Emotional labor of software engineers

Citation for published version (APA):
Abstract—The concept of emotional labor, introduced by Hochschild in 1983, refers to the “process by which workers are expected to manage their feelings in accordance with organizationally defined rules and guidelines”. For instance, judges are expected to appear impartial, nurses—compassionate and police officers—authoritative. While software development has been traditionally stereotyped as a nerdy “lone wolf” job less likely to induce emotional labor, nowadays software developers become more and more social, on the one hand, and are subject to increasing amount of behavioral expectations, e.g., formulated as codes of conduct.

In this position paper we stress that software developers are subject to emotional labor, envision how emotional labor can be identified based on emotion detection techniques applied in software engineering, suggest possible antecedents and consequences of emotional labor and discuss interventions that can be designed to address the challenges of emotional labor.

I. INTRODUCTION

Software complexity is not solely of technological nature but also defined by people and processes. This means that special attention has to be dedicated to well-being and job satisfaction of people involved in creation of software.

The concept of emotional labor, introduced by Hochschild in 1983 [1], [2], refers to the “process by which workers are expected to manage their feelings in accordance with organizationally defined rules and guidelines”. For instance, judges are expected to appear impartial, nurses—compassionate and police officers—authoritative.

Software development has been traditionally stereotyped as a nerdy “lone wolf” job less likely to induce emotional labor [3]. However, nowadays software developers become more and more social [4] and are expected to more and more communicate with their team mates. Moreover, indirect evidence of emotional labor of software developers is abound. Already in 1991 Riedl et al. [5] reported that when debugging experienced developers manage their emotional display, e.g., by “appearing puzzled and confused, if necessary” to arouse interest of their fellow developers if those might provide help with the debugging task. This can be seen as an example of surface acting, notion closely related to the emotional labor when “an employee changes his or her verbal, facial, and bodily expression of emotions without modifying his or her underlying feelings” [6]. Furthermore, the same study of Riedl et al. [5] stresses that this behavior does not come naturally, should be learned and not learning is experienced as clumsiness and a sign of lack of a novice. A more recent example of an organizational rule prescribing emotional behavior is the Contributor Covenant1 the most popular code of conduct on GitHub [7], the major platform open-source software development. Examples of positive behavior encouraged by the Contributor Covenant include “gracefully accepting constructive criticism” and “showing empathy towards other community members”, i.e., to suppress negative emotions that might have been triggered by criticism and amplify positive emotions towards the colleagues. Codes of conduct in open-source projects are experienced as problematic by certain software developers as witnessed by the opposing efforts known as “No Code of Conduct”2. Finally, exhaustion related to emotional labor has been shown to be one of the most important variables explaining IT career abandonment [8].

While attention to emotions expressed by developers is growing within the software engineering research community, the existing literature suggests the problem of emotional labor of software developers is understudied: it has gained limited attention from applied psychologists working on emotional labor due to the aforementioned stereotyping [9], [10], and has not been studied by software engineering researchers.

II. BACKGROUND

A. Emotional labor

Numerous studies have related emotional labor to such outcomes as employee well-being, e.g., job satisfaction [3] and burnout, as well as to organizational well-being, e.g., interpersonal performance and task performance [11].

Morris and Feldman [12] operationalize emotional labor along four dimensions: frequency of emotional display, attentiveness to required display rules, variety of emotions required to be expressed and emotional dissonance, i.e., “the conflict between genuinely felt emotions and emotions required to be displayed” [13]. In particular, emotional dissonance has been reported to have a strong and consistent relation with work exhaustion and job satisfaction [14]. More recent meta-analysis of 95 studies of emotional labor [15] confirmed this observation and further stressed positive correlation of emotional dissonance with emotional exhaustion, depersonalization, psychological strain, and psychosomatic complaints. Furthermore, the authors observed that surface acting correlates with the same variables. In an additional meta-analysis study of 105 studies Kammeyer-Mueller et al. [6] concluded that stress/exhaustion levels were most substantially related to perceived negative display rules, i.e., perceived requirements to suppress negative emotions, while for job satisfaction there was a substantial negative relationship with surface acting.


2https://github.com/domgetter/NCoC
B. Emotional labor and software engineers

Software development has been traditionally seen as a job with fewer interpersonal requirements [3] and, therefore, less likely to induce emotional labor. Not surprisingly software developers are absent from the Hochshild’s list of occupations most calling for emotional labor [1] that has influenced the emotional labor studies in the following years [16].

Several studies of emotional labor of software developers tend to lump them together with other kinds of IT professionals such as managers and support personnel [17], [18]. The studies show that for the IT professionals emotional dissonance predicts work exhaustion better than traditional predictors such as perceived workload; moreover, job satisfaction is influenced by work exhaustion and influences turnover intentions [17].

A complementary line of research focuses on software developers: the study of Rutner et al. distinguishes between different job types within IT [9], and of Gunsel targeted software developers [10]. Rutner et al. show that “perceptions of positive display rules and levels of political skill differed by job type, but that perceptions of negative display rules, surface acting and deep acting did not”, justifying individual studies of software developers as opposed to other IT job types [9]. Furthermore, the authors state that “programmers who, like other IT/IS professionals feel they should suppress negative emotional displays at work, also recognize the expectation to express positive emotions” [9]. Gunsel has studied the relation between emotional labor of the developers and the resulting quality of software and observed “a positive relationship between the variety of emotions displayed during the projects, operational effectiveness and flexibility”, while “emotional dissonance is found to be negatively associated with flexibility and responsiveness” [10].

C. Emotions and Software Engineering

While emotional labor has rarely been studied in the context of software engineering (Section II-B), the broader topic of the study of emotions expressed by software engineers has recently gained significant attention from the software engineering research community [19], [20], [21], [22], [23],[24], [25], [26], [27], [28], [29].

D. Shortcomings of the existing approaches

The literature overview presented above suggests that there is a gap between the existing studies of emotional labor, on the one hand, and studies of emotions in software engineering, on the other hand, beyond the obvious differences in the target populations. First of all, while in the software engineering realm attention is being predominantly dedicated to detection of natural emotions felt, e.g., by analyzing texts created during the software development process, studies of emotional labor are mostly based on surveying subjects, i.e., focus on the emotional labor perceived or rarely on emotions elicited [30]. Second, software engineering studies focus on emotions as experienced at a given moment, hic et nunc, studies of emotional labor focus on broader concepts related to personality such as dispositional affects [6] or political skill [9], or relatively extended time periods such as work shifts [31], [32] or their parts [33]. The momentary approach of Gabriel and Diefendorff is exceptional in this sense [30].

III. EMOTION LABOR OF SOFTWARE DEVELOPERS

In this position paper we stress that software developers are subject to emotional labor, envision how emotional labor can be measured based on emotion detection techniques applied in software engineering, suggest possible antecedents and consequents of emotional labor and discuss interventions that can be designed to address the challenges of emotional labor.

A. Identification of emotions

We start by discussing detection of emotions expressed by software engineers. As opposed to the existing techniques we aim at the momentary detection of the emotional dissonance as a gap between the emotion felt and emotion expressed. We investigate two groups channels used to communicate expressions: biometric channels that are more likely to reflect emotions genuinely felt and textual channels that are more likely to reflect emotions required to be displayed.

The first group of channels are physical reactions of the human body that can be measured by biometric devices [34]. Common biometric measurement techniques are electroencephalography (EEG), galvanic skin response measurement (GSR) and measurements obtained through an eye-tracker and face recognition techniques. EEG can measure valence of emotions, i.e., positive or negative, but also such cognitive processes as attention and perception [35]. Measuring cognitive processes is important for understanding the impact of emotional labor on job satisfaction and work exhaustion as they are likely to provide important confounding factors in the statistical models. Similarly, GSR can be used to measure arousal, emotional intensity and the direction of emotion [35]. Eye tracker can be used to fatigue and relaxation. Similarly to the cognitive processes, fatigue and relaxation can be expected to affect job satisfaction and work exhaustion. Tools for emotion detection based on face recognition [36] are capable of detecting such emotions as anger, contempt, disgust, fear, joy, sadness and surprise. We expect EEG, GSR and eye-tracker channels to be less regulated and therefore more adequately representing the emotions felt by the developers. Facial expressions are in general more regulated: non-surprisingly, Ekman and Frisen call the face “the major nonverbal liar’” [37]. However, since software developers work in virtual teams we do not expect the emotional display rules to affect the facial expressions. Furthermore, despite the general deceptiveness of the facial expression micro-facial displays can provide cues as to the authentic emotion felt by an individual [37]. Since application of biometric measurements might be error-prone due to their invasive character and sensor drift/noise it will be carried out in the controlled experiment setting.

The second group of channels are the texts produced by software developers such as code review comments, issue tracker reports, or questions and answers on Q & A platforms. Several techniques have been proposed for detection of emotion in
software engineering texts; from a broad range of emotions (anger, fear, joy, love, sadness and surprise) in the work of Ortu et al. [26] to techniques focusing on detection of anger and its direction in the work of Gachechiladze et al. [27]; from models of discrete emotions [26], [27] to continuous valence-arousal-dominance model by Mäntylä et al. [22].

As opposed to the emotions sensed by means of biometric instruments, emotions as expressed in texts are communicated towards the interlocutors, and therefore are more likely to be subject to regulation either explicit via codes of conduct [7] or via the perceived notion of professional conduct. Therefore, emotions discovered by analyzing software engineering texts are more likely to reflect emotional display rules induced by the project, and therefore, depend on the project culture, e.g., whether it is formal or not [38], whether insults are acceptable or not [39].

Existing emotions questionnaires such as PANAS [40], LEAS [41], [42], DEQ [43] can be used for validation.

B. From emotions to emotional labor

Next we plan to relate the individual measurements obtained by means of techniques developed in Section III-A to the conceptual framework of emotional labor, i.e., to propose an operationalization of the emotional labor constructs in terms of the emotions identified.

As a basis for such an operationalisation we consider the four dimensions of emotional labor proposed by Morris and Feldman [12], i.e., (1) frequency of emotional display, (2) attentiveness to required display rules, (3) variety of emotions required to be expressed and (4) emotional dissonance.

Frequency of emotional display has been operationalised as frequency of interactions [12] and further proposed to be measured as the number of interactions with different customers [44] or team mates [10]. Since software engineering texts have one or more addressees (e.g., individuals involved in reviewing a code change, or fixing a bug), ability to detect emotion in software engineering envisioned in Section III-A allows one also to quantify frequency of the emotional display. A similar argument can be made of the duration of emotional display, one of the components of the attentiveness to required display rules. The second component of the attentiveness to required display rules, i.e., intensity of the emotional display is related to the arousal component of the valence-arousal-dominance model of emotions [28], [29].

Variety of emotions required to be expressed calls for techniques capable of detecting different kinds of emotions, e.g., six discrete emotions detected by Ortu et al. [26] or the valence-arousal-dominance-based detection proposed Mäntylä et al. [22]. Results of the variety measurement can be compared with those obtained by the validated questionnaire [45].

As suggested in Section III-A emotional dissonance can be seen as a discrepancy between the emotion conveyed through the biometric channels and through the textual ones. To validate the emotional dissonance discovered in this way we would like to build on the existing psychological scales. However, emotional dissonance scale proposed by Cheung and Tang [46] is not suited for momentary evaluation. Therefore, we take leaf from the book of Gabriel and Diefendorff [30] and for continuous rating, i.e., we will record the session, replay the recording to the participants and ask them to rate to what extend did they feel emotional dissonance at a given moment. Once emotional dissonance has been validated at the level of a single moment, one should investigate how those momentary values of emotional dissonance can be aggregated to extended periods of time (cf. aggregation of software metrics [47], [48]). The aggregated values can be then compared with the measurements on the emotional dissonance scale [46].

C. Antecedents and consequents of emotional labor in software engineering

Identified emotional labor situations should lead to understanding antecedents and consequents of emotional labor in software engineering, i.e., aspects of developers’ personalities, roles played, project organization etc that can impact different aspects of emotional labor and the ways emotional labor affects software products created by developers as well as developers’ communities. This group of activities can convert the insights obtained so far into actions that can support software developers in their daily work.

1) Antecedents: Existing studies of emotional labor antecedents concerned such as personality variables extraversion and agreeableness [3] and positive/negative affectivity [6], and such demographics as gender [49], age [50], [51], race [52], [51] and national culture [53], [54]. All these variables can be expected to play a role also in the software engineering context. However, we focus on specifics of the software engineering task and keep personality and demographic variables as control. Specifically, we study the impact of a role played by a project contributor. Indeed, code reviewers, in particular, core code reviewers [55], [56], and bug triage masters can be expected to be involved in more interactions and more intensive interactions than regular developers. The same is likely to hold for project leaders and influential developers, closer to the center of the onion model [57], as well as for frequent contributors as opposed to the occasional ones [58]. Similarly, to roles we differentiate between analytic and synthetic software development tasks [59], e.g., identification of the bug cause vs. designing a bug fix, and study the impact of the kind of the task on emotional labor.

Furthermore, there seems to be little attention to the impact different organisational types can have on the emotional display rules, and, therefore, on the emotional labor. In software engineering, however, special attention has been given to identification of different organisational types, both in company-based and in open-source projects [38], [60]. We expect that distinguishing between different organisational types can help us to understand the differences between emotional display rules induced in different software development projects. In particular, we expect “community smells” [61], i.e., communication and collaboration anti-patterns reflecting undesirable community characteristics such as knowledge concentration or lack of communication, to incur negative emotions on the individuals involved and might increase emotional dissonance.
Finally, working on different kinds of artifacts might have different impact on the emotional state of the developer. Gungel [10] has shown the system complexity has a moderating effect on the relation between emotional labor and software quality. However, since software developers’ tasks involve working with software artifacts one could argue that the relation between system complexity and emotional labor is more intricate, as reviewing or modifying more complex code might not only be perceived as more intellectually challenging but also be expected to elicit more intensive emotions.

2) *Consequents:* We distinguish between two groups of consequents of interest: those related to the developers’ community and to quality of the software produced.

First, earlier studies relating emotional labor to such consequents as job satisfaction, organizational attachment/turnover intention, emotional exhaustion [3], [15], [62] should be replicated on software developers. Furthermore, models based on emotional labor should be compared against alternative models for turnover [63], [64], [65], [66], [67], [68], [69], [70] and burnout [22], [71] designed for software engineers.

We also plan to investigate the impact of emotional labor on the software quality: preliminary results of Gungel [10] suggest that such aspects of emotional labor as attentiveness, variety of emotions and emotional dissonance affect software quality as perceived by the developers with the project complexity moderating this relation. We would like to go beyond the perceived software quality to more objective measures of software quality such as the issue fixing time (cf. the study of Ortu et al. on affectiveness vs. issue fixing time [26] and of Jongeling et al. on the impact of the sentiment analysis tools in this context [72]). Furthermore, we would like to obtain a more refined understanding of the impact of emotional labor at individual activities such as introduction and removal of bugs, code smells and technical debt [73], [74], [75], [76]. We expect this relation between emotional labor and code quality since similar relations between code quality and single-out community factors have been established in the past, e.g., socio-technical congruence [77], truck-factors [78], and newcomer contributions vs. bad smells [74].

Finally, we plan to study the impact of emotional labor on developers’ productivity. The link between emotion and developers’ productivity has been suggested in the past [28], [22]. Our previous work covered other variables affecting productivity [66]: they should be included in statistical modeling.

### D. Designing interventions

Based on the understanding of the antecedents and the consequents of emotional labor of software engineers one can design *appropriate interventions*. Interventions can take place at the level of the project, of the individual developer, of their tasks and finally, at the level of the artifacts created. Some of the interventions can be supported by bots [79].

1) *Project:* If our expectation that community smells induce emotional labor then the corresponding mitigation techniques identified by Tamburri et al. [61], e.g., establishing a shared knowledge base (“social wiki”) or appointing certain developers to act as culture conveyors integrating previously disconnected sub-communities. Furthermore, recruitment policies can be designed or adapted to select candidates with self-expression congruent with emotional requirements [80]; if self-expression can at least partially be detected through software engineering texts as suggested in Section III-A such a congruence check can be integrated in the Social-Web candidate assessment advocated by Capiluppi et al. [81]. Finally, on a larger scale projects might consider changing their organisational type [38], [60], e.g., by opting for a more/less formal communication style.

2) *Individual developer:* Several emotional labor researchers suggested a possibility of offering trainings for emotion regulation, specifically for deep acting [11]. However, there are concerns related to hidden costs of deep acting [1] and to differences between deep acting learned through training as opposed to deep acting emerging naturally [11]. An alternative approach might be provided through implementation of mindfulness techniques [82], that have been shown to lead to significantly less emotional exhaustion and more job satisfaction. Application of mindfulness techniques is particularly promising since it has been recently successfully applied in the software engineering context as well [83].

3) *Task:* Different software development tasks can be expected to induce different kinds of emotional labor, e.g., the triage master is likely to have a higher frequency of emotional display. Similarly to recruitment task assignment should also take the risk of emotional dissonance into account, e.g., the triage master should not only be technically proficient and aware of responsibilities of individual subteams and developers, but also be capable of managing the aforementioned frequent emotional display. The same argument can be made for, e.g., the (core) code reviewers: one might wonder whether a recently observed high turnover of the core code reviewers [56] can be attributed to emotional labor.

4) *Artifacts:* Finally, if indeed as suggested in Section III-C1 maintaining or reviewing more complex systems induces more intensive emotions, this can be used as an additional argument supporting efforts reducing system complexity such as reengineering or refactoring.

### IV. Conclusions

In this position paper we have discussed the notion of emotional labor as studied in organizational psychology and argued that emotional labor is also experienced by software developers. We have outlined the ways emotional labor can be identified based on emotion detection techniques already applied in software engineering, as well as suggested possible antecedents and consequents of emotional labor. Based on the identified antecedents and consequents of emotional labor appropriate interventions can be designed.

### REFERENCES

