Flow Experience in Software Engineering

Saima Ritonummi
Faculty of Information Technology
University of Jyvaskyla
Jyvaskyla, Finland
saima.e.ritonummi@jyu.fi

Valtteri Siitonen
Faculty of Information Technology
University of Jyvaskyla
Jyvaskyla, Finland
valtteri.m.e.siitonen@jyu.fi

Markus Salo
Faculty of Information Technology
University of Jyvaskyla
Jyvaskyla, Finland
markus.t.salo@jyu.fi

Henri Pirkkalainen
Unit of Information and Knowledge Management
Tampere University
Tampere, Finland
henri.pirkkalainen@tuni.fi

Anu Sivunen
Department of Language and Communication Studies
University of Jyvaskyla
Jyvaskyla, Finland
anu.e.sivunen@jyu.fi

ABSTRACT

Software engineering (SE) requires high analytical skills and creativity, which makes it an excellent context for experiencing flow. Although previous work in the SE context has identified how positive affect and development tools can support the flow experience, there is still much to uncover about the characteristics of software developers' flow experiences. To address this gap in knowledge, we conducted a qualitative critical incident technique (CIT) questionnaire (n = 401) on the flow-facilitating factors and characteristics of flow in the SE context. The most important flow-facilitating factors in developers' work included *optimal challenge*, *high motivation*, *positive developer experience (DX)*, and *no distractions or interruptions*. The flow experiences were characterized by *absorption*, *effortless control*, *intrinsic reward*, and *high performance*. Our study identifies the features of flow commonly addressed in flow research; however, it also highlights how IT use, especially development tools that provide positive DX, as well as being able to work without excessive distractions and interruptions are important facilitators of developers' flow.

CCS CONCEPTS

• Human-centered computing → Human computer interaction → Empirical studies in HCI • Software and its engineering → Software creation and management

KEYWORDS

Software engineering, software development, flow experience

ESEC/FSE '23, December 3–9, 2023, San Francisco, CA, USA

ACM ISBN 979-8-4007-0327-0/23/12.
https://doi.org/10.1145/3611643.3616263

ACM Reference Format:
Saima Ritonummi, Valtteri Siitonen, Markus Salo, Henri Pirkkalainen, and Anu Sivunen. 2023. Flow Experience in Software Engineering. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '23) December 3–9, 2023, San Francisco, CA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3611643.3616263

1 INTRODUCTION

Flow is defined as enjoyment and absorption in an activity that requires high levels of skills, provides increasing levels of challenges, and is intrinsically motivating for an individual. An essential element of flow is an *autotelic* experience; that is, engaging in an activity that is rewarding in and of itself [1]. As working in software engineering (SE) requires high analytical skills [2], deep involvement in problem-solving [3], and creativity [4], developers' work can be expected to be a great source of flow. However, the technology-driven nature of the industry also has its unique characteristics, including continuous learning (e.g., updating individual skills to keep pace with constantly evolving tools) [5][6] and the importance of developer experience (DX) of the tools developers work with [4][7], which can affect work-related flow. Moreover, flow is important for work motivation. Motivated employees are more engaged in their tasks and perform better [8]. Motivated developers have been found to be more productive and remain in the organization, while demotivated software engineers are more likely to have turnover intentions and take sick leave [9]. Because the software industry is a substantial segment of the global economy [10], the motivation and productivity of those working in it impact not only individual well-being but also the performance of the sector.

For decades, flow has been a widely researched topic in psychology. More recently, it has been explored in SE. However, more research needs to be conducted in this field. Previous studies of flow in the SE context have examined how emotions can indicate being in flow [11], the barriers to experiencing flow [12][13], and, regarding information technology (IT) use, how DX affects flow (e.g., [7][14]). Scholars have also investigated how flow relates to developers' productivity [15] and the overall

assessment of their work performance and satisfaction [16]. While some aspects of developers' work that influence the experience of flow have received attention in recent years, developers' flow still requires a more holistic examination to gain a deeper understanding of its different dimensions in SE [2][4]. Furthermore, research addressing flow in this field has usually focused on students (e.g., college students participating in programming courses) [2][17] or part-time developers (e.g., scholars) [14], which calls for more studies on professional developers. In addition, although flow has been studied in the human–computer interaction (HCI) context since the 1990s [18][19][20], it can be argued that our understanding of the human-centric aspects of SE (e.g., how developers interact with IT and each other) [21] and of the patterns in employees' IT use that can facilitate flow [22] has been limited. Therefore, this study aims to comprehend flow in SE and, given the technology-intensive nature of developers' work, in the context of HCI.

We address the abovementioned research gaps by using a qualitative critical incident technique (CIT) approach. We ask the following questions: *What are the characteristics of software developers' flow experiences? How can software developers' flow experiences be facilitated?* We are also interested in which stages of the development process flow experiences occurred in most often and which technologies were involved in these experiences. To answer these questions, this study used a qualitative CIT questionnaire that included open- and closed-ended questions about an outstandingly strong flow experience in the respondent's work. The CIT is a useful method for collecting detailed descriptions of *critical incidents* that have significantly influenced an individual's activities—either positively or negatively [23]. Primary data collection was conducted via Prolific, an online panel that facilitates the collection of high-quality questionnaire responses [24][25]. We obtained 401 responses from individuals working in SE, who described their flow experiences. The findings highlighted that antecedents of flow, especially optimal challenge, and context-specific/situational factors, such as positive DX with integrated development environments (IDEs), help make work smoother and more enjoyable, facilitating flow.

This study contributes to research on the human aspects of SE (e.g., [26][27]) and the contextual understanding of flow [28] in SE. To the best of our knowledge, this is among the first empirical attempts to holistically examine flow experiences among professional software developers. The study's findings increase our knowledge of flow in SE and provide a foundation for further research on the topic. The practical implications for management and developers include the benefits of identifying flow-facilitating practices and tools. Applying these can provide opportunities to experience more frequent flow experiences in SE. Also, more flow can improve developers' performance, self-actualization [22], work motivation, and work engagement [8].

2 BACKGROUND AND RELATED WORK

In psychology, flow is defined as an optimal experience and a state of enjoyment and deep concentration associated with intrinsically motivated activities [1]. The characteristics of developers' work (e.g., complex problem-solving that requires a specific set of skills) enable assigning intrinsic rewards to work, which makes software development an excellent flow activity [2]. In this study, flow experiences refer to Csikszentmihalyi's conceptualization of flow [1][29][30]. Therefore, other concepts often used in the SE context (e.g., control flow and workflow) are not discussed here. These concepts describe *processes*, whereas flow experience describes an individual's *inner state*.

2.1 The Flow Experience

The concept of flow originates from positive psychology and studies on *autotelic activities* [30]. Autotelic activities are enjoyable and intrinsically rewarding—that is, they are rewarding in and of themselves and are not done for extrinsic rewards, such as money. In the literature, flow is often considered as having three conditions under which it can occur: 1) when the perceived challenges presented by the activity correspond to one's skills, 2) when there are clear proximal goals for what is being done, and 3) when the activity provides immediate feedback about the progress being made. The characteristics of being *in* flow include deep involvement in the activity (often described as intense and focused concentration), a merging of action and awareness, loss of reflective self-consciousness, a sense of control, an altered sense of time, and autotelic experience [29][30][31]. Together, the conditions and characteristics of flow constitute the dimensions of flow (see Figure 1) [29]. Some researchers have adopted a three-dimensional view in which flow is characterized by absorption, enjoyment, and intrinsic interest (e.g., [32][33]). In their review, Norsworthy et al. [28] suggest that most conceptualizations of flow conditions (or *flow antecedents*) seem to fall under *optimal challenge* (e.g., clear goals and immediate feedback) and *high motivation*, while the experiential dimensions of flow fall under *absorption* (e.g., intense concentration, merging of action and awareness, altered sense of time, and loss of reflective self-consciousness), *effortless control* (e.g., sense of control), and *intrinsic reward* (e.g., autotelic experience).

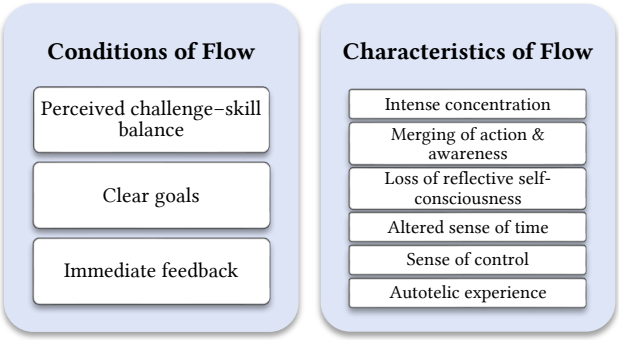

Figure 1: Dimensions of Flow

Flow is associated with goal-directed activities that provide increasing challenges [34]. Thus, flow usually occurs when the mind or the body (or both) is stretched to its limits, and something challenging and worthwhile is accomplished as a result [1]. In

addition to intrinsic motivation, the motivational aspect of flow is also linked to a sense of achievement [35][36][37] and creativity, as creative thinking typically requires the use of a considerable amount of psychic energy and perseverance (i.e., the ability to discover and formulate new problems) [37]. Although flow is often regarded as a motivational state, there are differences in how different personalities experience situations. Individuals with an *autotelic personality* are motivated by intrinsic rewards; they are more likely to feel in control of their actions and be motivated in situations that involve high challenges and require a high level of skills (for non-autotelics, these conditions can be demotivating) [1][29][38]. Since software development involves creativity and complex problem-solving [2], many of those working in the field likely have autotelic personalities.

This study focuses on the participants' descriptions of their experiences and the facilitators of their flow. Although flow conditions share commonalities with flow facilitators, they are not entirely synonymous. While the *conditions* of flow describe the interaction between the individual and the activity, *facilitators* include other circumstances, such as other people and situational factors, which can be context-specific (e.g., related to one's job characteristics) [39][40][41][42].

2.2 Flow and Related Concepts in HCI

Intrinsically motivated, goal-directed activities make IT use a great potential source of flow experiences. As is the case with any flow activity, the conditions of flow in the HCI context include tasks that provide enjoyable challenges [43]. Though there are different streams of flow research in this context, one common approach is the association of four dimensions to flow in IT use: control, curiosity, intrinsic interest, and attention [18][19]. These four dimensions have also been conceptualized as *technoflow* [22]; the last three have been termed *cognitive engagement* [44]. Similarly, Agarwal and Karahanna [45] introduced the concept of *cognitive absorption*, a state of deep involvement with software characterized by temporal dissociation, focused immersion, heightened enjoyment, control, and curiosity. The association of positive appraisals with IT use (e.g., being able to work faster or make fewer errors) has been described as *techno-eustress* [46]. Although they share similarities with flow dimensions, these concepts highlight the interaction between the user and IT. At the same time, as there are multiple streams of flow research in the HCI field, it has been suggested that a unified conceptualization of flow is needed to design better and more engaging artifacts as well as user experiences (UX) [47] that satisfy both functional and hedonic needs (e.g., fun, excitement, immersion, and flow) for interaction [48][49]. One challenge in examining flow in the HCI context has been viewing flow experience and technology as ontologically separate entities, even though the properties of individuals and technology are not separate but entangled (i.e., they both possess relational attributes that shape the activities that the individual and the artifact can perform) [22].

Furthermore, examining the flow enabled by IT use in the work context provides an opportunity to identify patterns in work practices that support employees' ability to experience flow [8]. Research has identified various outcomes of work-related IT use, including *techno-work engagement* [50], the positive and negative effects of technology-mediated interruptions [51], the autonomy paradox [52], and either increased or decreased job performance [53]. Although the overall effects of IT use at work seem to be paradoxical [54][55], this study addresses the positive outcomes of IT use for developers, especially how the use of different tools and other work characteristics are related to their flow experiences.

2.3 Flow in SE

Flow has received considerable attention in the work context (e.g., [8][32][33]) and the IT work context (e.g., [56]). In contrast, only a few empirical studies have focused on the psychological phenomenon of flow experience [57] in SE (e.g., [58]). There are even fewer studies on professional, full-time developers. Scholars have suggested that developers' emotions can correlate with their perceived progress in their tasks and indicate whether they are in flow [11][26]. Moreover, feeling creative has been attributed to being more productive and experiencing more flow during the workday [59][60]. Studies of developers' *productivity* have found that developers tend to feel productive when they have attainable, clearly defined goals or requirements and the possibility of working without excessive distractions, interruptions, or context switches, which can help reach flow (e.g., [15][61][62][63][64]).

In addition to the opportunities to experience flow, there are many flow-inhibiting factors in developers' work. These include a mismatch between the perceived challenge and skills, working with insufficient requirements (i.e., lack of clear goals), and interruptions [12], especially during immersive tasks [13]. Moreover, having problems with technology is a considerable flow barrier in developers' work [12]. An example is weaknesses in development tools' UX, which can be costly for work productivity and even prevent the adoption of tools [4]. However, it has been suggested that development tools that can provide positive DX support flow and intrinsic motivation in developers' work [65][66]. Much like the concept of UX, DX comprises the cognitive, affective, and intentional aspects of HCI, such as intrinsic motivation and flow [7]. Studies exploring DX have increased our understanding of developers' interactions with development tools [4], such as IDEs [66]; they have also provided concrete design guidelines for facilitating good DX for these tools (e.g., [7][58]).

3 METHOD

Qualitative studies of flow experiences are important for gaining a context-specific understanding of the phenomenon because they allow participants to describe these experiences in their own words [67]. To investigate flow in the context of SE, we used a CIT-based qualitative questionnaire. The CIT is an established qualitative method for collecting self-reports of critical incidents [23][68], such as outstandingly negative or positive experiences. This technique is especially suitable for capturing human behavior in contexts that are episodic in nature (e.g., work tasks) [69]. In addition to being widely used in the work context, this

method has been utilized in IT use research (e.g., [70][71]) and SE research [72].

CIT studies can be conducted either by making direct observations or by collecting retrospective critical incident reports [69]. Retrospective self-reporting is also common in flow studies (e.g., [8]). Because the flow state is characterized by deep absorption in an activity, retrospection is argued to be the most effective way to access this state [29]. Another important consideration in flow studies is determining which experiences count as flow and which do not. Doing so with predetermined measurements (e.g., the balance between perceived challenge and skills) can be challenging. Hence, one strategy that has been suggested is to provide respondents with a description of flow and ask them whether they have experienced it. If they have, they are asked to describe their experience in detail [34], which is similar to the CIT approach. In this study, we collected retrospective reports of critical incidents (i.e., flow experiences).

3.1 Data Collection

We designed a qualitative online questionnaire[1] in LimeSurvey to gain context-specific insights into flow experiences in developers' work. The questionnaire included open- and closed-ended questions based on established phrasings used in CIT research (e.g., [71]) and flow questionnaires (e.g., [8][32]). Initially, we provided a brief definition of flow to ensure that the respondents were familiar with the concept. This paper focuses on the open-ended questions, in which the participants were asked to recall and describe *an outstanding flow experience* they had had in as much detail as possible (Q2), the factors that facilitated the flow (Q3), and the technologies involved in the experience (Q1). The questionnaire included other questions not discussed in this study (e.g., queries regarding what prevented the respondent from experiencing more flow in their work).

The questionnaire also comprised closed-ended questions designed to collect additional information on the experiences, such as the duration of the experience (Q5), how long ago the experience occurred (Q6), how often the respondent experienced similar flow at work (Q8), and which stage(s) of the development process was the experience related to (Q4). The participants were allowed to choose all applicable stages (planning, development/programming, testing, implementation, and maintenance), in addition to an open field for other stages. It was important to allow this freedom because the stages can vary among different development methodologies (e.g., Agile, DevOps). Furthermore, the closed-ended questions included an assessment of the positive effect of the experience on the respondent's work (i.e., how critical the incident was perceived to be) (Q7) as well as statements about work-related flow (Q9) based on flow operationalizations in the work context [8][32]. These statements were used in the data analysis phase to help determine whether the experiences described in the responses counted as flow or not [73].

The questionnaire was created in Finnish and translated into English. The translation was proofread by an English language professional to ensure that the central concepts had been accurately translated. Before carrying out the study, the questionnaire was pretested by three developers to make sure that the questions were understandable.

The data were collected in two phases between December 2021 and April 2022. The pilot study was conducted by contacting international software companies, and 55 responses were obtained. After the pilot data were found to be of sufficient quality, primary data collection was conducted using the Prolific online panel, a useful service for collecting high-quality data (see e.g., [24][25]). We employed the same questionnaire design in both phases, but to target developers in Prolific, we applied prescreening criteria for people working in software-intensive industries.[2] We also added attention check questions, as suggested by the Prolific guidelines, and we only accepted responses from participants with a minimum of 20 previous submissions, with an approval rate of 97% or higher. This resulted in 437 responses, of which 401 were included in the final sample. See Table 1 for demographic information of the sample.

Table 1: Final Sample Demographics (n = 401)

Gender	**%**	**Employment Status**	**%**
Male	76.6	Full-timer	77.6
Female	22.2	Part-timer	8.7
Other	1.0	Freelancer	4.7
Not disclosed	0.2	Other	4.7
		Entrepreneur	4.0
		Not disclosed	0.1
Age	**%**	**Education**	**%**
≤ 25	32.2	Doctoral degree	1.5
26–35	41.4	Master's degree	31.2
36–45	17.0	Bachelor's degree	49.6
≥ 46	7.2	High school or equivalent	16.5
Not disclosed	2.2	Below high school	0.7
		Not disclosed	0.5

The weighted average of the respondents' ages was 31 years. The sample comprised 38 nationalities; the most common were Portuguese (15.2%), Finnish (14.0%), British (9.2%), Polish (8.0%), and Italian (7.5%). The most frequently mentioned job titles were software developer, software engineer, full-stack developer, web developer, front-end developer, and programmer.

3.2 Data Analysis

First, we established exclusion criteria for the responses. Following the CIT criteria [69] and suggestions made for research on flow [34][73], we excluded responses that did not describe *an outstandingly strong flow experience*. This included responses to

[1] http://r.jyu.fi/writing-task-about-flow.

[2] Software-intensive industries in Prolific include "information services and data processing," "software," and "video games."

the flow statements with an average of < 3.0 (Q9) (i.e., the characteristics of flow were not clearly present [73] in the description of the experience), experiences that did not have a particularly positive or effective outcome (Q7) [69], and respondents who could not declare how long ago the experience occurred (Q6) (i.e., "I don't know" responses). Participants who failed at least one attention check question or misunderstood the assignment were also excluded. Misunderstanding the assignment included describing a stressful experience or workflow, for example. While being in an intense flow is not always only a positive experience (flow can easily result in strain), the taxing experiences we excluded did not include any element of flow but rather described a *particularly demanding experience*. Descriptions about workflow depicted what happened in terms of process (e.g., "a continuous flow of tickets to the team"), but they did not describe a flow experience. We also excluded responses from full-time students whose experiences were unrelated to work. In total, we eliminated 36 questionnaires, which resulted in a final sample of 401.

The selected responses were analyzed following the content analysis approach and guidelines by Berg [74]. Content analysis is commonly used in CIT studies because it is useful for identifying, classifying, and categorizing the relevant factors that underlie the phenomenon of interest [23]. We started with open coding, and we labeled the emergent themes in a preliminary manner [75][76] to understand the characteristics of developers' flow experiences (Q2) and the facilitating factors of those experiences (Q3). Open coding was done primarily by the first author, and the preliminary codes were discussed among all authors. The responses that mentioned similar themes were coded with a descriptive label. Both new themes specific to flow in SE and categories consistent with flow research emerged from the data and the preliminary codes. Context-specific findings revealed in particular the importance of positive DX. The codes consistent with flow research included various dimensions of flow, such as optimal challenge (flow antecedent) and deep absorption in the activity (flow characteristic) (e.g., [1][28][29]). Most responses comprised many themes, and each response was coded with all the themes that were mentioned in it.

After coding the data and checking for frequency and patterns [23][74], the preliminary codes and their contents were discussed among all the authors and refined if needed. In establishing the refined codes (i.e., final categories), we maintained the data-driven nature of the findings and referred to flow research when applicable. Flow facilitators were grouped into two main categories—*flow antecedents* and *context-specific and situational facilitators*—while flow characteristics were grouped into *flow experience dimensions* and *flow outcomes* (although we did not specifically ask about outcomes, many participants described them; one example was being highly productive). In naming the categories that clearly referred to the antecedents, experiential dimensions, or outcomes of flow, we adopted the overarching constructs identified by Norsworthy et al. [28] in their scoping review of flow studies. This categorization helped us to create mutually exclusive categories and avoid possible overlaps between the constructs (e.g., intense concentration and merging of action and awareness), as opposed to using the nine-dimensional flow model. Table 2 summarizes the coding process of one respondent's response.

Table 2: Coding Process

Transcript Unit (Q2):		
While developing software for some motors at work, I ended up being very involved, and I really enjoyed coding in what seemed like a similar language to Assembly. During that phase, I was extremely excited to go to work.		
Flow Experience Characteristics:		
Preliminary Code	**Refined Code**	**Main Category**
Concentration	Absorption	Flow experience
Excitement	Intrinsic reward	Flow experience
Transcript Unit (Q3):		
The software and programming language combined with a challenge that I knew I was capable of solving.		
Flow-facilitating Factors:		
Preliminary Code	**Refined Code**	**Main Category**
Challenge–skill balance	Optimal challenge	Flow antecedent
Good DX	Positive DX	Context-specific facilitator

In addition to coding the descriptions with the respondents' *thoughts* about the experience, we also looked at the *activity* being performed when the flow occurred [77] (i.e., the stage[s] of the development process related to the experience) (Q4) and at the technologies and/or software involved in the experience (Q1).

4 RESULTS

This study identified the characteristics of developers' flow and the flow-facilitating factors in their work. First, we briefly examined the circumstances of the experiences, including the development process stage (i.e., the activity) and the technologies involved. As expected, development/programming was mentioned in the majority (83.3%) of the responses of the final sample (N = 401), followed by the rewarding feelings of implementation, testing, debugging, and conducting maintenance activities, such as code refactoring and updating legacy code (see Table 3).

Table 3: Flow and the Stages of the Development Process

Activity	**n**	**%**
Development/programming	335	83.5
Implementation	138	34.4
Testing	116	28.9
Planning	102	25.4
Maintenance	52	13.0
Other	7	1.7

Similarly, the technologies mentioned most often included tools used especially during the development stage, such as programming languages (mentioned by 41.9% of the respondents), IDEs (20.9%), frameworks (18.5%), platforms (12.5%), and DevOps tools (9.2%). The tools quoted most often included Visual Studio Code, Python, Java, JavaScript, and React. Hardware, mostly computers, was mentioned by 5.0% of the participants. In contrast, 1.5% stated that their flow was not related to the use of technology but to being able to use their creative and problem-solving skills, for example.

Over one-third (37.7%) of the respondents declared that their flow lasted for 1–12 hours, and 7.0% reported 13–24 hours, which means that the experience was over after the workday or within 24 hours. Surprisingly, many participants mentioned even longer flows of 1–7 days (19.5%), 1–4 weeks (14.2%), and even more than one month (15.7%), when they worked on projects that constantly provided opportunities to experience flow. However, the respondents also mentioned shorter micro flows that lasted less than an hour (4.2%). Regarding frequency, the majority reported experiencing flows *sometimes* (43.0%) or *often* (30.0%). Experiencing flow *very often* (12.5%), *rarely* (12.8%), or *very rarely* (1.8%) was less common.

4.1 Flow-facilitating Factors in Developers' Work

Flow-facilitating factors were grouped into two main categories, *flow antecedents* and *context-specific and situational facilitators*. Flow antecedents included ***optimal challenge*** (51.1%) and ***high motivation*** (26.9%). Context-specific and situational facilitators included ***positive DX*** (31.7%), ***no distractions or interruptions*** (13.2%), ***social support*** (12.7%), ***timetables*** (8.7%), ***other resources*** (3.7%), and ***positive mood and energy level*** (2.7%). The most significant flow-facilitating factors in developers' work were optimal challenge, high motivation, positive DX, and no distractions or interruptions (see Table 4).

Table 4: Flow-facilitating Factors in Developers' Work

Flow Antecedents	**n**	**%**
Optimal challenge	205	51.1
- *Challenge–skill balance*		
- *Clear goals*		
- *Immediate feedback*		
High motivation	108	26.9
- *Intrinsic motivation*		
- *Achievement motive*		
Context-specific and Situational Facilitators	**n**	**%**
Positive DX	127	31.7
No distractions or interruptions	53	13.2
Social support	51	12.7
Timetables	35	8.7
- *No time pressure*		
- *Sense of urgency*		
Other resources	15	3.7
Positive mood and energy level	11	2.7

The most prominent flow facilitator was ***optimal challenge***, which included, most importantly, appropriate *challenge–skill balance*. This was characterized by working on a task that offered the right amount of challenge (e.g., a good learning curve) and being able to reach flow because of one's skills, experience or expertise.

> The right proportion of challenge, skill, and passion (or excitement) about the topic. I may have challenges that are adequate to my skills, but they mean almost nothing without passion. For me, this is the key that enables a true feeling of flow. (Unity developer)

For some, optimal challenge meant very challenging tasks, problem-solving, and even being out of their comfort zone. Further, learning new skills and applying new knowledge (e.g., learning to use a new technology or acquiring skills needed for a task) was viewed as intriguing and flow-inducing.

> I was developing an AWS-based microservice architecture. It was a new topic for me and required a lot of learning. Still, I experienced strong moments of flow because I felt I had enough time and freedom to implement and explore the technology and its potential. (Senior software engineer)

Optimal challenges also included having *clear goals* and receiving *immediate feedback* about the progress being made. Although clear goals are important for any flow activity, for developers, this could mean, among other things, working on a project with well-defined requirements. Immediate feedback entailed getting into an efficient loop of positive feedback either with people or the technology and seeing the expected results emerge (i.e., receiving timely feedback).

> We got into a positive feedback loop with the tweaks we were making, which motivated everyone to stay focused until the end of the workday. People were able to clearly see the impacts of the decisions we made with each deployment.

High motivation was also an important flow facilitator for developers. This state was described as feeling motivated, dedicated, or passionate about a topic or having a strong curiosity about a topic. *Intrinsic motivation* could arise from doing something one enjoyed or felt was meaningful, working on an intriguing task, or being good at something right from the start. Some participants mentioned that they felt intrinsically motivated because they were contributing to a cause (e.g., national security or public health) or simply because they knew they were creating something useful for end-users.

> I was excited and interested in the subject. I worked furiously because I thought it was the best thing in the world at the time. (Software developer)

Achievement motive arose from wanting to reach high-standard results, making an impression, or showing off one's skills.

> It was my first task for that job, and I wanted to impress my boss. Also, I really enjoyed working on the assignment. (Software developer)

Although intrinsic motivation suggests that one experiences flow doing an activity they find intrinsically rewarding, intrinsic reward will be discussed in 24.2 as a characteristic of the flow experience (i.e., being *in flow*), whereas motivation is the reason to engage in the activity.

The most important context-specific/situational facilitator in developers' work was ***positive DX***. This included descriptions of development tools (e.g., IDEs and frameworks) that were easy to use/a joy to use, intuitive, helpful, and powerful, as well as tools that made work easier and/or faster (i.e., more efficient) by offering flexible and customizable features.

> I knew what I was doing, and everything worked out as I had planned. Unity and Visual Studio worked together as expected, and I felt I really knew how to use these technologies. (Software specialist)

> It [VS Code] works perfectly every time. It allows me to concentrate on my work and never lets me down. (Front-end developer)

> It [AWS] is simple, fast, and predictable. Everything worked like a charm. (Developer)

In some cases, positive DX was associated with *immediate feedback*, ***effortless control***, and ***high performance*** (e.g., increased productivity) (more about effortless control and high performance in 4.2).

> Good and correctly configured tools. Short feedback loop (i.e., the effects of the changes made could be quickly verified; a comprehensive test set to ensure that the changes were very unlikely to break any existing functionality). (Senior software developer)

> The features of the IDE were just what I needed to be able to focus on getting the code out quickly. It didn't get in the way and only enhanced my productivity. The keyboard shortcuts and all the built-in tools made it an easy one-stop shop for getting all the information I needed to be productive. (Software developer)

Another important situational facilitator was ***no distractions or interruptions***, which involved having distraction-free moments for focused work in a calm environment (e.g., a break from meetings, instant messages, or questions from colleagues, or not having too frequent context switches). One way to achieve interruption-free work and mitigate any possible distraction was to isolate oneself by using noise-canceling headphones and/or listening to music.

> I think that the over-ear noise-canceling headphones and the good playlist enabled the flow experience.

Social support was also a facilitator of many respondents' flow experiences. This entailed working in a supportive team where members were receptive to ideas, helped each other, worked together toward a shared goal, communicated efficiently, and had mutual respect for each other. A few participants mentioned that their flow occurred when they were pair programming.

> My colleague and I shared the work fairly equally and kept communicating regularly and constantly. We shared ideas and praised each other. The work was fun as it was difficult but rewarding. [Software engineer]

Social support also involved receiving encouragement from colleagues and online communities.

> A sense of pride in having colleagues over your back cheering you on (Senior software specialist)

Interestingly, ***timetables*** (8.7%) were shown to facilitate flow in two ways. In some circumstances, there was *no time pressure*, and it was possible to work with reasonable timetables or at one's own pace with the opportunity to get absorbed into the task ("Do it till it works").

> I was working with a colleague, and we were given enough time to be calm and focused and research the work properly. (Developer)

In other cases, there was a tight schedule or pressure due to being behind schedule, which created a *sense of urgency* that enabled the flow experience.

> My previous experience, the relative mix of easiness and things yet to learn… And I also was being pressured into doing this task quite fast. (Software developer)

Interestingly, working without time pressure and a sense of urgency enabled absorption in their own way; no time pressure provided the opportunity to explore and discover, and urgency forced the respondents to focus fully on the pressing issue.

Other resources included having the right kinds of tools (e.g., hardware, software, stable connections, and server resources), sufficient human resources, and budget. ***Positive mood and energy level*** were perceived as affecting the ability to get into flow. Being in the right kind of mood or having a "good mindset" (i.e., not being anxious or stressed), as well as being well rested, were experienced as helpful in reaching flow.

4.2 Characteristics of Developers' Flow

Based on the responses, we identified the topics that characterized the experiences the most. These were grouped into *flow experience* and *flow outcomes* (see Table 5). The dimensions of the flow experience were classified under ***absorption*** (38.4%), ***effortless control*** (37.9%), and ***intrinsic reward*** (33.7%). Many respondents also described ***high performance*** (25.9%) as an outcome of their flow.

Table 5: Characteristics of Developers' Flow

Flow Experience	**n**	**%**
Absorption - *Intense concentration* - *Time distortion* - *Merging of action and awareness*	154	38.4
Effortless control - *Effortless involvement* - *Sense of control*	152	37.9
Intrinsic reward - *Enjoyment* - *Satisfaction*	135	33.7
Flow Outcomes	**n**	**%**
High performance - *Increased productivity* - *Increased creativity*	104	25.9

Absorption was characterized by intense concentration, time distortion (e.g., losing track of time), and merging of action and awareness. Intense concentration was described as a full focus on the task, immersion or engagement in an activity, and complete absorption leading one not to hear or see anything around them. Many stated that this intensely focused state resulted in not wanting to take breaks and, in some cases, in being able to continue flowing even after a break. Intense concentration was often described as occurring when coding or programming.

> When I program automatic tests, I am completely absorbed in my work and very focused. I do not take breaks from work when I am thinking about a specific programming problem. (Junior test engineer)

Some respondents said that focus was achieved by being in an environment where one could concentrate in peace and quiet (e.g., isolating oneself or listening to music to block out distractions). However, for others, this intense involvement was not seen as entirely positive, as it is possible to get "too absorbed" in programming. Absorption also included descriptions of *time distortion*, meaning an *altered sense of time*. This was explained as losing track of time, time flying, days passing fast, and time standing still.

> I was able to move through the code with speed and finesse, losing track of time. (Senior software engineer)

Furthermore, some participants felt a *merging of action and awareness* in the state of absorption, which was described as forgetting everything around oneself, the environment becoming blurred or disappearing, "everything blending in," being "in the zone," or getting lost in coding. An interesting point put forward by one programmer was having no fear of making mistakes along the way as long as there was an interesting task that provided the right amount of challenge:

> The focus is just at the right level. The time and environment are nicely blurred, and things happen—not necessarily only the right things, but when you have a nice challenge, it doesn't matter. (Senior game programmer)

Developers' flow was also characterized by ***effortless control***, which included both effortless involvement in the activity and having a *sense of control* over one's actions. In some cases, this resulted in having increased willpower to accomplish the goal or feeling proud of being able to solve problems independently:

> I felt really good when I didn't have to ask for help and could solve the problems I encountered myself. I like my team members, and there is no shame in asking for help, but I just felt particularly good and was in a flow. (Back-end developer)

Effortless involvement was described as everything going smoothly, things working out as expected, tasks solving themselves, or "work flowing like water running through a tap."

> When you're in the flow of development, everything works as expected, and all is fine. (Software developer)

Interestingly, effortless control was often associated with positive DX. This heightened sense of control over one's actions and a feeling of effortless involvement in the activity resulted from tools that provide positive DX, which enabled smooth, effortless, and efficient working.

The respondents also reported another central aspect of flow, the autotelic experience (i.e., experiencing the activity as intrinsically rewarding). ***Intrinsic reward*** entailed *enjoyment* and *satisfaction*. *Enjoyment* was described as having a euphoric experience, a great rush, joy, being excited, having fun, and feeling energetic, among other things. *Satisfaction* entailed getting a rewarding feeling from certain tasks that induced flow. Many participants disclosed that they loved what they did for a living and could not get enough of coding.

> I lose track of time, hunger, and thirst, and my focus is completely on the task at hand. I do not think about anything else. This is quite a euphoric experience. It's one of the best feelings anyone can feel, in my opinion. I can spend hours at a time just lost in the moment. Nothing matters except my creation. I think this feeling is one of the best things about programming. (Unity developer)

> Things went exceptionally well; I felt motivated and ready for any challenge. I enjoyed my work more than usual, and I was eager to see the finished results. (Full-stack software engineer)

Satisfaction could also emerge from doing something because it was helpful in one way or another, such as re-platforming a legacy system, developing something useful for end-users, and working

on a project that was important for compliance, national security, or public health.

Finally, the developers' flow was characterized by ***high performance***. This included both *increased productivity* and *increased creativity*. *Increased productivity* was described as being productive or efficient (e.g., having a productive day or completing weeks' worth of work in one day), making progress (either massive strides or steady, step-by-step progress), finishing tasks, or, as some participants vividly put it, "being on fire."

> I worked for about two hours and got a lot of stuff done. I was completely absorbed in the code and the stuff I was creating. (Front-end developer)

> Clients requested features that I was able to pump out, test, document, and deliver quickly. When there is a quick turnaround, and I see directly how pleased the clients are with the new features, everything just seems "right." (Software engineer)

In many cases, productivity was described as not getting stuck or not encountering major issues while developing (i.e., no problems occurred that could have disrupted the flow and the focus on the task). Some respondents even mentioned the outcomes of these fecund sessions: an enhanced product, more powerful software, or a functioning solution with "all the goodies."

In addition to increased productivity, high performance included *increased creativity*. Creativity is an important aspect of flow activities. In the sample, it was characterized as using creative skills or creative thinking, turning thoughts into reality, "playing" with new technologies, exploration, and opening one's mind to new possibilities:

> Finally working out a piece of functionality on an app by understanding how the programming language worked in terms of saving data both in the device and on the cloud. Making progress after this realization was very rewarding, and I saw things in a new light. My mind was opened to new possibilities, and the software became more powerful as a result.

> Learning new stuff, adding more functionalities to the website, and all the endless possibilities that I started imagining. At last, doing the things I thought only geniuses could do. (Front-end developer)

In conclusion, many perceived to be high performing in flow and described the experience with absorption, effortless control over their actions, and a feeling of intrinsic reward.

5 DISCUSSION

This section discusses the theoretical and practical implications of this study. Our study contributes to research on the human-centric aspects of SE [21] by providing a contextual understanding of the flow experience in the SE context. In particular, it identifies the factors that can help facilitate said experience [28][37] in developers' work. While the characteristics of developers' flow are important, the most valuable input of this study is the identification of the most significant flow-facilitating factors in SE. A summary of its key contributions and the related literature is presented in Table 6. The practical implications include determining how flow can be facilitated at both the individual and organizational levels by fostering practices and using development tools that support flow.

Table 6: Key Findings and Related Work

Key Facilitators of Developers' Flow	**Related Work**
Optimal challenge	The findings highlight how optimal challenge [28][29][35] can emerge from IT use (e.g., interactions providing immediate feedback), improving developers' sense of control.
High motivation	While intrinsic motivation is essential for flow [1][29], achievement motive [36][37] was also important facilitator for many developers' flow experiences.
Positive DX	Negative DX [4][12] has been associated with less flow. This study identified positive DX as one of the most important facilitators of developers' flow, supporting research on the user experience of development tools [7][14][58][65][66].
No distractions or interruptions	A working environment with no distractions or interruptions to disrupt focus was perceived as important for facilitating flow, while excessive interruptions and distractions (both IT-mediated and from the physical environment) [12][13] have been identified as critical flow barriers in SE.
Key Characteristics of Developers' Flow	**Related Work**
Absorption Effortless control Intrinsic reward	In SE, the three-dimensional conceptualization of flow [28] is more descriptive than the nine-dimensional model [29][31]. Whereas some three-dimensional operationalizations include *enjoyment*, *absorption*, and *intrinsic motivation* [8][32], the findings suggest that motivation is an antecedent rather than a characteristic of flow [28][33].
High performance	High performance was considered an outcome of the flow experience [28][40]. It was also associated with IT use and, specifically, positive DX.

5.1 Theoretical Contributions

The results of this study highlight the importance of having tasks that present the right amount of challenge. The optimal balance between challenges and skills seems to be among the most important facilitating factors for experiencing flow in the SE context; in contrast, the lack of such balance constitutes a significant barrier [12]. The activity (in this case, work) needs to provide increasing challenges [34] in order to maintain the ideal state where flow occurs and avoid boredom due to excessively easy tasks as well as anxiety due to too difficult ones [29]. The optimal challenge, which includes the perceived challenge–skill balance, clear goals, and immediate feedback, is important for flow activities not only because accomplishing small-scale goals can increase enjoyment [39] but also because these goals serve the important purpose of working toward the project's overall aim. In particular, immediate feedback was intertwined with IT use and receiving unambiguous and timely feedback from the systems, which helps adjust subsequent actions [29]. This improves the sense of effortless involvement and the sense of control (i.e., effortless control) as well as developer experience. The developers associated effortless control also with using development tools that allowed them to feel their work progressing effortlessly and with control. The mentioned flow experiences often involved working with tools that provided positive DX; that is, the tools that make work easier, faster, and effortless. In other words, many of the respondents who had a positive DX with development tools when they experienced flow associated IT use with positive outcomes [46]. Positive DX (e.g., IDEs providing functionalities that made work smoother and more enjoyable) contributed to optimal challenge by providing timely feedback about the progress being made. This allowed the developers to feel effortless control over their actions. This positive DX may also increase enjoyment during the experience [7], which is an important aspect of flow. Previous studies have made hands-on recommendations for designs that support DX and flow (e.g., [14][58][66]). This study adopted a more holistic perspective on flow in developers' work. Positive DX is an important facilitating factor and part of developers' flow experiences, but there are also negative consequences of IT use, such as inopportune interruptions [13], which can result in the inability to reach flow [12]. The participants reported that not having distractions while working and proactively blocking out disturbances by using noise-canceling headphones and/or listening to music helped them reach and maintain flow.

As expected, being highly motivated was an important flow-facilitating factor in developers' work. The autotelic experience comprises a motivational component (being motivated) and an experiential component (enjoying the activity) [78]. The respondents vividly described their intrinsic motivation for what they do. This could be either because they worked with something they enjoyed or because they wanted to achieve a personal goal (e.g., produce high-quality work). Although flow activities are autotelic (i.e., rewarding in and of themselves) [1], the incentive to achieve something as a byproduct of flow has been investigated by flow researchers (e.g., [37]). The findings demonstrated that the achievement motive [35][36] was a flow-facilitating factor. They also documented motivation linked to situations involving difficult challenges (rather than easy ones) and requiring high levels of skills. Hence, it is possible that software development attracts autotelic personalities who feel confident in their skills and control over their actions [1][29][35]. For these individuals, being highly driven and working on something intrinsically motivating results in experiencing the activity as enjoyable. Further, intrinsic reward (i.e., enjoying what one does and having fun) proved to be an important aspect of developers' work. The distinction between intrinsic motivation and intrinsic reward can be hard to make, which explains the different conceptualizations of flow. Some models suggest that intrinsic motivation is subsumed in the experiential dimension of the *autotelic experience*, while others consider motivation to be an antecedent of flow. The findings suggested that motivation describes an incentive to act on perceived opportunities [29] rather than an experiential aspect of being in flow [28][33].

The nine-dimensional model of flow [29] views intense concentration, altered sense of time, the merging of action and awareness, and loss of reflective self-consciousness as separate dimensions. However, it has been suggested that "absorption" could be a better term [28] to cover the experience of intense and focused concentration [29] in which the environment "disappears" and one loses track of time and the feeling of being a social actor (not to mention that loss of reflective self-consciousness can be, by definition, hard to acknowledge and self-report). Indeed, we found that these experiences of intense flow were best described as *absorption*, which supports the view that this is the defining characteristic of the phenomenon [28]. Some participants reported that their flow lasted for less than an hour, which suggests that these could have been instances of micro flow [79]. However, because the purpose of a CIT study is to uncover particularly positive or negative experiences, the percentage of these short flow sessions was quite low. Surprisingly, we also found that many respondents experienced deep absorption and flow for several consecutive days or even weeks. This kind of long flow could indicate high levels of intrinsic motivation or outside pressure (e.g., tight production schedules).

High performance (i.e., being in a *high-functioning* state [28]) can be considered an outcome of flow, rather than a characteristic of being *in* flow. However, in this study, this characteristic is discussed among those of developers' flow experiences because the being highly productive or highly creative was reported by many respondents. High performance was attributed to the internal experience of facing the optimal level of challenge and intensely focusing on the task as well as to the aforementioned positive DX, which made it possible to be more productive. In addition, high performance was associated with increased creativity based on curiosity and exploration. Engaging in creative thinking [37], curiosity, and exploration are features of flow activities, especially in IT use [18][19][22], which is why considering UX design is essential in facilitating immersion and flow in the HCI context [48][49]. Overall, the characteristics of developers' flow were similar to those of flow experiences in other contexts, whereas the flow facilitators included more work- and

SE-specific factors, especially positive DX and the chance not to be distracted or interrupted.

5.2 Practical Implications

This study has two practical implications. First, the findings highlight possible ways to experience more flow at work. They demonstrate that flow occurs particularly when engaging in intrinsically motivating tasks that offer just the right amount of challenge. When this is the case, one can be confident about one's skills and feel that they are being somewhat tested. Therefore, working in a role or applying for a role or tasks that provide this kind of challenge is important for experiencing flow, as is a role where one finds the tasks motivating and exciting. The motivational aspect is important not only for facilitating more flow but also for keeping employees more engaged and improving their performance [8][9].

One of the most interesting aspects that emerged from the data was the positive effect of IT use on developers' work. In addition to the rather obvious observation that access to the required technological resources (i.e., hardware and software) is necessary, the findings emphasized the importance of positive DX. This was often associated with effortless control and working on tasks with ease and control. Providing developers with development tools that enable excellent DX and make work easier, faster, more enjoyable, and thus flow-inducing (or providing them with the opportunity to choose their preferred tools) is also essential in supporting their work performance.

The second managerial implication is facilitating more opportunities for focused problem-solving without too many distractions and interruptions (e.g., instant messages, meetings, and context switching). An essential aspect of any flow activity, especially in SE, is the ability to focus deeply on the task at hand. Although there are personal preferences regarding how this intense concentration can best be reached, it is clear that reducing unnecessary disturbances and context switches as well as providing tools to support concentrated work (e.g., noise-canceling headphones or an option to work remotely) can help developers have more control over their workdays and reach flow. However, planning, meetings, and other activities not directly related to development are often important for the progress and success of projects. Therefore, balancing tasks that serve different purposes and allocating time to focused work are key managerial considerations for facilitating flow.

5.3 Limitations and Future Directions

Flow is a very subjective experience. Therefore, directly measuring it [80] or defining a clear line between being in and not being in flow [33] is difficult. We addressed this problem by allowing the participants to describe experiences they felt were flow and by asking refining questions about these experiences to help assess whether they could be considered flow [34][73]. However, many flow questionnaires are very long because they attempt to comprehensively cover this experience; they have thus been critiqued for being tedious and taking considerable effort to fill out [66]. To avoid a tedious questionnaire and to mitigate respondent bias, we kept the questions concise and asked the participants to describe their experiences in their own words. The CIT method also has its limitations, including the recall bias related to retrospective reporting, the possible misinterpretation of these reports, and the ambiguity associated with coding rules and category labels [23]. Moreover, the definition of flow [1] provided at the beginning of the questionnaire could have inspired some respondents. Possible problems related to online panels include data quality issues (e.g., respondents not giving their full attention to the task when filling out the questionnaire) and sample-related issues, such as "super-users," who can distort results, and WEIRD sample barriers (though sample representativeness can be more easily reached using online panels) [81]. To address the risk of inattentive respondents, we added two attention check questions.

This study uncovers the context-specific characteristics of flow in SE and opens an avenue for further investigations. As more research on professional developers' flow is needed [2][14], an interesting aspect to explore is DX and how it can support flow (e.g., by designing IDEs that provide better feedback) [7]. Other topics to examine include the social aspect of developers' work and collective flow [40] in development teams, as well as the particularly positive/effective outcomes of flow [69] (i.e., other key outcomes, including more engagement in an activity or positive changes in one's well-being [28], in addition to being highly productive). The flow-facilitating factors identified in this study and flow barriers [12][13] could be analyzed more closely to compare their similarities and differences. Scholars could also study if there are differences in the flow-facilitating factors as well as in the ability to reach flow between junior developers and more experienced developers.

6 DATA AVAILABILITY

The data included many instances in which the respondents or their organizations could be identified. To protect their privacy, the participants were assured that only the research group would have access to their responses. Link to the questionnaire is provided in 3.1 Data collection.

ACKNOWLEDGMENTS

This research received funding from the Finnish Foundation for Economic Education and the Academy of Finland (341359).

REFERENCES

[1] Mihaly Csikszentmihalyi. 1990. *Flow: The Psychology of Optimal Experience.* HarperCollins, New York, NY.

[2] Jean A. Pratt, Liqiang Chen, and Carey Cole. 2016 The influence of goal clarity, curiosity, and enjoyment on intention to code. *Behav. Inf. Technol.* 35, 12 (Apr 2016), 1091–1101. https://doi.org/10.1080/0144929x.2016.1171399

[3] Jeff Gray and Bernhard Rumpe. 2017. The importance of flow in software development. *Softw Syst Model* 16, 4 (Sep 2017), 927-928. https://doi.org /10.1007/s10270-017-0621-x

[4] Reyhaneh Kalantari and Timothy C. Lethbridge. 2022. Preliminary results of measuring flow experience in a software modeling tool: UmpleOnline. In *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings (MODELS '22).* ACM, New York, NY, USA, 923–928. https://doi.org /10.1145/3550356.3559099

[5] Henry Edison, Nauman bin Ali, and Richard Torkar. 2013. Towards innovation measurement in the software industry. *J Syst Softw* 86, 5 (May 2013), 1390-1407. https://doi.org/10.1016/j.jss.2013.01.013

[6] Srinivas Kudaravalli, Samer Faraj, and Steven L. Johnson. 2017. A configural approach to coordinating expertise in software development teams. *MISQ* 41, 1 (Mar 2017), 43–64. https://doi.org/10.25300/misq/2017/41.1.03

[7] Kati Kuusinen, Helen Petrie, Fabian Fagerholm, and Tommi Mikkonen. 2016. Flow, intrinsic motivation, and developer experience in software engineering. In Helen Sharp and Tracy Hall (Ed.s.) *Agile Processes, in Software Engineering, and Extreme Programming. XP 2016. Lecture Notes in Business Information Processing*, vol 251, 104–117. Springer, Cham. https://doi.org/10.1007/978-3-319-33515-5_9

[8] Arnold B. Bakker. 2008. The work-related flow inventory: Construction and initial validation of the WOLF. *J Vocat Behav* 72, 3 (Jun 2008), 400-414. https://doi.org/10.1016/j.jvb.2007.11.007

[9] Sarah Beecham, Nathan Baddoo, Tracy Hall, Hugh Robinson, and Helen Sharp. 2008. Motivation in Software Engineering: A systematic literature review. *Inf Softw Technol.* 50, 9-10 (Aug 2008), 860-878. https://doi.org/10.1016/j.infsof.2007.09.004

[10] Viswanath Venkatesh, James Y. L. Thong, Frank K. Y. Chan, Hartmut Hoehle, and Kai Spohrer. 2020. How agile software development methods reduce work exhaustion: Insights on role perceptions and organizational skills. *Information Systems Journal* 30, 4 (Mar. 2020), 733–761. https://doi.org/10.1111/isj.12282

[11] Sebastian Müller and Thomas Fritz. 2015. Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress. In *Proceedings of the 37th IEEE International Conference on Software Engineering*, 688-699. https://doi.org/10.1109/icse.2015.334

[12] Saima Ritonummi, Valtteri Siitonen, Markus Salo, and Henri Pirkkalainen. 2022. Flow barriers: What prevents software developers from experiencing flow in their work. In *Proceedings of the 8th International Conference on Socio-Technical Perspective in IS development (STPIS'22)*, 247-264. https://ceur-ws.org/Vol-3239/paper21.pdf

[13] Vanessa Vella and Chris Porter. 2022. Wait a second! Assessing the impact of different desktop push notification types on software developers. In *Proceedings of the 33rd European Conference on Cognitive Ergonomics (ECCE'22)*. ACM, New York, NY, USA, Article 8. https://doi.org/10.1145/3552327.3552328

[14] Marc-Andre Kaufhold, Christian Reuter, and Thomas Ludwig. 2019. Flow Experience in Software Engineering: Development and Evaluation of Design Options for Eclipse. In *Proceedings of the 27th European Conference on Information Systems (ECIS)*. https://aisel.aisnet.org/ecis2019_rip

[15] Thomas Fritz and Sebastian Müller. 2016. Leveraging biometric data to boost software developer productivity. In *Proceedings of the 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, 66-77. https://doi.org/10.1109/saner.2016.107

[16] Michaela Greiler, Margaret-Anne Storey, and Abi Noda. 2022. An Actionable Framework for Understanding and Improving Developer Experience. *IEEE T SOFTWARE ENG* 49, 4 (Apr. 2023), 1411–1425. https://doi.org/10.1109/TSE.2022.3175660

[17] Benjamin U. Cowley, Arto Hellas, Petri Ihantola, Juho Leinonen, and Michiel Spape. 2022. Seeking Flow from Fine-Grained Log Data. In *IEEE/ACM 44th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, 247–253, https://doi.org/10.1145/3510456.3514138

[18] Linda K. Trevino and Jane Webster. 1992. Flow in computer-mediated communication: Electronic mail and voice mail evaluation and impacts. *Comm. Res.* 19, 5 (Oct 1992), 539-573. https://doi.org/10.1177/009365092019005001

[19] Jane Webster, Linda K. Trevino, and Lisa Ryan. 1993. The dimensionality and correlates of flow in human-computer interactions. *Comput. Hum. Behav.* 9, 4 (Winter 1993), 411-426. https://doi.org/10.1016/0747-5632(93)90032-N

[20] Jawaid A. Ghani and Satish P. Deshpande. 1994. Task characteristics and the experience of optimal flow in human—computer interaction. *J Psychol* 128, 4 (1994), 381-391. https://doi.org/10.1080/00223980.1994.9712742

[21] Gail C. Murphy. 2010. Human-centric software engineering. In *Proceedings of the FSE/SDP workshop on Future of software engineering research (FoSER '10)*. ACM, New York, NY, USA, 251–254. https://doi.org/10.1145/1882362.1882414

[22] Christopher B. Califf, T. S. Stumpf, and Joshua J. Frye. 2020. Revisiting Technology and Flow: A Call for an Alternative Perspective and Directions for the Future. In *Proceedings of the 53rd Hawaii International Conference on System Sciences 2020 (HICSS-53)*. http://hdl.handle.net/10125/64495

[23] Dwayne D. Gremler. 2004. The critical incident technique in service research. *J Serv Res* 7, 1 (Aug 2004), 65-89. https://doi.org/10.1177/1094670504266138

[24] Stefan Palan and Christian Schitter. 2018. Prolific.ac—a subject pool for online experiments. *J Behav Exp Finance* 17 (2018), 22–27. https://doi.org/10.1016/j.jbef.2017.12.004

[25] Eyal Peer, Laura Brandimarte, Sonam Samat, and Alessandro Acquisti. 2017. Beyond the Turk: Alternative platforms for crowdsourcing behavioral research. *J Exp Soc Psychol* 70 (2017), 153–163. https://doi.org/10.1016/j.jesp.2017.01.006

[26] Daniel Graziotin, Xiaofeng Wang, and Pekka Abrahamsson. 2015. Do feelings matter? On the correlation of affects and the self-assessed productivity in software engineering. *J Softw (Malden)* 27, 7 (Jul 2015), 467-487. https://doi.org/10.1002/smr.1673

[27] Luz M. Restrepo-Tamayo and Gloria P. Gasca-Hurtado. 2022. Human Aspects in Software Development: A Systematic Mapping Study. In *Proceedings of the International Conference on Collaboration Technologies and Social Computing (Collab Tech 2022)*, 1-22. https://doi.org/10.1007/978-3-031-20218-6_1

[28] Cameron Norsworthy, Ben Jackson, and James A. Dimmock. 2021. Advancing our understanding of psychological flow: A scoping review of conceptualizations, measurements, and applications. *Psychol Bull* 147, 8 (Aug 2021), 806–827. https://doi.org/10.1037/bul0000337

[29] Jeanne Nakamura and Mihaly Csikszentmihalyi. 2002. The concept of flow. In C. R. Snyder and Shane J. Lopez (Ed.s.) *Handbook of positive psychology*. Oxford University Press, New York, NY.

[30] Mihaly Csikszentmihalyi. 1975. *Beyond Boredom and Anxiety*, Jossey-bass Publishers, San Francisco, CA.

[31] Mihaly Csikszentmihalyi, Sami Abuhamdeh, and Jeanne Nakamura. 2014. Flow. In Mihaly Csikszentmihalyi (Ed.) *Flow and the Foundations of Positive Psychology*. Springer, Dordrecht. https://doi.org/10.1007/978-94-017-9088-8_15

[32] Lucia Ceja and Jose A. Navarro. 2012. 'Suddenly I get into the zone': Examining discontinuities and nonlinear changes in flow experiences at work. *Hum Relat* 65, 9 (Sep 2012), 1101-1127. https://doi.org/10.1177/0018726712447116

[33] Alma M. Rodríguez-Sánchez, Wilmar Schaufeli, Marisa Salanova, Eva Cifre, and Mieke Sonnenschein. 2011. Enjoyment and absorption: An electronic diary study on daily flow patterns. *Work Stress* 25, 1 (Apr 2011), 75-92. https://doi.org/10.1080/02678373.2011.565619

[34] Sami Abuhamdeh. 2020. Investigating the "flow" experience: Key conceptual and operational issues. *Front. Psychol.* 11, 158 (Feb 2020). https://doi.org/10.3389/fpsyg.2020.00158

[35] Stefan Engeser and Falko Rheinberg. 2008. Flow, performance and moderators of challenge-skill balance. *Motiv Emot* 32, 3 (Sep 2008), 158-172. https://doi.org/10.1007/s11031-008-9102-4

[36] Julia Schüler. 2010. Achievement incentives determine the effects of achievement-motive incongruence on flow experience. *Motiv Emot* 34, 4 (Dec 2010), 2-14. https://doi.org/10.1007/s11031-009-9150-4

[37] Mihaly Csikszentmihalyi. 2014. Motivation and Creativity: Towards a Synthesis of Structural and Energistic Approaches to Cognition. In Mihaly Csikszentmihalyi (Ed.) *Flow and the Foundations of Positive Psychology*. Springer, Dordrecht. https://doi.org/10.1016/0732-118X(88)90001-3

[38] Nicola Baumann. 2012. Autotelic Personality. In Stefan Engeser (Ed.) *Advances in Flow Research*. Springer, New York, NY. https://doi.org/10.1007/978-3-030-53468-4

[39] Mihaly Csikszentmihalyi, Sonal Khosla, and Jeanne Nakamura. 2017. Flow at work. In Lindsay G. Oades, Michael F. Steger, Antonella Delle Fave and Jonathan Passmore (Ed.s.) *The Wiley Blackwell Handbook of the Psychology of Positivity and Strengths-Based Approaches at Work*. John Wiley & Sons, Ltd. https://doi.org/10.1002/9781118977620.ch7

[40] Evangelia Demerouti. 2006. Job characteristics, flow, and performance: the moderating role of conscientiousness. *J Occup Health Psych* 11, 3, 266–280. https://doi.org/10.1037/1076-8998.11.3.266

[41] Marisa Salanova, Alma M. Rodríguez-Sánchez, Wilmar B. Schaufeli, and Eva Cifre. 2014. Flowing together: A longitudinal study of collective efficacy and collective flow among workgroups. *J. Psychol.* 148, 4 (Jan 2014), 435-455. https://doi.org/10.1080/00223980.2013.806290

[42] Charles J. Walker. 2010. Experiencing flow: Is doing it together better than doing it alone? *J Posit Psychol* 5, 1 (Jan 2010), 3-11. https://doi.org/10.1080/17439760903271116

[43] Parvaneh Sharafi, Leif Hedman, and Henry Montgomery. 2006. Using information technology: engagement modes, flow experience, and personality orientations. *Comput. Hum. Behav.* 22, 5 (Sep 2006), 899-916. https://doi.org/10.1016/j.chb.2004.03.022

[44] Jane Webster and Hayes Ho. 1997. Audience engagement in multimedia presentations. *Data Base Adv. Inf. Syst.* 28, 2 (Spring 1997), 63-77. https://doi.org.fi/10.1145/264701.264706

[45] Ritu Agarwal and Elena Karahanna. 2000. Time flies when you're having fun: Cognitive absorption and beliefs about information technology usage. *MISQ* 24 (4), (Dec 2000), 665-694. https://doi.org/10.2307/3250951

[46] Monideepa Tarafdar, Cary L. Cooper, and Jean-François Stich. 2019. The Technostress Trifecta - techno eustress, techno distress and design: Theoretical directions and an agenda for research. *Inf. Syst. J.* 29, 1 (Mar 2019), 6–42. https://doi.org/10.1111/isj.12169

[47] Raphael Rissler, Mario Nadj, and Marc T. P. Adam. 2017. Flow in Information Systems Research: Review, Integrative Theoretical Framework, and Future Directions. In *Proceedings of the 13th International Conference on Wirtschaftsinformatik (WI2017)*, 1051-1065. https://aisel.aisnet.org/wi2017/track10/paper/4/

[48] Marc Hassenzahl and Noam Tractinsky. 2006. User experience-a research agenda. *Behav. Inf. Technol.* 25, 2 (2006), 91-97. https://doi.org/10.1080/01449290500330331

[49] Effie Law. 2011. The measurability and predictability of user experience. In *Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems*, 1–10. https://doi.org/10.1145/1996461.1996485

[50] Jaana-Piia Mäkiniemi, Salla Ahola, and Johanna Joensuu. 2020. A Novel Construct to Measure Employees' Technology- Related Experiences of Well-Being: Empirical Validation of the Techno-Work Engagement Scale (TechnoWES). *Scand J Work and Organ Psychol* 5, 1 (Apr 2020), 1–14. https://doi.org/10.16993/sjwop.79

[51] Adela Chen and Elena Karahanna. 2018. Life interrupted: The effects of technology-mediated work interruptions on work and nonwork outcomes. *MISQ* 42, 4 (Dec 2018), 1023-1042. https://doi.org/10.25300/MISQ/2018/13631

[52] Melissa Mazmanian, Wanda J. Orlikowski, and JoAnne Yates. 2013. The autonomy paradox: The implications of mobile email devices for knowledge professionals. *Organ. Sci.* 24, 5 (Feb 2013), 1337-1357. https://doi.org/10.1287/orsc.1120.0806

[53] Sebastian Köffer, Kevin Ortbach, and Björn Niehaves. 2014. Exploring the relationship between IT consumerization and job performance: a theoretical framework for future research. *Commun. Assoc. Inf. Syst.* 35, 1 (2014). https://doi.org/10.17705/1CAIS.03514

[54] Leona Brust, Nicole J. Hartwich, Christoph Breidbach, and David Antons. 2022. How Deep is your Work? The Day-to-Day Effects of Information and Communication Technology Use on Deep Work of Employees. In *Proceedings of the International Conference on Information Systems (ICIS 2022)*. https://aisel.aisnet.org/icis2022/is_futureofwork/is_futureofwork/13

[55] Bin Wang, Yukun Liu, and Sharon K. Parker. 2020. How does the use of information communication technology affect individuals? A work design perspective. *Acad Manag Ann* 14, 2 (Aug 2020), 695-725. https://doi.org/10.5465/annals.2018.0127

[56] Pedro J. de Moura Jr. and Nayana de Oliveira Rosas. 2021. Perceptions About Flow and Boredom in the Information Technology Profession: Evidence of a Generational Issue. *IJHCITP* 12, 4 (2021), 1-17. https://doi.org/10.4018/IJHCITP.2021100101

[57] Rolf Mahnke and Thomas Hess. 2014. Flow Design Method: Four Steps towards Optimal User Experience. In *Proceedings der Multikonferenz Wirtschaftsinformatik 2014. https://www.researchgate.net/publication/260337581_Flow_Design_Method_Four_Steps_towards_Optimal_User_Experience*

[58] Gail C. Murphy. 2014. Getting to flow in software development. In Proceedings of the 2014 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software (Onward! 2014), 269-281. https://doi.org/10.1145/2661136.2661158

[59] Daniel Graziotin, Xiaofeng Wang, and Pekka Abrahamsson. 2015. How do you feel, developer? An explanatory theory of the impact of affects on programming performance. *PeerJ Comput. Sci.* 1, 18 (Aug 2015). https://peerj.com/articles/cs-18/

[60] Daniel Graziotin, Fabian Fagerholm, Xiaofeng Wang, and Abrahamsson. 2018. What happens when software developers are (un)happy. *J Syst Softw* 140 (Jun 2018), 32–47. https://doi.org/10.1016/j.jss.2018.02.041

[61] Zahra S. H. Abad, Mohammad Noaeen, Didar Zowghi, Behrouz H. Far, and Ken Barker. 2018. Two Sides of the Same Coin: Software developers' perceptions of task switching and task interruption. In *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering (EASE'18)*, 175-180. https://doi.org/10.1145/3210459.3214170

[62] Zahra S. H. Abad, Oliver Karras, Kurt Schneider, Ken Barker, and Mike Bauer. 2018. Task interruption in software development projects. In In *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering (EASE'18)*, 122-132. https://doi.org/10.1145/3210459.3210471

[63] 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)*, 19-29. https://doi.org/10.1145/2635868.2635892

[64] 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. *IEEE Trans. Softw. Eng.* 43, 12 (Dec 2017), 1178–1193. https://doi.org/10.1109/tse.2017.2656886

[65] Kati Kuusinen. 2016. Are Software Developers Just Users of Development Tools? Assessing Developer Experience of a Graphical User Interface Designer. *In Proceedings of the 6th International Conference on Human-Centered Software Engineering (HCSE 2016) and 8th International Conference on Human Error, Safety, and System Development (HESSD 2016)*. https://doi.org/10.1007/978-3-319-44902-9

[66] Zhiwen Zheng, Liang Wang, Yue Cao, Yuqian Zhuang, and Xianping Tao X. 2019. Towards non-invasive recognition of developers' flow states with Computer Interaction Traces. In *Proceedings of the 26th Asia-Pacific Software Engineering Conference (APSEC)*. https://doi.org/10.1109/apsec48747.2019.00048

[67] Susan A. Jackson and Hebert W. Marsh. 1996. Development and Validation of a Scale to Measure Optimal Experience: The Flow State Scale. *J Sport Exerc Psychol* 18, 1 (Jan 1996), 17–35. https://doi.org/10.1123/jsep.18.1.17

[68] John C. Flanagan. 1954. The critical incident technique. *Psychol Bull* 51, 4 (Jan 1954), 327–358. https://doi.org/10.1037/h0061470

[69] Janis Gogan, Mark-David McLaughlin, and Dominic Thomas. 2014. Critical Incident Technique in the Basket. In *Proceedings of the International Conference on Information Systems (ICIS 2014)* https://aisel.aisnet.org/icis2014/proceedings/ResearchMethods/2

[70] Markus Salo and Lauri Frank. 2017. User behaviours after critical mobile application incidents: the relationship with situational context. *Inf. Syst. J.* 27, 1 (Jan 2017), 5–30. https://doi.org/10.1111/isj.12081

[71] Markus Salo, Markus Makkonen, and Riitta Hekkala. 2020. The interplay of IT users' coping strategies: Uncovering momentary emotional load, routes, and sequences. MISQ 44, 3 (Sep 2020), 1143–1175. https://doi.org/10.25300/MISQ/2020/15610

[72] Aletta Nylén and Ville Isomöttönen. 2017. Exploring the critical incident technique to encourage reflection during project-based learning." In *Proceedings of the 17th Koli Calling International Conference on Computing Education Research*, 88-97. 2017. https://doi.org/10.1145/3141880.3141899

[73] Masato Kawabata and Rachel Evans. 2016. How to classify who experienced flow from who did not based on the flow state scale-2 scores: A pilot study of latent class factor analysis. *Sport Psychol* 30, 3 (Sep 2016), 267-275. https://doi.org/10.1123/tsp.2014-0053

[74] Bruce L. Berg. 2004. *Qualitative Research Methods for the Social Sciences* (5th. ed.). Pearson Education, Boston, MA.

[75] Howard Lune and Bruce L. Berg. 2017. *Qualitative Research Methods for the Social Sciences* (9th. ed.). Pearson Education, Boston, MA

[76] Michael D. Myers. 2019. *Qualitative Research in Business and Management* (3rd. ed.). SAGE, London.

[77] Mihaly Csikszentmihalyi and Reed Larson. 2014. Validity and Reliability of the Experience-Sampling Method. In Mihaly Csikszentmihalyi (Ed.) *Flow and the Foundations of Positive Psychology*. Springer, Dordrecht. https://doi.org /10.1007/978-94-017-9088-8

[78] Anne Landhäußer and Johannes Keller. 2012. Flow and Its Affective, Cognitive, and Performance-Related Consequences. In Stefan Engeser (Ed.) *Advances in Flow Research*. Springer, New York, NY. https://doi.org /10.1007/978-1-4614-2359-1_4

[79] Mihaly Csikszentmihalyi. 2014. Attention and the holistic approach to behavior. In Mihaly Csikszentmihalyi (Ed.) *Flow and the Foundations of Positive Psychology*. Springer, Dordrecht. https://doi.org/10.1007/978-94-017-9088-8_1

[80] Ryan W. Quinn. 2005. Flow in Knowledge Work: High Performance Experience in the Design of National Security Technology. *Adm Sci Q* 50, 4 (Dec 2005), 610–641. https://doi.org/10.2189/asqu.50.4.610

[81] Paul B. Lowry, John D'Arcy, Bryan Hammer, and Gregory D. Moody. 2016. ''Cargo Cult" science in traditional organization and information systems survey research: A case for using nontraditional methods of data collection, including Mechanical Turk and online panels. *J. Strateg. Inf. Syst.* 25, 3 (Oct 2016), 232–240. http://dx.doi.org/10.1016/j.jsis.2016.06.002