CHAPTER 10

Happiness and the Productivity of Software Engineers

Daniel Graziotin, University of Stuttgart, Germany

Fabian Fagerholm, Blekinge Institute of Technology, Sweden and University of Helsinki, Finland

Software companies nowadays often aim for flourishing happiness among developers. Perks, playground rooms, free breakfast, remote office options, sports facilities near the companies...there are several ways to make software developers happy. The rationale is that of a return on investment: happy developers are supposedly more productive and, hopefully, also retained.

But is it the case that *happy software engineers = more productive software engineers*[1]? Moreover, are perks the way to go to make developers happy? Are developers happy at all? These questions are important to ask both from the perspective of productivity and from the perspective of sustainable software development and well-being in the workplace.

This chapter provides an overview of our studies on the happiness of software developers. You will learn why it is important to make software developers happy, how happy they really are, what makes them unhappy, and what is expected for their productivity while developing software.

[1]In our studies, we consider a software developer to be "a person concerned with any aspect of the software construction process (such as research, analysis, design, programming, testing, or management activities), for any purpose including work, study, hobby, or passion." [4, page 326]. We also interchange the terms *software developer* and *software engineer* so that we do not repeat ourselves too many times.

C. Sadowski and T. Zimmermann (eds.), *Rethinking Productivity in Software Engineering*,
https://doi.org/10.1007/978-1-4842-4221-6_10

Why the Industry Should Strive for Happy Developers

We could think that happiness is a personal issue that individual developers are responsible for on their own time. In this line of thinking, software companies should focus on maximizing the output they get from each developer. However, to get productive output from a human, we must first invest. As humans, software developers' productivity depends on their skills and knowledge—but to access those, we need to create favorable conditions that allow the human potential to be realized. As noted in Chapter 5, developer satisfaction is important for productivity because reduced satisfaction can incur future costs; it follows that companies should be interested in the general well-being of their software developers. Furthermore, we believe we should simply strive to create better working environments, teams, processes, and, therefore, products.

What Is Happiness, and How Do We Measure It?

This is a very deep question that ancient and modern philosophers have aimed to answer in more than one book. However, present-day research does give us concrete insight into happiness and ways to measure it. We define happiness (as many others do) as a sequence of experiential episodes. Being happy corresponds to frequent positive experiences, which lead to experiencing positive emotions. Being unhappy corresponds to the reverse: frequent negative experiences leading to negative emotions. Happiness is the difference or balance between positive and negative experiences. This balance is sometimes called *affect balance*.

The Scale of Positive and Negative Experience (SPANE, [8]) is a recent but valid and reliable way to assess the affect balance (happiness) of individuals. Respondents are asked to report on their affect, expressed with adjectives that individuals recognize as describing emotions or moods, from the past four weeks. This provides a balance between the sampling adequacy of affect and the accuracy of human memory to recall experiences and reduce ambiguity. The combination of the scoring of the various items yields an affect balance (SPANE-B) score, which ranges from -24 (extremely unhappy) to +24 (extremely happy), where 0 is to be considered a neutral score of happiness.

Scientific Grounds of Happy and Productive Developers

While it is intuitive that happiness is beneficial for productivity and well-being, these ideas are also supported by scientific research. We have previously shown that happy developers solve problems better [1], that there is a relationship between affect and how developers assess their own productivity [2], and that software developers themselves are calling for research in this area [5]. We have also presented a theory that provides an explanation of how affect impacts programming performance [3]: events trigger affects in programmers. These affects might earn importance and priority to a developer's cognitive system, and we call them *attractors*. Together with affects, attractors drive or disturb programmers' focus, which impacts their performance. On a larger scale, our studies show that affect is an important component of performance in software teams and organizations [11]. Affect is linked to group identity—the feeling of belonging to the group—affecting cohesion and social atmosphere, which in turn are key factors for team performance and retention of team members.

We will now consider four important and ambitious questions.

- How happy are software developers overall?
- What makes them (un)happy?
- What happens when they are (un)happy?
- Are happy developers more productive?

Answering these questions is challenging. We spent a year designing a comprehensive study [4, 6] to address them. We needed data from as many software developers as possible. We also needed as much diversity as possible in terms of age, gender, geographical location, working status, and other background factors. We designed and piloted a questionnaire in such a way that the results could be generalizable (with a certain error tolerance) to the entire population of software developers. Our questionnaire had demographic questions, SPANE, and open-ended questions asking about developers' feelings of happiness and unhappiness when developing software. We asked them to describe a concrete recent software development experience, what could have caused them to experience their feelings in that situation, and if their software development was influenced by these feelings in any way, and, if so, how.

We obtained 1,318 complete and valid responses to all our questions.

How Happy Are Software Developers?

In Figure 10-1, you can see how happy our 1,318 participants were.

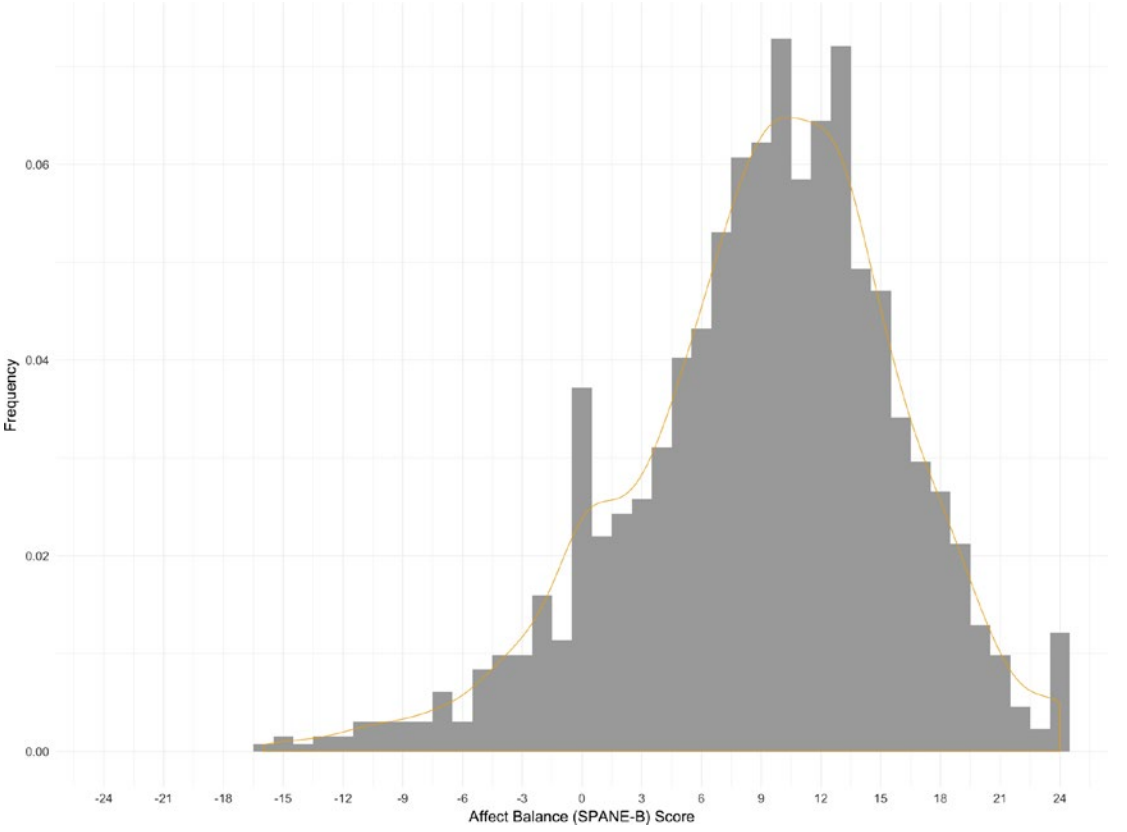

Figure 10-1. *Distribution of happiness of software developers (SPANE-B score)*

Our participants had a SPANE-B average score of 9.05, and we estimated the true mean happiness score of software developers to be between 8.69 and 9.43 with a 95 percent confidence interval. In other words, most software developers are moderately happy.

We compared our results with similar studies (Italian workers, U.S. college students, Singapore university students, Chinese employees, South African students, and Japanese college students). All results from other studies reported a mean SPANE-B score higher than 0 but lower than in our study. Software developers are indeed a *slightly happy* group—and they are happier than what we would expect based on knowledge about various other groups of the human population. This is good news, indeed, but there is room for improvement nonetheless. Some developers have a negative SPANE-B score, and there were many examples in the open responses about episodes of unhappiness that could be avoided.

What Makes Developers Unhappy?

Our analysis of the responses of our 1,318 participants uncovered 219 causes of unhappiness, which were mentioned 2,280 times in the responses [4]. We present here a brief summary of the results and the top three categories of things that make developers unhappy.

The causes of unhappiness that are controllable by managers and team leaders are mentioned four times as often as those being personal and therefore beyond direct managerial control. We also expected the majority of the causes to be related to human aspects and relationships. However, most of them came from technical factors related to the artifact (software product, tests, requirements and design document, architecture, etc.) and the process. This highlights the importance of strategic architecture and workforce coordination.

Being stuck in problem-solving and time pressure are the two most frequent causes of unhappiness, which corroborates the importance of recent research that attempts to understand these issues. We recognize that it is in software development's nature to be basically problem-solving under deadlines: we cannot avoid problem-solving in software development. However, developers *feel bad* when they are stuck and under pressure, and several detrimental consequences do happen (see the rest of this chapter). This is where researchers and managers should intervene to reduce the detrimental effects of time pressure and getting stuck. Psychological grit could be an important characteristic to train among software developers. Another could be how to switch your mind-set to get unstuck.

The third most frequent cause of unhappiness is to work with bad code and, more specifically, with bad code practices. Developers are unhappy when they produce bad code, but they suffer tremendously when they meet bad code that could have been avoided in the first place. As our participants stated, bad code can be a result of management decisions aiming to save time and effort in the short term. Similar negative effects were mentioned regarding third persons (such as colleagues, team leaders, or customers) who make developers feel inadequate with their work, forced repetitive mundane tasks, and imposed limitations on development. Many of the negative consequences can be avoided by rotating tasks, by making better decisions, and by actually listening to developers. Several top causes are related to perceptions of inadequacy of the self and others, validating recent research activities related to interventions that improve the affect of developers [3].

Finally, we see that factors related to information needs in terms of software quality and software construction are strong contributors to unhappiness among developers. Chapter 24 shows an example of how current software tools may overload developers with information and illustrates how problems related to information flow could be solved for individual developers, teams, and organizations. More research is needed on producing tools and methods that make communication and knowledge management in software teams easier and that help effortlessly store, retrieve, and comprehend information in all stages of the software development life cycle.

What Happens When Developers Are Happy (or Unhappy)?

We classified the answers to our open-ended questions and found dozens of causes and consequences of happiness and unhappiness while developing software [4, 6]. Developers in our study reported a variety of consequences of being unhappy. We have summarized these consequences in Figure 10-2. There is a pictogram for each major consequence, and they are divided into internal and external consequences. The internal consequences, pictured inside the mind of the developer, are directed toward developers themselves and have a personal impact. The external consequences are ones that have an effect outside the individual developer. They might impact a project, the development process, or a software artifact.

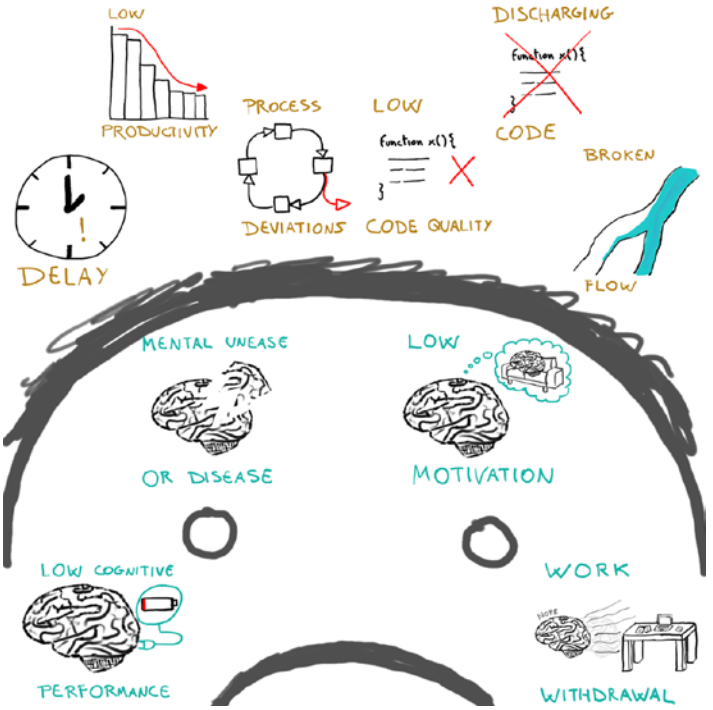

Figure 10-2. *Consequences of unhappiness while developing software. Available as CC-BY from Graziotin et al. [16]*

As you can see, developers reported several productivity-related consequences—and some even explicitly reported experiencing lower productivity. Other consequences include delays, process deviations, low code quality, throwing away code, and breaking the process flow in projects. These external effects are direct impacts on productivity and performance. Internal consequences, such as low motivation and reduced cognitive performance, indirectly affect productivity as well. Work withdrawal and mental unease, or, in the worst case, signs of disorders, are among the gravest consequences mentioned that impact developers personally.

For the purposes of this chapter, it is worth going into more detail on the consequences of happiness and unhappiness, because several of them are productivity-related and productivity was the most populated category of consequences. We are reporting them in an order that favors narrative, not by frequency of occurrence.

Cognitive Performance

We found that being happy or unhappy influences several factors related to cognitive performance, that is, how we efficiently process information in our brain. Happiness and unhappiness influence how we can focus while coding, as put by one participant: "[…] The negative feelings lead to not thinking things through as clearly as I would have if the feeling of frustration was not present." The opposite also holds true: "My software development is influenced because I can be more focused on my tasks and trying to solve one problem over another." As the focus can be higher when happy (or lower when unhappy), a natural consequence is that problem-solving abilities are influenced: "I mean, I can write codes and analyze problems quickly and with lesser or no unnecessary errors when I'm not thinking of any negative thoughts." Being happy while developing software brings higher learning abilities: "It made me want to pursue a master's in computer science and learn interesting and clever ideas to solve problems." However, being unhappy causes mental fatigue, and participants reported "getting frustrated and sloppy."

Flow

Participants mentioned how being unhappy caused breaks in their flow. *Flow* is a state of intense attention and concentration resulting from task-related skills and challenges being in balance (see more about that in Chapter 23). Unhappiness causes interruptions in developers' flow, resulting in adverse effects on the process. As put by a participant, "Things like that [of unhappiness] often cause long delays or cause one getting out of the flow, making it difficult to pick up the work again where one has left off." When happy, developers can enter a state of sustained flow. They feel full of energy and with strong focus. In such a state, they are "unaware of time passing." They can "continue to code without any more errors for the rest of the day" and "just knock out lines of code all day," with "dancing fingers." Flow is related to mindfulness, which is discussed in Chapter 25.

Motivation and Withdrawal

Motivation was often mentioned by our participants. They were clear in stating that unhappiness leads to low motivation for developing software: "[The unhappiness] has left me feeling very stupid, and as a result I have no leadership skills, no desire to participate, and feel like I'm being forced to code to live as a kind of punishment." The participants also stated that increased motivation occurred when they were happy.

Unhappiness and happiness are causes of work withdrawal and work engagement, respectively. Work withdrawal is a destructive consequence of unhappiness, and it emerged often among the responses. Work withdrawal is a family of behaviors that is defined as employees' attempts to remove themselves, either temporarily or permanently, from daily work tasks. We found varying degrees of work withdrawal, ranging from switching to another task ("[…] You spend like two hours investigating on Google for a similar issue and how it was resolved, you find nothing, and desperation kicks in.") to considering quitting developing software ("I really start to doubt myself and question whether I'm fit to be a software developer in the first place.") or even quitting the job. High work engagement and perseverance, on the other hand, were reported to occur when respondents were happy. This means, for example, pushing forward with a task: "I think I was more motivated to work harder the next few hours." This is slightly different from motivation, which is more about the energy directed to acting toward a goal. Work engagement is committing to the act of moving toward a goal.

Happiness and Unhappiness, and How They Relate to the Productivity of Developers

Finally, participants directly mentioned how unhappiness hinders their productivity. We grouped all responses related to performance and productivity losses. The responses within this category ranged from simple and clear ("productivity drops" and "[Negative experience] definitely makes me work slower") to more articulated ("[Unhappiness] made it harder or impossible to come up with solutions or with good solutions."). Unhappiness also causes delays in executing process activities: "In both cases [negative experiences] the emotional toll on me caused delays to the project." Of course, participants reported that happiness leads to high productivity: "When I have this [happy] feeling, I can just code for hours and hours," "I felt that my productivity grew while I was happy," and "The better my mood, the more productive I am." Here are more details on that by one participant: "I become productive, focused, and enjoy what I'm doing without wasting hours looking here and there in the code to know how things are hooked up together." An interesting aspect is that, when happy, developers tend to take on undesired tasks: "I think that when I'm in this happy state, I am more productive. The happier I am, the more likely I'll be able to accomplish tasks that I've been avoiding." On the other hand, unhappy developers could be so unproductive that they become destructive. We found some instances of participants who destroyed the task-related codebase ("I deleted the code that I was writing because I was a bit angry")

up to deleting entire projects ("I have deleted entire projects to start over with code that didn't seem to be going in a wrong direction."). Another intriguing aspect is about long-term considerations of being happy: "I find that when I feel [happy], I'm actually more productive going into the next task, and I make better choices in general for the maintenance of the code long-term. […] I'm more likely to comment code thoroughly."

Are Happy Developers More Productive?

But are happy developers *really* more productive? Whenever science attempts to show if a factor *X* causes an outcome *Y*, researchers design *controlled* experiments. Controlled experiments attempt to keep every possible factor constant (*A*, *B*, *C*, ...) except for the factors (*X*) that should cause a change to the outcome *Y*. You can find more about controlled experiments in Chapter 9. Whenever this control is not possible, we call these studies *quasi-experiments*.

Here is the issue with research on happiness: it is challenging to control the happiness (or the mood, the emotions) of people. One of the reasons is that a perfectly controlled experiment would need to be quite unethical to make the unhappy control group truly unhappy. The effects of asking participants to remember sad events, or showing depressing photographs, is negligible. Still, we set up two quasi-experiments to observe some correlations.

One of these studies [1] has received considerable media attention. We tested a hypothesis regarding a difference of intellectual (cognitive-driven) performance in terms of the analytical (logical, mathematical) problem-solving of software engineers according to how happy they were. We also wanted to perform a study where all the tools and measurements came from psychology research and were validated. So, we designed a quasi-experiment in a laboratory, where 42 BSc and MSc students of computer science had their happiness measured and then conducted a task resembling algorithmic design. For measuring happiness, we opted for SPANE (explained previously).

The analytic task was similar to algorithm design and execution. We decided to administer the Tower of London test (also known as Shallice test) to our participants. The Tower of London test resembles the Tower of Hanoi game. The test comprises two boards with stacks and several colored beads. There are usually three stacks per board, and each stack can accommodate only a limited number of beads. The first board presents predefined stacked beads. The participants received the second board, which has the same beads as the first board but stacked in a different configuration. The

participants have to re-create the configuration of the first board by unstacking one bead at a time and moving it to another stack. The Psychology Experiment Building Language (PEBL) is an open source language and a suite of neuropsychology tests [13, 14]. The Tower of London test is among them.

PEBL was able to collect the measures that let us calculate a score for the analytic performance. We compared the scores obtained in both tasks with the happiness of developers. The results showed that the happiest software developers outperformed the other developers in terms of analytic performance. We estimated the performance increase to be about 6 percent. The performance increase was not negligible, and we confirmed it by measuring Cohen's *d* statistic. Cohen's *d* is a number usually ranging from 0 to 2, which represents the magnitude of the effect size of a difference of means. Our Cohen's *d* for the difference between the two groups mean was 0.91—a large effect given that we did not obtain extreme cases of happiness and unhappiness. The margins could even be higher than that.

In another study [2], we did something more esoteric. We aimed to continue using psychology theory and measurement instruments for understanding the linkage between the real-time affect (let's say happiness) raised by a software development task and the productivity related to the task itself. Eight software developers (four students and four from software companies) worked on their real-world software project. The task length was 90 minutes (as it is about the typical length for a programming task). Each ten minutes, the developers filled a questionnaire formed by the Self-Assessment Manikin (SAM) and an item for self-assessing the productivity.

SAM is a scale for assessing an emotional state or reaction. SAM is peculiar because it is a validated way to measure the affect raised by a stimulus (like an object, or a situation) and it is picture-based (no words). SAM is simply three rows of puppets with different face expressions and body language. Therefore, it is quick for a participant to fill SAM, especially if implemented on a tablet (only three touches). We analyzed how developers felt during the task and how they self-assessed themselves in terms of productivity. Self-assessment is not a very objective way of measuring productivity, but it has been demonstrated that individuals are actually good at self-assessing themselves if they are observed alone [15]. The results have shown that high pleasure with the programming task and the sensation of having adequate skills are positively correlated with the productivity. This correlation holds over time. We also found that there are strong variations of affect in 90 minutes of time. Happy software developers are indeed more productive.

Potential Impacts of Happiness on Other Outcomes

Happiness influences so many things besides productivity, most of which are still related to development performance. Here we list three of them.

Unhappiness causes glitches in communication and a disorganized process: "Miscommunication and disorganization made it very difficult to meet deadlines." But happy developers can also mean more collaborative team members, leading to increased collaboration. Often, we saw a repeating pattern of willingness to share knowledge ("I'm very curious, and I like to teach people what I learned") and to join an effort to solve a problem ("We never hold back on putting our brains together to tackle a difficult problem or plan a new feature"), even when not related to the task at hand or the current responsibilities ("I was more willing to help them with a problem they were having at work.").

Being happy or unhappy influences not only the productivity of the code writing process but also the quality of the resulting code. Participants reported that "Eventually [due to negative experiences], code quality cannot be assured. So this will make my code messy, and more bug can be found in it," but also mentioned making the code less performant, or "As a result, my code becomes sloppier." Sometimes, being unhappy results in discharging quality practices ("[...] so I cannot follow the standard design pattern") as a way to cope with the negative experiences. Yet, being happy improves the quality of code. A participant told a small story about their work: "I was building an interface to make two applications talk. It was an exciting challenge, and my happy and positive feelings made me go above and beyond to not only make it functional but I made the UX nice too. I wanted the whole package to look polished and not just functional." When happy, developers tend to make less mistakes, see solutions to problems more easily, and make new connections to improve the quality of the code. A participant told us this: "When I'm in a good mood and I feel somehow positive, the code I write seems to be very neat and clean. I mean, I can write code and analyze problems quickly and with lesser or no unnecessary errors." As a result, the code is cleaner, more readable, better commented and tested, and with less errors and bugs.

The last factor we would like to report is mostly related to unhappiness, and it is quite an important one. It is about mental unease and mental disorder. We created this category to collect those consequences that threaten mental health. Participants reported that unhappiness while developing software is a cause of anxiety ("These kinds of situations make me feel panicky."), stress ("[The] only reason [for] my failure [is] due [to] burnout."), self-doubt ("If I feel particularly lost on a certain task, I may sometimes begin to question my overall ability to be a good programmer."), and sadness and feeling

depressed ("[…] feels like a black fog of depression surrounds you and the project."). In addition, we found mentions of feelings of being judged, frustration, and lack of confidence in one's ability.

What Does the Future Hold?

In 1971, Gerald Weinberg's book *The psychology of programming* [12] drew attention to the fact that software development is a human endeavor, and the humans doing it—the developers—are individuals with feelings. To this day, we still have more to understand about the human factor in software development. Software development productivity is still often managed as if it were about delivering code on an assembly line (see, e.g., Chapter 11). On the other hand, many companies do understand the importance of happy developers, invest in their well-being, and consider it to be worthwhile.

As we have shown, the link between happiness and productivity in software development is real. It is possible to quantify the happiness of software developers, and there are distinct patterns in the causes and consequences of their happiness.

What if we could include happiness as a factor in software development productivity management? In the future, an increasing number of people will work with digital products and services and perform tasks that are, in effect, software development. It would be worth investing in their happiness. It is important that we learn more about the relationship between well-being and software development performance. Rigorous research and educating practitioners on the research results are keys to improve the field. Besides sharp technical skills, we would like to give future software developers an understanding of the social and psychological factors that influence their own work.

Further Reading

In this chapter, we reported on several studies on the happiness of software engineers. Some of these studies [1, 2, 3, 5, 11] were self-contained and independent. Other studies [4, 6] are part of an ongoing project that we described in the section "Scientific Grounds of Happy and Productive Developers."

At the time of writing of this chapter, we still have to uncover all the categories, including those about what makes developers happy. We invite readers to inspect our open science repository [10], where we add new papers and results as we uncover them. The repository contains the entire taxonomy of what makes developers unhappy.

Key Ideas

Here are the key ideas from this chapter:

- Science says the industry should strive for happy developers.
- The overall happiness of software developers is slightly positive. Yet, many are still unhappy.
- The causes of unhappiness among software engineers are numerous and complex.
- Happiness and unhappiness bring a plethora of benefits and detriments to software development processes, people, and products.

References

[1] Graziotin, D., Wang, X., and Abrahamsson, P. 2014. Happy software developers solve problems better: psychological measurements in empirical software engineering. PeerJ. 2, e289. DOI=10.7717/peerj.289. Available: https://doi.org/10.7717/peerj.289.

[2] Graziotin, D., Wang, X., and Abrahamsson, P. 2015. Do feelings matter? On the correlation of affects and the self-assessed productivity in software engineering. Journal of Software: Evolution and Process. 27, 7, 467–487. DOI=10.1002/smr.1673. Available: https://arxiv.org/abs/1408.1293.

[3] Graziotin, D., Wang, X., and Abrahamsson, P. 2015. How do you feel, developer? An explanatory theory of the impact of affects on programming performance. PeerJ Computer Science. 1, e18. DOI=10.7717/peerj-cs.18. Available: https://doi.org/10.7717/peerj-cs.18.

[4] Graziotin, D., Fagerholm, F., Wang, X., and Abrahamsson, P. 2017. On the Unhappiness of Software Developers. 21st International Conference on Evaluation and Assessment in Software Engineering. 21st International Conference on Evaluation and Assessment in Software Engineering, 324–333. DOI=10.1145/3084226.3084242. Available: https://arxiv.org/abs/1703.04993.

[5] Graziotin, D., Wang, X., and Abrahamsson, P. 2014. Software Developers, Moods, Emotions, and Performance. IEEE Software. 31, 4, 24–27. DOI=10.1109/MS.2014.94. Available: https://arxiv.org/abs/1405.4422.

[6] Graziotin, D., Fagerholm, F., Wang, X., & Abrahamsson, P. (2018). What happens when software developers are (un)happy. Journal of Systems and Software, 140, 32-47. DOI=10.1016/j.jss.2018.02.041. Available: https://doi.org/10.1016/j.jss.2018.02.041

[7] Zelenski, J. M., Murphy, S. A., and Jenkins, D. A. 2008. The Happy-Productive Worker Thesis Revisited. Journal of Happiness Studies. 9, 4, 521–537. DOI=10.1007/s10902-008-9087-4.

[8] Diener, E., Wirtz, D., Tov, W., Kim-Prieto, C., Choi, D.-w., Oishi, S., and Biswas-Diener, R. 2010. New Well-being Measures: Short Scales to Assess Flourishing and Positive and Negative Feelings. Social Indicators Research. 97, 2, 143-156. DOI=10.1007/s11205-009-9493-y.

[9] Bradley, M. M. and Lang, P. J. 1994. Measuring emotion: The self-assessment manikin and the semantic differential. Journal of Behavior Therapy and Experimental Psychiatry. 25, 1, 49-59. DOI=10.1016/0005-7916(94)90063-9.

[10] Graziotin, D., Fagerholm, F., Wang, X., and Abrahamsson, P. 2017. Online appendix: the happiness of software developers. Figshare. Available: https://doi.org/10.6084/m9.figshare.c.3355707.

[11] Fagerholm, F., Ikonen, M., Kettunen, P., Münch, J., Roto, V., Abrahamsson, P. 2015. Performance Alignment Work: How software developers experience the continuous adaptation of team performance in Lean and Agile environments. Information and Software Technology. 64, 132–147. DOI=10.1016/j.infsof.2015.01.010.

[12] Weinberg, G. M. (1971). Psychology of Computer Programming (1 ed.). New York, NY, USA: Van Nostrand Reinhold Company.

[13] Piper, B. J., Mueller, S. T., Talebzadeh, S., Ki, M. J. 2016. Evaluation of the validity of the Psychology Experiment Building Language tests of vigilance, auditory memory, and decision making. PeerJ. 4, e1772. DOI=10.7717/peerj.1772. Available: https://doi.org/10.7717/peerj.1772.

[14] Piper, B. J., Mueller, S. T., Geerken, A. R, Dixon, K. L., Kroliczak, G., Olsen, R. H. J., Miller, J. K. 2015. Reliability and validity of neurobehavioral function on the Psychology Experimental Building Language test battery in young adults. PeerJ. 3, e1460. DOI=10.7717/peerj.1460. Available: https://doi.org/10.7717/peerj.1460.

[15] Miner, A. G., Glomb, T. M., 2010. State mood, task performance, and behavior at work: A within-persons approach. Organizational Behavior and Human Decision Processes. 112, 1, 43–57. DOI=10.1016/j.obhdp.2009.11.009.

[16] Graziotin, Daniel; Fagerholm, Fabian; Wang, Xiaofeng; Abrahamsson, Pekka (2017): Slides for the consequences of unhappiness while developing software. https://doi.org/10.6084/m9.figshare.4869038.v3.