Exploring Causes of Frustration for Software Developers

Denae Ford
Department of Computer Science
North Carolina State University
Raleigh, NC, USA
Email: dford3@ncsu.edu

Chris Parnin
Department of Computer Science
North Carolina State University
Raleigh, NC, USA
Email: cjparnin@ncsu.edu

***Abstract*—When learning to program, frustrating experiences contribute to negative learning outcomes and poor retention in the field. Defining a common framework that explains why these experiences occur can lead to better interventions and learning mechanisms. To begin constructing such a framework, we asked 45 software developers about the severity of their frustration and to recall their most recent frustrating programming experience. As a result, 67% considered their frustration to be severe. Further, we distilled the reported experiences into 11 categories, which include issues with mapping behaviors to code and broken programming tools. Finally, we discuss future directions for defining our framework and designing future interventions.**

I. Introduction

Developers must be able to constantly learn new technologies, adapt to new environments, and overcome challenges when learning and practicing their craft. However, failure to overcome obstacles in these situations can introduce a sense of mounting frustration in developers that can negatively impact learning outcomes [1] and influence retention in a field [2].

Researchers have studied frustration in psychology and software development. Dollard [3] defines frustration as an obstacle encountered in the pursuit of an expected goal. Ko et al. [4] describes six learning barriers that programmers face when learning to program. When these barriers are not overcome, developers can experience frustration and perceive their goal as increasingly insurmountable. In computer science education, several studies have been done to find connections with frustration and general learning goals [1], [5]. In addition, affective computing techniques have been used to detect frustration during programming tasks [6]. Overall, these techniques provide a useful but incomplete foundation for understanding frustration. Besides Eisenstadt's bug war stories [7], there has been a lack of a concerted effort in cataloging and categorizing the frustrating experiences developers face.

In this paper, we present findings from a survey of 45 software developers where we asked about their specific frustrating experiences. We organized our findings into 11 categories of causes of frustration for software developers.

II. Methodology

We conducted a survey of software developers to solicit frustrating scenarios they have encountered. Links to the survey were openly posted in computer science affinity groups and emailed out to self-identified groups of software developers. A total of 45 software developers completed the 16 question survey[1]. The responders consisted of 23 students, 12 industry employees, and 10 that classified as both. The survey asked the participants to rank the severity of their frustration during programming tasks on a Likert scale from 1 to 5. To identify the causes of frustration we focused on a single question: *When is the last time you were frustrated?* We used an open card sort to group each response into categories. *Card sorting* is a technique used to define taxonomies from data [8]. Each author randomly selected 10 responses, identified their themes, and compared these themes to the remaining responses until arriving at a steady 11 categories.

III. Categories Defined

From the survey, 67% of respondents acknowledged that frustration is a severe problem and their causes are reflected in the categories below. Categories are ordered from the highest to lowest frequency of cards. A response from the survey that best represents the category is listed after each description.

Mapping Behavior to Cause - Responses in this category refer to identifying what portion of the code was causing an issue. Not having a good mental model of the code resulted in frustration of misinterpretation of the task [9]. *"I was frustrated when I couldn't figure out why there was a random gap of space on my website..."*

Programming Tools - Learning curves exist when getting acclimated to a new tool: learning new features, shortcuts, etc. This learning curve is warped when the tool is broken. Respondents in this category acknowledged both of these frustrating occurrences with programming tools. *"I've been trying to transition to using an IDE I never have before..."*

Size - Respondents acknowledged three issues related to the size of their tasks: (1) Large goal spaces, high cognitive complexity: *Where do I begin?*, (2) Gulf of completion: *I've come a long way but there is a little thing blocking me*, and (3) Large artifacts: *There's so much to understand. "There were some logical errors in a big code base..."*

New Project Adjustment - Adjusting to a new project environment takes time. During this time the developer is

[1] https://github.com/alt-code/FrustrationExperiment/blob/master/Survey.md

unfamiliar with how to accomplish tasks, appropriate questions to ask, as well as configuration issues. *"I was trying to set up some software for a study I'm working on. I expected some configuration obstacles, but I became frustrated..."*

Unavailability of Resources - Documenting code is a recommended practice, but when missing, it creates confusion. In addition to documentation, this category acknowledges services that are supposed to be available to users but were not at their disposal. *"The server I needed for my files was not letting me log in and the only person who could fix it was too busy working on something else."*

Programming Experience - Respondents mentioned issues that arose with programming languages. These issues came from confusion with syntax of various languages and experience with new frameworks. *"I had to code something in Perl and I didn't know the language syntax."*

Simple Problem - Peers and management assign 'simple' tasks to individuals not fully comprehending the depth of the truly complex issue. Frustration arises when the expectation is not met and the 'simple' task is not as easy as recommended by others. *"Inability to code something that I know should be simple."*

Fear of Failure - The obsession over the fear of failure and not succeeding sets back individuals and overcomes them in the form of frustration. *"It builds a strong sense of anxiety. I feel like I may not solve the issue..."*

Internal Hurdles - Some respondents acknowledged that they were the cause of their own frustration. These respondents knew they have been putting this pressure on themselves; some pressures even leading to fatigue. *"The problem that I faced with frustration is I tend to procrastinate..."*

Limited Time - Respondents described a limited amount of time allotted to work on projects. In the short amount of time given it seems unreasonable to make significant progress and make lasting impressions to others on a team. *"I had to deal with an ambitious project in a limited time frame."*

Peers - Respondents referred to peers as a source of distraction from the task at hand. In addition, some respondents mentioned their peers as incompetent and being subordinate. *"Peers were terrible programmers, less experienced, and refused to use libraries/patterns to make things easier... "*

IV. Discussion

Each of the categories are intertwined in a way that show that one cause of frustration can be a part of the cause of another. For example, managers and *peers* can claim a task to be a *simple problem* which gives the developer high expectations for a task. In addition, these high expectations can cause developers to place pressure on themselves if the *simple problem* is not as easy as others claim it to be. There are many scenarios such as these that connect these categories.

Deriving these categories and connecting them through scenarios forms relationships between groups. These groups are internal pressures, external pressures, lack of experience, and perception. External pressures include categories where situations are out of the respondents control while internal refers to pressure put on themselves. Aside from these pressures placed on software developers there is a general lack of experience that comes with the territory; it takes time to get acclimated to new environments, programming languages, etc. Perception refers to how software developers perceive the problem and determines how to complete a task.

In addition to the defined categories, there was one response in which the participant acknowledged they have an issue with frustration. The respondent perceived frustration as inevitable and embraced it saying, *"My work revolves around it, so it's frequent and recent."* This *new norm* is expecting that there are going to be challenges that require a bit of a headache; expecting these obstacles is half the battle. The other half is how these challenges are approached. This perspective comes with experience and being able to reflect on them.

This corpus of data provides inspiration for interventions and a new perspective to address the causes of frustration. One intervention that can be derived is the unavailability of resources such as peers in combination with limited documentation presents the opportunity for crowd-based collaboration tools. For example, a tool that matched developers with similar issues in real time would aid this situation.

V. Conclusions & Future Work

Using an open card sort, we identified 11 areas for the research community to explore in comprehending the frustration of software developers. The results from this paper have demonstrated that there is still a problem space of frustration for developers and defined possible interventions. For software developers, studying their behavior will give us a better understanding of the obstacles they encounter. This paper encourages the continuance of exploration into this endeavor. In future work, we will rank the intensity of the causes of frustration, investigate the enhancers of frustration, and identify what coping mechanisms developers *think* are helpful for frustration.

References

[1] J. F. Grafsgaard, J. B. Wiggins, K. E. Boyer, E. N. Wiebe, and J. C. Lester, "Automatically recognizing facial indicators of frustration: a learning-centric analysis," in *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, 2013, pp. 159–165.

[2] J. T. Garner and L. T. Garner, "Volunteering an opinion: Organizational voice and volunteer retention in nonprofit organizations," *Nonprofit and Voluntary Sector Quarterly*, 2010.

[3] J. Dollard, N. E. Miller, L. W. Doob, O. H. Mowrer, and R. R. Sears, *Frustration and aggression.* Yale University Press, 1939.

[4] A. J. Ko, B. A. Myers, and H. H. Aung, "Six learning barriers in end-user programming systems," in *Visual Languages and Human Centric Computing, 2004 IEEE Symposium on*. IEEE, 2004, pp. 199–206.

[5] S. Hansen and E. Eddy, "Engagement and frustration in programming projects," *SIGCSE Bull.*, vol. 39, no. 1, pp. 271–275, Mar. 2007.

[6] M. M. T. Rodrigo and R. S. Baker, "Coarse-grained detection of student frustration in an introductory programming course," in *Proceedings of the fifth international workshop on Computing education research workshop*. ACM, 2009, pp. 75–80.

[7] M. Eisenstadt, "My hairiest bug war stories," *Communications of the ACM*, vol. 40, no. 4, pp. 30–37, 1997.

[8] A. Begel and T. Zimmermann, "Analyze this! 145 questions for data scientists in software engineering." in *ICSE*, 2014, pp. 12–13.

[9] S. Wiedenbeck, V. Fix, and J. Scholtz, "Characteristics of the mental representations of novice and expert programmers: An empirical study," *International Journal of Man-Machine Studies*, vol. 39, no. 5, pp. 793–812, 1993.