Fostering Software Developers' Productivity at Work Through Self-Monitoring and Goal-Setting

André N. Meyer
Software Evolution and Architecture Lab
University of Zurich, Switzerland
ameyer@ifi.uzh.ch

ABSTRACT

Software development organizations strive to enhance the productivity of their developers. While research has looked into various ways for improving developer productivity, little is known about the activities they pursue at work, how these activities influence the fragmentation of work, and how these insights could be leveraged to foster productivity at work. In my PhD thesis, I address software developer productivity by taking a mixed-method approach to investigate developers' perceptions of productivity in the field and to examine the individual differences of each developer's work. My goal is to increase developers' awareness about their own work habits and productivity, and to encourage productive behavior changes at work through the provision of two persuasive technologies, self-monitoring and goal-setting.

CCS CONCEPTS

• **Human-centered computing** → **User studies**; • **Software and its engineering** → Software creation and management;

KEYWORDS

Productivity, Quantified Workplace, Self-Monitoring, Goal-Setting, Interruptions, Context Switching

1 PROBLEM STATEMENT

There is a common refrain that repeats itself in similar forms every couple of years: our inability to produce enough software to satisfy the needs of the world. For example, in 1968, attendees at the first NATO software engineering conference coined the term software crisis [17] and in 2011, Andreessen wrote about software eating the world, expressing that the need for software keeps outstripping our ability to produce it [1]. There are a couple of ways of addressing the gap between software demand and supply. We could try to reduce

the demand, which seems unlikely to succeed. Or, we could try to increase the supply, namely our ability to produce software. In this PhD thesis, I focus on one way to increase the supply of software: the improvement of software developers' productivity.

A substantial amount of work researching developer productivity has been undertaken over the past four decades. The majority of this research introduces particular definitions of productivity, such as measuring productivity based on the number of lines of source code modified per hour [8] or the resolution time for modification requests [16]. Another body of research considers organizational factors associated with productivity, such as the effect of the workplace setting on productivity [7], and soft factors, such as interruptions and meetings (e.g., [11, 19, 23]). However, these studies on productivity are generally separate from studies of work fragmentation and do not take into account the individual differences in development work that might affect productivity.

Furthermore, Perry et al. showed that developers themselves are often not aware of these productivity factors and how they impact their own work [19]. One way to improve developers' awareness of their own behavior and foster productive behavior is to provide them with the means to self-monitor and to reflect about their actions, for example through visualizations [10, 20, 22]. This type of self-monitoring approach has been shown to foster behavior changes in other areas of life, such as physical activity (e.g., [5, 9]) and health (e.g., [2, 6]). Existing efforts to map the success of these self-monitoring approaches to the workplace, e.g., RescueTime [21] and Codealike [4], have largely focused on tracking and visualizing application usage on the computer that rarely allow insights into productive or unproductive behaviors at work. Only few of these tools have been evaluated (e.g., [22, 24]), limiting the knowledge of the overall value of self-monitoring in the workplace, the information that is of particular interest, and how these insights could be leveraged to foster productivity, for instance through personalized self-monitoring or goal-setting.

2 PROPOSED RESEARCH

The objective of my PhD thesis is to take a bottom-up approach to better understand developers' individual perceptions of productivity and the relationship to their work practices, as well as to investigate how the findings can be integrated into self-monitoring and goal-setting systems that aim to foster productive behavior changes. This yields the following hypothesis:

Models of developers' work and productivity can be used to foster productive behavior at work through a) self-monitoring and b) actionable insights for developers and their colleagues.

To validate this hypothesis, I will examine the following research questions:

[**RQ1**] *What does a software developer's work day look like in terms of activities and work fragmentation, and how does it relate to perceived productivity?*

[**RQ2**] *Can we develop a self-monitoring approach that raises developers' awareness about their work and productivity?*

[**RQ3**] *Can we devise approaches that foster productive behavior at work through the provision of concrete and actionable insights for a) software developers and b) their co-workers?*

Each research question is motivated, its method explained and selected results presented in the next sections. To evaluate the efficacy of the two persuasive strategies, self-monitoring and goal-setting, to foster productive behavior at work for developers, I developed WorkAnalytics, studied its impact on developer productivity, and refined it through feedback and observations made from several field studies. The insights from these field studies and the system can provide researchers and practitioners with a starting point for building tools that foster sustained productivity in the workplace.

3 RQ1: UNDERSTANDING DEVELOPER PRODUCTIVITY

Motivation. Little is known about how developers spend their time at work and how it influences their individual perception of productivity. In my work, I take a bottom-up approach to better understand developers' work habits, factors that influence their productivity, and study the relationship between their work activities, context switching, and their perceptions of productivity.

Method. To gain a better understanding of software developers' work and productivity, we deployed a monitoring application at 20 professional software developers' computers for an average of 11 full work days *in situ*. The monitoring application collects logs about developers' interaction with the computer (mouse and keyboard input, application usage) and self-ratings on perceived productivity at an hourly interval.

Results. Based on the analysis of the gathered data, our main observations are that developers' work habits and fragmentation of their work varies a lot across individuals and companies, as do their perceptions of productivity. Analyzing the activities developers pursue in a regular workday, we found that during an average workday of 8.4 hours, developers spend about one fourth of their time with coding related activities, and another fourth of their time with collaborative activities, such as meetings, emails and instant messaging. We observed notable differences across companies, e.g., time spent with emails ranges from less than a minute to more than an hour a day. Developers usually remain in an activity only between 20 seconds and 2.0 minutes before switching to another one. These

(a) Morning Person   (b) Low at Lunch   (c) Afternoon Person

Figure 1: 3 Types of Developers and their Perceptions of Productivity over the Course of a Work Day.

very short times per activity and the variety of activities a developer pursues each day illustrate the high fragmentation of software development work. By investigating participants' self-reported productivity ratings, we found that although there is a lot of variation between individuals' perceptions, each developer fits into one of the following three themes: morning person (20%), afternoon person (40%), and low-at-lunch person (35%) as visualized in Figure 1. For example, an afternoon person may be one who is industrious later in the day, or feels more productive as a result of having the majority of the work planned for the day completed. Amongst others, these results suggest that while information workers have diverse perceived productivity patterns, individuals do appear to follow their own habitual patterns each day. We further built explanatory models to describe which factors contribute to the self-reported productivity-ratings. Based on the analysis of stepwise linear regression models, we observed that productivity is a highly personal matter and perceptions of what is considered to be productive are different across developers. We found some overall themes, such as that many developers consider email, planned meetings and work unrelated browsing as less productive activities, and usually perceive themselves as more productive when they type more or use the mouse more.

These and other results published in [12, 15] show that while productivity is in fact a very personal matter, there are some observable themes and patterns across perceptions of productivity. Hence, to improve and quantify productivity, it is important to take into account a variety of factors that impact productivity, rather than a single measure.

4 RQ2: INCREASING AWARENESS ABOUT WORK WITH SELF-MONITORING

Motivation. Even though the impact of a variety of factors on developers' productivity has been well researched, most developers are often not aware of how their actions contribute to these factors and how they impact both their own productivity at work and the work of others [19]. One way to improve software developers' awareness of their own actions and foster productive behavior could be to provide them with the means to self-monitor and reflect about their actions. Even though several self-monitoring approaches at the workplace already exist (e.g. [10, 21, 24]), little is known about software developers' expectations of and experience with these tools. To address this gap, I aim to examine barriers towards the adoption of existing self-monitoring technologies and developers' requirements for information and features that such systems should include. I further want to explore how they are using and engaging with such a self-monitoring tool during their work and what its impact is on individuals' productivity and team collaboration.

Method. To explore developers' expectations of, the experience with, and the impact of self-monitoring in the workplace, we followed a mixed-methods approach. We used an iterative, user-feedback-driven development approach (N=20) and a survey (N=413) to infer design elements for workplace self-monitoring, which we implemented as a technology probe with WorkAnalytics. WorkAnalytics allows developers to self-monitor their time spent with different activities at work, their work patterns and productivity, and provides a retrospective view about their work day and work week (see Figure 2). After refining the prototype to accommodate the identified
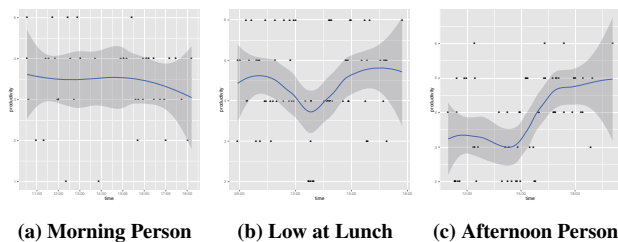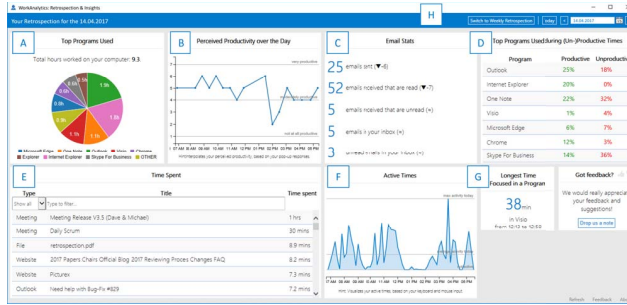
Figure 2: Screenshot of the Retrospection in WorkAnalytics.

design elements, we conducted a field study involving 43 professional software developers, to evaluate their experience with and the impact of WorkAnalytics during three work weeks.

Results. From the field study results, we derived six design recommendations for self-monitoring in the workplace, such as using experience sampling to increase the awareness about work, and that concrete and actionable insights, enriched with benchmarking data from co-workers, are likely needed to foster productive behavior change at work. Participants reported that the application helped them to make detailed observations into how they spend their time at work collaborating or working on tasks, when they usually feel more or less productive, and sort out misconceptions they had about their activities pursued at work, such as spending a surprisingly high amount of time collaborating with others via email. Based on the field study results, we inferred design recommendations for building tools for self-monitoring in the workplace. WorkAnalytics has also shown potential impact on developers' behavior at work, as 41% of the participants self-reported a change in their habits after reflecting on their work. This includes, e.g., optimizing how often and when they email, and actively trying to improve their focus and trying to avoid distractions. However, 13% of the participants stated that they were not sure what to change, as the visualizations and correlations were not concrete and actionable enough for knowing what or how to change, and they suggested to add recommendations to the tool to suggest concrete self-improvements at work. The remaining findings, resulting design recommendations, and much more detail about the study setup are published in [14].

5 RQ3: ACTIONABLE INSIGHTS TO FOSTER PRODUCTIVE BEHAVIOR

One result from studying RQ2 was that an increased self-awareness about work and productivity does not necessarily motivate a behavior change, similar to what related work has found in the health and physical activity domain [3, 22]. Hence, my goal is to extend the existing self-monitoring approach in WorkAnalytics with another persuasive strategy, goal-setting. I aim to evaluate how insights for developers and their teams can be made more concrete and actionable, to support the identification of meaningful goals and the definition of a realistic behavior change plan.

5.1 For Individual Developers
Motivation. To investigate how self-monitoring at the workplace could be enhanced to create more actionable insights for developers

that foster productive work, I aim to evaluate the efficacy of a semi-guided goal-setting system that facilitates the identification of a target behavior, supports measuring the progress towards the goal, and motivates maintaining the behavior change over time.

Method. I plan to pursue a mixed-method approach, where I first conduct an exploratory interview study to investigate developers' existing goal-setting practices, then define a goal-setting approach for developers based on the interview results, and finally evaluate its efficacy with developers in a field study. I am currently in the process of analyzing the results from the exploratory interview with 20 developers. Preliminary results suggest that developers are generally interested in improving their work practices, but have difficulties with identifying meaningful and realistic goals, and as a result, they also experience difficulties with defining a concrete behavior change plan. However, when I asked participants to self-reflect about their work and productivity from different perspectives in the interview, they usually identified several work behavior goals and perceived this self-reflection as very valuable.

As a next step, I plan to run a field study with 50 participants for six weeks and collect qualitative and quantitative data to evaluate a semi-guided goal-setting approach. The study will be designed to collect baseline data of developers' work during the first two weeks by running WorkAnalytics in the background on participants' computers. In a meeting with participants, I plan to ask them to self-reflect about their work and productivity from different perspectives using WorkAnalytics' retrospection, and support them with identifying goals and specifying a concrete behavior change plan. Participants will then be asked to continue their regular work while trying to reach their self-set goals and monitor their progress towards the goals with WorkAnalytics for about four work weeks. Regular guided self-reflection sessions and a final interview will give us qualitative insights into the experience with the system and the perceived impact it has from participants' perspective. Finally, I plan to analyze and compare the work behavior before and after the intervention quantitatively from the monitoring data I captured in the background using WorkAnalytics.

5.2 For Co-Workers.
Motivation. Previous work has shown that interruptions are one of the biggest impediments to productive work, distracting developers from their main task and resuming it takes on average about 15 minutes [13, 18]. My goal is to reduce interruptions at inopportune, costly moments by provisioning actionable insights to an individual developer that increase the awareness about co-workers' current focus at work.

Method. To achieve this, we developed and studied the Flow-Light (based on WorkAnalytics), an approach to reduce the cost of in-person interruptions by combining a physical interruptibility indicator in the form of a traffic-light like LED with an automated interruptibility measurement[1] based on a user's computer interaction. We evaluated the FlowLight and its effects at the workplace in a large-scale and long-term field study with 449 knowledge workers

---

[1]Interruptibility is defined as the availability for an interruption. The interruptibility measurement is based on computer interaction data and quantifies how interruptible (or focused) a user currently is. The FlowLight turns red when the user is focused and the interruptibility is low, green when the interruptibility is higher, and yellow when the user is not actively working on the computer.

from 12 countries and 15 sites at ABB. Each team used the Flow-Light over a minimum of five weeks that included the collecting of baseline data and self-logging of interruptions before and after the installation of the FlowLight.

Results. Analyzing participants' self-reported interruptions shows that the FlowLight significantly (Wilcoxon signed-rank test, Z=-5.0, p ¡ 0.000001) reduced the number of interruptions from 4.1 to 2.2 (46%) per participant and day, while having little impact on the important interruptions. Analyzing the qualitative data, including the survey results and interviews, we found that the FlowLight increased the awareness on the cost of interruptions within the workplace, participants felt more productive, and most participants continued using it for up to 13 months and even after the study. A more detailed analysis of the impacts can be found in [25].

6 CONTRIBUTIONS AND OUTLOOK

The goal of my research is to provide software developers, tool builders and researchers with insights into developers' work and productivity, and offer ways to foster productive behavior through self-monitoring and goal-setting. The expected contribution of this PhD thesis can be summarized as follows:

1. A better understanding of software developers' perceptions of productivity, the different factors influencing their work, and the relationship between activities at work and productivity.
2. Design recommendations for building self-monitoring systems for developers at work, the information and features they expect, and the insights they can get.
3. A long-term explorative study on creating actionable insights to foster productive work, using a combination of persuasive strategies, including self-monitoring and goal-setting.
4. A large-scale, long-term field study investigating an automated approach to increase productivity by reducing interruptions at inopportune moments.
5. Several approaches to foster productive behaviors in the workplace, in particular WorkAnalytics and FlowLight, which both have been evaluated through field-studies and are still actively used at several companies.

The results of the studies addressing RQ1 are published in the TSE'17 journal [12] and at the ESEM'17 conference [15]. The study and results of the self-monitoring tool for developers' work are published at the CSCW'18 conference [14] (RQ2). The large-scale study investigating the impact of a team-awareness tool on interruptions and productivity at work is published at the CHI'17 conference [25] (RQ3b). I plan to focus the remaining time of my PhD on RQ3a to investigate how developers can be motivated to foster productive behaviors with actionable insights and goal-setting. The target is to write one or more publications to suitable SE (or HCI) conferences such as FSE'19 or ICSE'19. Finally, I plan to finish my PhD thesis and defend it in spring 2019.

REFERENCES

[1] M. Andreessen. 2011. Why software is eating the world. *The Wall Street Journal*, August 20. (2011).
[2] Frank Bentley, Konrad Tollmar, Peter Stephenson, and Levy Laura. 2013. Health Mashups: Presenting Statistical Patterns between Wellbeing Data and Context in Natural Language to Promote Behavior Change. 20, 5 (2013), 1–27.
[3] Lora E. Burke, Jing Wang, and Mary Ann Sevick. 2011. Self-Monitoring in Weight Loss: A Systematic Review of the Literature. *Journal of the American Dietetic Association* 111, 1 (2011), 92–102.
[4] Codealike. 2017. codealike.com. (2017). retrieved January 27, 2018.
[5] Sunny Consolvo, Predrag Klasnja, David W. McDonald, Daniel Avrahami, Jon Froehlich, Louis LeGrand, Ryan Libby, Keith Mosher, and James A. Landay. 2008. Flowers or a Robot Army?: Encouraging Awareness & Activity with Personal, Mobile Displays. In *Proceedings of the 10th International Conference on Ubiquitous Computing (UbiComp '08)*. ACM, 54–63.
[6] Sunny Consolvo, David W. McDonald, Tammy Toscos, Mike Y Chen, Jon Froehlich, Beverly Harrison, Predrag Klasnja, Anthony LaMarca, Louis LeGrand, Ryan Libby, Ian Smith, and James A. Landay. 2008. Activity sensing in the wild: a field trial of ubifit garden. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, 1797–1806.
[7] Tom DeMarco and Tim Lister. 1985. Programmer performance and the effects of the workplace. In *Proceedings of the 8th international conference on Software engineering*. IEEE Computer Society Press, 268–272.
[8] Prem Devanbu, Sakke Karstu, Walcélio Melo, and William Thomas. 1996. Analytical and Empirical Evaluation of Software Reuse Metrics. In *Proceedings of the 18th International Conference on Software Engineering (ICSE '96)*. IEEE Computer Society, 189–199.
[9] Thomas Fritz, Elaine M. Huang, Gail C. Murphy, and Thomas Zimmermann. 2014. Persuasive Technology in the Real World: A Study of Long-term Use of Activity Sensing Devices for Fitness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, 487–496.
[10] Young-Ho Kim, Jae Ho Jeon, Eun Kyoung Choe, Bongshin Lee, Kwonhyun Kim, and Jinwook Seo. 2016. TimeAware: Leveraging Framing Effects to Enhance Personal Productivity. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. 272–283.
[11] Gloria Mark, Victor M Gonzalez, and Justin Harris. 2005. No task left behind?: examining the nature of fragmented work. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 321–330.
[12] André N. Meyer, Laura E Barton, Gail C Murphy, Thomas Zimmermann, and Thomas Fritz. 2017. The Work Life of Developers: Activities, Switches and Perceived Productivity. *Transactions of Software Engineering* (2017), 1–15.
[13] André N. Meyer, Thomas Fritz, Gail C. Murphy, and Thomas Zimmermann. 2014. Software Developers' Perceptions of Productivity. In *Proceedings of the 22Nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2014)*. ACM, 19–29.
[14] André N. Meyer, Gail C Murphy, Thomas Fritz, and British Columbia. 2017. Design Recommendations for Self-Monitoring in the Workplace: Studies in Software Development. In *CSCW*, Vol. 1. 1–24.
[15] André N. Meyer, Thomas Zimmermann, and Thomas Fritz. 2017. Characterizing Software Developers by Perceptions of Productivity. In *Empirical Software Engineering and Measurement (ESEM), 2017 International Symposium on*.
[16] Audris Mockus, Roy T. Fielding, and James D. Herbsleb. 2002. Two Case Studies of Open Source Software Development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology* 11, 3 (2002), 309–346.
[17] Peter Naur and Brian Randell. 1969. NATO Software Engineering Conference. *Scientific Affairs Division, NATO* (1969).
[18] Chris Parnin and Spencer Rugaber. 2011. Resumption strategies for interrupted programming tasks. *Software Quality Journal* 19, 1 (2011), 5–34.
[19] Dewayne E. Perry, Nancy A. Staudenmayer, and Lawrence G. Votta. 1994. People, Organizations, and Process Improvement. *IEEE Software* 11, 4 (1994), 36–45.
[20] James O. Prochaska and Wayne F. Velicer. 1997. The Transtheoretical Change Model of Health Behavior. *American Journal of Health Promotion* 12, 1 (1997).
[21] RescueTime. 2017. rescuetime.com. (2017). retrieved January 27, 2018.
[22] John Rooksby, Parvin Asadzadeh, Mattias Rost, and Alistair Morrison. 2016. Personal Tracking of Screen Time on Digital Devices. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 284–296.
[23] Janice Singer, Timothy Lethbridge, Norman Vinson, and Nicolas Anquetil. 2010. An Examination of Software Engineering Work Practices. In *CASCON First Decade High Impact Papers (CASCON '10)*. IBM Corporation, 174–188.
[24] Steve Whittaker, Victoria Hollis, and Andrew Guydish. 2016. Don't Waste My Time: Use of Time Information Improves Focus. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*.
[25] Manuela Züger, Christopher Corley, André N. Meyer, Boyang Li, Thomas Fritz, David Shepherd, Vinay Augustine, Patrick Francis, Nicholas Kraft, and Will Snipes. 2017. Reducing Interruptions at Work: A Large-Scale Field Study of FlowLight. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. 61–72.