Questions for Data Scientists in Software Engineering: A Replication

Hennie Huijgens
Delft University of Technology
Delft, The Netherlands
h.k.m.huijgens@tudelft.nl

Ayushi Rastogi
Ernst Mulders∗
Delft University of Technology
Delft, The Netherlands
a.rastogi@tudelft.nl
ernst@mulde.rs

Georgios Gousios
Arie van Deursen
Delft University of Technology
Delft, The Netherlands
g.gousios@tudelft.nl
arie.vandeursen@tudelft.nl

ABSTRACT
In 2014, a Microsoft study investigated the sort of questions that data science applied to software engineering should answer. This resulted in 145 questions that considered relevant for data scientists to answer, thus providing a research agenda to the community. Fast forward to five years, no further studies investigated whether the questions from the software engineers at Microsoft hold for other software companies, including software-intensive companies with different primary focus (to which we refer as software-defined enterprises). Furthermore, it is not evident that the problems identified five years ago are still applicable, given the technological advances in software engineering.

This paper presents a study at ING, a software-defined enterprise in banking in which over 15,000 IT staff provides in-house software solutions. This paper presents a comprehensive guide of questions for data scientists selected from the previous study at Microsoft along with our current work at ING. We replicated the original Microsoft study at ING, looking for questions that impact both software companies and software-defined enterprises and continue to impact software engineering. We also add new questions that emerged from differences in the context of the two companies and the five years gap in between. Our results show that software engineering questions for data scientists in the software-defined enterprise are largely similar to the software company, albeit with exceptions. We hope that the software engineering research community builds on the new list of questions to create a useful body of knowledge.

CCS CONCEPTS
• General and reference → Surveys and overviews.

KEYWORDS
Data Science, Software Engineering, Software Analytics.

∗Work completed during an internship at ING.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ESEC/FSE ’20, November 8–13, 2020, Virtual Event, USA
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7043-1/20/11...
https://doi.org/10.1145/3368089.3409717

1 INTRODUCTION
Software engineering researchers try solving problems that are relevant to software developers, teams, and organizations. Historically, researchers identified these problems from their experience, connections in industry and/or prior research. In 2014, however, a study at Microsoft [7] systematically analyzed software engineering questions that data scientists can answer and made it accessible to a wider audience.

Switching context, in the past few years ING transformed itself from a finance-oriented company to a software-defined, data-driven enterprise. From a software engineering perspective, this includes the implementation of fully automated release engineering pipelines for software development activities in more than 600 teams performing 2,500+ deployments per month for 750+ applications. These activities leave a trove of data, suggesting that data scientists using, e.g., modern machine learning techniques could offer valuable and actionable insights to ING.

To that end, ING needs questions that are relevant for their engineers which their data scientists can answer. As we started looking for existing resources, we came across the 145 software engineering questions for data scientists presented in the Microsoft study [7]. However, before adopting the list, we wanted to know:

RQ: To what extent do software engineering questions relevant for Microsoft apply to ING, five years later?

Microsoft is a large software company, while ING that is a FinTech company using software to improve its banking solutions (software-defined enterprise). Moreover, the two companies are at different scale. In 2014, Microsoft had more than 30,000 engineers while even today ING is almost half its size with approximately 15,000 IT employees (on a total of 45,000). More details on the differences in the context of the two companies are available in Table 1. We try to understand whether the questions relevant for a software company extend to a software-defined enterprise. We compare the results of the original Microsoft study [7] with our results at ING to understand the relevance of the questions beyond Microsoft but also as a guide for other software-defined enterprises that are undergoing their digital transformation. We further explore
whether the technological advances in the last five years changed the way we develop software. To answer this question, we carried out a replication of the original Microsoft study at ING. Similar to the original study, we conducted two surveys: one, to find data science problems in software engineering, and second, to rank the questions in the order of their relevance (see Figure 1). For the first survey, we randomly sampled 1,002 ING engineers and received 116 responses with 336 questions. We grouped the 336 questions on similarities resulting in 171 descriptive questions. We shared subsets of these 171 descriptive questions with another random sample of 1,296 ING engineers for ranking. In the end, we received 21,888 rankings from 128 ING engineers. These ranked 171 questions are the questions that engineers at ING would like data scientists to solve. Further, we compare our list of 171 questions to the original list of 145 questions to answer our research question. Our study shows that the core software development problems, relating to code (e.g., understanding code, testing, and quality), developer productivity (both individuals and team) and customer are same for the software company and the software-defined enterprise. Nonetheless, subtle differences in the type of questions point to changes in market as well as differences in the context of the two companies.

2 IMPACT OF THE MICROSOFT 2014 STUDY

In order to gain a good insight into the further course of the Microsoft 2014 study after it was published, including any implications for research, we conducted a citation analysis. In addition, we looked at studies that have not quoted the Microsoft study, but that are relevant to our study. Hence this section also serves as our discussion of related work. We investigated the 136 studies that, according to Google Scholar, quote the Microsoft study. First of all, we looked at the number of times that the 136 studies themselves were cited by other studies; we limited the further analysis to 70 studies with a citation per year greater than 1.00. We then characterized studies into empirical approach, reference characterization, SE topic, and machine learning (ML) topic (see Table 2). Note that one paper can belong to multiple topics. We made the following observations:

Microsoft itself is building on its study. 11% of the citations come from Microsoft studies itself, mostly highly cited studies on SE culture, such as [18, 41, 51]. We notice that all citing Microsoft studies use a survey among a large number of SE practitioners (ranging from 16 to 793 respondents with a median of 311), whereas other studies based on a survey generally reach substantially lower numbers of participants.

<table>
<thead>
<tr>
<th>Table 1: Context of Microsoft in 2014 and ING in 2019.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Microsoft 2014</strong></td>
</tr>
<tr>
<td>Branch Organization Size</td>
</tr>
<tr>
<td>Approx. 100,000 (in 2014), about 30,000 engineers</td>
</tr>
<tr>
<td>Team Structure</td>
</tr>
<tr>
<td>Development Model</td>
</tr>
<tr>
<td>Pipeline automation</td>
</tr>
<tr>
<td>Development Practice</td>
</tr>
</tbody>
</table>

Half of the citing studies analyze SE process data, and 24% uses a survey. Looking at the empirical approach (see the first sub-table in Table 2), indicates that 43% of the studies contain a quantitative component, in which analysis of SE process data in particular is part of the study. Good examples are [9, 28]. Furthermore, 24% of the citing studies uses a survey among SE practitioners, for example [18, 22, 45, 69, 75]. Ten percent is based on interviews with SE practitioners, such as [20, 41, 42, 50]. Seven percent contains a literature review, for example [12, 45, 73]. Another 7% conducts an experiment [33, 62], case study [49, 59], or field study [9, 10].

Only three out of 70 studies explicitly answer a question from the initial Microsoft study: The second sub-table in Table 2 shows that only 3 studies (4%) explicitly refer their research question to an initial Microsoft one: [16, 28, 33]. Nine studies (13%) partly try to answer a MS question: [8–10, 30, 52, 62, 64, 65, 70]. 29 studies (39%) refer to the original Microsoft study because they used it as an example for their own study [17, 59], either with regard to the study design [20, 22, 29, 37, 46, 48, 67], the rating approach (Kano) [51, 61], or the card sorting technique [19, 54, 60, 63]. Furthermore, a large part (38 studies, 54%) of the citing studies simply refers to the original Microsoft study in a simple related work way.

A majority of citing studies is about Software Analytics, Testing related studies, and SE Process. The third sub-table shows that most cited studies are about software analytics, often combined with a focus on the role of the software engineer and its perceptions, e.g. [42, 51]. In other cases the emphasis on software analytics is
combined with a more technical focus on machine learning, e.g. [21, 48]. Other studies within the topic software analytics are about a variety of methods, tools, and techniques [2, 3, 11, 14, 15, 27, 38, 47, 55, 71–73]. Many of the studies that cite the Microsoft study—and which are often quoted themselves—relate to testing or test automation. Fifteen studies (21%) are about testing [8–10, 13, 23, 24, 33, 45, 66], debugging [80] and code review [25, 46].

12 studies (17%) handle SE process related topics, such as productivity of software engineers [52], visualization [6, 31], and continuous delivery [74, 76]. In addition, studies also relate to continuous delivery pipelines and pipeline automation [74, 78]. Another frequent topic in citing studies is data and models, including aspects of cloud development [32, 49, 55]. Driven by a tendency toward automation of pipelines, software generates a large amount of data. Many different data sources—such as version control systems, peer code review systems, issue tracking systems, mail archives—are available for mining purposes [29, 79].

34% of the cited studies includes some form of Machine Learning. One third of the citing papers do include some form of machine learning (ML), ranging from applying a ML technique for analysis purposes to coming up with examples of the application of ML in practice. As the fourth sub-table in Table 2 shows, 8 studies include examples of applications of ML in practice, e.g. [11, 41, 55]. Text related techniques such as NLP occur 5 times, e.g. [23, 61], ensemble techniques 3 times [30, 37, 66], and instance-based and deep learning both 2 times [14, 21, 27, 48]. Four other techniques—neural networks, clustering, decision trees, and regression—occur one time. Perhaps this finding supports a trend that is visible in SE research, where more and more machine learning techniques are being used in SE analyzes and vice versa, also called AI-for-Software-Engineering [1, 40, 53].

13% are about the cultural aspects of software engineering. Software analytics is an area of extensive growth [56]. The original Microsoft 2014 study influenced ongoing research, looking at the 136 papers citing it gives the impression that it certainly did inspire other researchers and practitioners in setting up studies on software developers needs. Nine studies (13%) of the citing studies are about cultural aspects of software engineering, such as topic selection in experiments [58], characteristics of software engineers [20, 50, 67], causes for frustration [19], or challenges for software engineers [29, 63, 69].

3 STUDY DESIGN

Our study design comprises of two parts. In part one, we replicate the original Microsoft study at ING. We follow the step-by-step procedure prescribed in the original study, with slight modifications appropriate for our context. Figure 1 depicts the research methodology we followed; the figure is an exact copy of the approach used in the original Microsoft 2014 study with numbers from our study. In the next step, we compare the questions identified in the Microsoft study to ours for similarities and differences including addition of new questions and removal of previous questions to answer our research questions.

This figure is a copy from the original Microsoft 2014 study, with numbers from our study. The figure was re-used with permission of the Microsoft 2014 study authors.

Figure 1: Overview of the research methodology

3.1 The Initial Survey

We sent the initial survey to 1,002 ING software engineers randomly chosen from a group of 2,342 employees working within the IT department of ING in May 2018. Unlike the Microsoft study, we did not offer any reward to increase the participation. This is a deviation from the original study but aligns with the policy of ING. Out of the 1,002 engineers 387 engineers started the survey, 271 of them even filled the demographics but stopped when asked to write questions. In the end, we received 336 questions from 116 responses for a response rate of 11.6%. Table 3 shows the distribution of responses across discipline and role.

3.2 Coding and Categorization

Next we did an open card sort to group 336 questions into categories. Our card sort was open, meaning that we coded independently from the Microsoft study. To create independent codes, the first author who did a majority of the coding did not study the Microsoft paper before or during the replication. The other authors knew the paper from before and merely skimmed the methodology section for replication.

We let the groups emerge and evolve during the sorting process. This process comprised of three phases. In preparation phase, we created a card for each question. Questions 1 to 40 were tagged by
the second author. Questions 41 to 80 were tagged by the fourth author. Questions 81 to 90 were tagged by both the second and the fourth author. The tags of questions 1 to 90 were discussed by both the second and fourth author and based on their discussion final tags were prepared. The remaining questions 91 to 336 were then tagged by the first author, based on the tags from the previous step. We discarded cards that made general comments on software development and did not inquire any specific topic.

In execution phase, cards were sorted into meaningful groups and were assigned a descriptive title. Similar to the Microsoft study, the questions were not easy to work with; many questions were same or similar to one another, most were quite verbose while others were overly specific. We distilled them into a set of so-called descriptive questions that more concisely describe each category (and sub-category). In this step, out of the 336 questions, 49 questions were discarded and the remaining 287 questions were divided into 35 sub-categories. An example of reaching descriptive question is presented below:\footnote{A closed balloon indicates a respondent question; an open balloon indicates a descriptive question.}

\begin{itemize}
\item "What factors affect the composition of DevOps teams?"
\item "Would it be better to create specialized development teams instead of DevOps teams?"
\item "What is your idea of an ideal team that should develop software? How many and what kind of people should be part of it?"
\end{itemize}

Finally, in analysis phase, we created abstract hierarchies to deduce general categories and themes. In total, we created 171 descriptive questions, a full list of which is available in the technical report [4].

3.3 The Rating Survey

We created a second survey to rate the 171 descriptive questions. We split the questionnaire into eight component blocks (similar to the Microsoft study) and sent component blocks to potential respondents. The idea behind using the split questionnaire survey design is to avoid low response rate. Each participant received a block of questions along with a text "In your opinion, how important is it to have a software data analytics team answer this question?"

Finally, to rank each question, we dichotomized the ordinal Kano scale avoiding any scale violations [44]. We computed the following percentages for each descriptive question:

\begin{itemize}
\item Percentage of 'Essential' responses among all the responses:
\item Percentage of 'Essential' and 'Worthwhile' responses among all the responses (to which we refer as \textit{Worthwhile}):
\item Percentage of 'Unwise' responses among all the responses:
\end{itemize}

We rank each question based on the above percentages, with the top rank (\#1) having the highest percentage in a dimension (Essential, Worthwhile, or Unwise). Table 5 and Table 6 presents the most desired (Top 10 Essential, Top 10 Worthwhile) and the most undesired (Top 10 Unwise) descriptive questions. For all 171 questions and their rank, see the technical report [4].

3.3.1 Top-Rated/Bottom-Rated Questions. Finally, to rank each question, we dichotomized the ordinal Kano scale avoiding any scale violations [44]. We computed the following percentages for each descriptive question:

\begin{itemize}
\item Percentage of 'Essential' responses among all the responses: \textit{Essential}
\item Percentage of 'Essential' and 'Worthwhile' responses among all the responses (to which we refer as \textit{Worthwhile}):
\item Percentage of 'Unwise' responses among all the responses: \textit{Unwise}
\end{itemize}

We rank each question based on the above percentages, with the top rank (\#1) having the highest percentage in a dimension (Essential, Worthwhile+, or Unwise). Table 5 and Table 6 presents the most desired (Top 10 Essential, Top 10 Worthwhile+) and the most undesired (Top 10 Unwise) descriptive questions. For all 171 questions and their rank, see the technical report [4].

3.3.2 Rating by Demographics. Unlike the Microsoft study, we did not have employee database to rank responses based on demographics, and privacy regulations prevented us from asking people-related aspects such as years of experience (another deviation from the original study). Nonetheless, in both the initial and the rating survey, we asked the following professional background data from the participants:

\begin{itemize}
\item \textit{Discipline}: Participants were asked to indicate their primary working area: Development, Test, Project Management, Other Engineer (e.g. architect, lead), or Other Non-Engineer (only one selection was possible).
\item \textit{Current Role}: Participants were asked to indicate their current role: Individual Contributor, Lead, Architect, Manager, Executive, or Other (more selections were possible).
\end{itemize}

To investigate the relations of descriptive questions to professional background (discipline or current role), we built stepwise logistic regression models. We build our own models since the referenced study did not share scripts to run statistical tests although we did follow their procedure as is. Stepwise regression eliminated professional backgrounds that did not improve the model for a given question and a response. In addition, we removed professional backgrounds for which the coefficient in the model was not statistically significant at p-value < 0.01. For each of the 171 questions, we built...
We noticed that some of our categories directly match the Microsoft ware engineers want data scientists to answer. Replicating the As a preliminary analysis, we start by looking at the similarities and differences in the broader themes or categories in both the studies. Then for each theme, we see how the prominent questions in ING compare against the prominent questions at Microsoft.

To make the comparison systematic, we followed a two-step approach. First, we ran word count on the questions from both the companies presenting a text-based comparison to identify broad differences. Further, the first two authors manually analyzed top 100 essential questions from the two companies in detail. The authors drew affinity diagrams using Microsoft questions (see Figure 2) and appended related questions from ING to it. In case no cluster fits a question, a new cluster is created. This resulted in three types of clusters: match and no match (addition of ING questions and deletion of Microsoft questions). Analyses of the three clusters and the frequency distribution of questions (in addition to the previous three analyses) present insights into our research question.

4 RESULTS

The original Microsoft study came up with 145 questions that software engineers want data scientists to answer. Replicating the original study at ING, we identified 171 data science questions.

This section presents a comparison of the two sets of questions based on category, type of questions within categories, top-rated questions, bottom-rated questions, and questions relevant for different demographics. Next, we compare the questions from the two companies using word count and affinity diagrams to answer our research question.

4.1 Categories

We noticed that some of our categories directly match the Microsoft study. Other categories, however, can be mapped to one or more categories of the Microsoft study. No new emergent category in our study indicates that broadly there are no differences between the questions for a software-defined enterprise from a software company. For further analysis, we map our categories on to theirs, details on which are available in Table 4.

Next, we explore the essential questions at ING and their distinguishing link to the questions from the Microsoft study.

4.1.1 Bug Measures (BUG). The essential questions at ING relate to the effort spent on bugs, methods to prevent security-related vulnerabilities, and the relationship between bugs and specific ING-related development platforms.

- "How does the effort spent on fixing vulnerabilities and bugs relate to effort spent on writing software correctly from the start?"
- "What methods are most effective in preventing security-related vulnerabilities or bugs from being introduced in software code?"

4.1.2 Development Practices (DP). The performance and productivity of DevOps teams was found in a number of questions including team happiness and work pleasure (# 1 question), ways of decision making, non-overlapping development activities in the same environment, product ownership and business responsibilities, licensing of tools, and the choice of a data modeling approach.

- "What factors affect the performance and productivity of DevOps teams with regard to team happiness and pleasure in your work?"
- "What factors affect the performance and productivity of DevOps teams with regard to evidence-based decision-making versus decision-making based on expert opinions?"
- "What factors affect the performance and productivity of DevOps teams with regard to simultaneous slow and fast developments at the same time in the same environment?"

4.1.3 Development Best Practices (BEST). This category emphasized best (or worst) development practices relating to technology selection, effectiveness, and choice of tools.

- "How can we make sure that we build for re-usability and scalability?"
- "What factors affect high performance teams?"
- "When do you remove an old module that you think is not being used anymore?"

4.1.4 Testing Practices (TP). Questions here ranged from automated test data generation, on-demand provisioning of test environments, testing of high volumes, to question like "should we let loose Chaos Monkey" [35].

- "To what extent does on-demand provisioning of development and test environments, including up-to-date data affect delivery of software solutions?"
- "What factors affect performance testing on high data volumes?"
- "How can a system for (semi) automated CRUD test data generation improve delivery of software solutions?"
- "Should we let loose Chaos Monkey, like Netflix?"

4.1.5 Evaluating Quality (EQ). This category included questions on code analysis, ways to assess quality of software code, and effectiveness of testing practices.

- "What methods can be applied to analyze whether software code is working as expected?"
- "To what extent does testability of software code affect the quality of code?"

4.1.6 Customers and Requirements (CR). The essential questions related to measure customer value, requirement validation, and the use of formal models. Notably, questions relating to development trade-offs such as backward compatibility or the impact of testing in production appeared in the Microsoft study but not ours.

- "How to measure the customer value of a software product?"
- "How can requirements be validated before starting actual software development?"
- "How can user feedback be integrated in an efficient and effective way into software code?"

4.1.7 Software Development Lifecycle (SL). Questions in this category related to the effectiveness and performance in lead time,
### Table 4: ING categories and questions mapped on to the 12 Microsoft categories

<table>
<thead>
<tr>
<th>Category</th>
<th>ING 2019 Study</th>
<th>Microsoft 2014 Study</th>
<th>Difference ING 2019 compared to MS 2014</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teams and Collaboration</td>
<td>TC 14</td>
<td>16</td>
<td>↓ 8%</td>
</tr>
<tr>
<td></td>
<td>Subcategories</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4%</td>
<td>7%</td>
<td></td>
</tr>
<tr>
<td>Testing Practices</td>
<td>TP 32</td>
<td>15</td>
<td>↑ 5%</td>
</tr>
<tr>
<td></td>
<td>9%</td>
<td>3%</td>
<td></td>
</tr>
<tr>
<td>Services</td>
<td>SVC 3</td>
<td>1%</td>
<td>↓ 1%</td>
</tr>
<tr>
<td></td>
<td>1%</td>
<td>2%</td>
<td></td>
</tr>
<tr>
<td>Reuse and Shared Components</td>
<td>RSC 2</td>
<td>1%</td>
<td>↓ 1%</td>
</tr>
<tr>
<td></td>
<td>3%</td>
<td>3%</td>
<td></td>
</tr>
<tr>
<td>Customers and Requirements</td>
<td>CR 9</td>
<td>3%</td>
<td>↓ 6%</td>
</tr>
<tr>
<td></td>
<td>3%</td>
<td>2%</td>
<td></td>
</tr>
<tr>
<td>Software Development Lifecycle</td>
<td>SL 4</td>
<td>2%</td>
<td>↓ 2%</td>
</tr>
<tr>
<td></td>
<td>2%</td>
<td>4%</td>
<td></td>
</tr>
<tr>
<td>Development Practices</td>
<td>DP 51</td>
<td>15%</td>
<td>↑ 15%</td>
</tr>
<tr>
<td></td>
<td>15%</td>
<td>14%</td>
<td></td>
</tr>
<tr>
<td>Bug Measurements</td>
<td>BUG 5</td>
<td>3%</td>
<td>↓ 2%</td>
</tr>
<tr>
<td></td>
<td>2%</td>
<td>3%</td>
<td></td>
</tr>
<tr>
<td>Productivity</td>
<td>PROD 29</td>
<td>9%</td>
<td>↓ 16%</td>
</tr>
<tr>
<td></td>
<td>9%</td>
<td>8%</td>
<td></td>
</tr>
<tr>
<td>Evaluating Quality</td>
<td>EQ 38</td>
<td>11%</td>
<td>↑ 5%</td>
</tr>
<tr>
<td></td>
<td>11%</td>
<td>6%</td>
<td></td>
</tr>
<tr>
<td>Development Best Practices</td>
<td>BEST 49</td>
<td>15%</td>
<td>↑ 7%</td>
</tr>
<tr>
<td></td>
<td>15%</td>
<td>7%</td>
<td></td>
</tr>
<tr>
<td>Software Development Process</td>
<td>PROC 46</td>
<td>14%</td>
<td>↑ 8%</td>
</tr>
<tr>
<td></td>
<td>14%</td>
<td>7%</td>
<td></td>
</tr>
<tr>
<td>Discarded Cards</td>
<td>49</td>
<td>7%</td>
<td>↑ 8%</td>
</tr>
<tr>
<td></td>
<td>15%</td>
<td>7%</td>
<td></td>
</tr>
<tr>
<td>Total Cards</td>
<td>337</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>64</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Questions |

- "What factors affect the performance of DevOps teams and the quality of software code with regard to quantity and quality of environments?"
- "Are developers working in an open space with several teams more effective or less than developers working in a room with just their team?"
- "What makes a great coder? What aspects affect the performance of DevOps teams and the quality of software with regard to characteristics of an individual software engineer?"
- "What factors affect estimation of lead time, cost, and quality of software deliveries?"
- "Does a focus on quick release of features and bug fixes into production help to achieve confidence and agility?"

### Figure 2: Analysis of ING 2019 and MS 2014 questions.

cost, and quality (same as the Microsoft study) but also questions relating to security and risk from a management perspective.

- "What factors affect providing new technologies to consumers, and can implementations of new technology be internally and externally benchmarked?"
- "How can we improve the deployment process in DevOps teams?"

#### 4.1.8 Software Development Process (PROC).

Our questions related to development processes, technology selection, and deployment of software solutions. At Microsoft, in contrast, questions related to the choice of software methodology (e.g., ways in which agile is better than waterfall? and benefits of pair programming). We also noticed that at ING topics like the effects of automated continuous delivery pipeline popped up which were not seen in the Microsoft study.

- "How can we improve the deployment process in DevOps teams?"
- "Does a focus on quick release of features and bug fixes into production help to achieve confidence and agility?"

#### 4.1.9 Productivity (PROD).

This category had questions on the productivity of DevOps teams - but also individual developers, ranked essential. Notably, questions related to the measurement of individual developers (e.g., the questions mentioned below regarding "great coder" and "open spaces") were often ranked "Unwise". Quite unlike the Microsoft study, where respondents considered these questions as unwise, engineers at ING had a mixed opinion.

- "What makes a great coder? What aspects affect the performance of DevOps teams and the quality of software with regard to characteristics of an individual software engineer?"
- "What factors affect estimation of lead time, cost, and quality of software deliveries?"

### 4.1.10 Teams and Collaborations (TC).

Essential questions here are typically about dependencies between teams, team composition, team maturity, and knowledge sharing among teams.

- "To what extent do dependencies on other teams affect team performance?"
- "How does team maturity affect code quality and incidents?"
- "What factors affect the composition of DevOps teams?"

### 4.2 Top-Rated Questions

Table 5 shows top 15 "Essential" and top 10 "Worthwhile or higher" questions. Interestingly, only two out of the top 15 "Essential" questions were a part of the top 10 "Worthwhile or higher" questions and none vice-versa. This potentially means that our participants are more pronounced and opt for Essential or Worthwhile when they feel so. Culture can be another possible reason since all participants at ING are located in one country while participants of the Microsoft study were more diverse [34].

Our top questions are on development processes, technology selection, and deployment of software solutions. The top related questions at Microsoft, in contrast, relates to the choice of software methodology (e.g., ways in which agile is better than waterfall? and benefits of pair programming). We also noticed that in our study topics like the effects of automated continuous delivery pipeline popped up which were not seen in the Microsoft study.

Notably, a large fraction of the top 20 "Essential" or "Worthwhile or higher" questions at Microsoft (9 out of 20; including top 2) relates to customers. This suggests that for Microsoft customer benefit is most important or perhaps one of the most important question. Our study, in contrast, paints a very different picture.
Only two out of the 336 questions in the initial survey mentioned the word “customer” and only one of those questions made it to the top-20 (Q58 “How to measure the customer value of a software product” at rank 10 “Essential”). This question, in line with the Microsoft study, marked with icon ★, in Table 5.

Another eight “Essential” or “Worthwhile or higher” questions in the Microsoft study (marked with icon ⋆) focus on the engineer and the effects of software development practices and processes on her work. In our study, we identified nine questions with this icon. In addition to the focus on individual engineer, many of the questions in our study relates to the concept of the DevOps team. Overall, it seems that Microsoft has a big focus on customer while ING emphasizes on the engineering team itself. Finally, seven questions in the Microsoft study (marked with the icon ⋆) were about quality-related issues (same as ours with eleven questions).

### 4.3 Bottom-Rated Questions

Table 6 shows the top 10 unwise questions. The most "Unwise" question (Q27) at ING is the use of domain-specific language for use by non-experts. In the Microsoft study, the top five "Unwise" questions were all about a fear that respondents had of being rated. This effect can be seen in our study too (two of the top ten unwise questions - Q161 and Q30 - relate to measuring the performance of individual engineers), but not nearly as strongly as in the Microsoft study. Respondents in our study are torn on this topic; Q161 and Q30 are ranked as "Unwise" by respectively 22.2% and 20.0% of the respondents, but also ranked as "Essential" by another group of 44.4% and 40.0% of the respondents. Also, it was interesting to see that measuring and benchmarking time to market of software solutions (Q38) is one of the top 10 unwise questions. It indicates resistance against comparing departments based on key performance indicators like the time to market.

### 4.4 Rating by Demographics

Table 7 shows essential questions for different disciplines (Developer, Tester, Project Management) and roles (Manager, Individual Contributor, Architect). The complete inventory of questions for “Worthwhile or higher” and “Unwise” responses is present in the technical report [1].

#### 4.4.1 Discipline

Microsoft study showed tester as a specific discipline mainly interested in test suites, bugs, and product quality. We do not see the discipline “tester” in our study. This can be seen in Table 7 in which overall scores relating to “Test” are low and highest for "Development". Software engineers in the DevOps teams at ING consider themselves to be generic developers, and testing is an integrated part of the discipline "developer". Both developers and
Table 6: Questions with the highest "Unwise" percentages (opposition).

<table>
<thead>
<tr>
<th>Question</th>
<th>Category</th>
<th>Percentages</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q27 How can software solutions in one common language be developed in a way that it is applicable to every person, regardless of ones interest in software development?</td>
<td>CR</td>
<td>22.2% 55.6% 33.3%</td>
<td>121 152 1</td>
</tr>
<tr>
<td>Q39 How can Windows-server images be created in order to facilitate testing within a continuous delivery pipeline?</td>
<td>DP</td>
<td>9.1% 45.5% 27.3%</td>
<td>162 163 2</td>
</tr>
<tr>
<td>Q170 Why do many developers focus on the newest of the newest? Why don’t they leave this to a small group in order to use time and effort more efficiently?</td>
<td>DP</td>
<td>21.1% 47.4% 26.3%</td>
<td>128 161 3</td>
</tr>
<tr>
<td>Q161 What makes a great coder? What aspects affect the performance of DevOps teams and the quality of software with regard to characteristics of an individual software engineer?</td>
<td>PROD</td>
<td>44.4% 66.7% 22.2%</td>
<td>32 128 4</td>
</tr>
<tr>
<td>Q134 What factors affect TFS (Team Foundation Services) - a Microsoft product that provides source code management - with regard to working with automated pipelines?</td>
<td>BEST</td>
<td>38.9% 72.2% 22.2%</td>
<td>54 118 4</td>
</tr>
<tr>
<td>Q30 How can the performance of individual software engineers be benchmarked internally ING and externally with other companies?</td>
<td>PROD</td>
<td>40.0% 50.0% 20.0%</td>
<td>44 157 6</td>
</tr>
<tr>
<td>Q27 To what extent does changing of requirements during development affect the delivery of software solutions?</td>
<td>PROC</td>
<td>12.5% 68.8% 18.8%</td>
<td>150 124 7</td>
</tr>
<tr>
<td>Q21 How can PLI software code be converted to Cobol code, while maintaining readability of the code in order to simplify an application environment?</td>
<td>BEST</td>
<td>18.2% 36.4% 18.2%</td>
<td>140 169 8</td>
</tr>
<tr>
<td>Q38 How can we measure the time to market of software solutions delivered within a department at ING in order to benchmark the performance of that department against others?</td>
<td>DP</td>
<td>9.1% 54.5% 18.2%</td>
<td>162 155 8</td>
</tr>
<tr>
<td>Q149 What factors affect the use of machine learning in software development over a period of ten years?</td>
<td>DP</td>
<td>16.7% 66.7% 16.7%</td>
<td>143 128 10</td>
</tr>
<tr>
<td>Q28 How can the cost of data be identified, in order to sign a price tag to data?</td>
<td>DP</td>
<td>5.6% 50.0% 16.7%</td>
<td>168 157 10</td>
</tr>
</tbody>
</table>

Table is sorted on Rank Unwise.

Table 7: Statistically significant rating differences for the response "Essential" by professional background.

<table>
<thead>
<tr>
<th>Question</th>
<th>Category</th>
<th>Response</th>
<th>Dev</th>
<th>Discipline</th>
<th>Test</th>
<th>PM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q2 Are there practices of good software teams from the perspective of releasing software solutions into production?</td>
<td>PROC</td>
<td>Essential</td>
<td>66.7%</td>
<td>5.6%</td>
<td>11.1%</td>
<td></td>
</tr>
<tr>
<td>Q21 How can PLI software code be converted to Cobol code, while maintaining readability of the code in order to simplify an application environment?</td>
<td>BEST</td>
<td>Essential</td>
<td>66.7%</td>
<td>4.8%</td>
<td>0.0%</td>
<td></td>
</tr>
<tr>
<td>Q28 How can the cost of data be identified, in order to sign a price tag to data?</td>
<td>DP</td>
<td>Essential</td>
<td>72.7%</td>
<td>0.0%</td>
<td>0.0%</td>
<td></td>
</tr>
<tr>
<td>Q46 How do static code analysis tools such as Fortify and Sonar influence the quality of software engineering products?</td>
<td>BEST</td>
<td>Essential</td>
<td>36.6%</td>
<td>0.0%</td>
<td>27.3%</td>
<td></td>
</tr>
<tr>
<td>Q88 To what extent does testability affect the quality of software code?</td>
<td>EQ</td>
<td>Essential</td>
<td>68.4%</td>
<td>0.0%</td>
<td>0.0%</td>
<td></td>
</tr>
<tr>
<td>Q89 How does time spent - in terms of full-time versus part-time - of a Scrum master affect the delivery of software solutions?</td>
<td>PROC</td>
<td>Essential</td>
<td>66.7%</td>
<td>5.6%</td>
<td>11.1%</td>
<td></td>
</tr>
<tr>
<td>Q95 To what extent do dependencies on other teams affect team performance?</td>
<td>TC</td>
<td>Essential</td>
<td>68.4%</td>
<td>0.0%</td>
<td>0.0%</td>
<td></td>
</tr>
<tr>
<td>Q97 How does documentation during software maintenance affect delivery of software solutions?</td>
<td>TP</td>
<td>Essential</td>
<td>50.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td></td>
</tr>
<tr>
<td>Q110 What factors affect data analytics with regard to the use of external sources - such as market research reports and follow market trends - and let individual teams handle their local evolution?</td>
<td>PROC</td>
<td>Essential</td>
<td>66.7%</td>
<td>5.6%</td>
<td>11.1%</td>
<td></td>
</tr>
<tr>
<td>Q162 What methods are most effective in preventing security related vulnerabilities or bugs from being introduced in software code?</td>
<td>BUG</td>
<td>Essential</td>
<td>68.4%</td>
<td>0.0%</td>
<td>0.0%</td>
<td></td>
</tr>
</tbody>
</table>

The professional background with the highest rating is highlighted in **bold**. Questions that are also in Table 5 are shown in *italics*. The role "Manager" includes the responses for "Manager" and "Lead".

testers are for example significantly interested in the testability of software code, and the quality of software related to an agile way of working and working in DevOps teams. Other findings relate to developers being significantly interested in team performance, e.g.
regarding practices of good software teams from the perspective of releasing software into production, the use of data analytics to improve individual teams, and dependencies on other teams.

4.4.2 Role. More individual contributors (e.g., developers) than managers are interested in good practices for software teams to release software into production. More managers than individual contributors, on the other hand, are interested in how software can help realize new policies and changes in the way of working, the relationship between documentation and maintenance of software, and to what extent the use of static code analysis tools such as Fortify and Sonar can affect the quality of software.

4.5 Compare ING and Microsoft Questions

A comparison of the top 15 words from each company (see Table 8) shows that a majority of the popular themes are same (e.g., code, test, software, and quality.). The subtle differences, however, exists relating to rank (words in italics do not make it to top-15 in another company) and use of the word in another company (underlined).

A subset of these differences can be attributed to differences in terminology. For instance, Microsoft uses terms like employee/employees and team/teams, while its equivalent at ING are team/squad and engineer. Apart from this, Microsoft focused more on bugs, cost, time, customers, and tools while ING employees talked about version, problem, systems, process, and impact.

Next, we inferred 24 themes from the clusters in the affinity diagram organically merging into three broad categories: relating to code (like understanding code, testing, quality), developers (individual and team productivity) and customers (note that while customers did not make it to the top-15 in another company) and use of the word in another company (underlined).

The deployment-related questions at ING might be a result of the adoption of continuous delivery as a service. Surprisingly, we did not see such questions. Rather, we see questions relating to the functional aspects of agile and automated testing. Another subtle difference between the two companies is relating to code size. While not stated explicitly, from the nature of questions, it seems that the software teams at Microsoft are dealing with a large legacy codebase. This was reflected in questions relating to team awareness, code monitoring, backward compatibility, and refactoring. Such questions, however, did not occur in ING. Other than the above, we saw cloud-related questions appearing in the Microsoft study only, while deployment-related questions appeared in ING only.

In a nutshell, the core software development challenges of ING are consistent with Microsoft. There are although some nuanced differences which relate to the evolution of software market in the last five years as well as differences in the characteristics of the two companies.

### Table 8: Top 15 words from questions at ING and Microsoft

<table>
<thead>
<tr>
<th>Word / Coding</th>
<th>Count</th>
<th>Word / Coding</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>code / coding</td>
<td>48 (19%)</td>
<td>testing / debugging</td>
<td>92 (14%)</td>
</tr>
<tr>
<td>test / tests / testing</td>
<td>39 (16%)</td>
<td>code / coding</td>
<td>87 (13%)</td>
</tr>
<tr>
<td>software</td>
<td>31 (13%)</td>
<td>software</td>
<td>76 (11%)</td>
</tr>
<tr>
<td>employee / employees</td>
<td>16 (6%)</td>
<td>team / squad</td>
<td>72 (11%)</td>
</tr>
<tr>
<td>quality</td>
<td>13 (5%)</td>
<td>development</td>
<td>62 (9%)</td>
</tr>
<tr>
<td>bugs</td>
<td>13 (5%)</td>
<td>version / library</td>
<td>39 (6%)</td>
</tr>
<tr>
<td>development</td>
<td>12 (5%)</td>
<td>data</td>
<td>37 (6%)</td>
</tr>
<tr>
<td>cost</td>
<td>11 (4%)</td>
<td>incident, issue, problem</td>
<td>36 (5%)</td>
</tr>
<tr>
<td>team / teams</td>
<td>11 (4%)</td>
<td>security / risk</td>
<td>34 (5%)</td>
</tr>
<tr>
<td>time</td>
<td>10 (4%)</td>
<td>system / systems</td>
<td>34 (5%)</td>
</tr>
<tr>
<td>customer / customers</td>
<td>9 (4%)</td>
<td>quality</td>
<td>34 (5%)</td>
</tr>
<tr>
<td>impact</td>
<td>9 (4%)</td>
<td>production</td>
<td>21 (3%)</td>
</tr>
<tr>
<td>productivity</td>
<td>9 (4%)</td>
<td>engineer</td>
<td>14 (2%)</td>
</tr>
<tr>
<td>project</td>
<td>9 (4%)</td>
<td>process</td>
<td>14 (2%)</td>
</tr>
<tr>
<td>tools</td>
<td>7 (3%)</td>
<td>impact</td>
<td>13 (2%)</td>
</tr>
</tbody>
</table>

Top 15 words (sorted on count) from Microsoft 2014 and ING 2019 study. Words in the top-15 of one company and not the other are printed in italic. Words in one list and not the other are underlined.

5 DISCUSSION

In this section, we discuss potential explanations for the differences in the list of questions found in our study compared to the Microsoft study. We saw questions eliciting the need of agile in Microsoft study while at ING the questions related to functional aspects. Our hypothesis here is that in the last five years there is a change in the market: while in 2014, the questions on the adoption of agile and automated testing were common, in 2019 agile and automated testing became the norm. We noticed that many questions at Microsoft deals with the scale of legacy code while no such question appeared at ING. One potential explanation for the observation can be that software systems at ING are not of the same scale as Microsoft. Nonetheless, it remains a lesson that in the next 10 years, ING can also be dealing with the complexity of large code base as Microsoft is experiencing today. Finally, some questions appeared in only one organization. We believe that these observations have something to do with the individual practices followed at Microsoft and ING. The deployment-related questions at ING might be a result of the adoption of continuous delivery as a service. Surprisingly, we did not see any finance-related questions in the ING study. ING is a finance-based company and we expected to see some issues relating to both finance and software appear. We noticed that employees often talked about security, but no real finance-related questions appear. One explanation for this observation can be that the data science challenges relating to software development are independent of the actual field to which it is applied. Supporting this argument, 145 questions from Microsoft also did not bring up any product specific details. Another potential explanation can be that through our question we anchored our respondents into asking software development related questions only.
5.1 Implications
One of the key findings of this paper is a list of 171 questions that software engineers in a large, software-driven organization would like to see answered, in order to optimize their software development activities. From this, we see implications both in terms of practice and industry.

From a practical perspective, our study offers a new way of thinking to software development organizations who care about their development processes. The questions originally raised by Microsoft are not just relevant to one of the largest tech companies in the world, but also to large software-defined enterprises active outside the tech-sector proper. Inspired by these questions, an organization may select the most relevant ones, and seek ways to address them. While some questions are fundamentally hard to answer, organizations can make a starting point by collecting relevant data about their development processes. This, then, can help to make the development process itself more and more data-driven. This is exactly how ING intends to use the questions, and we believe companies around the world can follow suit.

From a research perspective, we have seen that the original Microsoft study has generated a series of papers that apply some form of Machine Learning to address the challenges raised in that study. In the research community, AI-for-Software-Engineering is an increasingly important topic, with many papers appearing that seek to apply machine learning to address software engineering problems. Our study aims to add urgency and direction to this emerging field, by highlighting not just which questions can be answered, but which ones should be answered, from a practitioner perspective.

5.2 Threats to Validity
While our study expands the external validity of the original study, the fact remains that the two lists of questions are based on just two companies, which are both large organizations with over 10,000 software developers. Our study highlights relevance to the FinTech sector, but it would be interesting to see further replications, for example in the automotive or health care sector, with different regulatory and additional safety constraints. We expect that many of the questions are also relevant to smaller organizations, especially given the agile way of working at ING. Nevertheless, it will be worthwhile to further explore this.

From a perspective of internal validity, creating codes independent of the prior study is challenging. It is possible that the similarities and differences seen compared to the Microsoft study relates to factors (e.g. researcher bias) other than the actual data. We tried mitigating it by limiting our exposure to the previous study, not involving authors from the Microsoft study, and multiple authors generating codes independently. Nonetheless, these biases are likely to exist.

For reasons of replication, we have used where possible the same survey questions, method of analysis and division into work area and discipline as in the Microsoft study [7]. Apart from positive effects, this choice also had a negative effect with regard to analysis of demographics, mainly due to the fact that ING uses a different way of working, including corresponding roles and team structure, than within Microsoft. Especially mapping the professional background "Discipline" of the original study on the demographic "Discipline" as applied within ING was challenging.

From a perspective of practical validity, the core challenges raised in our study are not only relevant for the Microsoft study, but apply to many software-defined enterprises outside the tech-sector proper. As a result, these questions are also relevant to smaller organizations, especially given the emerging field of developer-driven software development.

6 CONCLUSION
Conducted at ING—a software-defined enterprise providing banking solutions—the study presents 171 questions that software engineers at ING would like data scientists to answer. This study is a replication of a similar study at software company Microsoft, which resulted in 145 questions for data scientists. Further, we went a step beyond to investigate the applicability of Microsoft’s questions in ING, as well as changes in trends over the last five years.

We compared the two lists of questions and found that the core software development challenges (relating to code, developer, and customer) remain the same. Nonetheless, we observed subtle differences relating to the technology and software process developments (e.g., currently the debate about agile versus waterfall is now largely absent) and differences in the two organizations (e.g., Microsoft’s focus on solving problems with a large code bases and ING’s challenges with continuous deployment). We complete our analysis with a report on the impact Microsoft 2014 study generated, also indicating the impact that our study is capable to generate.

A thorough understanding of key questions software engineers have that can be answered by data scientists is of crucial importance to both the research community and modern software engineering practice. Our study aims to contribute to this understanding. We call on other companies, large and small, to conduct a similar analysis, in order to transform a software engineering into a data-driven endeavor addressing the most pressing questions.

ACKNOWLEDGMENTS
The authors thank ING and all participant engineers for sharing their experience and views with us. We thank the authors of the
original Microsoft study for the permission to reuse their research design figure.

REFERENCES


