2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)

Socio-Technical Work-Rate Increase Associates With Changes in Work Patterns in Online Projects

Farhana Sarker
University of California, Davis
fasarker@ucdavis.edu

Bogdan Vasilescu
Carnegie Mellon University
vasilescu@cmu.edu

Kelly Blincoe
University of Auckland
k.blincoe@auckland.ac.nz

Vladimir Filkov
University of California, Davis
vfilkov@ucdavis.edu

***Abstract*—Software developers work on a variety of tasks ranging from the technical, e.g., writing code, to the social, e.g., participating in issue resolution discussions. The amount of work developers perform per week (their work-rate) also varies and depends on project needs and developer schedules. Prior work has shown that while moderate levels of increased technical work and multitasking lead to higher productivity, beyond a certain threshold, they can lead to lowered performance.**

Here, we study how increases in the short-term work-rate along both the technical and social dimensions are associated with changes in developers' work patterns, in particular communication sentiment, technical productivity, and social productivity. We surveyed active and prolific developers on GitHub to understand the causes and impacts of increased work-rates. Guided by the responses, we developed regression models to study how communication and committing patterns change with increased work-rates and fit those models to large-scale data gathered from traces left by thousands of GitHub developers. From our survey and models, we find that most developers do experience work-rate-increase-related changes in behavior. Most notably, our models show that there is a sizable effect when developers comment much more than their average: the negative sentiment in their comments increases, suggesting an increased level of stress. Our models also show that committing patterns do not change with increased commenting, and vice versa, suggesting that technical and social activities tend not to be multitasked.

I. Introduction

Software developers who work in distributed, online projects often work on a variety of tasks [1], [2]. Some of those are technical, like coding, testing, and documenting code. Others are of the coordinating, or social, variety, like commenting on issues, agreeing on directions, responding to calls for bug fixing, etc. Both kinds of tasks are essential in healthy, multi-developer software projects, and they both count toward the work done per unit of time, or *work-rate* of developers. It is well-known that if the developer work-rate gets too high, it can contribute to work-related stress and lead to negative effects [3]–[5].

Figs. 1 and 2 show the weekly rates of commits and comments for a random prolific developer on GitHub. Clearly, the developer's contribution rates can frequently exceed 1 standard deviation above the developer's average contribution rates in both commits and comments. In general, keeping the work-rate in the "safe-zone" can result in healthier work environments and happier individuals and teams. Awareness of historical work-rates can help ensure work-rates do not reach unsafe levels.

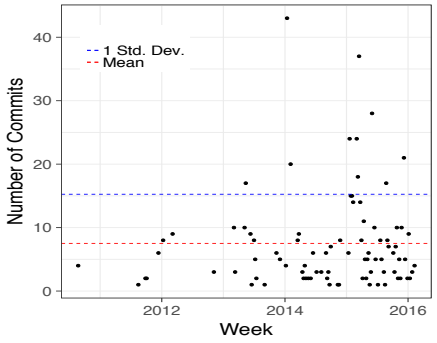

Fig. 1. Example number of commits per week for a prolific developer.

Traditional lines-per-day measures of work aggregate, to an extent, the work-rate of technical tasks. While coordination tasks figure in there too by association, it is not simple to unravel their contribution to those measures [2]. This is more so during some development stages, especially at early stages in the development of a feature, when coordination tasks may dominate a developer's time compared to at other stages, where development and testing dominate. In addition, many developers work on multiple projects at a time, and the tasks related to the context-switching overhead incurred contribute to their aggregate work-rates [1], [6]–[8]. Prior work has shown that multitasking has significant costs and may result in lowered technical performance if a work-rate threshold is crossed, mainly due to patterns in the task-switching overhead [1]. That work did not consider the multiple components involved in one's work; it only considered technical activity within multitasking developers' projects.

Thus, the many different tasks and the number of projects in which developers are involved should all be considered when calculating their work-rates. Knowing how the different work-rate dimensions interact and associate with observable developer work behavior can aid in recognizing when the work-rates indicate overload and/or may affect others. This knowledge can enable people to ameliorate or even preempt such situations. So, given that developers multitask on at least two types of tasks, technical and coordination, and experience varied rates of work, in what observable ways does their behavior change as their work-rates increase? And how can we use such data to model work patterns indicative of work-rate-related overload in software developers?

1558-1225/19/$31.00 ©2019 IEEE
DOI 10.1109/ICSE.2019.00099

Here, we study how developers' communication and committing behaviors change with significant work-rate increase. Given a large enough set of developers' commit and communication traces, the work-rates should form a smooth distribution, enabling analysis of its tails, for any sizable sample of developers. Our thesis is that the effects of work-rate changes over time may be identifiable from large amounts of longitudinal trace data of the variety of social and technical activities undertaken by developers. While different developers may have different steady (average) work-rates, significant increases in the work-rates compared to their average rates can observably modify their work behavior.

To show that, we undertake a mixed-methods, qualitative and quantitative study. First, we conducted a survey of GitHub developers to characterize the ways in which developers change their work behavior in response to increased technical and social work-rates, as well as the self-identified determinants of those reactions. The survey helped us identify relevant variables characterizing developers' actions in both the technical and social dimensions. Those included weekly commit and comment numbers, comment types (discussion and code review), comment text length and sentiment (positive, negative, and neutral), as well as measures of context-switching overhead identified in prior studies [1]. Then, using a dataset of developer trace data from a large number of GitHub projects, we built regression models over the above variables to determine how communication and committing behaviors change in response to work-rates that are higher than expected for each developer. We found the following:

- Many developers responded that they experience work-rate-related stress and deal with it in various ways.
- There is a long tail in both the normalized commit and comment rate distributions, indicating that there are more-than-expected number of weeks when developers have much higher-than-average work-rates.
- Notably, we find that when the number of comments increases significantly above a developer's average; the developer's negative comments, on average, increase as well, indicating an increased level of stress.
- Interestingly, we found that committing activities are not affected by increases in communication work-rate, and vice-versa for commenting activities, suggesting mutual independence between them.

II. Theory and Research Questions

In this section, we review relevant theories and prior work on effects of high work-rates and workload and the associated stress on developer performance, and we develop our research questions. Our work fits in the broader context of behavioral software engineering [9].

A. Social Coding Pressures

Developing software has always been a stressful activity, *e.g.,* because of constantly changing requirements, increasing user demands for agility, market pressure, unstable technology, and breaking changes, just to name a few. In a fast-paced, social-coding environment like GitHub, several additional factors may contribute to creating a more stressful environment.

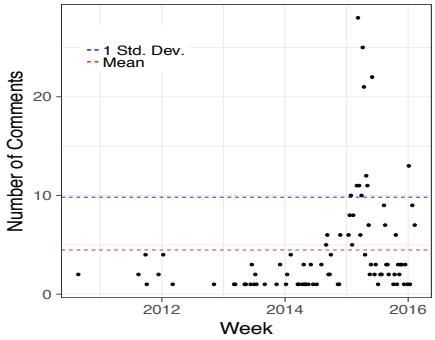

Fig. 2. Example number of comments per week for a prolific developer.

Social-coding sites like GitHub connect developers from around the world into a fast-paced, around-the-sun [10] development environment, enabling more online communication and coordination. A plethora of online communication media, including social media, are being used by software developers in their daily work [2], [11]–[14]; among the top channels for discussing development activities are code hosting sites (*e.g.,* discussions on pull requests and issues), private and public chats, private discussions (email), and discussion groups [2].

Still, despite the benefit of instant communication regardless of distance or time, such quick methods of communication can lead to **communication overload**, which has been associated with stress and diminishing productivity [3], [4], [15]. Communication load accumulates through the increasing number of messages sent and received over online communication tools like email and social media. In social coding, communication load increases still through the constant stream of pull requests and issues, which generate additional discussion. As social expectations often urge users to respond to messages within a socially acceptable timeframe to prevent negative feelings from communicating partners [15], users may develop a sense of social pressure to be constantly available in order to respond to their messages "on time" [4]. In open-source development, unresponsiveness from project maintainers to pull requests by newcomers can negatively affect the latter's motivation and engagement with the project [16]. Reinecke *et al.,* [4] found that the **social pressure to respond** to online messages leads to high levels of Internet multitasking (users concurrently respond to online messages while engaging in other activities) and communication overload, both of which are associated with burnout and anxiety [4]. Studies of communication overload in the workplace also show that people with stronger perceived notions of communication overload are likelier to experience burnout and to be less productive [3].

Also related is the perception that contributing to open-source projects on GitHub is like **being onstage** [17]. Because of the high transparency [18] of everyone's actions enabled by public GitHub profiles, active users experience a clear awareness of the audience for their actions, which influences how they behave. GitHub profiles are also used as signaling

mechanisms [19]–[21]. Users, contributors, and maintainers of open-source (inside the platform) and potential employers (outside the platform) all rely on signals extracted from GitHub activity traces to make rich inferences about GitHub participants and the quality of their work. The **signaling** value of GitHub activity traces to different stakeholders, characteristic of social-coding environments, creates incentives for people to be more active than otherwise and potentially more stressed.

Another dimension of social-coding stress comes from the interconnected nature of modern OSS work. As more code becomes available and is being reused, projects tend to form complex networks of interdependencies, *i.e.,* ecosystems [22], in which local changes can generate **network effects**, and stress, downstream. E.g., prior work has found that in such environments, issue reports often get tangled [23], [24], and breaking changes induce costs on users downstream [25].

The signaling value of high levels of GitHub activity and interconnected nature of modern-day OSS development create an environment ripe for people to multitask and switch focus often between different projects and activity types [1]. Yet, context switching between tasks incurs cognitive-switching costs for interrupting one task and resuming another [6], [7]. When switching costs are high, **multitasking** leads to mental congestion and decreased productivity; workers become overloaded [5], increasingly slower, and error-prone [8], [26]–[28].

Past studies have found that work-related stress is a growing problem for knowledge workers [29], [30]. Software developers are a subset of this population of people who "think for a living" [31]. Kristensen identified six job characteristics that influence stress: control or influence, predictability, social support, meaning, demands, and reward [32]. The demands of a job are related to both the amount and difficulty of tasks, which associate with the work-rate of an employee. Galbraith argued that organizational culture also influences the stress-levels of employees [33]. Mark *et al.,* also found that work interruptions lead to higher stress [34]. Other studies considered the impact of stress on knowledge workers, finding that stress can cause physical illnesses [35], [36]. Here, we focus on work-rate increase as a stressor in software developers.

B. Our Research Questions

The discussion above suggests that modern open-source software development, especially on social-coding platforms like GitHub, is a high-workload, high-work-rate, high-stress environment. High levels of activity may provide signaling benefits, and opportunities to engage in multiple activities arise naturally and constantly as open-source grows. Yet, attending to high volumes of issues and pull requests can be overwhelming, switching focus too much can be costly, and being constantly onstage can be draining.

We adopt a socio-technical view of open-source software development on GitHub to study how operating in a high-workload, high-work-rate, high-stress environment impacts developer performance along two dimensions: (1) discussions around code and issues and (2) commits to repositories. While prior research on software development productivity has focused mostly on technical contributions to projects, the social aspects of development are just as important. Online communication is an essential part of software development, and the frequency and social transparency that it imposes may cause social pressure, as discussed above. We hypothesize that high increases in work-rate, along both dimensions, are associated with abnormal work patterns, which could indicate stress. In turn, high stress could affect developers' linguistic patterns and committing/commenting behaviors on GitHub.

First, we examine the perceptions of developers on work-related stress and social pressure:

RQ 1: What are the perceived causes and impacts of stress and social pressure?

Guided by the RQ1 results, we investigate divergence from regular work patterns along both dimensions:

RQ 2: How prevalent are higher-than-average rates of commit and communication activities? How prevalent are they per individual?

Then, we model the effects of increased workload on developers' work patterns along each dimension:

RQ 3: What are the covariates of increased work activities that associate with negative or increased communication?

RQ 4: What are the covariates of increased work activities that associate with a higher number of commits?

III. Methodology

We used a mixed-methods approach with a sequential exploratory strategy [37]. To inform our understanding of the causes of social pressure and stress on GitHub, we first surveyed a set of active and prolific developers on GitHub. Based on the responses, we analyzed GitHub activity data via regression modeling to model changes in developers' work patterns in terms of technical and social work-rates.

A. GitHub Developer Dataset

For both our survey and trace data analysis, we used the January 1st, 2018 database dump provided by GHTorrent [38]. We first merged aliases of each existing developer into a single account using a set of heuristics (such as merging aliases with the same email address) [39]. We then chose to consider only active and prolific developers to ensure that the developers who we contacted for the survey were still active and experienced enough to give us reliable responses, and also to ensure that we would have enough developer activity data to conduct a thorough and meaningful empirical analysis of the variance around developers' mean levels of activity. Consequently, we chose a subset of developers who have a span of at least 5 years between their first and last commits and who made at least 500 commits to at least 10 distinct, non-forked, existing repositories, a definition of active and prolific developers that we adopted from prior work on data from GHTorrent [1]. To filter out rewritten commit histories with potentially incorrect timestamps, we considered only commits made between 2008 (the year GitHub went online) and January 1st, 2018 (the last date of the dump). Our final dataset consisted of 57,977 developers. 31.7% of them had at least 2 aliases, and the maximum number of aliases for a single developer was 13.

B. Qualitative Analysis: Survey

To develop our survey, a pilot survey was sent to 350 randomly selected developers from a set of prolific developers from a 2016 GHTorrent database dump. We filtered the developers similarly as in Section III-A. The pilot survey obtained 45 responses (a 13% response rate). It contained many open-ended questions and some closed-set questions where participants could also write-in their own responses. Some sample open-ended questions that we asked in the pilot survey include: How do you become aware of the stress of others on your team? In what ways are you impacted by the stress of others on your team? What causes you work-related stress? What do you do to try to reduce your stress? The closed-set responses were developed from findings of previous work [1], [4], but we allowed participants to provide their own responses so that we could develop a more comprehensive survey. Using the responses to these open-ended questions and the write-in responses, we used Thematic Content Analysis [40] to identify a closed-set of options for each question in our final survey to allow participants to rate each item using a Likert scale.

The thematic content analysis of the pilot survey data was done by one of the authors of this paper together with a clinical psychologist. It allowed us to create a list of options for these questions in the final version of our survey. For example, one participant in the pilot survey said that a cause of work-related stress was "the feeling of being behind on work, especially if I have verbally offered to do that work." This (and similar responses) resulted in having "time pressure" as a response option to this question in our final survey. The final set of options for each question is shown in the figures which summarize the results for each question in Section IV. For all questions with a closed-set of responses, we also allowed participants to select "other" and write in their own response.

We sent the final survey to 2,000 GitHub users randomly selected from the set of prolific developers described in Sec. III-A. We obtained 465 responses (a 23% response rate). All questions were optional. The survey included multiple-choice and Likert-scale questions. We asked the developers questions related to work-related stress and work-related social pressure. Respondents were asked to only consider the GitHub projects they currently contribute to when answering all questions. Participants had an average of 15 years of development experience and 7 years of experience with GitHub.

C. Quantitative Analysis: Online Activity Trace Data

We conducted a quantitative analysis on the technical and social trace activity of 3,000 developers randomly selected from the set of developers described in Section III-A. Before selecting the 3,000 developers, we first searched for keywords like *build, auto, exporter*, and *bot* within users' names, logins, and emails in order to find and remove bots and their activity. After manually inspecting the search results, we removed 21 users that were obviously bots. The 21 bots had contributed about 552,237 comments and about 2,070,653 commits.

1) Measuring Technical Activity: We consider commits as indicators and measures of technical contributions. Here we consider commits made between 2008 and 2018 to non-forked, existing repositories. The developers in our dataset contributed a total of 6,245,569 commits to 151,262 repositories.

2) Measuring Social Activity: Social activity within repositories on GitHub exists mainly in two forms: **discussion comments** and **code review comments**. Discussion comments exist on issues and pull requests. Code review comments exist on commits, which may be associated with pull requests or not. Code review comments differ from discussion comments in that a code review comment is made on specific lines of code within a Git diff, whereas a discussion comment is made within a single thread that is associated with either an issue or pull request. Code review comments made on individual commits not associated with pull requests have the additional option of being made on the entire commit rather than on specific lines within the Git diff. We consider all discussion and code review comments as developers' social activity.

To retrieve comments, we queried both GHTorrent's public MongoDB database and the GitHub API, as some data was missing from the MongoDB database [41]. From GHTorrent's MongoDB database, we retrieved about 98% of all developers' comments. We then queried the GitHub API using the PyGithub Python library to retrieve the remainder of the comments, raising our percentage of retrieved comments to about 99.7%. The rest of the comments appeared to have been deleted. Furthermore, we filtered the comments to take only those made between 2008 and 2018 to non-forked, existing repositories. In total, our final comment dataset consisted of 1,777,496 comments made to 68,516 repositories. The distribution of comment types are as follows: 47.5% discussion comments made on issues, 32.4% discussion comments made on pull requests, 17.9% code review comments made on commits within pull requests, and 2.2% code review comments made on commits not associated with pull requests.

3) Sentiment Analysis of Developers' Comments: Sentiment analysis has often been used to study the emotional polarity of pieces of text within the software engineering domain [42]–[44]. We use the sentiment of a comment as one method of measuring developers' feelings/potential stress.

Well-known sentiment analysis tools are often trained on social media corpora, making them unsuited for classifying text within the software engineering domain [45]. Comments made on social media outlets like Twitter are not representative of developers' technical conversations, and tools that are built on top of such corpora are known to misclassify software engineering text. In fact, a study of the effects of common, off-the-shelf tools like NLTK, SentiStrength, Alchemy, and Stanford NLP on technical text show that their classifications of software engineering text tend to disagree with human-labeled sentiments and also tend to disagree with each other [45]. Further studies show that existing tools tend to have an unrepresentative, negative bias when classifying technical texts [46], since some technical phrases such as "kill a process" and "create a patch" have negative connotations in layman terms [45], [46]. So, we found it necessary to explore tools that are better-suited to the software engineering domain.

TABLE I
SENTIMENT DISTRIBUTION/COMMENT TYPE (844,366 ISSUE DISCUSSION; 576,578 PR DISCUSSION; 318,189 PR CR; 38,363 COMMIT CR).

Comment Type	% Pos	% Neg	% Neu
Discussion on Issues	24.0	11.5	64.5
Discussion on PRs	32.5	10.0	57.6
CR on PRs	15.4	9.5	75.1
CR on Non-PR Commits	20.3	11.3	68.4
Total	**25.2**	**10.6**	**64.2**

There are few existing software-engineering-specific sentiment analysis tools [47]–[49]. Out of those, we chose to use Senti4SD [49], a post-level sentiment classifier that is trained on a set of about 4,000 Stack Overflow posts, manually-annotated by a group of trained computer science students. To compute the sentiment of a post, Senti4SD extracts a series of features, including lexicon-based features (number of tokens with positive/negative polarity, number of positive/negative emoticons, presence of exclamation marks, etc.), keyword-based features (number of uni-grams and bi-grams, number of elongated words, number of uppercase words, number of slang expressions like 'lol' that represent laughter, etc.), and semantic-based features (cosine similarity between post and prototype polarity vectors). These features were developed using polarity assignments within the SentiStrength lexicon [50]. Furthermore, the semantic-based features derive word meanings from a Distributional Semantic Model built on 20 million Stack Overflow comments, questions, and answers. These extensive strategies for feature extraction allow Senti4SD to significantly reduce the amount of text that is misclassified as negative compared to other tools [49].

Senti4SD has a slight disadvantage in that long posts can be misclassified if they contain higher distributions of excessive words whose calculated polarities do not convey the overall sentiment of the post [49]. To combat the effect of comment length on sentiment classification, we first preprocessed comments by removing portions that were not likely to contribute to the overall sentiment, such as code snippets, HTML tags, and URLs [49], as well as non-ASCII characters. We removed code snippets by using regular expressions to locate sections of comments surrounded by single or triple backticks, which users can use to highlight code segments within their comments. We also used regular expressions to locate and remove HTML tags and URLs. Furthermore, we removed inline references to other comments, which are prepended with the '>' character, placing sentiment analysis focus on the actual comment made. Our preprocessing steps reduced the average length of all comments by about 71.6 characters and reduced the percentage of comments that were classified as negative by about 1.9%.

The final distribution of sentiments within our comment dataset is: 25.2% positive, 10.6% negative, and 64.2% neutral. Table I shows a breakdown of sentiment per comment type, along with the number of comments per type. "PR" stands for "pull request," and "CR" stands for "code review."

Case Study of Negative Comments: We manually analyzed a set of 50 random comments that were classified as negative to better understand their context. We found a variety of different types of negative comments. Here we briefly describe the types and provide an illustrative example for each.

- Negative reviews. *"Given that view_or_value may transfer ownership, I'm very certain you want to say 'const view_or_value<View, Value>&' here."*
- Indicating work will not be done or will be delayed. *"This will have to wait till the weekend because I don't seem to have enough time this week to do a review; sorry"*
- Admitting mistakes. *"Odd, that must have happened when I copied it from one dev checkout to another, sorry about that. Fixing now."*
- Disagreements. *"I also find a bare 'I was told to do so' argument rather difficult to accept"*
- Lack of skills/understanding. *"I'm not sure what to do here, so I just added test data for mdstat from last posts"*

Thus, we find that Senti4SD successfully identified negative comments.

4) Data Corresponding to Threads: To assess the association of comment interdependence with developers' work patterns, we gather two additional measures per comment: the **thread position** of the comment (i.e., the position of the comment within a thread) and the **response time** of the comment.

We consider all discussion comments that were made on the same issue or pull request as belonging to the same thread, and we consider all code review comments that were made on the same Git diff position of the same commit as belonging to the same thread. Furthermore, we consider all code review comments that were made on the same, entire commit (rather than on a Git diff position of the commit) as belonging to the same thread. Thus, if the thread position of a comment is 5, then it was the 5th comment made within its thread.

We consider the response time of a comment to be the time difference between its timestamp and the timestamp of the most recent comment made before it on the same thread, measured in seconds. If a comment is the first comment in its thread, then we consider its response time to be the time difference between its timestamp and the timestamp of when its thread (*i.e.,* its issue, pull request, or commit) was created. We disregarded about 0.2% of all comments which had negative response times (probably due to incorrectly recorded timestamps and/or to later updates to earlier comments).

5) Noting Company Affiliations: To study whether company affiliations affect work patterns, we also determine whether each developer's account is associated with a company. If a developer's "company" section is filled out within the developer's GitHub profile, we consider this developer to be affiliated with a company.

6) Multitasking Measures: Prior work has shown that developers often work on multiple projects in parallel, to the extent that the level of context switching required by high levels of multitasking correlates with decreased productivity and poor performance [1]. Thus, multitasking and context-switching levels can further reveal the amount of workload a

developer is juggling and therefore indicate how much work pressure a developer is under. We also gather weekly multitasking and context-switching measures for each developer.

We adopt three measures of technical multitasking developed in prior work, where contributions are measured in commits ($cmits$): the average number of projects worked on per active day ($days\_active\_cmit$) within a week ($avg\_projs\_cmitted\_per\_day$), the diversity of focus switches among projects worked on within a week ($sfocus\_cmit$), and the diversity of day-to-day focus switches among projects worked on within a week ($sswitch\_cmit$) [1]. $sfocus\_cmit$ ranges between 0 and $log_2 N$, where N is the number of projects worked on during the week. The higher the value, the more equally a developer contributes to each project. $sswitch\_cmit$ uses focus shifting networks (FSNs) [51] to capture the repetitiveness of a developer's day-to-day focus-switching behavior among projects during a week. The lower the value, the less repetitive a developer's focus-switching behavior is from day to day.

Analogously, we define three measures of social multitasking: $avg\_projs\_cmnted\_per\_day$, $sfocus\_cmnt$, and $sswitch\_cmnt$, where contributions are measured in comments ($cmnts$) rather than commits.

7) Regression Modeling: We model how developers' committing and commenting behaviors change as a function of social and technical work-rate increases using hierarchical regression analyses. To gather a longitudinal dataset for this analysis, we first aggregate developers' commit and comment activity in each week that they are active. The resulting values describe developers' work-rates for each week. We then combine this data with the weekly multitasking measures described in Section III-C6.

We remove the top 1% of outliers from a number of variables ($cmits$, $projs\_cmitted$, $cmnts$, $projs\_cmnted$, avg_length, avg_thread_pos, and $avg\_response\_time$) that follow an exponential distribution to avoid potential high-leverage points issues. To avoid extreme values in the fixed effects, we *log* transform the *avg_response_time* variable, which has high values and variance.

Per-Developer Z-Normalization: Our research questions revolve around studying work-rate that is much higher than average. To that end, we calculate the average activity per week for each developer as a baseline, compared to which we track activity deviations in different weeks. We do this by calculating the per-developer average and standard deviation (over all their active weeks), and then for each active week i, z-normalized commit and comment activity:

$$\text{variable\_}z_i = \frac{\text{variable}_i - average(\text{variable})}{st.dev(\text{variable})}.$$

In this way, we can model deviations from the average in units of standard deviation. To identify high-activity weeks, we calculate for each week a binary indicator, or factor: 1 (true) when the variable's value is more than 1 standard deviation away from that developer's average on that variable, *i.e.,* variable_z score is greater than 1, and 0 (false) otherwise, *e.g.,* in Fig. 1, there are 10 weeks during which that developer exhibited a commit-rate more than one standard deviation higher than the developer's average commit-rate.

Below is a list of the z-standardized variable names we use within our models, along with their definitions:

- $\{cmit|cmnt\}\_z$: z-normalized number of commits/comments,
- neg_z: z-normalized number of negative comments,
- $projs\_\{cmit|cmnt\}\_z$: z-normalized number of projects committed to/commented on,
- $days\_activ\_\{cmit|cmnt\}\_z$: z-normalized number of days on which users committed/commented,

Developers who did not have much variation in their values for a certain variable had undefined values for the corresponding z-score values for that variable, *e.g.,* a developer who made 1 comment per week for every week commented will have undefined values for the $cmnt\_z$ variable because the standard deviation of $cmnts$ for each week is 0. We excluded these undefined values from our dataset, removing 152 (5.1% of) users and 24,062 (4.9% of) rows of data from our dataset.

The summary statistics of our final, filtered weekly dataset that contains all weeks where developers commit or comment are shown in Table II. This dataset, which we use for our models, contains 466,858 rows from 2,848 developers.

Linear Mixed-Effect Models: We use the R project library *lmer* that implements *linear mixed-effects regression* [52] (LMER) to measure the relationship between our outcomes (dependent variables) and our explanatory variables of interest, under the effect of various controls. LMER can be seen as an extension of standard ordinary least squares (OLS) linear regression, allowing for the addition of *random effects* on top of standard *fixed effects*. The latter can be interpreted the same way as coefficients in an OLS regression. Random effects are used to model (often sparse) factor groupings that may imply a hierarchy (nesting) within the data, or to control for *multiple observations* of the same subjects in time. In OLS regression, multiple observations can lead to multicollinearity, which can severely limit inferential ability [53]. LMER explicitly models correlation within (and between) random effect groupings, thus reducing the serious threat of multicollinearity when modeling longitudinal data using other methods. In our data, we use a random effect grouping for each *user_id* (using the term (1|user_id) in lmer) and allow for each grouping to have a separate, distinct regression intercept (while keeping the same slope across all groupings). This is different than fitting a standard linear regression to each user, as random effects allow us to model users with far fewer data points than would be allowed when fitting separate regressions.

We report variable significances using Satterthwaite's approximation for denominator degrees of freedom for regression variable t-tests, implemented using the *lmerTest* R library [54]. This is considered a reasonable approach when sample sizes are large [55]. We report psuedo-R^2 values, as described by Nakagawa and Schielzeth [56], called *marginal* R^2 and *conditional* R^2. The marginal R^2 can be interpreted as the variance described by the fixed effects alone and conditional R^2 as the variance described by both fixed and random effects.

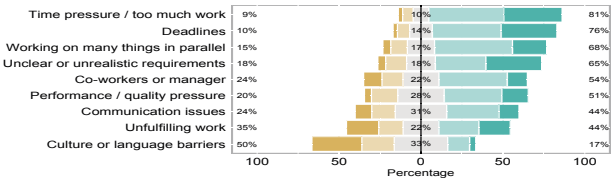

Fig. 3. Causes of work-related stress.

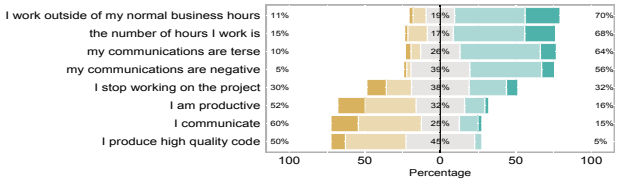

Fig. 4. Respondents' perceived behaviour changes when stressed.

We perform standard model diagnostic tests to evaluate model fit [57]; our models pass these tests.

We prune our fixed-effect control variables to account for multicollinearity by considering only variables with VIF (variance inflation factor) less than 5 [53], as having many fixed effects along with a complex design structure can introduce issues in model estimation. We also control for multiple hypothesis testing by correcting the p-values in the models for each variable using the Benjamini and Hochberg method [58].

TABLE II
SUMMARY STATISTICS FOR WEEKLY DATA

Statistic	Mean	St. Dev.	Min	Max
week	16,380.33	756.72	13,878	17,525
cmits	9.11	12.47	0	91
projs_cmitted	1.64	1.37	0	9
cmnts	2.24	4.87	0	36
disc	1.86	4.05	0	36
cr	0.39	1.82	0	36
projs_cmnted	0.67	0.99	0	5
avg_length	226.89	230.58	0.00	1,675.00
avg_sentences	2.28	1.47	0.00	37.00
pos	0.59	1.51	0	34
neg	0.25	0.78	0	23
neu	1.40	3.33	0	36
avg_thread_pos	3.82	3.85	1.00	30.20
avg_response_time	1,072,561	2,941,707	0	26,235,366
company	0.54	0.50	0	1
days_active_cmit	2.28	1.74	0	7
avg_projs_cmitted_per_day	1.11	0.62	0.00	9.00
sfocus_cmit	0.44	0.65	0.00	3.17
sswitch_cmit	0.21	0.44	0.00	3.17
days_active_cmnt	0.90	1.38	0	7
avg_projs_cmnted_per_day	0.48	0.60	0.00	5.00
sfocus_cmnt	0.16	0.42	0.00	2.32
sswitch_cmnt	0.06	0.23	0.00	2.18
cmit_z	−0.00	1.00	−2.57	12.78
cmnt_z	−0.00	1.00	−2.16	20.37
neg_z	0.00	1.00	−1.20	19.57
projs_cmit_z	0.00	1.00	−3.92	11.02
projs_cmnt_z	−0.00	1.00	−2.68	20.37
days_activ_cmit_z	−0.00	1.00	−3.67	5.37
days_activ_cmnt_z	0.00	1.00	−2.56	20.37

IV. RESULTS

A. RQ1: Survey Perceptions on Work-Related Stress

Our survey looked at downstream effects of work activities, especially work-related stress. We also investigated the social pressure to be constantly available and how it impacted work activities. The majority of our survey respondents reported experiencing work-related stress (66.7%). Fewer respondents reported feeling social pressure (37.8%).

1) Causes of Work-Related Stress: The survey investigated the work-activity-related causes of work-related stress. We asked respondents to indicate, using a five-point Likert-type scale, their agreement to a number of causes of work-related stress. These causes were identified through our pilot survey and are illustrated in Figure 3. For this image and all others depicting an agreement scale, the colors indicate: dark gold = strongly disagree, light gold = somewhat disagree, grey = neutral, light green = somewhat agree, and dark green = strongly agree. Most respondents agreed that too much work, deadlines, and high multitasking are causes of work-related stress. Thus, high workload is a leading cause of stress according to our respondents. Many also agreed that co-workers and communication issues are causes of stress.

We also asked participants what was more likely to cause their work-related stress: the social or technical aspects of their work. In the survey, we provided examples of social (e.g. participating in issues discussions) versus technical (e.g. writing code) to ensure a shared understanding. The majority of respondents said that the social aspects of their work are more likely to cause stress (144 of 187 or 77%).

2) Impacts of Work-Related Stress: Our survey also investigated how respondents perceived their behaviors to change when they are stressed (compared to when they are not stressed). Respondents reported working outside their normal business hours, working more hours, and having more terse and more negative communications. They also reported communicating less, being less productive, and producing less high-quality code. The summarized responses are in Figure 4. The colors indicate: dark gold = much less, light gold = somewhat less, grey = about the same, light green = somewhat more, and dark green = much more.

Similarly, respondents indicated that they were impacted by the stress of team members (see Fig. 6). They reported that this leads to increased communication problems, redistribution of work activities, friction within the team, and stress of others.

3) Identifying Stress in Others: We also asked the survey participants how they become aware of the stress of others on their team. Responses are in Figure 5. The options were created through our pilot survey. The response with the highest level of agreement (88%) was inferring stress from the tone, terseness, or emotion of their communication. Further, many agreed that stress can be inferred from a teammate's body language or demeanor (76%), from work assignments (61%), and from drops in productivity (61%) or quality (65%).

4) Reducing Stress: Our survey also asked what participants do to reduce their stress. Again, the options were created

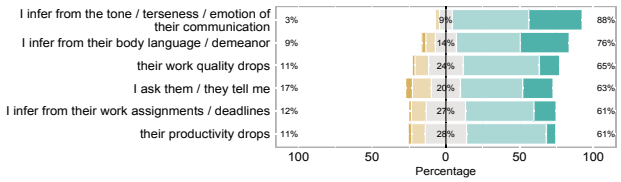

Fig. 5. Mechanisms to become aware of co-workers' stress.

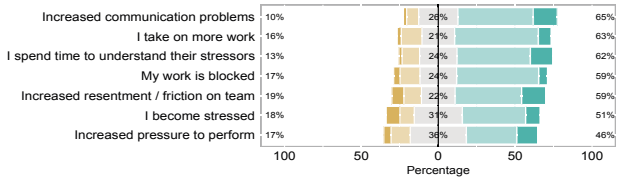

Fig. 6. Perceived impacts when other team members are stressed.

through our pilot survey. The responses with the highest levels of agreement are doing non-work activities and taking a break. Responses are illustrated in Figure 7.

5) Types of Social Pressure: For the respondents who reported feeling social pressure to be constantly available, our survey investigated the types of activities that lead to this feeling. The activities that create social pressure are shown in Fig. 8. Respondents agreed that most project-related activities are associated with social pressure. Keeping abreast of ongoing issue discussions had the highest level of pressure, followed closely by completing development work and participating in pull request discussions. Attending meetings was the only project activity where a majority of respondents did not agree that there was associated social pressure.

6) Impacts of Social Pressure: Our survey also investigated how respondents perceived their behavior to change by social pressure. Respondents felt they worked outside their normal business hours, fragmented their work, responded to online communications more quickly, became stressed, worked more hours than preferred, and communicated tersely. Most did not feel their productivity was impacted. Fig. 9 gives a summary.

RQ1 Answer: We identified many causes of work-related stress and social pressure for developers. Developers seem to perceive that stress and social pressure impact the way they communicate and perform their development work.

B. RQ2: Characterizing Work-Rates

We found that the frequency distributions of the z-normalized commit counts per week and comment counts per week are long-tailed, *i.e.,* they are not normally distributed. This means that there is a much higher probability than expected to find developers' commit and comment rates exceeding 1 standard deviation above their average rates.

RQ2 Answer: Developers often significantly exceed their z-normalized commit and comment rates.

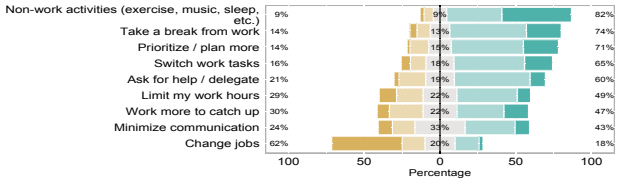

Fig. 7. Strategies to reduce stress.

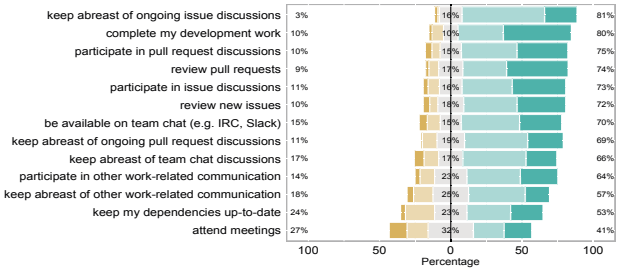

Fig. 8. Perceived social pressure.

C. RQ3: Work-rate Increase and Communication Changes

Here we investigate two facets of communication: the sentiment of comments and the amount of commenting during increased work-rates, both of them in terms of standard deviation units from the average behavior over all active weeks per developer (*i.e.,* z-score-normalized per developer).

Table III, left, shows the results of our first LMER model, where the response variable is neg_z, the per-developer, z-normalized number of negative comments per week. The fixed and random effects of the model together (conditional R^2) explain about 28% of the variability in the data.

Most notably, the factor $cmnt\_z > 1$, i.e., the deviation from the mean by more than 1 standard deviation in the commenting weekly numbers, has a significant and positive effect. This means that for each increase in commenting of more than 1 standard deviation units away from the average, the number of comments with negative sentiment grows by approximately 0.54 standard deviations on average (while holding all other predictors constant). On the other hand, $cmit\_z > 1$ is insignificant.

Other notable positive and significant factors are the number of projects commented on, and the days spent commenting. Interestingly, $sfocus\_cmnt$ has a significant and sizable negative effect. Thus, higher focus on fewer projects when commenting is associated with more negative sentiment within the comments, while holding all other predictors constant.

Interestingly and counterintuitively, having made no commits at all during the week and being associated with a company each have virtually no effect on the negative sentiment.

Table III, middle, shows that our second model, where the response variable is the per-developer, z-normalized number of comments per week, has a conditional R^2 of 94%. Here, the results show a significant, positive association between $days\_activ\_cmnt\_z > 1$, a factor indicating a significantly higher amount of commenting per week than average, and the

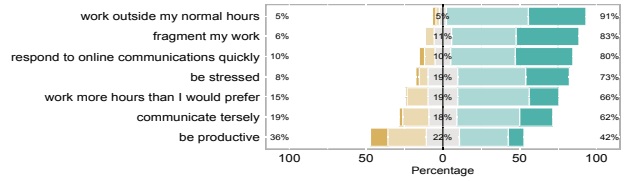

Fig. 9. Respondents' perceived behaviour changes when under social pressure.

	Sentiment_z	Comment_z	Commit_z
(Intercept)	−0.35 (0.03)***	0.67 (0.05)***	−0.78 (0.01)***
avg_length	0.00 (0.00)***	0.00 (0.00)***	−0.00 (0.00)***
cmits	−0.00 (0.00)*	−0.00 (0.00)	0.05 (0.00)***
cmnts	0.07 (0.00)***	0.12 (0.00)***	−0.00 (0.00)
avg_thread_pos	0.01 (0.00)***	0.00 (0.00)***	−0.00 (0.00)
log(avg_response_time + 0.05)	−0.01 (0.00)***	−0.00 (0.00)***	0.00 (0.00)***
avg_projs_cmitted _per_day	−0.01 (0.01)	−0.01 (0.00)	0.09 (0.00)***
avg_projs_cmnted _per_day	0.01 (0.01)	0.03 (0.01)***	0.00 (0.00)
days_active_cmit	−0.00 (0.00)	−0.01 (0.00)***	0.06 (0.00)***
days_active_cmnt	−0.02 (0.00)***	0.01 (0.00)***	−0.01 (0.00)***
sswitch_cmnt	−0.04 (0.01)**	−0.10 (0.01)***	−0.01 (0.00)**
sfocus_cmnt	−0.12 (0.01)***	0.03 (0.00)***	−0.01 (0.00)***
sswitch_cmit	−0.00 (0.01)	0.00 (0.00)	−0.05 (0.00)***
sfocus_cmit	0.01 (0.01)	0.00 (0.00)	−0.02 (0.00)***
cmnt_z>1	0.54 (0.01)***	0.82 (0.00)***	0.02 (0.00)***
cmit_z>1	0.01 (0.01)	0.05 (0.01)***	0.75 (0.00)***
cr>1	0.07 (0.01)***	0.15 (0.00)***	0.00 (0.00)
disc>1	0.15 (0.01)***	0.39 (0.00)***	0.01 (0.00)***
projs_cmit_z>1	−0.00 (0.01)	0.01 (0.00)	0.10 (0.00)***
projs_cmnt_z>1	0.23 (0.01)***	−0.01 (0.01)	0.01 (0.00)
days_activ_cmnt_z>1	0.21 (0.01)***	0.07 (0.00)***	0.01 (0.00)
days_activ_cmit_z>1	−0.00 (0.01)	0.02 (0.01)***	0.12 (0.00)***
cmits == 0	−0.00 (0.02)	−0.02 (0.01)*	−0.08 (0.00)***
company == 1	−0.03 (0.01)**	−0.47 (0.07)***	−0.00 (0.01)
AIC	637,606.75	336,611.59	138,869.79
Num. obs.	195,593	197,323	197,323
Num. groups: user_id	2,601	2,848	2,848
marginal R^2	0.24	0.25	0.85
conditional R^2	0.28	0.94	0.92

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

TABLE III
STATISTICAL MODELS

number of comments per week (all normalized). We also see a positive but small effect of the factor $cmit\_z > 1$.

There are also a few notable effects. First, the predictor $sswitch\_cmnt$ has a significant and sizable negative effect on the normalized number of comments. Thus, less predictable day-to-day focus-switching among multiple projects commented on corresponds with an increase in commenting. Further, the factor $company$ has a significant and sizable negative effect on the normalized number of comments, indicating that being affiliated with a company corresponds with a decrease of commenting.

RQ3 Answer: Periods of high levels of commenting are associated with more negative comments. Less focus-switching when commenting corresponds with more negative comments, and more unpredictable, day-to-day focus-switching when commenting corresponds with an increase in the number of comments. Commit activity does not seem to have a sizable effect on either the number of comments or negativity of comments. Company affiliation is associated with a decrease in commenting, but does not seem to affect the negativity of comments.

D. RQ4: Work-rate Increase and Committing Changes

Table III, right, shows that our third model, where the response variable is the per-developer, z-normalized number of commits per week, has a conditional R^2 of 92%.

Here, symmetrically to the previous model, the results are showing a clear association between some factors indicating a significantly higher amount of committing and the number of commits (both per week), *i.e.,* the factors $projs\_cmit\_z > 1$ and $days\_activ\_cmit\_z > 1$ have significant, positive effects on committing, holding all else constant. We also see a small, positive effect of the factor $cmnt\_z > 1$. There is no other sizable association with any other of the commenting factors.

The predictor $avg\_projs\_cmitted\_per\_day$ has a significant, sizable, positive effect, and the predictor $sswitch\_cmit$ has a significant, somewhat sizable, negative effect. Thus, committing to more projects per day on average within a week and less predictable day-to-day focus-switching among multiple projects committed to correspond with an increase in the normalized number of commits, indicating increasing effects of multitasking on committing beyond average.

Answer to RQ4: The amount of commits does not seem to be sizably associated with commenting. Multitasking activities appear to affect the amount of commits.

V. DISCUSSION

Our survey, as suspected, showed that work-related activities, and especially increase in work responsibilities, lead to work-related stress. Developers noted their various reactions to stress and their coping mechanisms. They cited both technical and social aspects as causes of this stress (time pressure, multitasking, co-workers, communication, etc.).

High Technical and Social Work-rates: Similarly, from the quantitative analysis, we see that developers often have periods when they commit and comment much above their average rates. These results suggest that software development is a demanding environment both technically and socially, making it likely that developers will experience stress.

Independence of Commenting and Committing: Although we expected to find that the increase in the two work-rate components, communication activities (social) and commit activities (technical), add up, we in fact found that those have largely independent effects on the associated changes in developer performance. One explanation is that the two types of activities are too different and difficult to multitask, so developers try to separate them, *e.g.,* by performing coding and testing during a certain week and code review and issue discussions during a different week. Future work can study this in detail.

Communication Overload and Negativity: In both the survey results and our regression analysis, we have found that increased social work-rates are associated with more negativity in communications. In the survey results, a majority of respondents agreed that the social aspects of their work are more likely to cause them stress than the technical, and that when they are stressed, their communications become more negative. Similarly, we observe that higher-than-average commenting, *i.e.,* social work-rate, associates with higher negativity in comments, with a significant and sizable effect size. Survey responders also agreed that they become aware of the stress of other teammates by inferring from the characteristics of their communication (tone, emotion, etc.). This indicates that when a team member's communications become negative, other teammates can tell and can be affected by it as well. From our analysis of a sample of negative comments in our dataset, we found quite a few reasons for the presence of negativity on GitHub (negative reviews, time constraints, disagreements, etc.), all of which could be products of stress and/or lead to more stressful situations. Being on the receiving end of these kinds of negativity in significant amounts could be detrimental to a developer's well-being. In fact, most survey respondents agreed that they take on more work when they see their team members are stressed and even become more stressed themselves, indicating a rippling effect in which developers' high workload and resulting stress can transfer to others who they communicate with, a direction for future study.

Company Affiliations and Decreased Commenting: We found that affiliation with a company is associated with a sizable decrease in commenting. In a company setting, developers may communicate via other, perhaps more effective means than GitHub, indicating that online-only communication may be less efficient. An interesting future work would be to further investigate these possibilities.

Multitasking and Changes in Work Patterns: We found that different kinds of technical and social multitasking are associated with increases in committing and commenting, respectively. Our models show that less focus-switching when commenting associates with more negativity, and more unpredictable, day-to-day focus-switching when commenting associates with increased communication. A plausible explanation is that focusing on fewer projects during a day allows for better critiquing/bug finding, and commenting within different projects in consecutive days allows time for those bugs to be fixed, yielding higher social productivity. Moreover, these findings in conjunction with prior work, which found that multitasking associates with higher technical productivity [1], suggest that technical and social work are multiplexed.

Recommendations Based on Findings: Our findings suggest good practices that could reduce stress and enable stronger and more efficient team work. First, balancing work activities could prevent high workloads and the need for excessive multitasking, consequently reducing the chance of stressful reactions like overworking and negative communications. Furthermore, as team members seem to pick up on others' stressful communications and become stressed and/or resentful as a result, moderating language when critiquing other members' work along with team building activities could reduce the chance of stress transfer and overall allow for more efficient communication and problem solving. Finally, our survey results suggest that taking breaks from work is a popular strategy for reducing stress.

VI. Threats to Validity and Conclusion

Threats: We collected developers' information and activity from GHTorrent, so our data is limited by how completely GHTorrent captures developer data. Also, we do not consider activity outside of GitHub that may influence stress levels. But due to our large sample size, and since people's average behaviors tend to be similar, other contributors to stress are likely normally distributed, making the chance of bias unlikely.

We only looked at commits to measure technical work-rate, and discussion and code review comments to measure social work-rate. Other technical and social contributions can be included in future work, like issue and pull request events, comments made on other communication platforms, etc.

Senti4SD has drawbacks that could affect our work. First, the classifier was trained on Stack Overflow posts, which may not accurately represent GitHub comments. Also, the authors used the SentiStrength lexicon, which has shown to have limited performance when applied to the SE domain.

Conclusion: Guided by the ever-increasing number and diversity of tasks in the portfolios of today's software developers, we sought to model the effects of work-rate increase on developer communication and performance outcomes. We adopted a multidimensional view of work-rates comprised of (1) a coordinating, or social, dimension and (2) a technical dimension. We approached the problem with a mixed qualitative (survey) and quantitative (contextualized modeling) approach.

Our survey showed that developers do react to stress caused by work-rate increase in various ways, and they do employ coping mechanisms so that the stress does not spill out into the social or technical activities.

We expected to find that the increase in the communication and commit work-rates add up, but we found that those have largely independent effects on the associated changes in developer behavior, suggesting that they are not multitasked together. Notably, our data analytics studies showed that high social work-rate increase does sizably associate with an important change in behavior, which also correlates with high-stress: increased sentiment negativity in comments.

We demonstrated that the effects of work-rate increase can be measured, and the analytics is readily usable for downstream decision making. To individual developers, having such feedback can be invaluable in helping them modify their behavior before it starts affecting the community. To a team, such analytics can be useful in keeping track of and evaluating increase in stress-related workloads associated with group tasks. A future work based on these methods and findings would be to develop tools that keep track of average behavior and semaphore unhealthy changes in real time.

REFERENCES

[1] B. Vasilescu, K. Blincoe, Q. Xuan, C. Casalnuovo, D. Damian, P. Devanbu, and V. Filkov, "The sky is not the limit: multitasking across github projects," in *Proceedings of the International Conference on Software Engineering (ICSE)*. IEEE, 2016, pp. 994–1005.

[2] M.-A. Storey, A. Zagalsky, F. Figueira Filho, L. Singer, and D. M. German, "How social and communication channels shape and challenge a participatory culture in software development," *IEEE Transactions on Software Engineering*, vol. 43, no. 2, pp. 185–204, 2017.

[3] K. Reinke and T. Chamorro-Premuzic, "When email use gets out of control: Understanding the relationship between personality and email overload and their impact on burnout and work engagement," *Computers in Human Behavior*, vol. 36, pp. 502–509, 2014.

[4] L. Reinecke, S. Aufenanger, M. E. Beutel, M. Dreier, O. Quiring, B. Stark, K. Wölfling, and 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*, vol. 20, no. 1, pp. 90–115, 2017.

[5] A. Zika-Viktorsson, P. Sundström, and M. Engwall, "Project overload: An exploratory study of work and management in multi-project settings," *International Journal of Project Management*, vol. 24, no. 5, pp. 385–394, 2006.

[6] J. P. Borst, N. A. Taatgen, and H. van Rijn, "What makes interruptions disruptive? a process-model account of the effects of the problem state bottleneck on task interruption and resumption," in *CHI*. ACM, 2015, pp. 2971–2980.

[7] C. Parnin and S. Rugaber, "Resumption strategies for interrupted programming tasks," *Software Quality Journal*, vol. 19, no. 1, pp. 5–34, 2011.

[8] J. P. Borst, N. A. Taatgen, and H. van Rijn, "The problem state: a cognitive bottleneck in multitasking." *Journal of Experimental Psychology: Learning, memory, and cognition*, vol. 36, no. 2, p. 363, 2010.

[9] P. Lenberg, R. Feldt, and L. G. Wallgren, "Behavioral software engineering: A definition and systematic literature review," *Journal of Systems and Software*, vol. 107, pp. 15–37, 2015.

[10] P. Devanbu, P. Kudigrama, C. Rubio-González, and B. Vasilescu, "Timezone and time-of-day variance in GitHub teams: an empirical method and study," in *Proceedings of the 3rd ACM SIGSOFT International Workshop on Software Analytics (SWAN)*. ACM, 2017, pp. 19–22.

[11] S. Black, R. Harrison, and M. Baldwin, "A survey of social media use in software systems development," in *Proceedings of the 1st Workshop on Web 2.0 for Software Engineering*. ACM, 2010, pp. 1–5.

[12] A. Begel, R. DeLine, and T. Zimmermann, "Social media for software engineering," in *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research*. ACM, 2010, pp. 33–38.

[13] M.-A. Storey, C. Treude, A. van Deursen, and L.-T. Cheng, "The impact of social media on software engineering practices and tools," in *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research*. ACM, 2010, pp. 359–364.

[14] Y. Tian, P. Achananuparp, I. N. Lubis, D. Lo, and E.-P. Lim, "What does software engineering community microblog about?" in *Proceedings of the 9th IEEE Working Conference on Mining Software Repositories*. IEEE Press, 2012, pp. 247–250.

[15] Y. M. Kalman and S. Rafaeli, "Online pauses and silence: Chronemic expectancy violations in written computer-mediated communication," *Communication Research*, vol. 38, no. 1, pp. 54–69, 2011.

[16] I. Steinmacher, T. Conte, M. A. Gerosa, and D. Redmiles, "Social barriers faced by newcomers placing their first contribution in open source software projects," in *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. ACM, 2015, pp. 1379–1392.

[17] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Social coding in github: transparency and collaboration in an open software repository," in *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*. ACM, 2012, pp. 1277–1286.

[18] ——, "Leveraging transparency," *IEEE Software*, vol. 30, no. 1, pp. 37–43, 2013.

[19] J. Marlow, L. Dabbish, and J. Herbsleb, "Impression formation in online peer production: activity traces and personal profiles in github," in *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*. ACM, 2013, pp. 117–128.

[20] J. Marlow and L. Dabbish, "Activity traces and signals in software developer recruitment and hiring," in *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*. ACM, 2013, pp. 145–156.

[21] A. Trockman, S. Zhou, C. Kästner, and B. Vasilescu, "Adding sparkle to social coding: An empirical study of repository badges in the npm ecosystem," in *Proceedings of the International Conference on Software Engineering(ICSE)*. ACM, 2018.

[22] K. Blincoe, F. Harrison, and D. Damian, "Ecosystems in github and a method for ecosystem identification using reference coupling," in *Proceedings of the International Conference on Mining Software Repositories (MSR)*. IEEE, 2015, pp. 202–207.

[23] W. Ma, L. Chen, X. Zhang, Y. Zhou, and B. Xu, "How do developers fix cross-project correlated bugs?: A case study on the GitHub scientific Python ecosystem," in *Proceedings of the International Conference on Software Engineering (ICSE)*. IEEE, 2017, pp. 381–392.

[24] Y. Zhang, Y. Yu, H. Wang, B. Vasilescu, and V. Filkov, "Within-ecosystem issue linking: a large-scale study of Rails," in *Proceedings of the 7th International Workshop on Software Mining*. ACM, 2018, pp. 12–19.

[25] C. Bogart, C. Kästner, J. Herbsleb, and F. Thung, "How to break an api: cost negotiation and community values in three software ecosystems," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. ACM, 2016, pp. 109–120.

[26] J. S. Rubinstein, D. E. Meyer, and J. E. Evans, "Executive control of cognitive processes in task switching." *Journal of Experimental Psychology: Human Perception and Performance*, vol. 27, no. 4, p. 763, 2001.

[27] S. J. Gilbert and T. Shallice, "Task switching: A PDP model," *Cognitive Psychology*, vol. 44, no. 3, pp. 297–337, 2002.

[28] C. Rosen, "The myth of multitasking," *The New Atlantis*, vol. 20, no. Spring, pp. 105–110, 2008.

[29] C. Ipsen, "Knowledge work and work-related stress," in *Congress of the International Ergonomics Association*, 2006.

[30] M. Mogensen, V. Andersen, and C. Ipsen, "Ambiguity, identity construction and stress amongst knowledge workers: developing collective coping strategies through negotiations of meaning," in *International conference on Organizational Learning, Knowledge and Capabilities, Copenhagen, Denmark*, 2008.

[31] T. H. Davenport, *Thinking for a living: how to get better performances and results from knowledge workers*. Harvard Business Press, 2005.

[32] T. S. Kristensen, "Challenges for research and prevention in relation to work and cardiovascular diseases," *Scandinavian Journal of Work, Environment & Health*, pp. 550–557, 1999.

[33] J. R. Galbraith, "Designing organizations: an executive guide to strategy," *Structure*, 2002.

[34] G. Mark, D. Gudith, and U. Klocke, "The cost of interrupted work: more speed and stress," in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, 2008, pp. 107–110.

[35] J. K. Dua, "Job stressors and their effects on physical health, emotional health and job satisfaction in a university," *Journal of Educational Administration*, vol. 32, no. 1, pp. 59–78, 1994.

[36] J. Bakker, L. Holenderski, R. Kocielnik, M. Pechenizkiy, and N. Sidorova, "Stess@ work: From measuring stress to its understanding, prediction and handling with personalized coaching," in *Proceedings of the 2nd ACM SIGHIT International health informatics symposium*. ACM, 2012, pp. 673–678.

[37] S. Easterbrook, J. Singer, M.-A. Storey, and D. Damian, "Selecting empirical methods for software engineering research," in *Guide to advanced empirical software engineering*. Springer, 2008, pp. 285–311.

[38] G. Gousios and D. Spinellis, "Ghtorrent: Github's data from a firehose," in *Mining software repositories (msr), 2012 9th ieee working conference on*. IEEE, 2012, pp. 12–21.

[39] B. Vasilescu, A. Serebrenik, and V. Filkov, "A data set for social diversity studies of github teams," in *Proceedings of the 12th Working Conference on Mining Software Repositories*. IEEE Press, 2015, pp. 514–517.

[40] V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative research in psychology*, vol. 3, no. 2, pp. 77–101, 2006.

[41] "Querying mongodb programmatically," http://ghtorrent.org/raw.html, accessed: 2018-04-27.

[42] E. Guzman, D. Azócar, and Y. Li, "Sentiment analysis of commit comments in github: an empirical study," in *Proceedings of the 11th Working Conference on Mining Software Repositories*. ACM, 2014, pp. 352–355.

[43] D. Pletea, B. Vasilescu, and A. Serebrenik, "Security and emotion: sentiment analysis of security discussions on github," in *Proceedings of the 11th working conference on mining software repositories*. ACM, 2014, pp. 348–351.

[44] M. Ortu, G. Destefanis, B. Adams, A. Murgia, M. Marchesi, and R. Tonelli, "The jira repository dataset: Understanding social aspects of software development," in *Proceedings of the 11th international conference on predictive models and data analytics in software engineering*. ACM, 2015, p. 1.

[45] R. Jongeling, P. Sarkar, S. Datta, and A. Serebrenik, "On negative results when using sentiment analysis tools for software engineering research," *Empirical Software Engineering*, vol. 22, no. 5, pp. 2543–2584, 2017.

[46] N. Novielli, F. Calefato, and F. Lanubile, "The challenges of sentiment detection in the social programmer ecosystem," in *Proceedings of the 7th International Workshop on Social Software Engineering*. ACM, 2015, pp. 33–40.

[47] M. R. Islam and M. F. Zibran, "Leveraging automated sentiment analysis in software engineering," in *Proceedings of the 14th International Conference on Mining Software Repositories*. IEEE Press, 2017, pp. 203–214.

[48] T. Ahmed, A. Bosu, A. Iqbal, and S. Rahimi, "Senticr: A customized sentiment analysis tool for code review interactions," in *Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering*. IEEE Press, 2017, pp. 106–111.

[49] F. Calefato, F. Lanubile, F. Maiorano, and N. Novielli, "Sentiment polarity detection for software development," *Empirical Software Engineering*, pp. 1–31, 2017.

[50] M. Thelwall, K. Buckley, and G. Paltoglou, "Sentiment strength detection for the social web," *Journal of the Association for Information Science and Technology*, vol. 63, no. 1, pp. 163–173, 2012.

[51] Q. Xuan, A. Okano, P. Devanbu, and V. Filkov, "Focus-shifting patterns of oss developers and their congruence with call graphs," in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*. ACM, 2014, pp. 401–412.

[52] R. H. Baayen, D. J. Davidson, and D. M. Bates, "Mixed-effects modeling with crossed random effects for subjects and items," *Journal of memory and language*, vol. 59, no. 4, pp. 390–412, 2008.

[53] J. Cohen, *Applied multiple regression/correlation analysis for the behavioral sciences*. Lawrence Erlbaum, 2003.

[54] A. Kuznetsova, P. B. Brockhoff, and R. H. Christensen, "lmertest package: Tests in linear mixed effects models," *Journal of Statistical Software*, vol. 82, no. 13, pp. 1–26, 2017.

[55] A. Gelman and J. Hill, *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press, 2006.

[56] S. Nakagawa and H. Schielzeth, "A general and simple method for obtaining r2 from generalized linear mixed-effects models," *Methods in Ecology and Evolution*, vol. 4, no. 2, pp. 133–142, 2013.

[57] J. D. Singer and J. B. Willett, *Applied longitudinal data analysis: modeling change and event occurrence*. Oxford University Press, 2003.

[58] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: a practical and powerful approach to multiple testing," *Journal of the royal statistical society. Series B (Methodological)*, pp. 289–300, 1995.