ ☐ ☐

(13.f) we need to measure user behavior to decide what the software should be like. ☐ ☐ ☐ ☐ ☐

NOTIFYING AND INVOLVING USERS

14. How much do you agree with the following statements regarding notifying users about experiments? Please answer according to your personal beliefs.

completely agree →
← completely disagree

(14.a) Users do not need to know they are involved. ☐ ☐ ☐ ☐ ☐

(14.b) If we collect personal information, users need to be notified. ☐ ☐ ☐ ☐ ☐

(14.c) If no laws are being broken, users do not need to be notified. ☐ ☐ ☐ ☐ ☐

(14.d) Users can be involved in an experiment without their knowledge if we let them know afterwards. ☐ ☐ ☐ ☐ ☐

(14.e) Users should always be notified when they are being involved in an experiment. ☐ ☐ ☐ ☐ ☐

(14.f) It is ok not to disclose all the experiment details to users involved. ☐ ☐ ☐ ☐ ☐

(14.g) It is ok to intentionally deceive or mislead the user if experiment results depend on it. ☐ ☐ ☐ ☐ ☐

15. How much do you agree with the following statements about involving users in experiments? Please answer according to your personal beliefs.

completely agree →
← completely disagree

(15.a) I cannot trust that the results will be correct. ☐ ☐ ☐ ☐ ☐

(15.b) Involving users in experiments is time-consuming. ☐ ☐ ☐ ☐ ☐

(15.c) Our company does not have the needed technical infrastructure. ☐ ☐ ☐ ☐ ☐

(15.d) Users would not like to be part of software experiments. ☐ ☐ ☐ ☐ ☐

(15.e) Users have to be convinced of the benefits before taking part. ☐ ☐ ☐ ☐ ☐

(15.f) Experiments give users false expectations. ☐ ☐ ☐ ☐ ☐

(15.g) Experiments reveal secrets about the product strategy. ☐ ☐ ☐ ☐ ☐

References

[1] Pearson correlation coefficient, 2017, (https://en.wikipedia.org/wiki/Pearson_correlation_coefficient). Accessed: 2018-09-30.

[2] U. Abelein, B. Paech, Understanding the influence of user participation and involvement on system success–a systematic mapping study, Empir. Softw. Eng. 20 (1) (2015) 28–81.

[3] R. Agrawal, R. Srikant, et al., Fast algorithms for mining association rules, in: Proc. 20th int. conf. very large data bases, VLDB, 1215, 1994, pp. 487–499.

[4] A. Antonov, Importance of variables investigation, 2016, (https://mathematicaforprediction.wordpress.com/2016/01/11/importance-of-variables-investigation/). Accessed: 2018-02-02.

[5] V.R. Basili, The role of experimentation in software engineering: past, current, and future, in: Software Engineering, 1996., Proceedings of the 18th International Conference on, IEEE, 1996, pp. 442–449.

[6] J. Björk, J. Ljungblad, J. Bosch, Lean product development in early stage startups., in: IW-LCSP@ ICSOB, 2013, pp. 19–32.

[7] J. Bosch, Building products as innovation experiment systems, in: International Conference of Software Business, Springer, 2012, pp. 27–39.

[8] N.M. Bradburn, S. Sudman, B. Wansink, Asking Questions: The Definitive Guide to Questionnaire Design–for Market Research, Political Polls, and Social and Health Questionnaires, John Wiley & Sons, 2004.

[9] V. Braun, V. Clarke, Using thematic analysis in psychology, Qualitat. Res. Psychol. 3 (2) (2006) 77–101, doi:10.1191/1478088706qp063oa.

[10] V.J. Callan, Individual and organizational strategies for coping with organizational change, Work Stress 7 (1) (1993) 63–75.

[11] J. Creswell, Research Design: Qualitative, Quantitative, and Mixed Methods Approaches, 3rd, SAGE Publications Inc., 2009.

[12] A. Fabijan, P. Dmitriev, H.H. Olsson, J. Bosch, The benefits of controlled experimentation at scale, in: Software Engineering and Advanced Applications (SEAA), 2017 43rd Euromicro Conference on, IEEE, 2017, pp. 18–26.

[13] A. Fabijan, H.H. Olsson, J. Bosch, Customer feedback and data collection techniques in software r&d: a literature review, in: International Conference of Software Business, Springer, 2015, pp. 139–153.

[14] F. Fagerholm, A.S. Guinea, H. Mäenpää, J. Münch, Building blocks for continuous experimentation, in: Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering, in: RCoSE 2014, ACM, New York, NY, USA, 2014, pp. 26–35.

[15] F. Fagerholm, A.S. Guinea, H. Mäenpää, J. Münch, The RIGHT model for continuous experimentation, J. Syst. Softw. 123 (2017) 292–305, doi:10.1016/j.jss.2016.03.034.

[16] J.A. Gliem, R.R. Gliem, Calculating, interpreting, and reporting cronbach's alpha reliability coefficient for likert-type scales, Midwest Research-to-Practice Conference in Adult, Continuing, and Community Education, 2003.

[17] J.D. Gould, C. Lewis, Designing for usability: key principles and what designers think, Commun. ACM 28 (3) (1985) 300–311.

[18] J. Grudin, Utility and usability: research issues and development contexts, Int. Comput. 4 (2) (1992) 209–217.

[19] M. Gutbrod, J. Münch, M. Tichy, How do software startups approach experimentation? Empirical results from a qualitative interview study, in: International Conference on Product-Focused Software Process Improvement, Springer, 2017, pp. 297–304.

[20] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference and Prediction, 2nd, Springer, 2009.

[21] J. Järvinen, T. Huomo, T. Mikkonen, P. Tyrväinen, From agile software development to mercury business, in: International Conference of Software Business, Springer, 2014, pp. 58–71.

[22] B.A. Kitchenham, T. Dyba, M. Jorgensen, Evidence-based software engineering, in: Proceedings of the 26th International Conference on Software Engineering, IEEE Computer Society, 2004, pp. 273–281.

[23] R. Kohavi, R. Longbotham, D. Sommerfield, R.M. Henne, Controlled experiments on the web: survey and practical guide, Data Min. Knowl. Discov. 18 (1) (2009) 140–181.

[24] S. Kujala, User involvement: a review of the benefits and challenges, Behav. Inf. Technol. 22 (1) (2003) 1–16.

[25] E. Lindgren, J. Münch, Software development as an experiment system: a qualitative survey on the state of the practice, in: International Conference on Agile Software Development, Springer, 2015, pp. 117–128.

[26] D.I. Mattos, J. Bosch, H.H. Olsson, Challenges and strategies for undertaking continuous experimentation to embedded systems: industry and research perspectives, in: J. Garbajosa, X. Wang, A. Aguiar (Eds.), Agile Processes in Software Engineering and Extreme Programming, Springer International Publishing, Cham, 2018, pp. 277–292.

[27] M.J. Muller, J.H. Haslwanter, T. Dayton, Participatory practices in the software lifecycle, in: Handbook of Human-Computer Interaction (Second Edition), Elsevier, 1997, pp. 255–297.

[28] H.H. Olsson, H. Alahyari, J. Bosch, Climbing the "stairway to heaven"–a multiple-case study exploring barriers in the transition from agile development towards continuous deployment of software, in: Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on, IEEE, 2012, pp. 392–399.

[29] H.H. Olsson, J. Bosch, Towards continuous validation of customer value, in: Scientific Workshop Proceedings of the XP2015, ACM, 2015, p. 3.

[30] E. Ries, The Lean Startup: How Today's Entrepreneurs use Continuous Innovation to Create Radically Successful Businesses, Crown Books, 2011.

[31] O. Rissanen, J. Münch, Continuous experimentation in the B2B domain: a case study, in: Proceedings of the Second International Workshop on Rapid Continuous Software Engineering, in: RCoSE '15, IEEE Press, Piscataway, NJ, USA, 2015, pp. 12–18.

[32] R. Ros, P. Runeson, Continuous experimentation and a/b testing: a mapping study, in: Proceedings of the 4th International Workshop on Rapid Continuous Software Engineering, ACM, 2018, pp. 35–41.

[33] D.I. Sjøberg, J.E. Hannay, O. Hansen, V.B. Kampenes, A. Karahasanovic, N.-K. Liborg, A.C. Rekdal, A survey of controlled experiments in software engineering, IEEE Trans. Softw. Eng. 31 (9) (2005) 733–753.

[34] D. Tang, A. Agarwal, D. O'Brien, M. Meyer, Overlapping experiment infrastructure: more, better, faster experimentation, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2010, pp. 17–26.

[35] J. Tsalikis, D.J. Fritzsche, Business ethics: a literature review with a focus on marketing ethics, J. Bus. Ethics 8 (9) (1989) 695–743, doi:10.1007/BF00384207.

[36] N.G. Vinson, J. Singer, A practical guide to ethical research involving humans, in: Guide to Advanced Empirical Software Engineering, Springer, 2008, pp. 229–256.

[37] P. Wei, Z. Lu, J. Song, Variable importance analysis: a comprehensive review, Reliab. Eng. Syst. Saf. 142 (2015) 399–432.

[38] S. Yaman, F. Fagerholm, M. Munezero, H. Mäenpää, T. Männistö, Notifying and involving users in experimentation: ethical perceptions of software practitioners, in: 2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), 2017, pp. 199–204, doi:10.1109/ESEM.2017.31.

[39] S.G. Yaman, F. Fagerholm, M. Munezero, J. Münch, M. Aaltola, C. Palmu, T. Männistö, Transitioning towards continuous experimentation in a large software product and service development organisation–a case study, in: Product-Focused Software Process Improvement: 17th International Conference, PROFES 2016, Trondheim, Norway, November 22–24, 2016, Proceedings 17, Springer, 2016, pp. 344–359.

[40] S.G. Yaman, M. Munezero, J. Münch, F. Fagerholm, O. Syd, M. Aaltola, C. Palmu, T. Männistö, Introducing continuous experimentation in large software-intensive product and service organisations, J. Syst. Softw. 133 (2017) 195–211.

[41] S.G. Yaman, T. Sauvola, L. Riungu-Kalliosaari, L. Hokkanen, P. Kuvaja, M. Oivo, T. Männistö, Customer involvement in continuous deployment: a systematic literature review, in: International Working Conference on Requirements Engineering: Foundation for Software Quality, Springer, 2016, pp. 249–265.

[42] J. Yu, H. Cooper, A quantitative review of research design effects on response rates to questionnaires, J. Market. Res. (1983) 36–44.