CHAPTER 22

Fitbit for Developers: Self-Monitoring at Work

André N. Meyer, University of Zurich, Switzerland
Thomas Fritz, University of Zurich, Switzerland
Thomas Zimmermann, Microsoft Research, USA

Self-Monitoring to Quantify Our Lives

Recently, we have seen an explosion in the number of devices and apps that we can use to track various aspects of our lives, such as the steps we walk, the quality of our sleep, or the calories we consume. People use devices such as the Fitbit activity tracker to increase and maintain their physical activity level by tracking their behavior, setting goals (e.g., 10,000 steps a day), and competing with friends. Generally, the miniaturization of self-tracking devices and their ubiquitousness make it possible to carry them around all the time and track more and more aspects of our lives. At the same time, studies have shown that these approaches can successfully encourage people to change their behavior, often motivated through persuasive technologies, such as goal-setting, social encouragement, and sharing mechanisms [3].

Notably, the interest for self-monitoring tools at the workplace is also increasing, and approaches to get insights into one's behavior and habits during work have emerged. Tools, such as RescueTime, allow users to get insights into the amount of time they spend in different applications on their computer, or Codealike visualizes to developers how they spent their time inside the IDE working in different code projects. Yet, little is known about developers' expectations of, their experience with, and the experience of self-monitoring in the workplace.

C. Sadowski and T. Zimmermann (eds.), *Rethinking Productivity in Software Engineering*,
https://doi.org/10.1007/978-1-4842-4221-6_22

Self-Monitoring Software Developers' Work

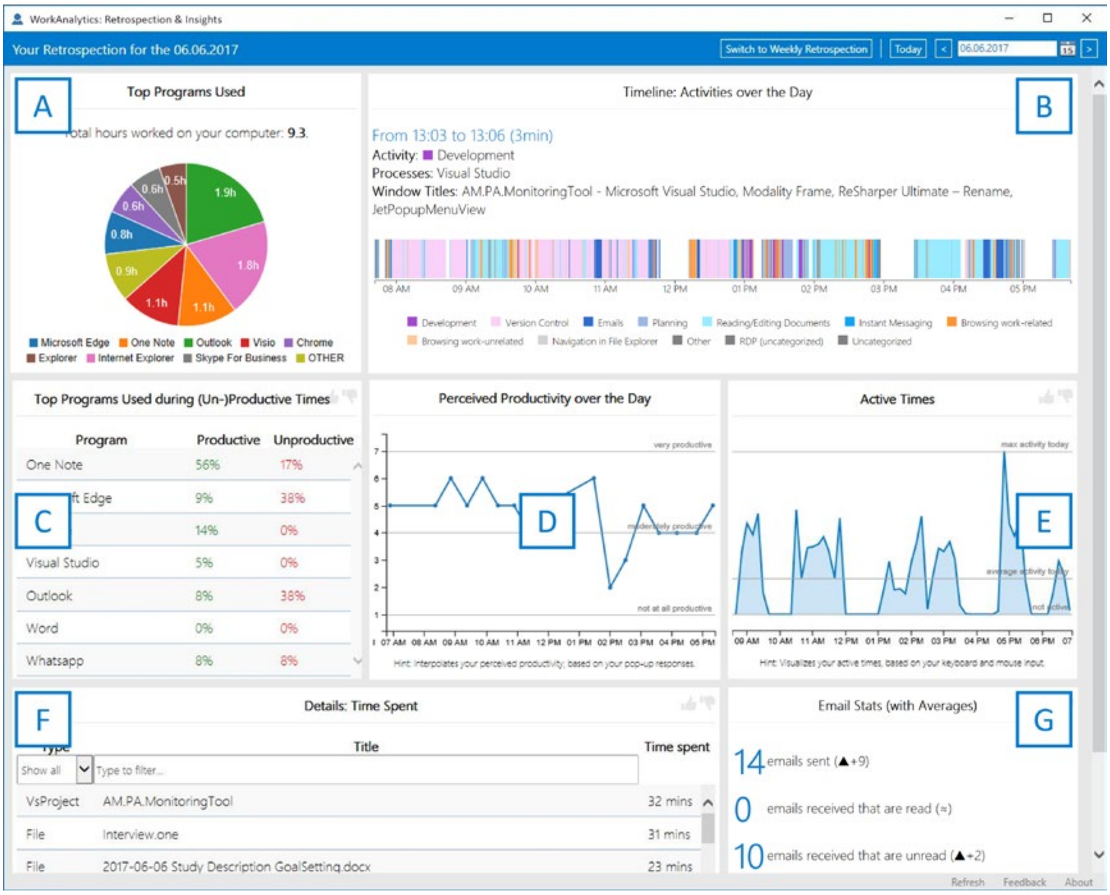
There are numerous factors that impact a software developers' success and productivity at work: interruptions, coordinating work with the team, requirements that change, the infrastructure and office environment, and many more (see Chapter 8). Developers are often not aware of how these factors impact both their own productivity and the work of others [1]. The success of self-monitoring approaches in other domains suggests that self-monitoring can improve the awareness of developers about their work. Developers can reflect about their actions and factors that increase or decrease their productivity and make informed decisions to improve their productivity. The captured data about developers' work and productivity could further allow developers to compare themselves to other developers with similar job profiles.

This idea is related to Watts Humphrey's work on the Personal Software Process (PSP) that aims to help developers better understand and improve their performance by tracking their estimated and actual development of code [2]. The research conducted to evaluate PSP showed promising results, including more accurate project estimations and higher code quality. Today, with sensors and data trackers being more ubiquitous and accurate, we can give developers the ability to measure their work and behavior changes automatically and provide a much broader set of insights.

To learn the requirements and best practices for self-monitoring systems for software developers, we ran a mixed methods study: a literature review, a survey with more than 400 developers, and an iterative feedback-driven approach with 5 pilot studies and a total of 20 software developers. The study revealed developers' expectations of features, measures of interest, and possible barriers toward the adoption of self-monitoring systems. We then built PersonalAnalytics, a self-monitoring tool targeted to developers and studied its impact and use with 43 professional software developers who used it during three workweeks.

PersonalAnalytics consists of three components: the monitoring component, the self-reporting pop-up, and the retrospection. The monitoring component captures information from various individual aspects of software development work, including application use, documents accessed, development projects worked on, websites visited, and collaborative behaviors from attending meetings, as well as using e-mail, instant messaging, and code review tools. The data collection runs nonintrusively in the background, requiring no additional input from the developer. In addition, PersonalAnalytics prompts developers to reflect on their work periodically and to-self report their perceived productivity using a pop-up. To enable more multifaceted insights, the captured data is visualized in a daily retrospection (see Figure 22-1), which also provides a higher-level overview in a weekly summary and allows users to relate various data with each other.

In this chapter, we share the lessons that we learned from building and evaluating PersonalAnalytics and the insights that users received from using the tool. We describe why these insights are sometimes not enough for a behavior change. Chapter 16 further extends the discussion on dashboards in software engineering, by debating about their need and risks.



Total hours worked on your computer: 9.3

From 13:03 to 13:06 (3min)
Activity: Development
Processes: Visual Studio
Window Titles: AM.PA.MonitoringTool - Microsoft Visual Studio, Modality Frame, ReSharper Ultimate – Rename, JetPopupMenuView

Program	Productive	Unproductive
One Note	56%	17%
ft Edge	9%	38%
	14%	0%
Visual Studio	5%	0%
Outlook	8%	38%
Word	0%	0%
Whatsapp	8%	8%

Hint: Interpolates your perceived productivity, based on your pop-up responses.

Hint: Visualizes your active times, based on your keyboard and mouse input.

Type	Title	Time spent
Show all	Type to filter...	
VsProject	AM.PA.MonitoringTool	32 mins
File	Interview.one	31 mins
File	2017-06-06 Study Description GoalSetting.docx	23 mins

14 emails sent (▲+9)

0 emails received that are read (≈)

10 emails received that are unread (▲+2)

Figure 22-1. *Daily retrospection in PersonalAnalytics. (A) displays the distribution of time spent in the most used programs, (B) shows a timeline of time spent in different activities, (C) depicts the most used programs and the amount of time the user self-reported feeling productive/unproductive while using them, (D) illustrates the user's self- reported productivity over time, (E) visualizes the user input from mouse and keyboard, (F) shows a detailed breakdown of how much time was spent on different information artefacts (including web sites, files, e-mails, meetings, code projects, code reviews), and (G) summarizes e-mail-related data such as the number of e-mails sent/received.*

Supporting Various Individual Needs Through Personalization

In our preliminary studies, developers expressed an interest in a large number of different measures when it comes to the self-monitoring of their work. To support these individually varying interests in work measures, we included a wide variety of measures into PersonalAnalytics and allowed users to personalize their experience by selecting the measures that were tracked and visualized. To capture the relevant data for these measures, PersonalAnalytics features multiple data trackers: the *Programs Used tracker* that logs the currently active process and window titles every time the user switches between programs or logs "idle" in case there was no user input for more than two minutes; the *User Input tracker* that collects mouse clicks, movements, scrolling, and keystrokes (no key logging, only time-stamp of any pressed key); and the *Meetings and E-mail trackers* that collect data on calendar meetings and e-mails received, sent, and read using the Microsoft Graph API of the Office 365 Suite [5].

After using PersonalAnalytics for several weeks, two-thirds of our users wanted to personalize and better fit the retrospection to their individual needs. They also wanted even more data on other aspects of their work. For instance, they wanted to compare themselves with their team members, get high-level measures such as their current focus or progress on tasks, and correlate their data with biometric data, such as their heart rate, stress level, sleep, and exercise.

The diverse requests for extending PersonalAnalytics with additional measures and visualizations emphasize the importance for personalization and customization of the experience to increase satisfaction and long-term engagement. While it might seem surprising that developers requested many development-unrelated measures to understand their work, this can be explained by the relatively low amount of time they usually spend with development-related activities, on average just between 9 percent and 21 percent, versus other activities such as collaborating (45 percent) or browsing the Web (17 percent) [4].

Self-Reporting Increases Developers' Awareness About Efficiency

PersonalAnalytics asks users to answer a pop-up survey once an hour on their computer. The collected data allows us to learn more about productivity and the tasks that developers work on. During the pilot studies, users expressed aversion toward the pop-up, as it included too many questions. After refining the pop-up to include only one question asking users to self-report productivity for the past hour, most started to like the pop-up. Two-thirds of the users mentioned that the brief self-reports increased their awareness about work and helped them assess whether they had spent their past work hour effectively, whether they had spent it working on something of value, and whether they had made progress on their current task:

"The hourly interrupt helps to do a quick triage of whether you are stuck with some task/problem and should consider asking for help or taking a different approach."

PersonalAnalytics does not automatically measure productivity but rather lets users self-report their productivity. This was highly valued by users as many do not think an automated measure can accurately capture an individual's productivity, similar to what is discussed in Chapters 2 and 3.

"One thing I like about [PersonalAnalytics] a lot is that it lets me judge if my time was productive or not. So just because I was in a browser or Visual Studio doesn't necessarily mean I was being productive or not."

These findings emphasize that self-reporting can be of value to users as it increases their awareness about work. It is yet to be seen how long the positive effects of self-reporting last and whether users lose interest at some point.

Retrospection About Work Increases Developers' Self-Awareness

The users of PersonalAnalytics liked the ability to self-reflect on work and productivity with the retrospection that visualizes a personalized list of measures; 82 percent said that the retrospection increased their awareness and provided novel insights. The insights included how developers spend their time collaborating or making progress on tasks, their productivity over the course of a day, or the fragmentation at work. The time spent further rectified some misconceptions users had about their work, such as how much time they actually spent with e-mails and work-unrelated browsing (for example, Facebook):

"[PersonalAnalytics] is awesome! It helped confirm some impression I had about my work and provided some surprising and very valuable insights I wasn't aware of. I am apparently spending most of my time in Outlook."

"I did not realize I am as productive in the afternoons. I always thought my mornings were more productive but looks like I just think that because I spend more time on e-mail."

Actionable Insights Foster Productive Behavior Changes

Naturally, most users of self-monitoring tools don't just want to learn about themselves but also want to improve themselves. We asked the users of PersonalAnalytics about what behaviors they changed. Interestingly, this study resulted in ambivalent responses. Roughly half of the users changed some of their habits based on what they learned from reflecting about their work. This includes trying to better plan their work, e.g., by taking advantage of more productive afternoons, trying to optimize how they spend their time with e-mails, or trying to focus better and avoid distractions, e.g., by closing the office door or listening to music when the background noise is distracting. However, the other half of our users didn't change their behavior, either because they didn't want to change something or because they were not sure what to change. These users reported that some of the new insights were not concrete and actionable enough for knowing what or how to change:

"While having a retrospection on my time is a great first step, I gained interesting insights and realized some bad assumptions. But ultimately, my behavior didn't change much. Neither of them have much in way of a carrot or a stick."

"It would be nice if the tool could provide productivity tips, ideally tailored to my specific habits and based on insights about when I'm not productive."

To improve the actionability of the insights, users asked for specific recommendations that encourage more focused work, e.g., to start a focused work block using the Pomodoro technique, to recommend a break from work for when they were stuck on the same task for too long, all the way to intervening and blocking certain applications or websites for a certain time:

"Warnings if time on unproductive websites exceeds some amount, and perhaps provide a way for the user to block those sites (though not forced)."

Besides providing developers with personalized recommendations for improvements based on their work behavior, allowing them to benchmark and compare themselves with their team or other developers could lead to insights that are actionable enough to change a behavior. For example, PersonalAnalytics could collect anonymized measures about developers' work habits, such as fragmentation, time spent on activities, and achievements; correlate the measures with other developers with similar job profiles; and present the comparisons to the developer. Insights could reach from letting a developer know that others spend more time reading development blogs to further educate themselves, all the way to informing them that they spend way more time in meetings than most other developers.

Increasing Team Awareness and Solving Privacy Concerns

One drawback of giving developers insights only into their own productivity is that their behavior changes might have negative impact on the overall team productivity. As an example, a developer who blocks out interruptions at inopportune times to focus better could be blocking a co-worker who needs to ask a question or clarify things. Also receiving insights into how the team coordinates and communicates at work could help developers make more balanced adjustments with respect to the impact their behavior change might have on the team. For example, being aware of co-workers' most and least productive times in a workday could help to schedule meetings during times where everybody is the least productive and where interrupting one's work for a meeting has the least effect. Being more aware of the tasks each member of the team is currently working on and how much progress they are making could also be useful for managers or team leads to identify problems early, e.g., a developer who is blocked on a task or uses communication tools inefficiently, and take appropriate action.

However, these additions to a workplace self-monitoring tool would require aggregating and analyzing the data from multiple developers, which could result in privacy concerns given the possibly sensitive nature of the data. When creating tools that include data from multiple users, tool builders need to ensure privacy, e.g., by giving users full control over what data is being captured and shared, by properly obfuscating the data, and by being transparent about how the data is being used. If not done properly, this could severely increase pressure and stress for developers.

A recurring theme during the pilots and initial survey was the users' need to keep sensitive workplace data private. Some users were afraid that sharing data with their managers or team members could have severe consequences on their employment or increase pressure at work. To account for privacy needs at work, PersonalAnalytics, among other precautions, stores all logged data only locally on the user's machine, rather than having a centralized collection on a server. This enables users to retain full control of the captured data. While a few users were initially skeptical and had privacy concerns, no privacy complaints were received during the study, and the majority even shared their obfuscated data with us for analyzing it. While some users mentioned that they voluntarily exchanged their visualizations and insights with teammates to compare themselves, others mentioned that they would start to game the tool or go as far as leave the company, in case their manager would force them to run a tracking tool that would ignore their privacy concerns.

We think that the chances of misuse of the data and developers' sensitivity will decline if managements establish an environment where the data is used for process improvements only and not for HR-related evaluations. Also, making comparisons across teams with absolute data might lead to wrong conclusions since conditions can differ so much between different teams, projects, and systems. Hence, the delta improvements such as behavior changes and trends are important to consider. Nonetheless, further research is required to determine how workplace data can be leveraged to improve team productivity, while respecting and protecting employee privacy, including data protection regulations such as the GDPR [7]. This topic is explored in more depth in Chapter 15.

Fostering Sustainable Behaviors at Work

One way to foster software developers' productivity is to increase their self-awareness about work and productivity through self-monitoring. We found that regular self-reflection using the retrospection and minimal-intrusive self-reports allows developers to increase their awareness about time spent at work, their collaboration with others, their productive and unproductive work habits, and their productivity in general. You also learned that developers are interested in a large and diverse set of measurements and correlations within the data and that the insights gained from looking at the visualized data is not always concrete and actionable enough to motivate behavior changes. Detailed descriptions of the studies and more findings can be found in the corresponding paper [6]. In the future, we could imagine that self-monitoring tools for developers at their workplace will be extended to include an even richer set of measures that can be correlated with each other. For example, by allowing integrations with development tools (e.g., GitHub, Visual Studio, or Gerrit) and biometric sensors (e.g., Fitbit), developers could be warned to carefully review their changes again before checking in a breaking code change after having slept badly in the night before. Another possibility to foster productive behavior changes is goal-setting. Workplace self-monitoring tools could be extended to not only enable developers to gain rich insights but also motivate them to identify meaningful goals for self-improvements and allow them to monitor their progress toward reaching them. Finally, anonymized or aggregated parts of the data could be shared with the team, to increase the awareness within the team and reduce interruptions, to improve the scheduling of meetings, and to enhance the coordination of task assignments.

We open-sourced PersonalAnalytics on Github (`https://github.com/sealuzh/PersonalAnalytics`), opening it up to contributions and making it available for use.

Key Ideas

Here are the key ideas from this chapter:

- Self-monitoring personal behavior at work can improve developers' performance for a substantial proportion of developers.
- Self-reporting productivity allows developers to briefly reflect about their efficiency and progress at work and take timely actions that improve productivity.
- Developers have a diverse interest in measures about their work, ranging from development related data to data about their collaboration in the team, all the way to biometric data.

References

[1] Dewayne E. Perry, Nancy A. Staudenmayer, and Lawrence G. Votta. 1994. People, Organizations, and Process Improvement. IEEE Software 11, 4 (1994), 36–45.

[2] Watts S. Humphrey. 1995. A discipline for software engineering. Addison-Wesley Longman Publishing Co., Inc.

[3] 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 International Conference on Human Factors in Computing Systems.

[4] 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.

[5] Microsoft Graph API. `https://graph.microsoft.io`.

[6] André N. Meyer, Gail C Murphy, Thomas Zimmermann, and Thomas Fritz. 2018. Design Recommendations for Self-Monitoring in the Workplace: Studies in Software Development. To appear at CSCW'18, 1–24.

[7] European General Data Protection Regulation (GDPR). 2018. `https://www.eugdpr.org`.