Ramp-up Journey of New Hires

Tug of War of Aids and Impediments

Ayushi Rastogi[*], Suresh Thummalapenta[†], Thomas Zimmermann[‡], Nachiappan Nagappan[‡] and Jacek Czerwonka[†]
[*]Indraprastha Institute of Information Technology, New Delhi, India
[†]Microsoft Corporation, Redmond, USA
[‡]Microsoft Research, Redmond, USA
ayushir@iiitd.ac.in, {suthumma, tzimmer, nachin, jacekcz}@microsoft.com

Abstract*—Hiring top talent is essential for any software company's success. After joining the company, new hires often spend weeks or months before making any major contribution and attaining the same productivity level as existing employees. We use the term *ramp-up journey* to refer to this transition of new hires from novice to experts. There can be several factors, such as lack of experience or lack of familiarity with processes unique to the new company, which influence the ramp-up journey. To understand such aids and impediments in the ramp-up journey, we conducted a study by analyzing data extracted from version control systems of eight large and popular product groups in Microsoft with several thousand software developers. In particular, we studied two aspects of the ramp-up journey. First, we studied time taken to make the first check-in into the version control system, an important milestone in the ramp-up journey indicating the first contribution. Second, we analyzed the time taken to reach the same productivity level as existing employees in terms of check-ins. We further augmented our quantitative study with qualitative results derived by surveying 411 professional developers. Our study produced promising results, including factors such as having a mentor, prior knowledge of required skill sets, and proactively asking questions, that could help reduce the ramp-up journey of new hires.

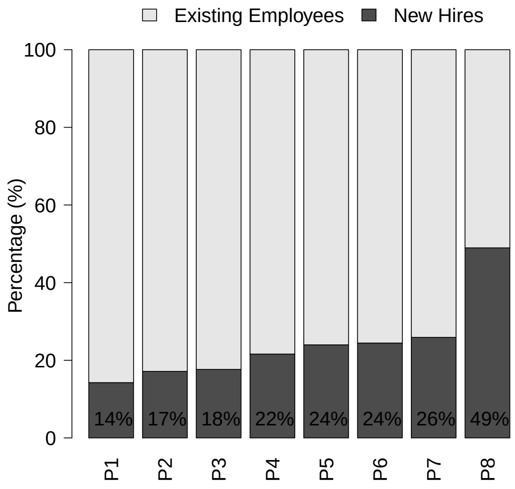

Fig. 1. Percentage of new hires in different product groups

I. Introduction

Hiring new talent is one of the core competencies of any software company, to meet evolving business requirements and to stay ahead of its competitors. For instance, over the past decade, Microsoft hired several thousand software developers each year. These new hires range from fresh college graduates without any prior industry experience to professional developers with several years of experience. Soon after onboarding the company, new hires undergo rigorous training in getting familiar with not only their assigned project, but also with processes and the overall culture of the company. Therefore, new hires take some time before making any contribution to their project and becoming as productive as existing employees. We use the term *ramp-up journey* to refer to this time period spent by new hires in transitioning from novice to experts and becoming as productive as existing employees.

There exist several reasons why ramp-up journey of new hires can span up to several weeks or months. In the case of college graduates, despite of best curricula, there are still gaps between what graduates learn in the college and what they need to know to be productive in a typical work environment [1] [2]. For instance, graduates are trained in various skills such as programming and development methodologies (Agile or Extreme Programming). However, college graduates often lack communication and teamwork skills, and are also not prepared to deal with various aspects such as complex development processes, legacy code, and tight deadlines [3]. On the other hand, experienced professionals, although familiar with some aspects such as working under tight deadlines, also face challenges in a new company. The primary reason is that different companies use different processes, technologies, and tools. For example, IBM primarily uses Rational Team Concert [4] as a version control system, whereas Microsoft primarily uses Source Depot, Team Foundation Server [5], or Git [6]. Therefore, experienced professionals need to master new processes and technologies before becoming as productive as existing employees [7].

To understand the aids and impediments in the ramp-up journey of newly hired software developers, in this paper, we conducted a large-scale study that uses mixed data analysis. Our analysis combines software engineering data extracted from version control systems and qualitative data from surveys and interviews. We analyzed recent releases of eight large and popular product groups in Microsoft with several thousand software developers. These eight product groups account for the majority of engineering workforce at Microsoft. We observed that 18-49% of all software developers in product teams considered in our study are new hires (refer Figure 1). In Figure 1, the horizontal axis shows different product groups and the vertical axis shows percentage of new hires for the duration of product release under analysis. We use P1 to P8 to identify different product groups and to anonymize the results for confidentiality. Percentage of new hires in product groups as high as 49% makes it interesting to understand factors that influence the time it takes for new hires to become productive. To gain deeper understanding of the factors that influence the ramp-up journey of new hires with different experience levels, we used three levels to represent new hires: entry, middle, and senior. 'Entry Level' represents developers such as college graduates without any prior industry experience. Middle level represents developers with 1-3 years of experience or have higher qualification such as doctorate degree. Finally, senior level represents developers with more than 3 years of experience.

We analyze the ramp-up journey of new hires on two aspects. The first milestone in the ramp-up journey of new hires is achieved when new hires make first check-in into the version control system. Specifically, we examine check-ins in the master branch or shippable branch. The choice of master or shippable branch ensures that all check-ins analyzed in the study mark significant contribution and are not test check-ins. Thus, first check-in ensures that new hires have attained a basic understanding of the engineering system used by the project and also some basic knowledge of the project to achieve the task. Finally, new hires ramp-up when they attain the productivity level of existing employees. In this study, we measure the time to first check-in and the ramp-up time to analyze the productivity of new hires and understand factors that influences the ramp-up journey.

Our motivation is that the results of our study help fix the problems from a two pronged approach for both industry and academia. From the industry perspective, this information help managers and business analysts fix some of the bottlenecks in the existing processes and improve useful practices to help new employees ramp-up faster, increase morale, improve productivity, etc. From an academic perspective, the study can help faculty understand the skills needed for the students to succeed in the industrial environment. To summarize, in this study, we try to answer two broad sets of research questions:

1) The factors that influence the ramp-up journey of new hires; and
2) The amount of time it takes for new hires to become productive.

To the best of our knowledge, our study is the first that attempts to combine both quantitative and qualitative data to address these questions. There have been prior studies [8] [9] [10] discussed in related work that approach this from different perspectives like interviews, surveys, etc. However, we are the first to quantitatively combine both software engineering data and data from developers' opinions. Noteworthy to add that some of the findings from this study may already be known in the industry, however, there exists no empirical evidence drawn from a systematic analysis of software engineering data. We believe that product teams can use the results from this study to adopt best practices.

Our results indicate that the ramp-up journey is influenced by various factors in the company. We discovered that while having a mentor, prior knowledge of required skill sets, etc. help in increasing the productivity; lack of proper documentation, trying to get access and permissions, etc. reduce the productivity of new hires. We complete the story by presenting a comprehensive list of activities that new hires engage in and present their suggestions to improve the productivity.

II. Related Work

There exists a large body of research on integrating freshly recruited university graduates into the organization. Begel and Simon conducted a qualitative analysis of fresh university graduates. They observed that a large fraction of problems encountered by university graduates is due to their inexperience with the corporate environment [8]. Dagenais et al. provided an initial theory on project landscapes to help new hires familiarize faster with the team [9]. Their study emphasized the relevance of mentor and good documentation to help new hires adjust and perform in team. Sim and Holt interviewed new hires to identify patterns in which they familiarize themselves with the project and the environment, and discussed its implication [11]. Besides these, Ostroff investigated the role of mentoring in the learning process of newcomers [10]. The author observed that newcomers with mentor have a better understanding of organizational issue and practices compared to others. Another class of study provide recommendations to bridge gaps between the understanding of college graduates and the requirements of the industry. Raderwacher and Walia conducted a systematic literature review to identify the most common areas of deficiency in university graduates from academia or industry job perspectives [12]. Similarly, Begel and Simon observed difficulties in the transition from college graduates to experienced software engineers and suggested changes in curricula and software engineering courses [2]. Brechener goes on to recommend courses that might help bridge the expectation gaps between academia and industry [1]. The existing studies have largely focused on understanding problems encountered by fresh university graduates and attended this question from the qualitative analysis perspective. In relation to existing studies, our study makes the following two novel contributions:

1) We study the ramp-up journey of new hires ranging from fresh university graduates to professionals with prior job experience.
2) We quantitatively analyze the software engineering data and augment the results with qualitative analysis.

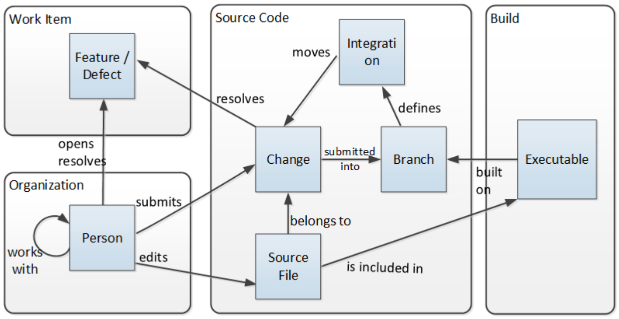

Fig. 2. Data collected by CodeMine

III. Background

CodeMine [13] provides a data collection framework for all major Microsoft development teams. It collects information from source code repositories, irrespective of their format, and stores data on changes, sources, branches, and code integrations in a normalized schema. It does the same for builds, work item repositories, and organizational data. After normalizing into a common schema (see Figure 2), it creates relationship between artifacts. For example, it creates relationship between work items and source changes to capture the changes made in response to a work item. Similarly, it creates relationship between builds and source code to capture the changes which appeared for the first time in a particular build. When the relationships are built, CodeMine exposes all collected and interpreted data as a service. As a result, the tool is ideal for learning about the practices used across teams and developing a set of metrics that can be used for generally characterizing branch structures.

IV. Research Methodology

A. Data Collection

In this section, we describe the data collection methodology for our study. We use a mixed data analysis technique combining software engineering data extracted from version control systems and bug repositories, and qualitative data from surveys and interviews, to answer our research questions. Table I summarizes the products from which we analyzed new employees and the start-end date of the releases under analysis.

To determine the 'new hires' in our analysis, used in the rest of the paper, the employee in the software development role had to satisfy one or more of the following four criteria:

1) The employee is a university recruit or this is the first company of the employee.
2) The employee joined the company as an intern or vendor and converted to a full time position.
3) The employee left the company and joined again after at least a year.
4) The employee worked for other companies in the past.

In point 3, we consider returning employees as new hires because they have to adapt to the rapidly changing technology and practices in the company, in addition to the other factors.

TABLE I
Products and start-end date of the releases

Products	Start Date	End Date
Azure	2011-01-01	2013-12-31
Bing	2009-12-30	2013-11-12
Exchange	2010-10-01	2013-08-31
Office	2011-09-01	2013-08-31
SQL Server	2009-07-01	2012-03-06
Windows	2009-10-22	2013-07-01
Windows Phone	2010-07-03	2014-02-03
Windows Server	2009-10-22	2013-07-01

The rest of the software developers, including internal transfers, are termed as existing employees for this study. Next, we explain below the quantitative and qualitative analysis methodology.

B. Quantitative Analysis

The core part of our quantitative analysis is formed by the CodeMine data [13]. We use the version control system database and the employee information database as the starting point in our analysis. In the ramp-up journey, for new hires to transition from novice to experts, they have to attain the productivity level of existing employees. To do so, we first define a baseline for productivity of software developers in the context of our study.

Software developers perform various activities like writing code, reviewing code, debugging code, etc. The objective of these activities is to generate useful features, provide fixes to existing bugs, etc. In Microsoft, all software developers are expected to make code check-ins into the version control system as part of developing new features or making bug fixes. So, one denomination to measure the contribution of software developers is code check-ins. In this study, we examine various factors related to code check-in to suggest the productivity of software developers.

The first milestone in the ramp-up journey of new hires is when they make their first check-in into the version control system. First check-in marks the first useful contribution, and time to first check-in indicates the time it takes to make a useful contribution. Since, the first check-in alone does not suggest that the developer is productive, we also measure the time to ramp-up. We quantify the ramp-up time as the time it takes for a new hire to reach the average productivity level of existing employees. To measure the time to ramp-up, we examine the frequency of check-ins as an indicator of the familiarity with the process that comes with initial experience. Thus, frequency of check-ins measure the familiarity with the process. However, it falls short to comment on the efforts and the span of knowledge required to implement the change. To capture these features, we examine artifacts related to source code like lines changed and files changed. Lines changed indicate efforts of software developers, where the contribution can be in the form of code, comments, or documentation. Similarly, files changed indicates that the developer has acquired an understanding of a set of files and their relationships. These measures, that is, check-ins counts, lines of code changed, and files changed have also been used in literature to measure developer's contribution [14] [15].

In this section, we measure the time to first check-in and examine the ramp-up time using frequency of check-ins, lines changed, and files changed. Here, it is important to note that the three metrics used to compute ramp-up time are not exhaustive. However, the three metrics capture the key contributions of software developers and gives a fair picture of the ramp-up journey. In addition to this, we examine the influence of experience and the product team of the new hire on the time to first check-in and the ramp-up time. In this study, we try to answer the following research questions:

RQ1 Does the product group of new hires influence the time to first check-in?

RQ2 Does prior job experience, within or outside the company, influence the time to first check-in?

RQ3 Does early check-in correlates with early ramp-up?

RQ4 Is ramp-up journey a function of experience and product?

C. Qualitative Analysis

To complete the understanding of factors that influence the ramp-up journey of new hires, we augment quantitative analysis with qualitative results. Here, qualitative results include a small set of interviews and a broad deployment of a survey directed towards new hires at Microsoft. To get a flavor of factors that influence the ramp-up journey of new hires and to help with designing the survey, we conduct a small scale interview of four software developers. Three out of the four interviewees were 'Entry Level' software developers and one of them was 'Senior Level' software developer. We interviewed software developers for a half hour each and presented them with two open-ended questions.

1) What factors supported or undermined their attempts to make early first check-in and reduced the time to ramp-up?
2) What could have been done to reduce the time to first check-in and the ramp-up time?

We use the responses from the four interviews to formulate two sets of multiple-choice questions, in addition to the demographics and open ended questions. The first set of questions try to understand the influence of specified factors on the time to first check-in. Similarly, the second set of questions try to understand the effect of specific factors on the time to ramp-up. The two sets of questions are evaluated on a 5-point Likert scale, along with an additional field titled 'I don't know'. The field 'I don't know' is intended to address cases where the survey respondents have no understanding of the situation described as factors influencing the ramp-up journey. Both sets of questions are followed by an open ended question to capture factors that are not mentioned in the list. Next, we asked software developers the list of activities, other than code check-in, that demands their time and efforts. Finally, we requested their suggestions on practices that may help reduce the ramp-up time.

TABLE II
STATISTICAL SIGNIFICANCE OF SURVEY RESULTS

Parameter	Increase	No effect	Decrease	p-value
Lack of proper documentation for the project	267	63	75	<0.001***

To conduct the survey, we identified software developers from the eight product groups based on the following criteria:

- Developers have some minimum experience with the aids and impediments that can be faced during ramp-up.
- The aids and impediments encountered during the ramp-up journey are still fresh in their minds.
- Have a reasonable sample of new hires, as we anticipated a response rate of $\approx 20\%$ based on previously conducted studies.

Based on the above criteria, we identified 1,189 software developers with 6 to 13 months of experience at Microsoft on the date of analysis. The survey was sent to all identified software developers representing different roles, career stage paths, and nationalities. We received 411 completed responses (34.57% response rate), 1 partially filled response, and no disqualified response. 99.8% of the responses we received were from individual contributors. The other roles asked in the survey were lead and manager.

As we present the qualitative results, we first analyze the usefulness of the summaries of survey by computing statistical significance using chi-square test at 0.05 significance level. To compute statistical significance, we convert the ordinal scale to its nominal equivalent. We merge 'Strong Increase' with 'Moderate Increase' and collectively present the result as 'Increase'. Similarly, we merge 'Strong Decrease' with 'Moderate Decrease' as 'Decrease'. The rest two categories, 'I Don't Know' and 'No Effect' are considered as one and are titled 'No effect/I don't know'. For each statistically significant result, we compute the central tendency as the most frequent response or 'mode' and present the results. Table II presents a sample question asked in the survey that tries to understand the impact of 'Lack of proper documentation for the project' on the time to first check-in. A detailed discussion of the complete list of factors is given in the next two sections. In Table II, 267 survey respondents said that lack of proper documentation for the project increases the time to first check-in. 63 survey respondents either did not encounter this problem or considered that it had no effect, and 75 responses suggested that it decreases the time to first check-in. We conducted chi-square test and observed p-value<0.001. P-value<0.001 gives a very strong presumption against the null hypothesis, thereby suggesting that the result is statistically significant. The claim made by central tendency summaries, as identified by mode, is that *'Lack of proper documentation increases the time to first check-in'*.

In the next two sections, we present an analysis of the time to first check-in and the ramp-up time respectively. We quantitatively analyze the time to first check-in and the

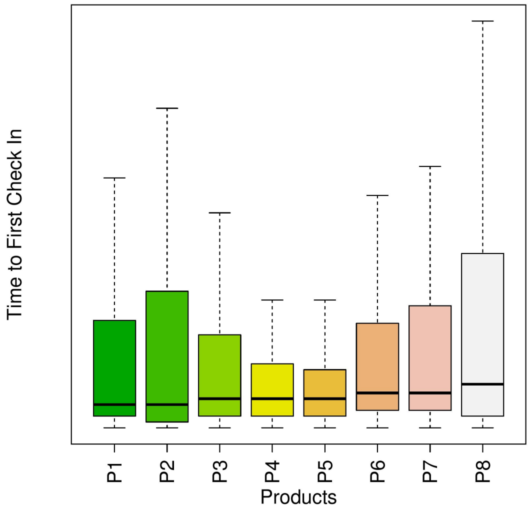

Fig. 3. Product groups and time to first check-in

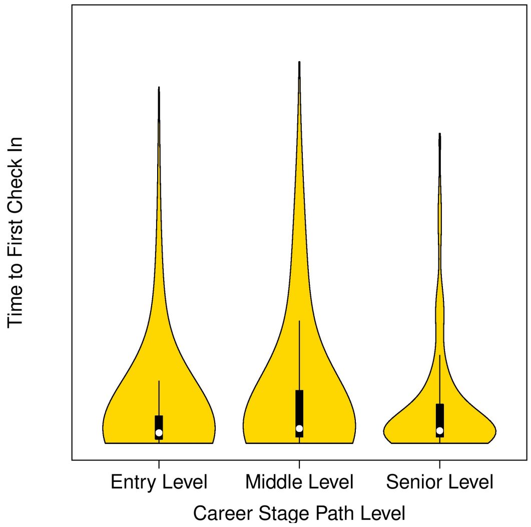

Fig. 4. Experience and time to first check-in

ramp-up time, followed by the summaries of the multiple choice questions asked in the survey. Further, to complete the analysis, we present the opinions of survey respondents presented in the open ended questions. We present the list of activities that claim the time and efforts of new hires and also suggestions from the survey respondents to improve the ramp-up journey. We card sort the opinions of the new hires in the open-ended questions and present it in the non-increasing order of the frequency of occurrence.

V. Time to First Check-In

The time to first check-in marks the first step in the ramp-up journey of new hires. We measure the time to first check-in as the duration from the starting date at Microsoft until the time the new hire makes first check-in into the master branch of the version control system. For confidentiality reasons, we obscure the unit of time used in the study and report the results.

A. *Quantitative Analysis*

1) RQ1: Does the product group of new hires influence the time to first check-in?: Microsoft is a large software company with multiple product divisions. Each product division at Microsoft is slightly different from other divisions in terms of tools, technologies, or processes being used. Here, we are interested to know whether working with some specific product group help new hires make early first check-in into the system. By answering this question, we can identify some of the best practices in product groups which can then be transferred to other product groups. To do so, we measure the time to first check-in for all new hires in the eight product teams and compute quartiles. The choice of quartiles for the study ensures that our results are not affected by outliers. Figure 3 shows the boxplot of the time to first check-in for the new hires in the eight product teams. Here, the horizontal axis shows the product groups and the vertical axis shows the time to first check-in measured in weeks. In Figure 3, we observe that the median of the population of new hires across all product divisions take ≈4-10% of the maximum time to first check-in. The analysis of eight product groups at Microsoft suggests that working on some specific product group has no significant impact on the time to first check-in. However, we see that the third quartiles of the product teams show marked differences. Figure 3 shows that new hires in products P2 and P8 take longer (≈34% and ≈43% of the maximum time to first check-in across products respectively) relative to other product groups (minimum ≈14% in product P5). Further investigation is required to understand the cause.

2) RQ2: Does prior job experience, within or outside the company, influence the time to first check-in?: Microsoft recruits several thousand new hires every year, ranging from fresh university graduates to professionals with prior job experience. Here, we are interested to know whether prior job experience help new hires make early first check-in into the system. To investigate the impact of prior job experience on the time to first check-in, we analyze the influence of the career stage path or job title, an indicator of experience, on the time to first check-in. At Microsoft, professionals are hired with job-titles that conform to their prior job experience. For instance, professionals with 1 to 3 years of experience are hired as 'Middle Level' software developers. Thus, in this study, we analyze the role of job title on the time to first check-in. To start with, we classify job titles in Microsoft into 'Entry Level', 'Middle Level' and 'Senior Level' software developers. 'Entry Level' include software developers with job titles 'Software Development Engineer (SDE)' and 'IT SDE'. 'Middle Level' include job titles 'SDE 2' and 'IT SDE 2', and 'Senior Level' include job titles 'Senior SDE', 'Principal SDE', 'Partner SDE' , and 'Distinguished Engineer'. For each career stage path, we compute the time to first check-in for all new hires and plot the results.

Figure 4 shows the viola plot where the horizontal axis shows the three career stage paths of software developers and the vertical axis shows the time to first check-in measured in weeks. The plot in Figure 4 is a combination of box plot and smoothened density function where the density function indicates the developer distribution pattern. Thus, broader the width; higher the fraction of new hires that takes specific time to make first check-in. In Figure 4, we observe that the percentage increase in the median time to first check-in for middle and senior level software developers is ≈20% relative to entry level software developers. Least median time to first check-in for entry level software developers imply that developers with no or less than a year of prior job experience makes early first check-in compared to experienced new hires. Further, we observe that middle level and senior level software developers have the same median time to first check-in with different density distributions. The density distribution for senior level software developers peaks at median, relative to the spread of middle level software developers. This distribution implies that senior level software developers perform consistently and make early first check-ins compared to middle level software developers. This observation calls for further investigation to understand the factors that influence the time to first check-in for all levels of software developers.

B. Qualitative Analysis

To present a comprehensive list of factors that influence the ramp-up journey, we present the opinions of new hires at Microsoft on the factors that influence the time to first check-in. We asked new hires the effect of a set of factors on the time to first check-in. These questions are based on the interviews conducted with four new hires with different career stage paths. Table III shows the impact of these factors on the time to first check-in. The opinions of new hires are presented on the scale of 'Strong Increase' to 'Strong Decrease'. We use an additional field titled 'I don't know' to account for cases where survey respondents have no opinion on the impact of the factor asked in the survey. Table III present the opinions of new hires in 7 (out of 8) product groups as there was no survey response from one product group we analyzed in the study. These observations can be used by the product groups to identify and eliminate bottlenecks in the processes and help improve existing practices.

To visualize trends across product groups, we color code the observations in Table III. Here, we present areas of improvement as shades of red and good practices as shades of green. The factors on which survey respondents have no opinion are presented as yellow. We leave the factors with no effect colorless. Also, '-' indicates that we received insignificant responses from the product division on the influence of the factor to support interpretation. Further, darker the color; stronger is the impact. Thus, dark red (on a relative scale) means that the parameter strongly increases the time to first check-in. In Table III, all results, except for 'Availability of Resources on Arrival' are statistically significant. We present summaries of statistically significant results and open ended responses.

In Table III, we see that lack of proper documentation increases the time to first check-in. The lack of proper documentation strongly increases the time to first check-in in 5 (out of 7) product groups. While for the rest 2 product groups, it moderately increases the time to first check-in. Also, getting access and permissions, working on codes with dependencies, and working on legacy codes moderately increases the time to first check-in for majority of product teams. 4 out of 7 product teams say that joining the team near product release has no effect on the time to first check-in, while the other 2 teams did not comment on it. It is noteworthy to see that new hires in product P4 says that joining the team near product release strongly increases the time to first check-in. Besides these, changing product teams, writing new code compared to fixing the issues, identifying the reviewers for the code, changes in team composition, and making check-ins in branches other than the main branch have no effect for the majority of product teams. One key observation from this analysis is that not all product teams are influenced by all the parameters stated above. Also, the degree of influence varies substantially across product teams.

In the open-ended question that followed, we asked survey respondents to enumerate factors, other than the ones listed in the survey, which influence the time to first check-in. We card-sorted the responses in the open-ended question and found the following themes that influence the masses. The themes are presented in the non-increasing order of occurrence.

1) *Mentorship*: Software developers stressed the importance of having a manager, mentor or lead, to talk to, during the initial days. They said that mentors can assist new hires in getting unstuck and make early first check-ins. Also, software developers who were not assigned mentors experienced that absence of mentor resulted in significant loss of time.
2) *Documentation*: Software developers feel that lack of detailed documentation of products and processes strongly increase the time to first check-in. To add to, the documents are stored at different places, in different formats, and some documentation are out of date.
3) *Process*: Software developers feel that engineering processes need some improvement. They believe that the use of standard components, for which documentations and manuals are available widely, will help reduce the time to first check-in.
4) *Access and Permissions*: New hires feel that it takes some time to figure out the desired access and permissions. They suggest that it will be helpful to associate access and permissions with the team and not the individuals.
5) *System setup*: Software developers say that they spend a considerable amount of time to set-up environment and configurations, which can be improved.

Besides these, developers find that the lack of confidence to make changes, large size of the code base to study, lack of

TABLE III
INFLUENCE OF THE FOLLOWING FACTORS ON THE TIME TO FIRST CHECK-IN [SI: STRONG INCREASE; MI: MODERATE INCREASE; NE: NO EFFECT; MD: MODERATE DECREASE; SD: STRONG DECREASE; DK: DONT KNOW]

How do the following items affect the time to first check-in?	P1	P2	P3	P4	P5	P6	P7	Statistical Significance
Lack of proper documentation for the project	SI	SI	SI	SI	SI	MI	MI	<0.001***
Getting access and permissions	MI	MI	MI	SI	SI	MI	MI	<0.001***
Working on a code with dependencies to others' work	SI	MI	MI	MI	-	MI	MI	<0.001***
Working on preparatory tasks (like code review, coding assignments, etc.)	MI	MI	MI	SI	MI	MI	MI	<0.001***
Working on legacy code	MI	MI	NE	MI	MI	MI	MI	<0.001***
Availability of resources (like desktop, task related equipment(s) on arrival)	NE	NE	NE	SI	SI	-	MI	0.25
Join the team near product release	NE	NE	NE	SI	DK	NE	DK	<0.001***
Changing products, such as moving from Windows	DK	DK	DK	NE	NE	NE	DK	<0.001***
Writing new code than fixing issues	NE	NE	NE	MI	MI	NE	DK	<0.001***
Identifying the reviewers for the code	NE	NE	NE	MI	NE	NE	NE	<0.001***
Changes in team composition, such as change of immediate manager	NE	NE	NE	NE	NE	NE	NE	<0.001***
Making check-ins in branches other than the main branch	NE	NE	NE	NE	NE	NE	MI	<0.001***

TABLE IV
CORRELATION BETWEEN TIME TO FIRST CHECK-IN AND RAMP-UP TIME AFTER FIRST CHECK-IN

	P1	P2	P3	P4	P5	P6	P7	P8
Commit Counts	-0.06	-0.01	-0.09	-0.22	+0.04	-0.00	-0.04	+0.13
Lines Changed	-0.07	-0.09	-0.24	-0.39	-0.13	+0.10	-0.34	-0.31
Files Changed	-0.13	-0.18	-0.18	-0.52	-0.21	-0.00	-0.16	-0.30

technical skills required for the job, development environment, meetings with a broader scope (not targeted), and frequent manager changes increase the time to first check-in.

VI. TIME TO RAMP-UP

A. Quantitative Analysis

As summarized in Mythical Man-Month, adding personnel to the project decreases productivity in the short term [16]. New hires take time to reach the productivity level of existing employees. The amount of time taken to ramp-up influences resource planning, effort estimation, and hence the productivity of the team. Therefore, managers and business analysts might be interested to know the time to ramp-up and study its impact. In this study, we define ramp-up time of new hires as the time required to reach the median productivity level of existing employees. We measure the time to ramp-up on three parameters, namely the frequency of check-ins, lines changed, and files changed. To establish the baseline, we measure the median check-in counts, lines changed, and files changed for existing employees in each product per unit time. We then measure the unit of time it takes for the new hires to reach the median productivity level of existing employees. In this section, we are interested to find answers to the following research questions:

1) RQ3: Does early first check-in correlates with early ramp-up?: Managers and business analysts might be interested to understand the best practices that help reduce the ramp-up time. So in this context, we analyze whether early first check-in help new hires ramp-up faster, compared to others who take longer to make first check-in. For all product teams, we compute the correlation between the time to first check-in and the time to ramp-up after first check-in. We compute the correlation on check-in counts, lines changed, and files changed using Spearman's rank correlation coefficient. For all three parameters, we observe negligible to weak correlation ranging from +0.10 to -0.39 (refer Table IV). Negligible to weak correlation between the time to first check-in and the ramp-up time suggests that the time to ramp-up is not a function of the time to first check-in. However, the negative magnitude of correlation implies that new hires who take longer to make first check-in do not necessarily take longer to ramp-up after first check-in.

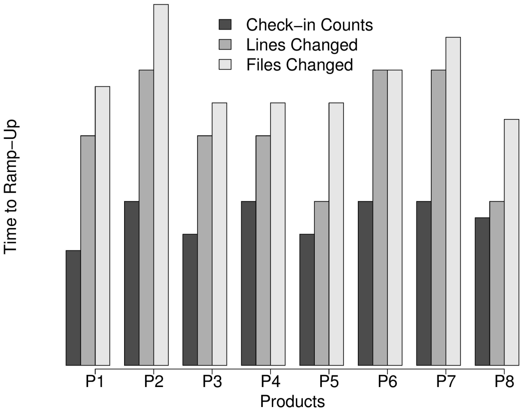

Fig. 5. Product groups and time to ramp-up

2) RQ4: Is ramp-up journey a function of experience and product?: In the previous section, we examined the impact of experience and product group on the time to first check-in. We observed that the median time to first check-in is similar across all product groups analyzed. Also, we saw that middle level software developers' take longer than entry level and senior level software developers to make first check-in. In this study, we are interested to know that similar to the time to first check-in, does experience and product team influence the ramp-up time?

For each product group, we compute the median time to ramp-up and present the results (as shown in Figure 5). In Figure 5, the horizontal axis shows the product groups and the vertical axis shows the time to ramp-up in months. The legend refers to the three parameters we used to measure the time to ramp-up. In Figure 5, we observe that new hires ramp-up on the three parameters stepwise. First, new hires ramp-up on check-in counts, followed by lines changed, and files changed. This follows the intuition that making changes to multiple files require broad understanding of the task, and hence takes longer than ramping-up on check-in counts, or lines changed. Also, we observe that the median time to ramp-up on eight product groups is similar for check-in counts (≈32-45% of the maximum time to ramp-up measured in months). However, it varies significantly for lines changed (≈45-81%) and files changed (≈68-100%). A large variance in the ramp-up time on lines changed and files changed indicate differences in ramp-up time across products. This information can be used by managers and business analysts in understanding the productivity of new hires across products, make better effort estimations, conduct resource planning, and take appropriate actions.

Figure 6 shows the time to ramp-up based on experience. In Figure 6, the horizontal axis shows experience as identified by the title and the vertical axis shows the time to ramp-up measured in months. We analyze the median time to ramp-up on check-in counts, lines changed, and files changed. We see that the percentage increase in the median time to ramp-up on check-in counts for different experience levels is ≈5% (measured in months). Thus, experience has no impact on ramp-up time on check-in counts. However, for ramp-up on files changed and lines changed, we see that middle and senior level software developers take marginally longer than entry level software developers to ramp-up. The percentage increase in the median ramp-up time on lines changed for middle level and senior level software developers is ≈13% and ≈6% respectively relative to entry level software developers. Similarly, the percentage increase in the median time to ramp-up on files changed is ≈22% for middle level and senior level software developers relative to entry level software developers.

B. Qualitative Analysis

We present the opinions of new hires at Microsoft on the factors that influence the time to ramp-up. We asked the new hires the effect of the factors (mentioned in Table V) on the time to ramp-up. These questions are based on the interviews of new hires who followed different career stage paths. Table V shows the central tendency summaries, as presented by mode, for the 7 (out of 8) product groups analyzed in the study. Similar to the previous section on qualitative results, here red indicates areas of improvement, green indicates good practices, yellow implies don't know, and colorless means no effect. Also, darker the color means stronger the impact. In Table V, all results, except for 'Communicating Technical Prerequisites', are statistically significant.

In Table V, we observe that prior knowledge of programming languages, programming environment, and tools help decrease the ramp-up time. Similarly, proactively asking questions, prior familiarity with the process, and having a mentor decrease the time to ramp-up. Though, the influence is different for products P4 and P5. Contrary to this, new hires say that maintaining documentations, to do lists, and working on preparatory tasks increase the ramp-up. When asked about the impact of prior familiarity with the team on ramp-up time, new hires expressed mixed opinions across product teams. Also, 5 (out of 7) teams say that active participation in social events has no effect on the time to ramp-up. The rest 2 teams say that it moderately increases the time to ramp-up. In addition to the parameters stated above, software developers feel that the following practices influence the time to ramp-up.

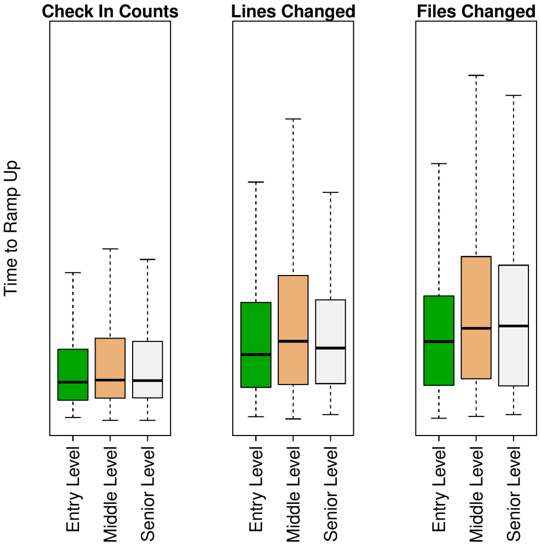

Fig. 6. Experience and time to ramp-up

1) *Team Interaction*: New hires say that verbal communications in team and pair programming are the most effective ways to ramp-up. They find that spending more time with the manager and the team during first 1-2 months is helpful. They add, that recently ramped-up employees reduce the ramp-up time of new employees the most.
2) *Training*: Software developers say that training programs like boot camp, etc. are very helpful.
3) *Overview of the system*: Software developers say that well-chosen starting tasks that gives a complete overview of the system helps reduce the ramp-up time.
4) *Proximity to release*: Developers find that joining the team after product release increases the time to ramp-up as there is not much code to write.

Other than these, developers' say that familiarity with people, customer, process, product, and code helps improve the ramp-up journey.

VII. Other New Hire Activities

In the above two sections, we presented different factors that influence the ramp-up journey and measured time to ramp-up using various features of check-ins. However, new hires perform various activities, other than code check-ins, thereby

TABLE V

Influence of the following factors on the time to ramp-up [SI: Strong Increase; MI: Moderate Increase; NE: No Effect; MD: Moderate Decrease; SD: Strong Decrease; DK: Don't Know]

How do the following item affect the time to ramp-up?	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**Statistical Significance**
Prior knowledge about programming languages (such as C#), programming environments (such as Visual Studio), or tools (such as versioning tools)	SD	MD	SD	-	MD	-	SD	<0.001***
Proactively asking questions to your manager, mentor or team	SD	MD	MD	-	-	SD	MD	<0.001***
Prior familiarity with processes (such as how effort estimation is done or review process)	MD	MD	MD	MI	MI	MD	MD	<0.05**
Having a mentor	SD	SD	SD	SI	SI	SD	DK	<0.001***
Maintaining documentation, to do lists, introductory videos for different employee titles	SI	NE	SD	SI	MI	MI	MI	<0.001***
Preparatory tasks (such as code review, building prototypes) or training programs (like boot camp)	MI	MI	MD	SI	SI	MI	SI	<0.001***
Communicating the relevant technical prerequisites (such as tools and languages used) of an employee's title, prior to joining	MD	NE	NE	MI	SI	MI	MI	0.66
Prior familiarity with team (such as the cases of moving from an intern to a full time position)	NE	DK	DK	MI	MI	DK	DK	<0.001***
Active participation in social events, such as team lunches etc.	NE	NE	NE	MI	MI	NE	NE	<0.001***

making it important to understand the activities that are not captured in the study. We asked the survey respondents the list of activities they perform and present the themes arranged as different stages in the ramp-up journey of new hires.

New hires who relocated from other countries or states say that it takes time to settle in a new country or state. Further, they add that on arrival, they have to set-up the system, get the required access and permissions, enroll for benefits, get HR/staffing set up, etc. Once the required resources are made available, new hires find themselves trying to understand the existing system and identify their role in the team. They say that they undergo training programs, attend meetings, and study all kinds of documentation to acquire the required technical and functional knowledge. They add, that sometimes they are assigned code reviews, prepare prototypes or demos, or even read legacy code to get a better understanding of the system. Further, in the process of knowledge transfer, they spend substantial time in proactively asking questions. New hires emphasize that while they are learning and trying to ramp-up, they also engage in other activities like testing, bug fixing, debugging, bug triaging, identifying and resolving dependencies, etc.

New hires also identify some miscellaneous activities that claim their time and effort. They say that they spend time in planning, writing proposals, estimate time for the task, perform production related duties, add work items, participate in events like 'Hackathon', etc. While this list is not exhaustive, it gives a fair understanding of the various activities that new hires perform other than code check-ins. This list of activities suggest that while code check-in is a good indicator of productivity, it does not present a complete picture.

We requested new hires for suggestions on practices that might help improve the productivity of new hires. We believe that this information can help companies in improving the ramp-up journey of new hires. New hires suggest that improvements in engineering systems like applying companywide coding standards, improved code base and documentation, easy tools, etc. will increase the productivity of new hires. These findings align with Microsoft's recent initiatives such as One Engineering System to have one common system across all projects. They emphasized the usefulness of training tools and sessions, and guided work for a few weeks during ramp-up time. They also added that centralization of all information and clearly communicated expectations are other factors that accelerate the ramp-up journey.

VIII. Threats to Validity

A. Internal Validity

- *Data accuracy:* The accuracy of the results of this study depends on the accuracy of the data on which it is built, e.g., some data may be missing or incomplete. We believe that this is only a minor threat. For the study, we used the CodeMine tool, which attempts to capture software development activities as accurately and completely as possible. Several production systems at Microsoft are built on top of CodeMine and its accuracy has been extensively verified.

B. Construct Validity

- *Activities in other product groups:* We analyze commits in the eight product groups, which constitute a vast majority of the Microsoft workforce. However, if developers engage in activities in product groups other than the ones analyzed here, we are not able to capture their contribution.
- *Activities other than code check-ins:* We compute the ramp-up journey of new hires in terms of code check-ins. However, new hires engage in a wide variety of activities other than code check-ins (refer Section VII). Also, different product groups emphasize on different set of activities during the ramp-up journey of new hires. These two factors may influence the observed time to first check-in and the time to ramp-up. So, while comparing product groups on the time to first check-in and the time to ramp-up, these two factors should be taken into consideration.

C. External Validity

- *Application of results to product divisions within and outside Microsoft:* We analyzed eight large, popular product teams, which constitute the majority of Microsoft's engineering workforce. We, therefore, believe that the results are widely applicable to product divisions in Microsoft. We do not claim that the findings and recommendations presented in this study extend to any organization and product team. For example, we expect that the findings may not generalize to organizations with bootstrap mechanism different from Microsoft or organizations that hire software developers for testing purposes only. While findings may not generalize, the research methodology can be applied to other contexts as long as there are sufficient data points to compute a baseline productivity of existing employees.
- *Geographic differences:*The survey included participants from different countries with different cultures and working hours. In addition, Microsoft relocates new hires from other countries to new countries. We did not collect enough data to analyze and control for this effect. We, therefore, caution the reader that our findings may not apply to arbitrary countries and cultures.

IX. Conclusion

We conducted quantitative and qualitative analysis to understand the factors that influence the ramp-up journey of new hires. Our results reiterate some of the knowledge already known to the industrial world by mining software engineering data. We analyzed eight large product groups at Microsoft and observed that the time to first check-in, a milestone in the ramp-up journey of new hires, is invariant to the product group analyzed. In terms of experience, as indicated by the career stage path levels, entry level software developers make faster check-ins compared to middle and senior level software developers. To complete the analysis, we asked the opinions of new hires to understand the factors that influence the time to first check-in. We observed that among other factors, lack of proper documentation, getting access and permissions, etc. increase the time to first check-in. Further, we computed the ramp-up time of newly hired software developers on check-in counts, lines changed, and files changed. We observed that first new hires ramp-up on check-in counts, followed by lines changed, and files changed. We also found that the time to first check-in is weakly, if at all, correlated with the ramp-up time. The negative correlation implies that new hires who take longer to make first check-in do not necessarily take longer to ramp-up thereafter. Also, we see that ramp-up time is a function of experience and product on lines changed and files changed. In addition to this, survey results suggest that prior knowledge of required technical skills, proactively asking questions, and familiarity with the process help reduce the ramp-up time along with other factors. We also list activities, other than the code check-in, that claim developers' time and efforts. We conclude the study with suggestions of new hires to help improve the productivity of new hires.

In future, we would like to study the influence of cultural and work hour differences in different nations on the ramp-up journey of new hires. Also, we would like to investigate the ramp-up journey of internal transfers to understand whether internal transfers help organization.

Acknowledgment

We would like to offer our special thanks to Christian Bird, Michaela Greiler, Trevor Carnahan, and Fei Huang for reviews and Alok Bhayana for the data. We would also like to thank the conference reviewers for insightful and constructive reviews.

References

[1] E. Brechner, "Things they would not teach me of in college: what microsoft developers learn later," in *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*. ACM, 2003, pp. 134–136.

[2] A. Begel and B. Simon, "Struggles of new college graduates in their first software development job," in *ACM SIGCSE Bulletin*, vol. 40, no. 1. ACM, 2008, pp. 226–230.

[3] D. K. Taft, "Programming grads meet a skills gap in the real world," *Retrieved September*, 2007.

[4] "Rational Team Concert," http://www-03.ibm.com/software/products/en/rtc/, [Online; accessed 11-March-2015].

[5] "Team Foundation Server," http://msdn.microsoft.com/en-us/vstudio/ff637362.aspx, [Online; accessed 11-March-2015].

[6] "git–distributed-even-if-your-workflow-isnt," http://www.git-scm.com/, [Online; accessed 11-March-2015].

[7] "Survey Identifies Greatest Challenges When Starting a New Job," http://accountemps.rhi.mediaroom.com/new-job-challenges, [Online; accessed 11-March-2015].

[8] A. Begel and B. Simon, "Novice software developers, all over again," in *Proceedings of the Fourth international Workshop on Computing Education Research*. ACM, 2008, pp. 3–14.

[9] B. Dagenais, H. Ossher, R. K. Bellamy, M. P. Robillard, and J. P. De Vries, "Moving into a new software project landscape," in *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1*. ACM, 2010, pp. 275–284.

[10] C. Ostroff and S. W. Kozlowski, "The role of mentoring in the information gathering processes of newcomers during early organizational socialization," *Journal of Vocational Behavior*, vol. 42, no. 2, pp. 170–183, 1993.

[11] S. E. Sim and R. C. Holt, "The ramp-up problem in software projects: A case study of how software immigrants naturalize," in *Software Engineering, 1998. Proceedings of the 1998 International Conference on*. IEEE, 1998, pp. 361–370.

[12] A. Radermacher and G. Walia, "Gaps between industry expectations and the abilities of graduates," in *Proceeding of the 44th ACM technical symposium on Computer science education*. ACM, 2013, pp. 525–530.

[13] J. Czerwonka, N. Nagappan, W. Schulte, and B. Murphy, "Codemine: Building a software development data analytics platform at microsoft," *Software, IEEE*, vol. 30, no. 4, pp. 64–71, 2013.

[14] G. Robles, S. Koch, J. M. GonZÁlEZ-BARAHonA, and J. Carlos, "Remote analysis and measurement of libre software systems by means of the cvsanaly tool," in *Proceedings of the 2nd ICSE Workshop on Remote Analysis and Measurement of Software Systems (RAMSS)*. IET, 2004, pp. 51–56.

[15] A. Alali, H. Kagdi, and J. I. Maletic, "What's a typical commit? a characterization of open source software repositories," in *Program Comprehension, 2008. ICPC 2008. The 16th IEEE International Conference on*. IEEE, 2008, pp. 182–191.

[16] F. P. Brooks, *The mythical man-month*. Addison-Wesley Reading, MA, 1975, vol. 1995.