The Emergence of Technostress in Software Development Work: Technostressors and Underlying Factors

Valtteri Siitonen [a], Saima Ritonummi [a], Markus Salo [a] and Henri Pirkkalainen [b]

[a] *Faculty of Information Technology, University of Jyväskylä, P.O Box 35, 40014 Jyväskylä, FINLAND*
[b] *Unit of Information and Knowledge Management, Faculty of Management and Business, Tampere University, P.O. Box 527, 33101 Tampere, FINLAND*

Abstract

Software development is a fast-growing market that is heavily tied to information technology (IT) use. Despite the high utilization of IT and high stress levels of software developers, research has largely neglected the effect of IT use on stress experienced by software developers. To address this gap in the research, we employ the concept of technostress. Prior technostress research has found many technostress-creating factors that cause severe negative consequences for both organizations and their employees. Despite these advancements, little is known about how technostress emerges in different organizational contexts and the underlying factors behind the technostress creators in these contexts. We conducted a qualitative study using the critical incident technique to uncover these factors and relevant technostress creators in the context of software development. We utilized a questionnaire with open-ended and closed-ended questions to collect descriptions of technostress experiences from 406 software developers. The current research identifies 21 influencing factors and 10 technostress creators to be relevant in software development, hence contributing to the technostress literature. We also contribute to software development research by explaining how the use of IT contributes to the stress experienced by software developers.

Keywords

Technostress, Software development, Stress, Stressor

1. Introduction

Software development is an important aspect of modern technology-driven business and is quickly becoming one of the biggest markets in the world [1]. It is important for organizations engaged in software development to be able to produce high-quality software. To accomplish this, it is important that developers can work with technologies that support their needs and ways of work to stay productive and efficient [2]. However, in the hectic and complex environment of software work [3], there are many hurdles that can obstruct these goals. With the high focus on IT use in software development, one possible hurdle is the stress caused by IT use, which is known as technostress [4].

The ever-increasing focus on IT use has sparked a light on the plethora of negative aspects of IT use in organizations including constant availability, hard-to-use technologies, and the fast pace of technological change [5, 6]. These negative aspects have made it difficult to keep up with the evolution of IT and to not experience some level of exhaustion or fatigue caused by the use of IT [7, 8]. The proliferation of technostress also brings forth the harmful consequences it can cause to both individual employees and organizations as a whole including increased turnover rate and burnout [7, 9]. These negative effects of IT use are especially relevant in occupations in which the use of IT is highly integrated into ways of work, such as software development. Overall, software development work can be a highly stressful area of work, and those working as software developers have been found to

8th International Workshop on Socio-Technical Perspective in IS Development (STPIS 2022), August 19–21, 2022, Reykjavík, Iceland
EMAIL: valtteri.m.e.siitonen@student.jyu.fi (A. 1); saima.e.ritonummi@student.jyu.fi (A. 2); markus.t.salo@jyu.fi (A. 3); henri.pirkkalainen@tuni.fi (A. 4)
ORCID: 0000-0001-7610-4584 (A. 1); 0000-0001-6708-7639 (A. 2); 0000-0001-5229-0300 (A. 3); 0000-0002-5389-7363 (A. 4)
© 2022 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-3239/paper22.pdf

encounter the highest levels of occupational stress compared with other occupations [10, 11]. The reasons for this are manifold and are caused by, for example, the common failure of software projects placing a burden on the developers, the high pace of change in the software development landscape, and the high workload experienced by software developers [3, 12, 13], resulting in diminished work motivation [14], burnout [15], and increased error rate [16].

Despite the high utilization of IT, need for extensive technical knowledge, and high stress levels of software developers [10, 13], little is known about how the use of IT can cause stress for those working in software development. Some studies on occupational stress in software development have mentioned that elements of IT use, such as the pace of technological change [11] and technical constraints [13], could affect the stress experienced by software developers. However, IT use has not been the focus of prior research and the technostress experienced by software developers has not been directly examined [e.g., 11, 13, 15]. Thus, there still exists a gap in research related to the relevant technostressors in software development work and why these technostressors emerge. Organizational technostress research has also presented a call for research regarding the emergence of technostress in specific contexts of using IT [4]. Despite the wide understanding of the different technostressors in the organizational environment, such as the complexity of IT and overload caused by IT use [5, 9], little is known about the underlying factors behind these technostressors and how they manifest in different contexts of IT use [17]. Uncovering these underlying factors is crucial to understand the issues in the IT use environment and mitigate the harmful effects of technostress. To address these gaps in research, our current study aims to answer the following research question: *How does the use of IT cause technostress in software development work?* To tackle this research question, we aim to identify both the relevant technostressors in the context of software development and the specific factors underlying these technostressors to explain why these technostressors emerge.

We conducted a qualitative study using the critical incident technique (CIT) [18] to build an understanding of technostress in the context of software development. The CIT is a research method for data collection and analysis to understand human behavior through critical events from the perspective of the respondent [18, 19]. We used the CIT to bring forth the voice of the respondents and understand the software developers' technostress experiences as described in their own words [19]. We collected 406 critical incident descriptions from those working in software development utilizing an online questionnaire with open-ended and closed-ended questions. Based on our findings, software developers experience technostress caused by multiple technostress creators, such as techno-overload, techno-uncertainty, and technology malfunctions. Some of the underlying factors in the IT use environment affecting technostressors are, for example, poor documentation, merge conflicts, and multitasking.

Our research contributes to both the technostress and software development literature. When it comes to the technostress literature, our study introduces the emergence of technostress in a previously unmapped context. We identify the relevant technostressors, such as techno-complexity and techno-uncertainty, and their underlying factors, in the context of software development, which broadens the understanding of these technostressors in different contexts of IT use [e.g., 4, 5, 8, 20]. By addressing the underlying factors affecting the technostressors, we contribute to an area of technostress research which has often been left uninvestigated in prior work [17]. Regarding the software development literature, our study begins to explain how the use of IT contributes to the high stress levels experienced by software developers. As a practical implication, our study recognizes technostress-creating IT use practices, suggesting actions to take to avoid them in software development work.

2. Background

Lazarus and Folkman define stress as a "relationship between the person and the environment that is appraised by the person as taxing or exceeding his or her resources and endangering his or her wellbeing" [21 p. 19]. In other words, both the imbalance between an individual and their surroundings and the individual's interpretations of what they find taxing affect whether an individual finds a particular event stressful or not [22]. This definition follows the transactional view of stress introduced in the field of psychology. Following the transactional view of stress, two important concepts can be realized: stressors and strains. Stressors are those factors causing stress to an individual, and strains are

the consequences or outcomes of the stressors [21, 22]. It is important to note that stress is a highly subjective experience, and two individuals can perceive the same situation in different ways, seeing it as either negative stress (distress) or positive stress (eustress) [23, 24, 25]. In the current study, we focus on negative stress (distress) because of its harmful consequences on those working with IT in the organizational context [e.g., 4, 9, 26, 27].

2.1. Technostress

Technostress is defined as stress caused by the use of IT [4]. Interest in technostress research has grown rapidly [28], and extensive research has been conducted on technostress in the organizational context [29]. The research in this area has mainly focused on the type of technostress generally experienced in organizations through different technostress creators (technostressors) and their consequences (strains). Some of the most commonly found technostressors in the organizational context are, for example, techno-overload, techno-uncertainty, and techno-complexity. Despite these advancements, little research has been conducted to recognize the underlying factors that affect the emergence of technostress creators in different organizational contexts, such as software development. Next, we examine the relevant technostressors, causes for them, and strains found to affect knowledge workers and IT professionals in the organizational environment [e.g., 4, 9, 30, 31].

Techno-overload refers to situations in which users must work harder and longer because of the demands imposed by IT use [27]. IT professionals have been found to experience overload caused by IT use for multiple reasons, including multitasking, productivity demands, and information load [6, 9, 32]. These factors can result in extended workdays and harmful consequences for the employee's well-being. Overall, techno-overload has been found to include similar aspects with *role overload* [9] that is discussed later in this section.

The hectic and unpredictable work environment can introduce a feeling of *techno-invasion*. The line between work and free time can become blurry because of both the pressure of constant availability and work IT making its way into the daily lives of employees [27, 32]. Thus, maintaining the distinction between a work role and family role can become increasingly difficult, leading to increased turnover intention and burnout [33]. This pressure of conflicting demands between work and family, which has only been exacerbated by the growing ease of working from home and IT use, can also introduce stress caused by *work–home conflict* [6].

The constant change in IT has introduced its own problems to employees in organizations. The constant change and learning of new IT can create *techno-uncertainty* among users, making them hesitant about learning and using new IT. With the constant introduction of new IT, learned skills can become obsolete and make users feel frustrated and anxious [30]. Learning new IT can also take a significant amount of time and feel increasingly complex and difficult [31]. The difficulty of using and learning IT is referred to as *techno-complexity* [27]. Complexity can make employees feel inadequate and question their skills in using IT, leading to frustration and helplessness. IT professionals have also been found to experience technostress caused by *techno-insecurity.* Here, techno-insecurity refers to one's fear of losing their job because of being replaced by more skilled IT users [30].

IT malfunctions have been found to cause technostress through systems breakdowns and unreliability [6, 34, 35]. IT malfunctions can leave employees frustrated and unable to do their jobs, causing work overload and interruptions to their workflow. Fischer et al. [35] also suggest *monitoring* as a possible technostress creator. Here, monitoring refers to the tracking of IT use, threatening the individual's feeling of privacy and potentially causing stress.

IT use has increased the number of *interruptions* users encounter in their work and interruptions have been connected to technostressors such as techno-overload [36]. Interruptions can be caused by, for example, different communication channels, such as Teams and Slack, notifications, reminders, and so forth. Employees are often required to use such channels and react swiftly to different alerts and notifications, taking their concentration away from the task they are working on. Interruptions taking a large chunk of the employees' time can make them struggle to complete their primary responsibilities, leaving them frustrated [37].

Finally, the use of IT has been connected to role stress caused by *role ambiguity* [6, 38] and *role overload* [39, 40]. Role ambiguity refers to role performance unpredictability and the lack of

information needed to perform said role [22]. Ayyagari et al. [6] show that IT-related factors, such as the pace of change and technology presenteeism, affect the level of role ambiguity experienced by employees. Role overload, on the other hand, is caused by having too many roles to fulfill, leaving employees overwhelmed by the amount of work they must accomplish [22]. IT use aspects linked with role overload have been found to include, for example, the complexity of IT and the changing and increasing demands because of the use of IT [27, 40].

Technostress can have many adverse consequences (strains) which can be either behavioral, psychological, or physiological [22]. The behavioral strains found in previous technostress research include, for example, absenteeism, turnover, and decreased use intention [9, 30, 41, 42]. These behavioral strains can have major negative effects on the organizations themselves through, for example, decreased productivity and increased costs [41, 43, 44]. Psychological strains include, for example, decreased job and user satisfaction, organizational and continuance commitment, burnout, exhaustion, and concentration problems [7, 8, 30, 41, 45, 46]. These kinds of strains can hurt the well-being of the individual experiencing technostress and decrease their ability to work. Technostress can also have physiological symptoms similar to normal stress; these can include headaches and increased stress hormone levels and heart rate [34, 36, 47, 48].

2.2. Software development and stress

Software development is often considered a hectic area of work with tight schedules, changing requirements, and a need for deep technical knowledge [11, 13]. This kind of work environment can expose employees to a high amount of pressure, leading to occupational stress and adverse outcomes such as burnout and increased turnover intention [11, 15]. Even though research on technostress in the context of software development is limited, research on the challenges and occupational stress encountered in software development has considered many different stressors that can cause stress in software development work. Next, we go over these stressors to create an overview of the challenges that software developers experience in their work. These findings will also be mirrored with the technostress creators described in the previous section.

Rajeswari and Anantharaman [13] propose 10 occupational stressors that are present in software development work: fear of obsolescence, individual team interactions, client interactions, work–family interface, role overload, work culture, technical constraints, family support, workload, and technical risk prosperity. Some of these stressors, such as fear of obsolescence, individual team interactions, and technical constraints, could be linked to the use of IT. These links manifest through a lack of skills to use the latest IT and constraints imposed by the software and hardware used. Thus, there are similarities with the technostressors described earlier, such as techno-complexity and IT malfunctions.

Other studies have also found developers experiencing stress because of overload caused by high workloads or difficulties balancing the demands of work and family [15, 49, 50, 51]. However, these studies explored the stressors on a more general level, and the use of IT was not examined as a possible reason for overload and the conflicts between work and family demands.

Software development work is highly subject to change and, thus, requires the ability to adapt to and tolerate change [52, 53]. The constant change in requirements, project scope, and introduction of new technologies have been found to affect the stress experienced by software developers and have been noted as a possible cause for software project failure [12, 13, 54]. The swift change of IT can make it difficult to keep up to date with the latest IT and skills in using them, which resembles the technostressors of techno-uncertainty and techno-complexity discussed in the previous section.

Software development can also be highly subject to interruptions that cause delays in one's workflow. Dealing with interruptions has been found to be an issue inside software development teams [55, 56, 57]. The causes for interruptions can be, for example, interactions with customers and team members, changes in the workflow through changing requirements, and environmental factors such as breaks and telephone calls [55]. This could imply that the interruptions caused by IT use are also a relevant stress-creating factor in software development.

With large, complex projects, software developers can also be required to perform different tasks and roles. Winderler, Maurping, and Venkatesh [3] show that the size, complexity, and volatility of software projects increase the stress caused from both role ambiguity and role conflict. Singh et al. [51]

also find that role ambiguity and role conflict contribute to exhaustion experienced by Indian software developers.

Occupational stress experienced by software developers has been connected to a variety of adverse outcomes for both developers and organizations. Sonnentag et al. [15] connect the stress experienced in software development teams to burnout. Other studies have also connected the stress experienced in software development to burnout and other well-being-related consequences, such as anxiety and decreased social activity [49, 51]. Stress has been found to negatively affect the performance of both software development teams and individuals [3, 11, 58], along with the organizational commitment of developers [51]. The pressure caused by stress can also lower the quality of work, and stress experienced in software development has been found to contribute to the number of development errors [16].

In summary, only a few connections to IT use have been made in the occupational stress literature among software developers. To the best of our knowledge, there is no research employing the concept of technostress in the context of software development. With the high utilization of IT in software development work, it is important to consider the effects of IT use on stress as experienced by those working in software development. Next, we present the design and results of our CIT study to identify the key technostressors and underlying factors affecting the emergence of these technostressors.

3. Methods

We needed rich data to understand the emergence of technostress in software development work. Thus, we used the CIT to collect self-reported descriptions of the technostress experienced by software developers in their work. The CIT is a qualitative research method that consists of "a set of procedures" for collecting and analyzing human behavior through critical incidents as reported by participants [18 p. 327]. The CIT is based on inspecting the phenomenon from the respondent's perspective, giving the respondent the freedom to decide the incident they feel is the most relevant for the phenomenon under interest [19]. Hence, in the current study, the CIT allowed the developers to bring forth their real-life experiences for an in-depth understanding of the phenomenon without restricting them to a specific framework or terminology [19, 59]. A critical incident can be described as an event that is "complete enough that it allows inferences and predictions to be made about the person performing the act" [18 p. 327] and that it has a significant negative or positive effect on the individual [19, 60]. The collection of critical incidents helped us gather detailed descriptions of the technostress experiences because critical incidents can be easier to remember [61]; they also fit well with the purpose of the current study, considering the significant negative effects that stress can have on an individual. Overall, the CIT has been found to produce reliable data [see 61, 62] and be useful for gathering insights into human behavior and IT use [e.g., 59, 63, 64, 65].

3.1. Data collection

We collected our data via an online questionnaire that included both open-ended questions to describe the critical stressful incident in detail and closed-ended questions to further evaluate the impact and outcomes of the incident. Adapting the formatting used in previous CIT studies [e.g., 59, 63, 66], we asked the respondents to recall "an exceptionally burdensome/stressful experience related to the use of technology or software in their work." We instructed the respondents to take their time when recalling the incident so as to increase the accuracy of the incident descriptions and minimize recall bias [66]. In the open-ended questions, the respondents were asked to provide a detailed description of the incident in their own words, describe the technology involved in the experience, and the factors that made the experience stressful for them. In the closed-ended questions, we asked the respondents to describe the strain, date, and duration of the incident. The questionnaire also included other questions that were left out of the scope of the current study. All the relevant questions used in the current study are listed in Appendix A. The questionnaire was carefully translated from Finnish to English to avoid changing the meaning of the questions and to maintain integrity. Three pretesters and a professional proofreader were also used to ensure that the questionnaire was coherent and understandable.

In the first phase of data collection, we created a pilot study in which we contacted software development organizations both in Finland and internationally. We asked the organizations to distribute

the questionnaire to their software developers. In total, we received 63 full responses. The pilot study was conducted between December 2021 and February 2022. The respondents were able to answer the questionnaire in both Finnish and English. Based on the pilot study, the quality and detail of the incident descriptions were redeemed sufficient. Only minor changes to the questionnaire were made when moving to the next phase of data collection.

Primary data collection was conducted using the online research platform Prolific between March 2022 and April 2022. Online research platforms such as Prolific have been found to be useful in collecting reliable data in large quantities [see 67, 68]. We used the prescreening criteria provided by Prolific to gather data from those working in the software-intensive industry sector[2], including software, information services, and video games. Respondents fulfilling the prescreening criteria were able to answer the questionnaire on a first come, first served basis. To identify those working in software development, we asked the respondents to describe their role in the industry and their job titles. To enhance the quality of the responses, we used two attention check questions, as suggested by Prolific, to ensure that the respondents understood what they were being asked in the questionnaire. We also set criteria to accept only responses from those with an approval rate of 97 or higher and a minimum of 20 previous submissions on the platform.

In total, we received 447 responses (including responses from the pilot study) from those working in software development of which 406 were included in the final sample (the exclusion criteria are discussed in the following section). Of those included in the sample, 312 were men, 89 were female, four were other genders, and one preferred not to disclose their gender. The age range of the sample was between 18 and 73 years old, with an average age of 31 years. In total, 39 different nationalities were present in the data, with the most common being Portuguese (n=64), Finnish (n=59), British (n=35), Polish (n=32), Italian (n=29), and South African (n=27). Of the 406 respondents, 327 reported having a bachelor's degree or higher. Most respondents (84.7%) were working full- or part-time software development jobs in an organization. Some (9.6%) were working as freelancers or entrepreneurs in software development, and some (4.9%) were primarily students, with most of them having part-time employment in a software development organization. A summary of the respondents' background information is presented in Table 1.

Table 1
Background information of the respondents (n=406)

	Frequency		**Frequency**
Gender		**Education**	
Male	312 (76.8%)	Less than a high school education	3 (<1%)
Female	89 (22.0%)	High school education or equivalent	74 (18.3%)
Other	4 (<1%)	Bachelor's degree	198 (48.9%)
I prefer not to disclose	1 (<1%)	Master's degree	124 (30.4%)
		Doctoral degree	5 (1.2%)
		I prefer not to disclose	2 (<1%)
Age		**Employment status**	
19 and under	6 (1.5%)	Employed full-time (30+ hours a week)	305 (75.1%)
20–29	211 (52.1%)	Employed part-time (<30 hours a week)	39 (9.6%)
30–39	110 (27.2%)	Entrepreneur	19 (4.7%)
40–49	46 (11.1%)	Freelancer	20 (4.9%)
50–59	18 (4.4%)	Student	20 (4.9%)
60 and over	3 (<1%)	Unemployed	2 (<1%)
I prefer not to disclose	12 (2.9%)	I prefer not to disclose	1 (<1%)

[2] The data collection was done as a part of a larger inquiry focusing on those working in the area of software intensive industry. Software intensive industry can be defined as the area of industry in which the dependability, utilization, and rate of change of software is high [69, 70]. However, in this study we focused only on those working specifically in software development.

3.2. Data analysis

When going through the incident descriptions, we used a content analysis approach in our analysis. We adapted the suggestions for content analysis presented by Berg [71] and identified overarching categories and data-driven categories, determined a coding scheme, and coded the data into the fitting categories. We maintained a data-driven focus by first going through the data and themes that emerged from them, after which we connected these themes to findings in the literature [72]. Our analysis was divided into two distinct phases.

In the first phase of analysis, two of the authors read through the data to identify incident descriptions to be included in the final data sample. In practice, this was done by going through the data and establishing possible exclusion criteria. For an incident description to be included in the final sample, it had to include a clear description of the incident, with sufficient detail of what happened and why it was critical [19]. Thus, 17 incident descriptions were excluded because of the reported strain being either "fairly low" or "very low" (on a 5-point scale), and four were excluded because of overall insufficient incident descriptions[3]. Additionally, the incident had to be relevant to the context of our study, namely, the use of IT in the context of software development work. Thus, 15 incident descriptions were excluded because of the incident not involving the use of IT and five because of the incident not being related to the work context. In total, 41 incident descriptions were excluded from the analysis. This left us with 406 out of 447 incident descriptions to be analyzed in the final stage.

In the second phase of our analysis, one of the authors went through the incident descriptions to identify the potential factors affecting the emergence of technostressors in software development work. In practice, one of the authors went through 100 incident descriptions, forming preliminary data-driven categories utilizing open coding. Open coding was utilized as different factors can appear as threatening depending on the environment that the individual operates in [24] and only using codes from prior research could have limited our ability to uncover the factors specific to software development. The categories were then labeled based on the factors identified in the IT use environment that emerged from the data. For example, incidents describing issues with using IT caused by documentation were put under a category labeled as "poor documentation". Labels from prior research were also used if the results included similar findings. For example, incidents describing software crashes and hardware failures were put under a category labeled "system breakdowns" [present in e.g., 34].

After this, the data were reread by the same author, during which all the incident descriptions were coded into the established preliminary categories. During the coding process, we constantly compared the new findings to our established categories to verify the categories [46, 72] and to identify possible new patterns present in the data [19, 71]. If an incident did not fit into one of the categories, a new category was established and labeled appropriately. After no new categories emerged [73], the rest of the data were coded into their fitting categories and all the categories were discussed between the authors resulting in only minor changes in the labels of the categories.

Finally, the established categories of factors were compared with each other and with findings from prior research to link the factors to different technostress creators [72]. For example, factors describing difficulties with using IT caused by "complex and difficult to use systems," "poor documentation," "old and legacy systems," and "poor code quality" were linked with the technostress creator "usability issues" [present in, e.g., 31]. It is also important to note that the same factors could affect multiple technostress creators, thus causing the respondent to suffer from the stress caused by multiple technostress creators simultaneously. For example, the factor "learning new IT" affected the technostressor "techno-uncertainty" because of the fear of new IT obsoleting previous knowledge and affected the technostressor "techno-overload" because of the amount of learning causing exhaustion.

[3] An incident description was redeemed insufficient if it did not provide any details of the context of the incident or why it was stressful (e.g., "It was very awkward" and "I can't focus on a stressful experience right now I get stressed")

4. Results

In the closed-ended questions, to create an overview of the characteristics of the incidents, we asked the respondents to provide the level of strain, length, and date of the incidents. Most of the respondents described the strain caused by the incident to be either "fairly high" (n=165) or "very high" (n=161), with 80 respondents describing the strain as "moderate." Less than half of the respondents (n=143) described a more acute incident lasting less than 24 hours, whereas over half of the respondents (n=257) described an incident lasting more than this time frame, with many incidents lasting over a month (n=73). This would imply that technostress experienced by software developers is not only caused by acute IT use-related issues but also by more chronic stressful incidents from using IT in their work. This could also partly explain the high number of incidents that occurred over a month ago (n=292) relative to answering the questionnaire. As discussed earlier, critical incidents can also be easier to remember, and a highly stressful experience can linger in the mind for a long time. An overview of the technostress incident characteristics is presented in Table 2.

Table 2
Overview of technostress incident characteristics

Reported strain	n	For how long did the experience last?	n	How long ago did the experience occur?	n
Very high	161	Less than an hour	13	Less than two weeks ago	57
Fairly high	165	1–12 hours	91	2–4 weeks ago	53
Moderate	80	13–24 hours	39	1–3 months ago	87
		1–7 days	106	4–12 months ago	94
		1–4 weeks	78	Over one year ago	111
		Over one month	73	I don't know	4
		I don't know	6		

Overall, our results show that multiple factors in the IT use environment affect the technostress creators and, hence, technostress experienced by software developers in their work. The incident descriptions included stress caused by technostress creators, such as usability issues, interruptions, and techno-uncertainty. The software development-related factors in the IT use environment affecting technostress creators were, for example, poor documentation, poor code quality, rapid technological change, and multitasking. The respondents also described different kinds of strains that they suffered from, including negative emotions such as frustration, anger, and helplessness, and negative effects on, for example, their work motivation, productivity, well-being, and work quality. Some also described far-reaching consequences for their well-being, which left them thinking about quitting their job or experiencing long-lasting exhaustion or burnout. Next, we go over the factors in the IT use environment and technostress creators identified in the incident descriptions affecting the emergence of technostress in the software developers' use of IT. The main results are summarized at the end of this section in Table 3.

4.1. Usability issues (techno-complexity)

Many of the respondents reported that **complex and hard-to-use systems** affect the stress caused by usability issues. Hard-to-use systems slowed the work pace and caused frustration. As some described, figuring out how to use the system took longer than the actual work task itself. Working with complex systems made doing smaller, simpler tasks take more troublesome, increasing the possibility of making mistakes in the software development process. Some respondents described these kinds of systems to include tedious design choices that made the system more difficult to use and that using the system required either trial and error or workarounds to perform the task. In the following example, a developer described the stress caused by the use of a program that was aimed at helping the software development team keep track of the progress of their projects:

The program was too complex, required too many inputs for it to produce a work structure. It was developed to help us track our progress better as a team, but it took more time figuring it out than doing the actual work.

One aspect of stress caused by usability issues brought up in the incident descriptions was working with **old and legacy systems.** The developers described how working with legacy technologies caused a variety of problems because of the technologies being hard to understand and not used often anymore. It was described as difficult to create workflows with legacy systems and programming languages because solutions needed to be pondered from an "old point of view." Older systems were also often described as slow, cumbersome, and more complex compared with newer systems, as demonstrated by the following example:

We have some older products that are developed in C++, which is an older language. Dealing with this language and this older project has always been very slow and cumbersome compared with newer languages such as C#. Whenever I have to deal with this project, it seems like it is always a pain to compile and do things that are much simpler in the newer projects.

Documentation was mentioned as an important part of the software development process. **Poor documentation** was described as hindering the use and development of otherwise usable systems by making their use more cumbersome. In a way, documentation was seen as a user manual of sorts to understand how to use the systems and the code on which they were built. Thus, poor documentation made the use of systems more difficult, time-consuming, and, in some cases, even impossible. If the documentation were lacking, it could leave the developer alone with the problem that they were trying to solve.

The fact that there is no documentation or help on the web for what I was doing and so I had to try to solve the problem all by myself and try to figure out why I was having terrible performance.

Poor code quality was also referred to as a factor causing usability issues. Bad quality code made the systems harder to understand and use, producing issues similar to poor documentation. Issues took longer to solve, leaving developers frustrated and angry about the time wasted to figure out the system. Sometimes, the work had to be started over, and all progress was lost because of issues with the quality of the code.

E2E tests in cypress were failing a lot, I was doing an investigation of the issues and if possible, also fixes. But tests were very poorly written … difficult to understand and even more difficult to fix … After some time, you just realized that the best thing would be to delete all of it and start over.

4.2. Techno-uncertainty

Learning new IT was often described as frustrating and difficult. When it came to learning new IT, the respondents also had to learn new ways of working, which affected the overall software development process and their productivity. The developers described that the hectic work environment with the pressure of deadlines and tight schedules increased the pressure to learn new IT quickly and efficiently. Learning also took its toll on the respondents. Learning new IT was often described as exhausting because of the increased workload (linking it closely with techno-overload), the number of technologies to learn, and the pace of change. One respondent described having to work "double the hours," and another "thought about quitting the job" because of excessive learning tasks. The stress caused by learning was described by one of the respondents as follows:

Web apps aren't my field, but I was forced to learn everything in 1 week only. For this reason, I had to study and work for about 20 hours a day, from morning to night. It was totally too hard for me.

Another aspect closely related to learning was **technological change**. The developers described that it was difficult to get used to the pace of change and that software development was too focused on adopting new IT over productivity. For example, one of the respondents described being stressed by "the constant change of (technical) frameworks" and suffering from "change fatigue." The constant change in IT was often described as a trigger for learning and many developers described that they were not listened to when changes in IT were being made. The constant change in IT could also invalidate earlier learning activities, which made the work feel pointless.

The focus (of software development) is on developing the tools for development … The appearance of solutions and the trendiness of the technologies take precedence. … As a developer, this causes frustration and the work to feel pointless. … The time invested in learning them (new tools and systems) is wasted when they are discarded and moved on to the next trending technology.

4.3. Techno-overload

Many respondents mentioned experiencing overload and exhaustion because of the **piling up of work and overtime**. Multiple smaller factors were mentioned as affecting the piling up of work caused by IT use, such as fixing broken IT, learning new IT, the complexity of the IT used, and strict deadlines. Many respondents described having to work off hours, which they reported to cause many well-being-related issues, such as lack of sleep, restlessness, and even burnout. Some respondents also described that there was just too much to do, and the workload was becoming too much, leaving them overwhelmed and exhausted. For example, one developer described a project in which they had too much to do with the IT in question:

I had a lot to code, from the temperature control to the gas flow inside the oven. … This was all done in a few weeks, leaving me feeling burnout. I worked a lot of extra hours for the project to be successful and on time.

Another described working with a faulty alpha release as follows:

… almost no sleep, cognitive impairment due to sleep deprivation, stress, and exhaustion. Feelings of helplessness and frustration.

Specific **taxing tasks** when using IT were also recognized as an aspect of stress caused by techno-overload. These kinds of tasks were described by the respondents as being mentally draining and difficult, which required, for example, a lot of creativity and problem-solving skills. Another aspect described by the respondents was that these tasks had to be completed swiftly in a limited amount of time which increased the pressure put onto the developers. One of the respondents described an exceptionally taxing task with a Java application when working as a coding instructor, as follows:

I don't exactly recall the project details, but I was required by a student to assist them in building a project using Java. It was a burdensome experience as it required a lot of creativity, out-of-the-box thinking and extensive knowledge of Java. The project problem did not have an immediate solution I could think of, and every part of it required significant mental effort.

Less-mentioned techno-overload-creating factors were **multitasking** and **information load**. Software developers mentioned having to work on multiple projects at the same time, with many projects using different tools and technologies. Working with different technologies required doing

different tasks simultaneously, increasing the risk of experiencing fatigue, and being exhausted. For example, one of the respondents described experiencing stress and eye strain caused by working on multiple tasks and monitors simultaneously:

> *I have been asked to handle many open tickets due to bugs on the main platform. The time to complete the operation was really short and, in the meantime, I had to continue with a task assigned to me. I use two monitors at work, my eyes got tired.*

A few respondents also described incidents in which they were flooded with information. The understanding of large and complex systems was mentioned as requiring a lot of information to take in, leaving the developers overwhelmed and exhausted because of information overload. For example, one of the developers described working on a web application as follows:

> *I worked many hours a day to finish on time, and I was very stressed because there was a lot of information to process. … Besides the lack of time, web development is a heavy and complex task, because you have to process a lot of data …*

4.4. Role overload and role ambiguity

Insufficient IT use skills affected the stress caused by role overload and role ambiguity. Many respondents described incidents in which they had to perform a variety of roles that were not their area of expertise while using IT they had limited experience working with. Sometimes, it was assumed that the developers could handle new IT and fill in a variety of roles in the software development process. The lack of skills and resulting workload had a negative effect on the respondents, decreasing productivity and increasing role overload. In the following example, a web developer described an incident in which they were tasked to work as a graphics designer with limited knowledge of the tools, in addition to their usual job description:

> *I was tasked to make animations for a website, I am not a graphics designer, I am a website developer hence I did not find it amusing at all. I do like seeing ideas into life, yes, but I prefer doing that with code, not with graphics designing skills that I barely possess.*

The developers also described incidents in which they had to work with limited knowledge and a lack of technological skills to perform the tasks in their usual role description. These incidents were often related to having to work with IT that the developers had no prior experience using and were common among those who had just started out in a new position. Limited knowledge with the IT and the role made the work feel burdensome and frustrating and many developers described being stressed by having no idea what they were supposed to be doing. This resulted in some doubting their IT use skills and even their ability to work as a developer because of role ambiguity, as demonstrated by the following quote:

> *The fact that I wasn't used to the particular software nor languages in hand. This already had me kind of frustrated, so when we couldn't figure out what was the problem for some time, I was really struggling and felt like I didn't know enough for the job.*

4.5. Work–home conflict (techno-invasion)

A few respondents mentioned **work IT invading their free time** and **constant availability** as a source of stress linked to work–home conflict. The respondents described constant connectivity through different IT as enabling—and in a way requiring— them to be constantly available. One reason for this kind of requirement was mentioned as being the increasing competition in the software development sector. Constant availability was seen as blurring the boundaries between work and personal time due to having to work off hours, leaving respondents frustrated, angry, and exhausted, which can also be

linked to techno-overload. In the example, a respondent described how the pressure of constant connectivity through IT forced them to work on their personal time:

> *Now a days with internet-enabled phones we are fully connected. However, the same thing can be burdensome when you really want to have quite time with friends and family. … As competition increases, one needs to become more and more responsive even after office hours and on weekends. We went camping with friends. The plan was to have some isolation from work and have personal fun time. However, the production system had an issue and I got immediate chat and lots of emails. I had to spend close to 4 hours on that till the issue got resolved.*

4.6. Interruptions

The respondents described different kinds of interruptions caused by IT use. Interruptions were found to be stressful because of them disturbing the development workflow in an already hectic work environment. Interruptions were their own technostress creator because of some respondents reporting stress solely caused by interruptions, but they were also closely connected as a factor with other technostress creators, such as IT malfunctions and techno-overload. Examples of **interruptions caused by technical issues** were the downtime of a system or IT not functioning properly.

> *Getting stuck not being able to work for an entire day because the backend wouldn't work was incredibly frustrating.*

These kinds of issues were often not resolvable by the developers, which caused some to feel helpless. The respondents described having to wait for the issue to be resolved before they could continue their work and having to claw back the time that was wasted, leading to extended workdays. On the other hand, in some cases, the respondents felt like they could solve the issue and found themselves looking for the solution, wasting time, and unable to continue with their primary work tasks.

> *I spent the whole day trying to submit one piece of code (because of faulty validation checks), ending up running the code/content validation builds time after time after time. Sounds like a chill day, but I also have to take into account the knock-on effect that has on the other work I need to complete, and the effect that has on people who need to use my work. Hell.*

Interruptions were also caused by the **use of certain IT** in the software development process. This kind of IT was used for communication between team members and in performing development activities such as testing and implementation. The respondents described the reasons behind these kinds of interruptions as how the IT was used rather than the IT on its own. These reasons included restrictions based on IT use and how communication was arranged between team members. One of the respondents described being constantly disturbed by others through internal communication channels, and in the following example, another described their work being delayed because of the restrictions placed on the use of their testing environment only once a day:

> *There were manual jobs included in the application that enabled the system to run the transactions automatically at a certain point in time of the day. … As a tester, I needed the jobs and transactions to run as soon as possible for testing purposes. Instead, I had to wait for the jobs to finish …*

The final cause of interruptions mentioned by the respondents was **merge conflicts**. Merge conflicts were stressful because of them taking a significant amount of time to resolve and running the risk of deleting work in progress. Merge conflicts were reported as hampering productivity and causing helplessness and exhaustion among the respondents.

I recently had a situation where I had to merge my work into a branch someone else had also made changes to, so I was faced with a huge merge conflict, which took me a lot of time to unpick.

4.7. IT malfunctions

On their own, **system breakdowns,** such as crashes and hardware failures were mentioned as a common source of stress caused by IT malfunctions. System breakdowns were also often mentioned with many other technostress creators, such as interruptions and techno-overload. Crashes and hardware failures were described as making the development process more tedious and exhausting because of catching up, especially if work progress had been lost in the crash.

I was working on our internal tool. Suddenly, the tool crashed, and all my progress was gone. I had to start the task from the beginning, which wasted a lot of my time.

Common aspects of **buggy and unreliable software** were poor system performance and glitches, leading to difficulties in performing the tasks the software was intended to be used for. Some respondents described stress as being caused by long-lasting issues related to the unreliable behavior of the software that should have been resolved a long time ago. Yet that still hindered the use of IT and workflow. Buggy and unreliable software were also mentioned, with stress caused by other technostress creators, such as usability issues and interruptions. Overall, bugs and unreliability caused frustration and hindered the respondents' ability to perform their work tasks.

My whole team is working from home. We had an issue where this internal software developed a bug where it wouldn't update what co-workers were uploading what they had done throughout the day. … When this happened, nobody had any clue what had been done.

4.8. IT–user misfit

IT not fitting the needs of the user or solution was described by many as a source of stress caused by misfit. IT–user misfit caused a plethora of different issues, also linking it with other technostress creators such as techno-overload and IT malfunctions. The respondents described incidents in which they had to work with IT that did not fit the purpose of the work they were doing, causing frustration and the work feeling tedious and pointless. Both forms of IT-user misfit resulted in tasks taking a lot of extra time, causing exhaustion and an increased pace of work. The right tools that fit the needs of the user and the task were seen as an essential part of productivity to enable task performance and avoid costly mistakes and workarounds in the development process. One respondent described a stressful incident in which the computer they had to work on did not meet the specifications necessary for their work, resulting in workarounds and unstable performance:

The programs I develop handle lots of data, and when I started out, my company computer had less than the ideal amount of RAM, making the workflow laggy and I was even forced to have to use 2 different computers at once, so that the company computer did not crash because of too many applications being open. … So trying to develop code and have many applications open at once was frustrating and took a lot of time. Lots of crashes happened.

Another aspect of IT–user misfit was **incompatibility** between different technologies. Issues with incompatibility were described as making the user experience more stressful and creating technical difficulties such as crashes. Incompatibilities led to wasting developers' time, having to use workarounds, or the IT not fulfilling the tasks it was meant to perform.

... The stressful part was that there was no compatibility between Visual Studio 2019 and Crystal Reports, so my user experience when manipulating reports was exponentially more stressful, like the VS19 would crash randomly, features of CR would be unavailable or impossible to use.

4.9. Monitoring

The **use of IT intended for monitoring** was also mentioned by a few respondents as a cause of stress. Monitoring and controlling the respondents' use of IT was seen as particularly stressful by the few who mentioned these kinds of incidents. Monitoring made the respondents feel uncomfortable and pressured which affected their productivity, sense of privacy, and work motivation. The use of monitoring software was also enforced, so the respondents had no say in whether they wanted to use that kind of software or not. For example, one respondent described having to use software intended for monitoring:

The boss wanted to measure how much time we spent on the computer and what we did during it, so they installed that monitoring software for the 8 hours every day, and if you don't type, it sent you alerts and took screenshots for a company server. I felt hyper watched.

Table 3
Summary of the technostressors and factors affecting them in software development

n	Factors in the IT use environment of software development [a]	Technostressors
53	Complex and hard-to-use systems	Usability issues (techno-complexity)
27	Poor documentation	
25	Poor code quality	
16	Old and legacy systems	
32	Learning new IT	Techno-uncertainty
17	Technological change	
45	Piling of work and overtime because of IT use	Techno-overload
13	Taxing tasks using IT	(information overload, work overload)
10	Multitasking	
10	Information load	
52	Insufficient IT use skills	Role overload and role ambiguity
10	Constant availability because of IT use	Work–home conflict (techno-invasion)
32	Interruptions caused by technical issues	Interruptions
16	Interruptions caused by IT use	
7	Merge conflicts	
37	System breakdowns	IT malfunctions
29	Buggy and unreliable software	
28	IT not fitting the needs of the user	IT–user misfit
19	IT not fitting the solution	
16	Technical incompatibility	
7	Use of IT intended for monitoring	Monitoring

[a] = *many of the factors were also linked with other technostressors*

5. Discussion

The aim of our study was to examine the emergence of technostress in the previously unmapped context of software development. We used the CIT to build an understanding of the technostress creators and context-specific factors of software development that contribute to the technostress creators experienced by software developers. This approach contributes to the call for research

examining technostress through a more context-focused lens in the organizational environment [4]. Our research provides new insights into how and why technostress emerges in software development work, hence making theoretical contributions to both the technostress and software development literature.

5.1. Contributions to research

We contribute to technostress research in the following ways: First, we recognize multiple factors in the software development IT use environment that contribute to the emergence of technostress creators and technostress in the context of software development. These include, for example, poor documentation, poor code quality, and working with older or legacy technologies for techno-complexity, merge conflicts for interruptions, and the effect of the hectic and volatile nature of software development on techno-overload and techno-uncertainty. Examining the specific factors underlying technostress creators has often been overlooked in current technostress research [17]. Thus, our findings provide new dimensions for the inspection of these technostressors for future studies. For future research, we suggest that the software development factors and their effect on technostress creators could be further validated using a quantitative approach. Second, our study extends the overall body of knowledge about technostress in an organizational environment. We have found many previously studied technostress creators, such as techno-overload, techno-uncertainty, and usability issues, to be relevant in the context of software development as well. These findings further enhance the relevancy of these technostress creators in the organizational context of IT use [e.g., 4, 5, 9, 29, 31]. We also have connected a less-studied technostress creator of monitoring suggested by Fischer et al. [35] to the emergence of technostress in software development work. Additionally, we found IT–user misfit to contribute to technostress experienced by software developers. This result contradicts previous findings because the fit between technology and the task has mostly been considered as a technostress inhibiting factor [74]; therefore, this provides a new perspective to be considered in technostress research. To further enhance the knowledge of technostress in the context of software development, we suggest that future research examine both the individual and organizational ways to mitigate the harmful effects of technostress in the context of software development.

Our research also contributes to the software development literature, specifically to the research on occupational stress experienced by software developers in their work. First, the impact of IT use on stress experienced by software developers has rarely been mentioned in relation to other occupational stressors [11, 13]. Our research employs the concept of technostress to further explain the stress experienced by those working in software development. For example, our results bring in the aspect of IT use to explain stress caused by overload and usability issues. Our findings also contribute to earlier findings related to challenges caused by the pace of technological change [11] and learning of new IT [53] through technostress creators such as techno-uncertainty. We recommend that the research focusing on stress in software development also consider the stress caused by IT use and its role on the high stress levels experienced by software developers. Finally, we ponder a possible connection between technostress and its resulting outcomes in software project failure. Compared with the literature on software project failure, the incident descriptions included similar themes related to the lack of skills, being overworked, and the use of new IT [e.g., 54, 75, 76]. With our results suggesting that technostress can affect, for example, the motivation, productivity, and work quality of developers, we suggest that the possible connection between technostress and failed software projects should be examined in future research.

Our findings can also be linked to ideas of the socio-technical approach to systems design. Based on our results, failing to optimize the needs between human and technical factors [2, 77] can lead to adverse outcomes of technostress demonstrated by, for example, the technostressors IT malfunctions and IT-user misfit. This emphasizes the importance of not forgetting the needs of the user when introducing technical solutions [2] and could lead to interesting avenues for future research such as if a stronger emphasis on socio-technical ideas for system design could mitigate the harmful effects of technostress.

5.2. Practical implications

Our study also has practical implications that should be taken into consideration in organizations engaged in software development. First, the realization of technostress in the software development context shows the struggles and frustration that developers face in adapting to the fast pace of technological change and resulting issues with using IT. The constant adoption of new trending technologies can break the workflow of developers, causing change fatigue and excessive workloads. More problems arise if the adopted IT does not fit the needs of the user or the tasks they are trying to accomplish. Thus, we recommend that software development organizations listen to the needs of their developers and make them more involved in the selection of tools and technologies that best suit the purpose of the development task. The pace of change should also be kept reasonable to avoid adverse outcomes, such as lowered productivity and negatively impacted well-being of developers.

Second, it became evident in the incident descriptions that developers are sometimes thought of as fluid technology users who can be put into performing a variety of roles using different technologies. However, this is often seen as an irritating practice that can cause stress, frustration, and confusion among developers, hampering both their motivation and productivity. Organizations should make sure that developers have a clear understanding of the role and demands of their work to avoid excessive overload and multitasking caused by the use of IT.

Third, developers face interruptions caused by the use of IT and suffer from harmful IT use practices, such as monitoring and the use of work IT in their personal time. Developers should be given quiet time to focus on their work without disturbances and should not feel obligated to be connected to their work IT in their free time. We encourage software development organizations and individuals working in software development to avoid these harmful IT use practices and apply clear policies on using IT to avoid excessive interruptions and work–home conflict.

Finally, the presented factors affecting the emergence of technostress in software development could help software developers to recognize these factors in their own organizational IT use environment. Recognizing these factors and the possible technostress caused by them could help software developers to point out these issues and drive towards fixing them, improving their well-being and quality of work. Fixing these issues could not only alleviate the adverse outcomes of technostress, but also improve the atmosphere of the working environment of the organization.

5.3. Limitations and conclusion

As with all research, we recognize important limitations with our study. First, the use of CIT has limitations that should be discussed. Even though CIT has been found reliable for collecting accurate incident descriptions [61, 62], self-reported data are still suspect to recall bias [19]. Many of the described incidents had occurred some time ago, and as time passes, the individual might reinterpret a past incident. However, CIT research has confirmed that recall is more accurate when reporting on critical behavior [61], and critical details might be missed if only recent incidents are included in the analysis [18]. Second, we recognize that the collection of the most critical technostress incidents might not bring forth the more minor and continuous stressful aspects of IT use that are present in the everyday work of software developers. It is also possible that the recognized factors in the IT use environment affect other technostress creators that were not present in our data set. Finally, because our mode of inquiry was a questionnaire, the researcher could not ask subsequent questions [78]. Thus, it is possible that some details were missed that could have affected the interpretation of the incidents.

Software development is an important part of modern organizations' business. It is key that software development produces high-quality software and that this process goes hand in hand with the happiness and well-being of those working in the field of software development. In the current study, we employed the concept of technostress and saw how many factors can endanger the well-being of software developers and affect the productivity of both developers and organizations alike. Understanding these issues is crucial to avoiding them in the future and figuring out ways to decrease the impact of unhealthy IT use practices in software development work.

6. Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful comments in improving the quality of the paper. This research has been funded by the Foundation for Economic Education, Finland, and partially by the Academy of Finland (341359).

7. References

[1] Gartner Research, Software Market View 2020-2021, 2021. URL: https://www.gartner.com/en/documents/4004846

[2] E. Mumford, Socio-technical design: an unfulfilled promise or a future opportunity?, in: Organizational and Social Perspectives on Information Technology, Springer, Boston, MA, 2000, pp. 33-46. doi:10.1007/978-0-387-35505-4_3

[3] J. B. Windeler, L. Maruping, V. Venkatesh, Technical Systems Development Risk Factors: The Role of Empowering Leadership in Lowering Developers' Stress, Information Systems Research 28 (2007) 775–796. doi:10.1287/isre.2017.0716.

[4] M. Tarafdar, C. L. Cooper, J. Stich, The technostress trifecta - techno eustress, techno distress and design: Theoretical directions and an agenda for research, Information Systems Journal 29 (2019) 6–42. doi:10.1111/isj.12169.

[5] T. Fischer, R. Riedl, Technostress Research: a Nurturing Ground for Measurement Pluralism?, Communications of the Association for Information Systems 40 (2017) 375–401. doi: 10.17705/1CAIS.04017.

[6] R. Ayyagari, V. Grover, R. Purvis, Techostress: Technological Antecedents and Implications, MIS Quarterly 35 (2011) 831–858. doi:10.2307/41409963.

[7] F. Gaudioso, O. Turel, C. Galimberti, The mediating roles of strain facets and coping strategies in translating techno-stressors into adverse job outcomes, Computers in Human Behavior 69 (2017) 189–196. doi:10.1016/j.chb.2016.12.041.

[8] L. Reinecke, S. Aufenanger, M. E. Beutel, M. Dreier, O. Quiring, B. Stark, K. Wölfling, K. W. Müller, Digital Stress over the Life Span: The Effects of Communication Load and Internet Multitasking on Perceived Stress and Psychological Health Impairments in a German Probability Sample, Media Psychology 20 (2017) 90–115. doi:10.1080/15213269.2015.1121832.

[9] T. S. Ragu-Nathan, M. Tarafdar, B. S. Ragu-Nathan, Q. Tu, The Consequences of Technostress for End Users in Organizations: Conceptual Development and Empirical Validation, Information Systems Research 19 (2008) 417–433. doi:10.1287/isre.1070.0165.

[10] M. Kumashiro, T. Kamada, S. Miyake, Mental stress with new technology in the workplace, in: M.J. Smith and G. Salvendy (Eds.), Proceedings of the Third International Conference on Human–Computer Interaction, Elsevier Science, New York, NY, 1989, pp. 270–277.

[11] M. A. Chilton, B.C. Hardgrave, D. J. Armstrong, Person-Job Cognitive Style Fit for Software Developers: The Effect on Strain and Performance, Journal of Management Information Systems 22 (2005) 193–226. doi:10.1080/07421222.2005.11045849.

[12] I. Crnkovic, Component-Based Software Engineering—New Challenges in Software Development, Software Focus 2 (2001) 127–133. doi:10.1002/swf.45

[13] K.S. Rajeswari, R. N. Anantharaman, Development of an instrument to measure stress among software professionals: Factor analytic study, in: Proceedings of the 2003 SIGMIS Conference on Computer Personnel Research: Freedom in Philadelphia--leveraging differences and diversity in the IT workforce, Association for Computing Machinery, New York, NY, 2003, pp. 34–43. doi: 10.1145/761849.761855.

[14] D. Graziotin, F. Fagerholm, X. Wang, P. Abrahamsson, What happens when software developers are (un) happy, Journal of Systems and Software 140 (2018) 32–47. doi:10.1016/j.jss.2018.02.041.

[15] S. Sonnentag, F. C. Brodbeck, T. Heinbokel, W. Stolte, Stressor-burnout relationship in software development teams, Journal of Occupational and Organizational Psychology 67 (1994) 327–341. doi:10.1111/j.2044-8325.1994.tb00571.x.

[16] T. Furuyama, Y. Arai, K. Iio, Analysis of fault generation caused by stress during software development, Journal of Systems Software 38 (1997) 13–25. doi:10.1016/s0164-1212(97)00064-2.
[17] M. Salo, H. Pirkkalainen, C. E. H. Chua, T. Koskelainen, Formation and Mitigation of Technostress in the Personal Use of IT, MIS Quarterly 46 (forthcoming, 2022). doi:10.25300/MISQ/2022/14950.
[18] J. C. Flanagan, The critical incident technique, Psychological Bulletin 51 (1954) 327–358. doi:10.1037/h0061470.
[19] D. D. Gremler, The critical incident technique in service research, Journal of Service Research 7 (2004) 65–89. doi:10.1177/1094670504266138.
[20] A. M. Fuglseth, Ø. Sørebø, The effects of technostress within the context of employee use of ICT, Computers in Human Behavior 40 (2014) 161–170. doi:10.1016/j.chb.2014.07.040.
[21] R. S. Lazarus, S. Folkman, Stress, appraisal, and coping, Springer Publishing Company, New York, NY, 1984.
[22] C. L. Cooper, P. J. Dewe, M. P. O'Driscoll, Job-Related Sources of Strain, in: Organizational stress: A Review and Critique of Theory, Research, and Applications, Sage Publications Inc., Thousand Oaks, CA, pp. 27–60, 2001. doi: 10.4135/9781452231235.n2.
[23] R. S. Lazarus, From Psychological Stress to the Emotions: A History of Changing Outlooks, Annual Review of Psychology 44 (1993) 1–22. doi:10.1146/annurev.ps.44.020193.000245
[24] M. Le Fevre, J. Matheny, G. S. Kolt, Eustress, distress, and interpretation in occupational stress, Journal of Managerial Psychology 18 (2003) 726–744. doi:10.1108/02683940310502412.
[25] C.B. Califf, S. Sarker, S. Sarker, The Bright and Dark Sides of Technostress: A Mixed-Methods Study Involving Healthcare IT, MIS Quarterly 44 (2020) 809–856. doi:10.25300/misq/2020/14818.
[26] P. Borle, K. Reichel, F. Niebuhr, S. Voelter-Mahlknecht, How Are Techno-Stressors Associated With Mental Health and Work Outcomes? A Systematic Review of Occupational Exposure to Information and Communication Technologies within the Technostress Model, International Journal of Environmental Research and Public Health 18 (2021) 8673. doi:10.3390/ijerph18168673.
[27] M. Tarafdar, Q. Tu, B. S. Ragu-Nathan, T. S. Ragu-Nathan, The Impact of Technostress on Role Stress and Productivity, Journal of Management Information Systems 24 (2007) 301–328. doi:10.2753/mis0742-1222240109.
[28] C. Salazar-Concha, P. Ficapal-Cusí, J. Boada-Grau, L. J. Camacho, Analyzing the evolution of technostress: A science mapping approach, Heliyon 7 (2021) 1–15. doi:10.1016/j.heliyon.2021.e06726.
[29] A. S. Nisafani, G. Kiely, C. Mahony, Workers' technostress: a review of its causes, strains, inhibitors, and impacts, Journal of Decision Systems 29 (2020) 243–258. doi:10.1080/12460125.2020.1796286.
[30] M. Tarafdar, Q. Tu, B. S. Ragu-Nathan, T. S. Ragu-Nathan, Crossing to the dark side: Examining Creators, Outcomes, and Inhibitors of Technostress, Communications of the ACM 54 (2011) 113–120. doi:10.1145/1995376.1995403.
[31] C. Sellberg, T. Susi, Technostress in the office: a distributed cognition perspective on human–technology interaction, Cognition, Technology & Work 16 (2014) 187–201. doi:10.1007/s10111-013-0256-9.
[32] H. Yun, W. J. Kettinger, C. C. Lee, A new open door: The smartphone's impact on work-to-life conflict, stress, and resistance, International Journal of Electronic Commerce 16 (2012) 121–152. doi:10.2753/jec1086-4415160405.
[33] K. J. Harris, R. B. Harris, M. Valle, J. Carlson, D. S. Carlson, S. Zivnuska, B. Wiley, Technostress and the entitled employee: impacts on work and family, Information Technology & People 35 (2022) 1073–1095. doi:10.1108/itp-07-2019-0348.
[34] R. Riedl, H. Kindermann, A. Auinger, A. Javor, Technostress from a Neurobiological Perspective, Business & Information Systems Engineering 4 (2012) 61–69. doi:10.1007/s12599-012-0207-7.
[35] T. Fischer, A. Pehböck, R. Riedl, Is the technostress creators inventory still an up-to-date measurement instrument? Results of a large-scale interview study, in: Proceedings of the 14th International Conference on Wirtschaftsinformatik. Siegen, 2019, pp. 1834–1845.

[36] P. Galluch, V. Grover, J. B. Thatcher, Interrupting the Workplace: Examining Stressors in an Information Technology Context, Journal of the Association for Information Systems 16 (2015) 1–47. doi:10.17705/1jais.00387.
[37] S. Tams, J. B. Thatcher, V. Grover, Concentration, Competence, Confidence, and Capture: An Experimental Study of Age, Interruption-Based Technostress, and Task Performance, Journal of the Association for Information Systems 19 (2018) 857–908. doi:10.17705/1jais.00511.
[38] M. Salanova, S. Llorens, E. Cifre, The dark side of technologies: Technostress among users of information and communication technologies, International Journal of Psychology 48 (2013) 422–436. doi:10.1080/00207594.2012.680460.
[39] K. Wang, Q. Shu, (2008). The Moderating Impact of Perceived Organizational Support on the Relationship Between Technostress and Role Stress, in: The 19th International Workshop on Database and Expert Systems Applications, IEEE, 2008, pp. 420–424. doi:10.1109/DEXA.2008.67.
[40] M. A. Alam, Techno-stress and productivity: Survey evidence from the aviation industry, Journal of Air Transport Management 50 2016 62–70. doi:10.1016/j.jairtraman.2015.10.003.
[41] M. Tarafdar, Q. Tu, T. S. Ragu-Nathan, Impact of Technostress on End-User Satisfaction and Performance, Journal of Management Information Systems 27 (2010) 303–334. doi:10.2753/mis0742-1222270311.
[42] C. Maier, S. Laumer, C. Weinert, T. Weitzel, The effects of technostress and switching stress on discontinued use of social networking services: A study of Facebook use, Information Systems Journal 25 (2015) 275–308. doi:10.1111/isj.12068.
[43] M. Tarafdar, E. B. Pullins, T. S. Ragu-Nathan, Technostress: negative effect on performance and possible mitigations, Information Systems Journal 25 (2015) 103–132. doi:10.1111/isj.12042.
[44] G. La Torre, V. De Leonardis, M. Chiappetta, Technostress: how does it affect the productivity and life of an individual? Results of an observational study, Public Health 189 (2020) 60–65. doi:10.1016/j.puhe.2020.09.013.
[45] K. Wang, Q. Shu, Q. Tu, Technostress under different organizational environments: An empirical investigation, Computers in Human Behaviour 24 (2008) 3002–3013. doi:10.1016/j.chb.2008.05.007.
[46] M. Salo, H. Pirkkalainen, T. Koskelainen, Technostress and social networking services: Explaining users' concentration, sleep, identity, and social relation problems, Information Systems Journal 29 (2019) 408–435. doi:10.1111/isj.12213.
[47] B. B. Arnetz, C. Wiholm, Technological stress: Psychophysiological symptoms in modern offices, Journal of Psychosomatic Research 43 (1997) 35–42. doi:10.1016/s0022-3999(97)00083-4.
[48] R. Riedl, On the biology of technostress: literature review and research agenda, the DATABASE for Advances in Information Systems 44 (2012) 18–55. doi:10.1145/2436239.2436242.
[49] E. Mellblom, I. Arason, L. Gren, R. Torkar, The Connection Between Burnout and Personality Types in Software Developers, IEEE Software 36 (2019) 57-64. doi:10.1109/MS.2019.2924769.
[50] S. Sarker, S. Sarker, Exploring Agility in Distributed Information Systems Development Teams: An Interpretive Study in an Offshoring Context, Information Systems Research 20 (2009) 440–461. doi:10.1287/isre.1090.0241.
[51] P. Singh, D. Suar, M. P. Leiter, Antecedents, Work-Related Consequences, and Buffers of Job Burnout Among Indian Software Developers, Journal of Leadership & Organizational Studies 19 (2012) 83–104. doi:10.1177/1548051811429572.
[52] L. Williams, A. Cockburn, Agile Software Development: It? s about Feedback and Change, Computer 36 (2003) 39–43. doi:10.1109/mc.2003.1204373.
[53] R. Florea, V. Stray, Software tester, we want to hire you! An analysis of the demand for soft skills, in: J. Garbajosa, X. Wang, A. Aguiar (Eds), Agile Processes in Software Engineering and Extreme Programming, Springer, Cham, 2018, pp. 54–67. doi:10.1007/978-3-319-91602-6_4.
[54] L. Wallace, M. Keil, A. Rai, Understanding software project risk: a cluster analysis, Information & Management 42 (2004) 115–125. doi:doi.org/10.1016/j.im.2003.12.007.
[55] M. Wiesche, Interruptions in Agile Software Development Teams, Project Management Journal 52 (2021) 210–222. doi:10.1177/8756972821991365.

[56] I. S. Stjerne, J. Söderlund, D. Minbaeva, Crossing times: Temporal boundary-spanning practices in interorganizational projects, International Journal of Project Management 37 (2019) 347–365. doi:10.1016/j.ijproman.2018.09.004.
[57] T. D. LaToza, G. Venolia, R. DeLine, Maintaining mental models: a study of developer work habits, in: Proceedings of the 28th International Conference on Software Engineering, Association for Computing Machinery, New York, NY, 2006, pp. 492–501. doi:10.1145/1134285.1134355.
[58] A. Rezvani, P. Khosravi, Emotional intelligence: The key to mitigating stress and fostering trust among software developers working on information system projects, International Journal of Information Management 48 (2019) 139–150. doi:10.1016/j.ijinfomgt.2019.02.007.
[59] M. Salo, M. Makkonen, R. Hekkala, The Interplay of IT Users' Coping Strategies: Uncovering Momentary Emotional Load, Routes, and Sequences, MIS Quarterly 44 (2020) 1143–1175. doi:10.25300/misq/2020/15610.
[60] B. Edvardsson, I. Roos, Critical incident techniques: Towards a framework for analysing the criticality of critical incidents, International Journal of Service Industry Management 12 (2001) 251–268. doi:10.1108/EUM0000000005520.
[61] B. E. Andersson, S. G. Nilsson, Studies in the reliability and validity of the critical incident technique, Journal of Applied Psychology 48 (1964) 398–403. doi:10.1037/h0042025.
[62] W. W. Ronan, G. P. Latham, The reliability and validity of the critical incident technique: A closer look, Studies in Personnel Psychology 6 (1974) 53–64.
[63] A. N. Islam, Sources of satisfaction and dissatisfaction with a learning management system in post-adoption stage: A critical incident technique approach, Computers in Human Behavior 30 (2014) 249–261. doi:10.1016/j.chb.2013.09.010.
[64] M. L. Meuter, A. L. Ostrom, R. I Roundtree, M. J Bitner, Self-Service Technologies: Understanding Customer Satisfaction with Technology-Based Service Encounters, Journal of Marketing 64 (2000) 50–64. doi:10.1509/jmkg.64.3.50.18024.
[65] A. Serenko, O. Turel, Rigor and Relevance: The Application of the Critical Incident Technique to Investigate Email Usage, Journal of Organizational Computing and Electronic Commerce 20 (2010) 182–207. doi:10.1080/10919391003711050.
[66] M. Salo, L. Frank, User behaviours after critical mobile application incidents: the relationship with situational context, Information Systems Journal 27 (2017) 5–30. doi:10.1111/isj.12081.
[67] E. Peer, L. Brandimarte, S. Samat, A. Acquisti, Beyond the Turk: Alternative platforms for crowdsourcing behavioral research, Journal of Experimental Social Psychology 70 (2017) 153–163. doi:10.1016/j.jesp.2017.01.006.
[68] P. B. Lowry, J. D'Arcy, B. Hammer, G. D. Moody, "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, The Journal of Strategic Information Systems 25 (2016) 232–240. doi:10.1016/j.jsis.2016.06.002.
[69] P. Rodríguez, A. Haghighatkhah, L.E. Lwakatare, S. Teppola, T. Suomalainen, J. Eskeli, T. Karvonen, P. Kuvaja, J. M. Verner, M. Oivo, Continuous deployment of software intensive products and services: A systematic mapping study, Journal of Systems and Software 123 (2017) 263–291. doi:10.1016/j.jss.2015.12.015.
[70] M. Hölzl, A. Rauschmayer, M. Wirsing, Engineering of software-intensive systems: State of the art and research challenges, in: M. Wirsing, J. P. Banâtre, M. Hölzl, A. Rauschmayer (Eds), Software-Intensive Systems and New Computing Paradigms: Lecture Notes in Computer Science, Springer, Berlin, 2008, pp. 1–44. doi:10.1007/978-3-540-89437-7_1.
[71] B. L. Berg, Qualitative Research Methods for the Social Sciences, 5th. ed., Pearson Education, Boston, MA, 2004.
[72] M. D. Myers, Qualitative Research in Business and Management, 3rd. ed., Sage Publications, London, 2019.
[73] J. L. Gogan, M. D. McLaughlin, D. Thomas, Critical Incident Technique in the Basket, in: Proceedings of the Thirty Fifth International Conference on Information Systems, Association for Information Systems, Auckland, 2014, pp. 1–18.
[74] R. Ayyagari, Impact of information overload and task-technology fit on technostress. in: Proceedings of the Southern Association for Information Systems Conference, Association for Information Systems, Atlanta, GA, 2012, pp. 18–22.

[75] L. A. Kappelman, R. McKeeman, L. Zhang, Early Warning Signs of IT Project Failure: The Dominant Dozen, Information Systems Management 23 (2006) 31–36. doi:10.1201/1078.10580530/46352.23.4.20060901/95110.4.
[76] J. Verner, J. Sampson, N. Cerpa, What factors lead to software project failure?. in: Second International Conference on Research Challenges in Information Science, IEEE, Morocco, 2008, pp. 71–80. doi:10.1109/RCIS.2008.4632095.
[77] G. Baxter, I. Sommerville, Socio-technical systems: From design methods to systems engineering, Interaction with Computers 23 (2011) 4-17. doi:10.1016/j.intcom.2010.07.003
[78] M. D. Myers, M. Newman, The qualitative interview in IS research: Examining the craft, Information and Organization 17 (2007) 2-26. doi:10.1016/j.infoandorg.2006.11.001

Appendix A: Questionnaire questions regarding the critical incident using IT

Burdensome/stressful experience

Please take a moment to recall **an exceptionally burdensome/stressful experience** related to the **use of technology or software in your work.**

You can take a few minutes to recall. This time is allowed for in the duration of the survey.

Technology and/or software that was involved in the experience:

[open question]

Please describe the burdensome/stressful experience in as much detail as possible in your own words:
[open question]

What exactly made the experience so burdensome/stressful? (e.g., what were you attempting to do with the technology/software and how was it related to your work tasks?)
[open question]

For how long did the experience last?
- Less than one hour
- 1–12 hours
- 13–24 hours
- 1–7 days
- 1–4 weeks
- Over one month
- I don't know

How long ago did the experience occur?
- Less than two weeks ago
- 2–4 weeks ago
- 1–3 months ago
- 4–12 months ago
- Over one year ago
- I don't know

How would you evaluate the strain caused by the experience?
- Very low
- Fairly low
- Moderate
- Fairly high
- Very high
- I don't know