

Encouraging Expressions of Gratitude in Open-Source Software

Olivia Xu

CMU-CS-23-143

January 2024

Computer Science Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Geoff Kaufman, Chair
Chinmay Kulkarni
Bogdan Vasilescu

*Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science.*

Copyright © 2024 **Olivia Xu**

Keywords: Open Source, Gratitude, VS Code Extension

For my family

Abstract

Can channels through which open-source software is consumed strengthen expressions of gratitude between users and developers? Open-source software is the infrastructure on which all of digital society relies on. It is vital to the functioning of governments, private companies, and individual lives. Yet, ways to show support to those who work on open-source software — often unpaid volunteers — are few. An important reason could be that the way in which software is packaged and distributed alienates the software from the developer. Indeed, this is what allows for the rapid distribution and efficient use of this software, but it also re-inscribes a kind of software-centrism, centering attention on the technical capabilities of the software rather than the labor involved. As a result, users are far more likely to reach out only when they have a complaint. This lack of positive feedback can be a major cause for burnout, which risks slowed innovation, or worse, maintainers discontinuing work that much of our digital infrastructure is critically reliant on.

We introduce a VS Code extension, *Hug Reports*, that detects packages imported by a user within their code and renders an inline button for sending a thanks message to the developers of those packages, from within VS Code. We deployed *Hug Reports* with 20 participants (“expressers”) over a two-week period to observe its use in situ and to understand how and when participants experienced and expressed gratitude. To investigate how the gratitude would be received and interpreted, we aggregated thanks messages logged during the course of our deployment and sent these to the developers they were addressed to (“recipients”), inviting them to participate in a survey and interview. We present our findings from this formative deployment and discuss implications for designing appreciation systems.

This thesis contributes (1) a technical system that aims to give users a way to express appreciation; (2) empirical insights about how gratitude is experienced and communicated within software development workflows and how gratitude is received and interpreted within the culture of open-source projects; (3) design implications of an appreciation system in the context of open source.

Acknowledgments

First and foremost, I would like to thank Professor Geoff Kaufman for being my advisor and for his guidance and mentorship alongside Professor Chinmay Kulkarni throughout my research career. I would also like to thank Pranav Khadpe for introducing me to HCI research and for his advice, as well as everyone else in the research groups I worked with along the way. I would like to further thank Professor Bogdan Vasilescu for being a part of my thesis committee. Finally, I would like to thank all of my friends and family for their support.

Contents

- 1 Introduction** **1**

- 2 Related Work** **3**
 - 2.1 Open Source 3
 - 2.2 Giving and Receiving Gratitude 4
 - 2.3 Appreciation Systems 4

- 3 System Design** **7**
 - 3.1 User Scenario 7
 - 3.2 Design Features 7
 - 3.2.1 Realizing reliance on other developers' code 7
 - 3.2.2 Ability to say a quick thanks and write a longer message 8
 - 3.2.3 Not seeing recipient names 9
 - 3.2.4 Thanking the 20 most recent contributors 9

- 4 Deployment Study** **11**
 - 4.1 Participants 11
 - 4.2 Procedure 11

- 5 Findings** **13**
 - 5.1 Extension Data 13
 - 5.1.1 Most users felt a positive effect from saying thanks 13
 - 5.1.2 Thanking at a package / class / function level 13
 - 5.1.3 Thanking near the top of the file 14
 - 5.2 Participant Interviews 15
 - 5.2.1 Users started to notice their reliance on and gratitude for other developers' code 15
 - 5.2.2 Users tended to not write long-form thanks messages 16
 - 5.2.3 Users tended to say thanks reactively after using code rather than proactively, and all at once 17
 - 5.2.4 Users felt that their thanks was spontaneous and not coerced 17
 - 5.2.5 Users felt that their thanks was quickly discharged and only accumulated in new use cases 18
 - 5.2.6 Users were intentional about moments in time for saying thanks 18

5.3	Developer Survey Data	19
5.3.1	Developers were surprised to receive these thanks messages	19
5.3.2	Developers want to receive more detailed thanks messages	19
5.3.3	Developers sometimes felt that the thanks messages were misdirected	21
5.4	Maintainer Interviews	21
5.4.1	Maintainers appreciated receiving thanks from people	21
5.4.2	Granularity of appreciation should be project-specific	22
5.4.3	Preferred form of receiving thanks	22
6	Discussion	23
6.1	Incorporating a Two-Tier Thanking System	23
6.2	Improvements to <i>Hug Reports</i>	24
6.3	Challenges in Building an Appreciation System for Open Source	24
6.4	Future Work	24
7	Conclusion	27
	Bibliography	29

List of Figures

- 3.1 (a) The extension displays a raised-hands icon in the gutter next to each line of code that imports or makes use of an imported package, class, or function. (b) Right-clicking on one of the raised-hands icons in the gutter opens an options menu that allows the user to “Say Thanks” for that code. 8
- 3.2 (a) After a user clicks “Say Thanks”, they are presented with an option to write a long-form message. (b) Clicking “Say More” brings the user to this form where they can write a more detailed message of thanks to the developer(s). 9
- 5.1 All user responses to how sending a thanks message made them feel. -5 corresponds to “much more negative than normal”, 0 corresponds to “no different than normal”, and 5 corresponds to “much more positive than normal”. 14
- 5.2 All user responses to how they anticipated the recipient(s) of their thanks message would feel. -5 corresponds to “much more negative than normal”, 0 corresponds to “no different than normal”, and 5 corresponds to “much more positive than normal”. 14
- 5.3 All user responses to how surprised they anticipated the recipient(s) of their thanks message would be. 0 corresponds to “not at all surprised” and 10 corresponds to “extremely surprised”. 15
- 5.4 What users chose to say thanks for (package / class vs. function). Users said thanks for a package / class a similar number of times as for a function. 15
- 5.5 Distribution of line numbers at which users said thanks for a package, class, or function. Users tended to say thanks at the beginning of a file. 16
- 5.6 All contributor responses to how surprised they were to receive the thanks users sent. -5 corresponds to “not at all surprised” and 5 corresponds to “extremely surprised”. 20
- 5.7 All contributor responses on how they felt upon receiving the thanks sent by users. -5 corresponds to “much more negative than normal”, 0 corresponds to “no different than normal”, and 5 corresponds to “much more positive than normal”. 20

Chapter 1

Introduction

Open-source projects are typically maintained by communities of volunteers. Today’s open-source ecosystem has accelerated the development and use of many crucial pieces of software, all without the need for connecting with the authors, except for the occasional bug reports or feedback on how to improve [13, 20]. In turn, developers often receive little to no positive feedback and an overwhelming barrage of negative feedback [13], toxic comments [8, 17], and sometimes even abuse [1, 31]. As a result, developers have reported feeling discouraged and underappreciated for the large amounts of work it takes for them to maintain and grow an open-source project [13, 15]. In many situations, this has contributed to various open-source projects not being sustainable for long periods of time as maintainers have little incentive to and / or cannot withstand this environment [32]. Some projects have attempted to combat these issues by introducing a code of conduct [27], but this has not always been successful given the difficulty in identifying, applying, and then enforcing these rules [7]. Additionally, codes of conduct only attempt to minimize the negative feedback maintainers receive and do not necessarily increase the positive feedback they receive.

This environment has been perpetuated due to the technical practice of modularization. With the ubiquity of open-source software, writing modular code is crucial to ensure that the software can be efficiently developed and then easily distributed for others to use [18]. This idea of modularity has led users to view others’ code as a necessary good [30] as they use it and inevitably leads to a detachment between developers and users of that software, where maintainers are often underappreciated. Even for those who do want to express their appreciation to a developer, the high amount of effort it currently takes (i.e. identifying the author of the relevant piece of code, finding out how to contact the author, and then finally expressing gratitude) is another barrier.

In this thesis, we introduce *Hug Reports*, a new system that supports expressions of gratitude by enabling users to communicate their appreciation for software written by other developers directly within VS Code. The name *Hug Reports* was inspired by the idea that we wanted our extension to counteract the persistent culture of bug reports in open source. Built as a VS Code extension, it allows a user to express their appreciation to the author(s) of any imported code directly through the IDE as they are coding, with the option of sending a long-form message of gratitude with further details. *Hug Reports* attempts to increase a user’s awareness of their reliance on others’ code and subsequently provides an easy way for them to express their gratitude within VS Code itself. We hypothesize that by increasing users’ awareness of their reliance

on and appreciation for others' code, this will increase their perspective-taking with maintainers. Moreover, *Hug Reports* seeks to improve appreciation for and feelings of being valued and capable of contributing among maintainers through authentic expressions of appreciation.

To understand how *Hug Reports* can better promote and support expressions of gratitude among users and developers, we conducted a technology probe in the form of a small-scale field deployment study and enlisted 20 participants to try out our extension for two weeks. Participants were users who regularly coded in Python and / or JavaScript / TypeScript. Additionally, half of the participants were required to say thanks at least 2 times a day, whereas the other half was not. Afterwards, we interviewed both the users and the developers who were thanked, focusing on how both sides felt during and after writing and receiving the messages.

By analyzing usage data and interviews with both users and recipients, our results indicate initial success with our system — *Hug Reports* — in allowing users to more easily express their gratitude to developers, in addition to developers feeling more recognized for their contributions. However, the main contribution is a clearer understanding of the design space for an appreciation system with open-source communities — that is, what criteria are necessary when designing such a system. Moreover, while our extension was built with the open-source community in mind, we believe our system can be applicable outside of this context and adapted to other similar settings, such as volunteer communities.

Chapter 2

Related Work

We draw upon literature in the fields of human-computer interaction and social psychology to motivate our work. We begin by examining open-source culture, as well as appreciation as an action, before discussing a broad range of systems geared towards expressing appreciation.

2.1 Open Source

Open-source software has quickly become ubiquitous in recent years and is crucial for the development of many important applications. Yet, the future of these open-source projects is often largely dependent on its maintainers — the developers who oversee the project, through organizing the project and its road map to bringing in new code contributions and creating project growth. Without maintainers, open-source projects would likely be unable to successfully function and grow. These projects are dependent on these maintainers for fostering an open-source community while ensuring continued project development [22]. However, the responsibility that maintainers face is not without challenges. A critical issue that many open-source projects experience is maintainer burnout [23] and high turnover rates [6, 16].

Maintainers are tasked with a wide range of responsibilities, including responding to and addressing bug reports, improving documentation, implementing new features, and so on. Coupled with the fact that many of these maintainers are often volunteering to fill these roles highlights why this often leads to burnout. There has been work focused on simplifying a maintainer’s responsibilities, such as through the creation of a dashboard that carefully aggregates metrics useful to a maintainer in maintaining a healthy open-source community [22]. Even still, maintainers face negative environments where they encounter toxicity and pushback from software users. Maintainers may receive insults and attacks while software users concurrently demand new features and bug fixes. In turn, this environment then leads to maintainers leaving projects and difficulties in retaining maintainers, resulting in a high turnover rate. Recent work has focused on studying how to mitigate these negative aspects of being a maintainer, such as through building a toxicity detector aimed at measuring and detecting toxic discussions in GitHub issues [23]. Similarly, there has been work done on automatically detecting cases of toxic language and unnecessary pushback faced by maintainers to prevent and mitigate interpersonal conflict within code reviews on GitHub [21].

2.2 Giving and Receiving Gratitude

The power of gratitude is often undersold. Along with this, a lack of gratitude can have negative consequences. Expressing gratitude improves the well-being of both the expresser and recipient, positively impacting feelings of connectedness with others and self-improvement motivation [2, 14, 29]. However, expressers tend to undervalue the positive impact that appreciation can have on the recipient(s), and this often leads expressers to refrain from expressing gratitude [11]. Not only does this prevent recipients from receiving gratitude that would improve their well-being, but it also prevents expressers from maximizing their own well-being. In a study that had participants write gratitude letters and predict how happy recipients would feel upon receiving these letters, it was found that expressers significantly underestimated how positive recipients would feel [11]. Similarly, expressers tend to overestimate the cost of expressing gratitude [10]. Consequently, this leads to unwarranted barriers in expressing gratitude, mitigating the positive outcomes from expressing gratitude.

While any expression of gratitude can benefit both the expresser and recipient, it has been found that expressions in the form of long-form essays and / or letters resulted in greater subjective well-being than short lists [24]. However, any expression of gratitude is still more beneficial for both ends than no expression at all. From the perspective of the recipients, what matters most is some form of expression in the first place.

2.3 Appreciation Systems

Appreciation systems that serve as a platform for users to exchange thanks have been explored in a few different cases. Most commonly, they have been deployed within the workplace as a way for employees to send appreciation amongst themselves, while simultaneously providing an additional metric for managers to access and consider [12, 25]. Many of these systems were designed with the idea of addressing the problem of recognition in mind. One such example of these is Yammer, which is an enterprise social networking system where employees can give praise to each other, resulting in an increased praise count on a per-employee basis [4, 25]. Another example is Bonus.ly, a peer bonus system that gives each employee a fixed budget for showing appreciation for a colleague [25, 26].

Given that these systems were designed with the workplace as the intended setting, the exchange is able to reach other people, including the managers of either or both the sender and receiver. Furthermore, these systems often place an emphasis on acknowledging contributions among peers that would otherwise not occur [25]. Specifically, they aim to highlight contributions that would typically be unnoticed by a manager but still had an impact on peers, for instance. On the other hand, many of these systems differ from each other in terms of who the audience is and the visibility of the appreciation. In Yammer, the audience only consists of the receiver [4, 25], and any appreciation also appears on their corresponding profile page, but in Bonus.ly, the audience is the entire company [25, 26]. Additionally, receiving appreciation within Yammer is the sole result, whereas there is some compensation in the form of a bonus (e.g. monetary) for those that receive appreciation within Bonus.ly. This highlights the different design decisions that can be made within these appreciation systems. Designing an effective appreciation system

requires consideration of these different affordances.

Chapter 3

System Design

In this section, we present an example user scenario, as well as the key elements of the system design of *Hug Reports*. The name *Hug Reports* was inspired by the idea that we wanted our extension to counteract the persistent culture of bug reports in open source.

3.1 User Scenario

Tim is new to web development and recently started building a personal website, so he has been doing a lot of coding using React. Tim uses VS Code to do his coding and has found himself using a lot of built-in web development features provided by React’s open-source code. He recently installed the *Hug Reports* extension to try it out and notices how often he has been using React’s built-in components, indicated by the many raised-hands icons he sees in the gutter next to the line numbers (*realizing reliance on other developers’ code*) (Figure 3.1(a)). Tim realizes that he would not have been able to build his personal website so quickly without these functions and wants to express some appreciation to the developers of them, so he right-clicks on a raised-hands icon next to a line of code and clicks “Say Thanks” (Figure 3.1(b)). Following this, Tim is prompted with a popup asking if he wants to send a more personalized message to the developers, and he decides to write a message in the linked form expressing how their code has allowed him to build a website for the first time (*ability to both easily send a quick thanks and compose a longer message, if desired*) (Figure 3.2). Tim continues to use *Hug Reports* as he codes in VS Code and has found it much easier to thank developers whenever he wants to.

3.2 Design Features

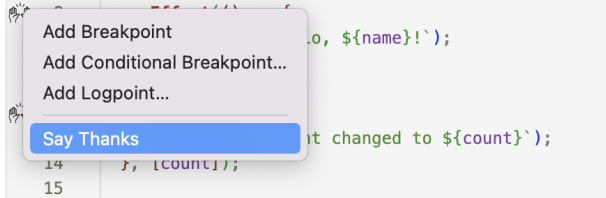
3.2.1 Realizing reliance on other developers’ code

Users have come to take open-source software for granted [5] and are often unaware of how reliant they are on such software [28]. Compounded with the fact that it is not an established norm for users to thank developers in the open-source community for their code, users do not think to express their appreciation in many of these scenarios. As a result, *Hug Reports* visually shows a raised-hands icon in the gutter (i.e. the area to the left of the line numbers) on every

```
1 import React, { useState, useEffect } from 'react';
2
3 function App() {
4   const [name, setName] = useState('');
5   const [greeting, setGreeting] = useState('');
6   const [count, setCount] = useState(0);
7
8   useEffect(() => {
9     setGreeting(`Hello, ${name}!`);
10  }, [name]);
11
12  useEffect(() => {
13    console.log(`Count changed to ${count}`);
14  }, [count]);
15
```

(a) Raised-hands icons

```
1 import React, { useState, useEffect } from 'react';
2
3 function App() {
4   const [name, setName] = useState('');
5   const [greeting, setGreeting] = useState('');
6   const [count, setCount] = useState(0);
7
8   useEffect(() => {
9     setGreeting(`Hello, ${name}!`);
10  }, [name]);
11
12  useEffect(() => {
13    console.log(`Count changed to ${count}`);
14  }, [count]);
15
```



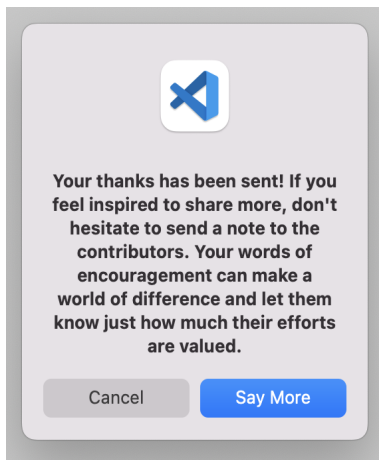
(b) Ability to say thanks

Figure 3.1: (a) The extension displays a raised-hands icon in the gutter next to each line of code that imports or makes use of an imported package, class, or function. (b) Right-clicking on one of the raised-hands icons in the gutter opens an options menu that allows the user to “Say Thanks” for that code.

line that uses code that has been written by others (e.g. an imported package, an imported class, etc.). Rather than only rendering this icon once at the top and having the user specify the specific code they want to thank, we chose to render it on every relevant line in order to provide a signal showing the user their reliance on other developers’ code. Moreover, we render a raised-hands icon given its commonly interpreted meaning of expressing gratitude, in addition to it being easy to discern in the small gutter space.

3.2.2 Ability to say a quick thanks and write a longer message

If a user wants to express gratitude for a particular package, class, or function, they are able to do this with a quick click of the “Say Thanks” option when the corresponding raised-hands icon is right-clicked. In order to promote more expressions of gratitude, this option requires very little effort upfront. If a user is inspired to say more to the developers of the corresponding code, they are subsequently able to do this in a separate form. This gives the user multiple options: 1) they can easily express their gratitude for a particular package, class, or function directly in VS Code;



(a) Option to say more

Say Thanks

Send a personal note to the contributors and let them know how their code has helped you!

Write your note here:

(b) Form for sending more detailed thanks

Figure 3.2: (a) After a user clicks “Say Thanks”, they are presented with an option to write a long-form message. (b) Clicking “Say More” brings the user to this form where they can write a more detailed message of thanks to the developer(s).

2) they can elaborate on their appreciation if desired. By splitting these actions into two, the user is always able to express their gratitude with relatively little effort. Moreover, the effort required for the user to say more than just a quick thanks is still minimized as the user does not have to seek out the developers to thank themselves, nor do they have to figure out how to get in contact with these developers. In doing this, we aim to provide the user flexibility in how they show their appreciation, all while needing minimal effort.

3.2.3 Not seeing recipient names

When a user tries to say thanks, they do not see the names of the developers they are expressing gratitude for. Showing the names of the people getting thanked may result in some potential bias by the users writing the messages [3, 9, 19], and we try to minimize this. If users would like to find out who they are saying thanks to, they can still find the corresponding code online and its most recent developers.

3.2.4 Thanking the 20 most recent contributors

When a user sends a message of thanks for a particular package, class, or function, that message gets sent to the 20 most recent contributors who worked on the corresponding code. We chose the heuristic of the 20 most recent contributors as this number is able to capture the most relevant collaborators on the code. Typically in open-source projects, there are often many developers working on the same piece of code, and so in order for most of their efforts to be represented, we chose to have the 20 most recent contributors get thanked. If there are fewer than 20 developers who have ever worked on the corresponding code, then all of them will get thanked. Furthermore, core developers for any corresponding code are likely to be continuously working on the project

and the ones who worked on the project most recently, and so they would likely be within the 20 most recent contributors.

Chapter 4

Deployment Study

We conducted a small-scale field deployment study to investigate usage of our extension by users and how it was received by the maintainers who were thanked.

4.1 Participants

We recruited 18 participants to try out our extension for two weeks. Convenience sampling was used to recruit these participants, primarily students at Carnegie Mellon University. Participants were aged between 23 and 30, with a median age of 26, and included 12 men and 6 women. Furthermore, a majority of participants were intermediate to advanced programmers. Half of these participants were instructed to use the extension at least 2 times a day on the days they were coding, whereas the other half was free to use it whenever and for however many times they wanted to. This was to ensure we had enough data to analyze. Participants were randomly given a survey every few times they clicked “Say Thanks” asking about their motivations in sending this message, how sending this message made them feel, and how they anticipate the recipients will feel upon seeing this message. All participants were compensated \$20.

4.2 Procedure

Participants indicated whether or not they would be willing to participate in a 30 to 60 minute interview following their usage of the extension for two weeks. Out of the 18 participants, 7 agreed to an interview. These participants were compensated an additional \$15. All interviews followed the same protocol and focused on (1) why users decided to say thanks when they did and the context behind it; (2) when users decided to say thanks; (3) what users decided to say thanks for (i.e. at the package level vs. a more specific level); (4) whether users felt that their thanks was spontaneous or coerced; (5) when users would say thanks again for the same code. To help participants recall what they said thanks for and any messages they wrote, examples were pulled up during the interviews. All interviews were recorded.

Following the completion of the two-week deployment for each of these participants, we aggregated the number of thanks and the specific messages for each package, class, or function that

was thanked. We contacted the 20 most recent developers individually for each of these and presented the aggregated information for the corresponding package, class, or function. Rather than presenting the aggregated thanks to all of the 20 developers together, this information was sent privately to each contributor in order to reduce potential bias and to make it more personalized (each contributor could still figure out who else the thanks went to, as we described the heuristic we used for who was receiving these thanks). Developers were given a survey to complete asking about their experience when receiving these thanks and also indicated whether they would be willing to participate in a 30 to 60 minute interview. By the time of this thesis, we contacted a total of 200 developers, with 25 responding to our survey (12.5% response rate), and out of those who responded to our survey, we interviewed two of these developers. All interviews were recorded. Developers were not compensated for filling out the survey, but those who participated in an interview were compensated \$15.

Chapter 5

Findings

5.1 Extension Data

We collected data on every thanks message sent across the two weeks each participant used our extension for. For each click of “Say Thanks”, we logged the participant’s ID, the line number, the line of code, and the timestamp. We also collected every response for any long-form message of thanks, in addition to every response to the survey asking the participant about the thanks message they just sent. We aggregated the usage data and removed the first entry for each participant, as the first entry corresponded to a trial attempt by each participant from when we helped them with installing the extension. We discuss patterns present in this cleaned usage data and the survey results below.

5.1.1 Most users felt a positive effect from saying thanks

All participants reported feeling either no different than normal or at least some positive effect from saying thanks, with none saying they felt more negative than normal (Figure 5.1). Along with this, most participants anticipated that the developers receiving these thanks would feel at least somewhat more positive than normal upon seeing these thanks (Figure 5.2). Many of these participants thought that the developers would be happy to know that other people were using their code and were finding it useful, with one participant saying the developers “might appreciate that people find their package useful and that it is relevant to new technological trends” and another participant saying that these developers would be “motivated to continue their open source contributions”. Many participants also anticipated that the developers would be at least somewhat surprised to receive these thanks (Figure 5.3).

5.1.2 Thanking at a package / class / function level

Our extension allows a user to say thanks for any piece of imported code, regardless of whether it is an entire package or just a class or function. Saying thanks for a package corresponds to clicking “Say Thanks” on the line of code that imports that package, and saying thanks for a class or function corresponds to either clicking “Say Thanks” on the line of code that imports that class or function, or the line of code that instantiates that class or calls that function. We

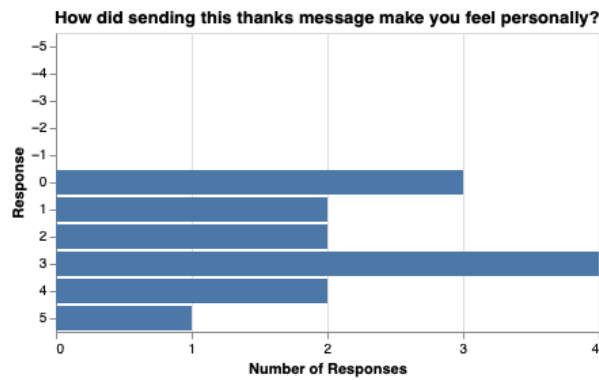


Figure 5.1: All user responses to how sending a thanks message made them feel. -5 corresponds to “much more negative than normal”, 0 corresponds to “no different than normal”, and 5 corresponds to “much more positive than normal”.

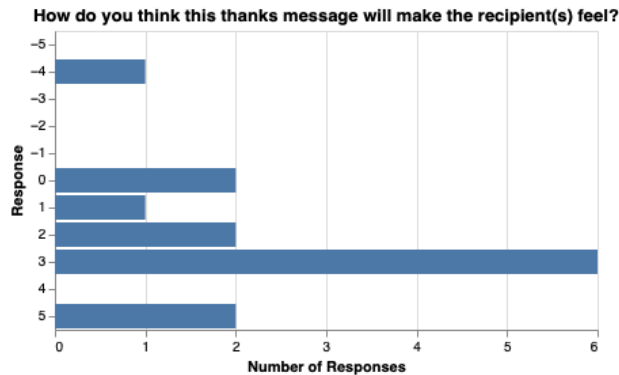


Figure 5.2: All user responses to how they anticipated the recipient(s) of their thanks message would feel. -5 corresponds to “much more negative than normal”, 0 corresponds to “no different than normal”, and 5 corresponds to “much more positive than normal”.

found that users tended to say thanks for an entire package or class on par with only a function. Figure 5.4 shows that 53% of thanks was given to a package or class, and the remaining thanks was for a specific function.

5.1.3 Thanking near the top of the file

A user is able to say thanks for any imported package, class, or function on either the line that code gets imported or where it is actually used in the code. We found that even though a user is able to say thanks from any part of the file that makes use of imported code, participants tended to say thanks (i.e. click “Say Thanks”) near the top of the file where external code is typically imported. Figure 5.5 shows the distribution of the number of participants who said thanks at various parts of the file. It is likely the case that participants had varying levels of file lengths, but even so, Figure 5.5 highlights that the majority of thanks were sent from within the first 20 lines of any code file.

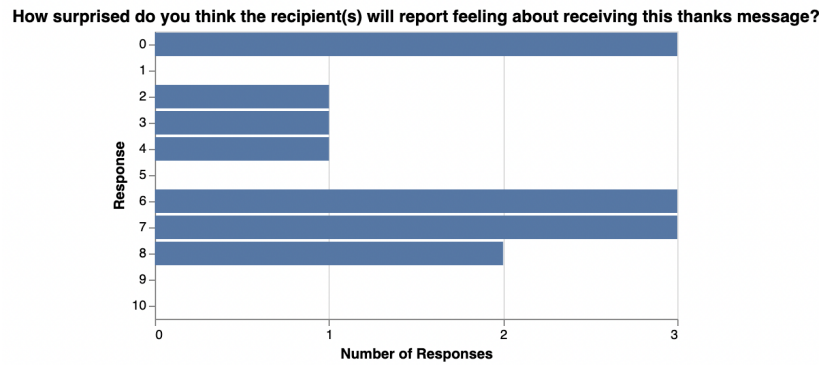


Figure 5.3: All user responses to how surprised they anticipated the recipient(s) of their thanks message would be. 0 corresponds to “not at all surprised” and 10 corresponds to “extremely surprised”.

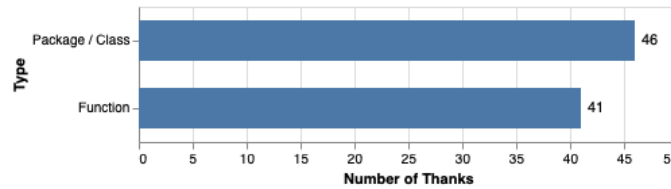


Figure 5.4: What users chose to say thanks for (package / class vs. function). Users said thanks for a package / class a similar number of times as for a function.

5.2 Participant Interviews

Across our conversations with the participants, we identified common patterns that we describe in this section. We discuss these patterns below and accompany them with quotes that reflect points raised by multiple participants.

5.2.1 Users started to notice their reliance on and gratitude for other developers’ code

Our extension places a raised-hands icon next to any line of code that uses an imported package / class / function, and so users are able to explicitly see all instances where their code is making use of another developer’s code. Participants highlighted how seeing this icon served as a reminder that they were using and relying on other developers’ code.

Maybe subliminally, I was realizing just how much [code] I use that’s not my own.
(P4)

One participant mentioned how the extension also raised awareness about how they actually felt when using other developers’ code:

[The extension] kind of raised my awareness. I think it kind of reminds me to express my gratitude. I realized that I just took some of the code packages for granted because there are so many people using them, but they’re also free, and they’re open

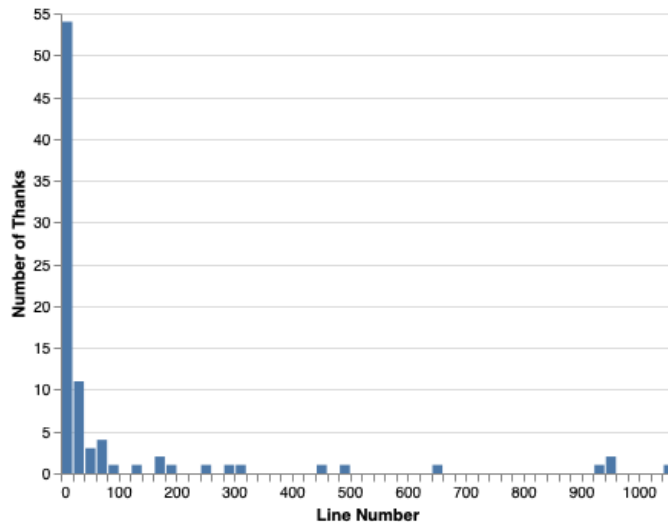


Figure 5.5: Distribution of line numbers at which users said thanks for a package, class, or function. Users tended to say thanks at the beginning of a file.

source, so I can imagine how many hours those programmers have spent behind the scenes. Those little [icons] just served as a daily reminder. (P3)

Participants felt as if were made more aware of their reliance on other developers' code through the extension by seeing every location of this reliance through the raised-hands icon, and this subsequently made them feel gratitude for being able to use this code.

5.2.2 Users tended to not write long-form thanks messages

Our extension allows a user to send a quick message of thanks with the click of the “Say Thanks” option, but they can also opt to send a long-form message following this. When asked if they wrote any long-form messages, many participants reported not doing this for a few reasons. Participants tended to not write long-form thanks messages in the interest of the maintainers. Participants did not want to bother the maintainers with long messages, as they assumed that maintainers would already be busy enough without having to read additional messages:

I thought [the maintainers] must be busy, and they would be happier if they received a message, but they don't have much time to read through all the messages they receive. (P2)

Similarly, P5 said that they were not sure when their thanks message would get sent and that they do not want to bother the maintainer with “a random message at 2am in the morning”. Participants also cited how they were unsure of what to write in a long-form thanks message as a reason for not writing them that often or at all. For instance, P5 mentioned how “I didn't know exactly what I should say to them because I didn't know who they are”, and P7 said that “to offer more detailed thanks, I think I would have to appreciate the nuances, and I think I'm not a good enough developer or expert enough developer to actually appreciate a lot of those things”. Moreover, one of the participants pointed how it was much more difficult to be specific about positive feedback than negative feedback:

I think this [function] is good, but honestly, I feel like it's a lot easier to write something when things are wrong, like 'this failed', ... it's much easier like 'here are my inputs, here's what failed, and here's what you need to work on' versus like 'this was good for my application'. (P6)

We found that even when given the option of writing a long-form message to accompany a simple thanks, users tended to forego this in the interest of saving both their own time and the maintainers' time, as well as being unsure of what to exactly express.

5.2.3 Users tended to say thanks reactively after using code rather than proactively, and all at once

While using our extension, participants noted that they were more likely to engage with it and say thanks for a package, class, or function after they had successfully made use of it. In other words, most participants only said thanks upon successful use of a particular piece of code, such as completing the setup for an application or writing code that properly runs using imported code, rather than preemptively. For instance, P1 mentioned that they said thanks after they had 'run the first couple of things and set it up', and P3 noted that 'success is actually a key metric' for them in deciding whether or not to thank a package, class, or function. However, one participant mentioned that when they would say thanks depended on how well they already knew the package, class, or function:

I think it depends. For packages that I've known for a while, and now that I am able to thank them, I'll just thank them preemptively. For packages I haven't used before, I'll probably try to use it before I thank them. (P2)

Furthermore, participants found that they tended to say their thanks all at once, especially if their thanks was reactive rather than proactive. Whereas users are able to say thanks however many times they want and whenever, many participants reported using the extension multiple times at a given point in time, usually when they suddenly remembered they wanted to say thanks. For example, P7 said they were "speed thanking a bunch of people" at one point. This usually occurred when a user remembered the extension existed:

I think [sending thanks] was very batch-like. When I remembered that [the extension] existed, I would do a bunch. (P6)

Another participant reported a very similar sentiment:

I think I tried to batch them like at once, or at least that's how I remember it. I tried to do it for some JavaScript and npm packages. It feels a little bit awkward to say thanks to the same team repeatedly, I guess, so whenever I remember, I just, you know, thank all of them, or thank them multiple times or something. (P2)

Users tended to have similar patterns of using the extension by sending thanks reactively and in batches.

5.2.4 Users felt that their thanks was spontaneous and not coerced

When using our extension, participants felt that they were able to use the extension in the moment and whenever they felt like expressing thanks. Participants expressed how the extension allowed

them to say thanks spontaneously and that they did not feel coerced or obligated to say thanks:

I don't feel coerced at all. I don't remember me feeling obligated to thank them. It felt more like a nice opportunity for me to thank them without having to find out how to contact them. (P2)

Some participants also mentioned that the extension could be even more obvious in reminding them to say thanks. P1 said that “something more aggressive on very specific occasions would be better”, such as “a simple threshold-based thing where after you've imported something ten times, you either show a little popup in the status bar or some glowing icon”, because they found the icon becoming “background noise” due to how often their code made use of other developers' code.

5.2.5 Users felt that their thanks was quickly discharged and only accumulated in new use cases

Our extension does not limit the number of times a user can say thanks for a particular package, class, or function. Yet, participants often described feeling as if they were able to discharge their gratitude in a single setting (P1, P3, P4), with P4 saying that they “got it out of [their] system”. One participant also said that they did not feel the need to say thanks again because they did not want to bother those receiving the messages and that it would take additional effort:

I feel like in our culture, we just don't like bothering other people, and also, I think I'm sort of more of an introvert. I think saying 'thank you' once is already good enough. I think if you say it multiple times, first of all, I have to spend extra effort on it. (P3)

Additionally, participants noted how they would only say thanks again for a particular package / class / function if they were using it again in a new setting (e.g. a new project), or if they discovered a novel feature offered by the same package, for instance:

I think if there is a unique functionality that hadn't been captured before, so like you know, I have use case A, and then I find out there's a use case B, I would definitely thank again. It wouldn't be like I think for use case A, ten weeks go by, and I think for use case A again. It would have to be a separate use case. (P6)

After thanks for a particular package / class / function has been expressed, participants typically did not feel the need to say thanks for it again unless they found a new reason to.

5.2.6 Users were intentional about moments in time for saying thanks

The extension always displays a raised-hands icon next to any line of code that uses other developers' code, and so users of the extension can choose exactly when they actually say thanks. There was a common pattern that emerged regarding when participants chose to and chose not to say thanks (either a quick thanks or a long-form thanks). Specifically, a majority of the participants noted that they refrained from saying thanks whenever they were in the middle of completing a task (P1, P4, P5, P6, P7). Participants reported that stopping to say thanks, especially to write a long-form message, would disrupt them if they were in the flow of working:

It just takes more time, and if I'm doing something which requires a lot of my head space, I wouldn't be very inclined to give that time and mental energy to write a longer thanks message. (P1)

On the contrary, participants found that it was a good time to express thanks whenever there was a pause in their coding or if they were stuck:

I got stuck in the moment like in my coding, and then, I was like my cognitive load wants to get some rest, and I thought it was a good moment to play around with the system, and then I composed a longer thanks message. It's kind of a refreshing moment. (P5)

Participants made use of the extension at the most convenient times for them and also refrained from using the extension when it was inconvenient for them, and they conformed their usage of the system to their own programming patterns.

5.3 Developer Survey Data

The survey given to developers whose code was thanked focused on (1) how receiving the thanks and the thanks messages themselves made them feel and (2) what else they would have liked to know upon receiving these thanks. We discuss trends and common points raised by the developers in the survey responses below.

5.3.1 Developers were surprised to receive these thanks messages

Out of the developers who responded to our survey, most of them responded saying they were at least somewhat surprised to receive this hug report (i.e. the thanks sent by the participants in our study), with over 20% of respondents saying they were extremely surprised (Figure 5.6). This is somewhat aligned with how surprised the participants anticipated these developers would be (Figure 5.3), except developers tended to be somewhat more surprised (the values for the scales are unaligned, but they correspond to the same range of “not at all surprised” to “extremely surprised”). Additionally, most of these developers reported feeling either no different than normal or at least somewhat more positive than normal, with five developers saying this hug report made them feel much more positive than normal (Figure 5.7). This aligned very closely with how participants anticipated the recipients would feel, with most saying that the recipients would either feel no different than normal or at least some level of positive effect (Figure 5.2).

Most developers also mentioned that they either never or rarely received thanks. Only a few developers responded saying they had received some messages of thanks in the past, with one saying they received ‘stars on GitHub projects once every week and issues on GitHub once every couple of months’ and another saying they have received ‘a few thanks and messages on GitHub and Discord’.

5.3.2 Developers want to receive more detailed thanks messages

While the developers reported appreciating the idea of receiving messages of thanks from other people using their code, with one developer responding that they ‘felt happy and feel motivated

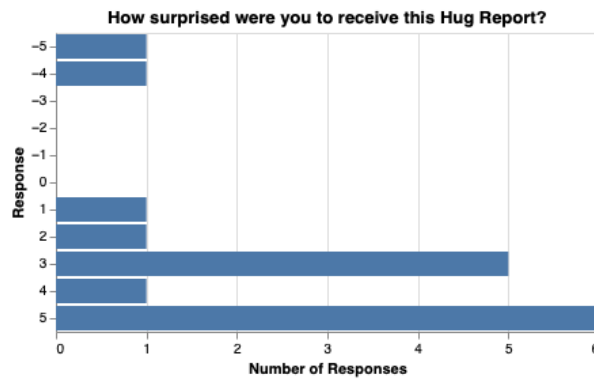


Figure 5.6: All contributor responses to how surprised they were to receive the thanks users sent. -5 corresponds to “not at all surprised” and 5 corresponds to “extremely surprised”.

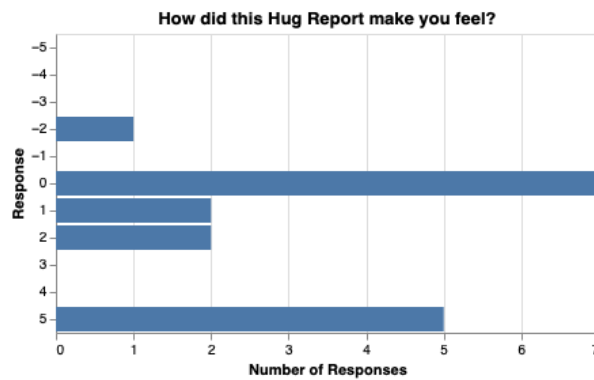


Figure 5.7: All contributor responses on how they felt upon receiving the thanks sent by users. -5 corresponds to “much more negative than normal”, 0 corresponds to “no different than normal”, and 5 corresponds to “much more positive than normal”.

to do more’, many of them also noted that the thanks they received were either too generic or unclear about what the users were thanking them for. Developers were hoping to see more substance and details in the messages of thanks:

There is nothing personalized about the report, so all it is telling me is that people are using that code, and that is no surprise. I much prefer hearing when a piece of code made a true difference to someone, e.g. if they wrote ‘this function in SciPy shaved two months off of my PhD’ or ‘this library you wrote completely transformed the way we were able to execute our project’ or even ‘I love the API you’ve designed so much!’.

Similarly, developers felt ‘bemused’ about what they were being thanked for because the report did not feel personalized enough, as these developers received an aggregate count of the number of users who wanted to thank them and potentially some messages of thanks without much detail:

Receiving the hug report was a wonderful surprise and brightened my day. However, the specific things I’m being thanked for honestly make no sense at all. I’m going to choose to believe that they really are happy about specific things I’ve done, anyway

:.).

Overall, while the developers expressed feeling surprised and happy to receive these messages of thanks, they would rather see more detailed messages specific to them.

5.3.3 Developers sometimes felt that the thanks messages were misdirected

All thanks messages for a particular package, class, or function were directed at the 20 most recent developers. However, in some cases, developers reported feeling as if the hug report they received should rather be directed to others and not themselves:

I am not the main author of this work, nor have I made significant contributions.
This report should go to others.

One developer also gave suggestions as to who exactly should be receiving the hug report instead:

This is really nice, but I think you should use the author list in the project release or the .mailmap file.

Some of these developers did not feel as if their relevant contributions were significant enough for them to be receiving the hug report, with one developer even saying that the hug report seemed “kind of spammy” because they were not very involved with the relevant piece of code.

5.4 Maintainer Interviews

At the time this thesis was written, we completed two interviews of developers, who are also maintainers, who received messages of thanks. We discuss findings from these two interviews, and we plan to conduct more interviews to supplement these results.

5.4.1 Maintainers appreciated receiving thanks from people

Maintainers juggle a multitude of responsibilities, ranging from fixing bugs and developing new features to overseeing a project’s success. However, the maintainers we interviewed mentioned that they rarely, if ever, receive any form of appreciation from their end users and that they appreciated what *Hug Reports* was trying to do:

If you look at the entire ecosystem, the truth is, most maintainers, like I said, for most maintainers in most of the other projects I maintained, it is like what, I think 2 thanks, or whatever, but in general, you don’t even hear from your user base unless there’s a bug, right, so I certainly think [*Hug Reports*] is a good idea.

The maintainers noted that it felt nice to receive these messages of thanks because it adds a ‘human touch’ to the process:

it feels good to know that somebody’s using the stuff you put out there. Often you have no idea. ... So it’s nice to hear from a human.

Maintainers did appreciate that they received thanks from actual people who were saying thanks for the work they had done, especially since this was not a common occurrence for them. Moreover, they found value in these messages of thanks being directed at specific people.

5.4.2 Granularity of appreciation should be project-specific

Regarding who should receive the messages of thanks sent by the participants, we decided to send them out to the 20 most recent developers for the relevant package, class, or function, and we did not modify this heuristic based on different projects. From our interviews with these maintainers, we found that they felt that it would be better to tailor who the thanks should go to depending on the project structure and community. One maintainer mentioned that it might not make sense to be thanking the 20 most recent developers for a specific package, class, or function, especially in a large project because it could be the case that it does not accurately reflect the effort of all developers involved. In this case, the maintainer suggested looking at the release notes instead to find the people appropriate for receiving these messages of thanks.

Furthermore, one maintainer mentioned that this heuristic of sending out the messages of thanks to the 20 most recent developers leads to “misdirected” thanks. This maintainer felt that these messages “should have gone to people who have made significant contributions”, instead, which we did not account for; in our current model, as long as a developer made any contribution (e.g. even changing a single line of code), they would show up as having made a contribution and would receive these messages of thanks. They also noted that these “misdirected” thanks could dilute the value of them:

If I get something like this [hug report], and I feel like it’s misdirected, then it feels like it’s less valuable if I was a core contributor, because you’re also thinking some rando like that guy who didn’t really do anything, why is he getting just as much thanks as I do?

Maintainers thought that the granularity at which they received messages of thanks would have to depend on the context for their contribution, as well as the project community.

5.4.3 Preferred form of receiving thanks

The maintainers we interviewed had a few thoughts regarding their preferences for receiving messages of thanks, in terms of how often and how it would appear. One maintainer suggested giving a developer the option of deciding whether or not they wanted to receive these messages:

It’ll vary between individuals whether they want to receive [the hug report] or not, and it’s very hard to make a generalization, right? It might be better to sort of aggregate those metrics at a project level and say, here, you can find them if you want. You do that once, and then they can opt in if they want to get it the next time, or if they just want to know that it’s there somewhere.

Another maintainer commented on how it is important to balance between the thoughtfulness of the messages with the amount of time it would take to consume them, especially in the interest of time for busy maintainers:

You have to balance the need for making someone feel good, which is, you know, very much appreciated, and the need for, you know, reducing the amount of stuff that someone has to look at, which is already pretty high.

Overall, the maintainers also noted that they found it useful to receive both a metric regarding the number of thanks they got, in addition to the long-form messages that were more personal.

Chapter 6

Discussion

Through our deployment, we found out how users interacted with *Hug Reports*, in addition to how developers on the receiving end of these thanks perceived these expressions of appreciation. We identified aspects of our extension that worked well, along with areas open for improvement. Here, we reflect on opportunities and open challenges in the design of an appreciation system within the open source ecosystem.

6.1 Incorporating a Two-Tier Thanking System

Having a two-tier thanking system that gives a user the option to send both a lightweight thanks with a single click and a long-form message is beneficial to both the user and developer who receives the thanks. From the user perspective, having the option to be able to say thanks with a single click encourages users to say thanks more often given the low amount of effort it takes. By lowering the barrier to expressing gratitude, a user is more likely to send their thanks to developers whose code they use, which is all the more important in an environment where these developers rarely receive any thanks. On the other hand, developers can still see that users are using their code and are appreciative of it. It is a metric that the developer can make use of, whether it be for personal recognition or professional purposes.

Research has shown the benefits — positive feelings of connectedness and self-improvement motivation, to name a few — that giving gratitude has on both the expresser and recipient [24]. By allowing users to send a message detailing their gratitude, this not only benefits them but also the recipients. Moreover, developers who receive these long-form messages of thanks are able to feel a greater personal connection with those using their work, and they can see messages that give them something more than just a metric, as provided by the lightweight thanks users can send. However, it is important to note that by having this option of sending a long-form message, recipients expect to receive messages that actually convey a user’s gratitude. The developers who responded to our survey upon receiving the thanks directed at them mentioned wanting to see messages with more substance and detail that get to why the user was grateful and what exactly they were grateful for. In order to address this, a potential solution would be to provide some scaffolding, prompting questions, or examples to users regarding what a message should contain.

6.2 Improvements to *Hug Reports*

In order to build a useful appreciation system, it must work as a user would expect. Most users we interviewed noted that the extension in its current form allowed them to perform the functionalities they wanted when expressing appreciation. However, a few participants mentioned that they encountered quite a few false positives where the extension placed a raised-hands icon next to code that was not written by another developer. These false positives should be reduced and completely removed, if possible, in an improved design.

Additionally, while most participants found the extension suitable for their needs, they mentioned that it would be nice to be able to thank code that has been written by others but was not imported (e.g. using code from a cloned package). While not as critical, addressing this would allow even more developers to receive thanks for their work.

If *Hug Reports* were to be deployed at a large scale and integrated into an existing version control system, such as GitHub, it should give users the option of choosing when to make use of this extension with additional configurations relating to when a user wants to be reminded to say thanks, information they receive regarding their thanking patterns, etc.

6.3 Challenges in Building an Appreciation System for Open Source

In designing and building an appreciation system for open source, there are still a few open challenges. One critical challenge is finding a way to balance the differences between various open-source projects. As noted in the interviews with the maintainers, open-source projects can vastly differ in size and how they operate, and ensuring who the appropriate people for receiving these messages of thanks are will differ based on these factors. Additionally, determining how these developers should be receiving these messages of thanks is another challenge that arises from these differences. There are many possibilities for determining who receives these messages and how they should be receiving these messages, and figuring out which solution is the most appropriate will require further iterations of this system paired with understanding how it is received by such developers. In order to make these challenges feasible to tackle, it might be helpful to start by focusing on a specific subset of open-source projects before generalizing to the open source ecosystem, especially as there will likely be differences in how developers perceive these messages of thanks depending on the environment they operate in.

6.4 Future Work

The research presented in this thesis is ongoing. Moreover, this is a design space that still has a lot of opportunity for exploration. As discussed above, we found that certain features in such an appreciation system are desirable, in addition to features that are not. The first step would be to incorporate the feedback that users of the extension and developers who received messages of thanks gave into our system. In order to build an extension that will be used and beneficial to both the users and developers, an iterative design process is necessary. More concretely, this will

involve cleaning up the extension to minimize errors in detecting use of other developers' code, as well as balancing the way developers want to receive messages of thanks. Upon redesign of our system to take into account the feedback, the next step will be to perform another deployment, and this process should be repeated until user and recipient needs are satisfied. The ultimate goal will be to deploy this extension broadly, so that any user can make use of this extension, and for developers to subsequently feel more appreciated for all of the work they do.

Chapter 7

Conclusion

This thesis presents *Hug Reports*, a VS Code extension that gives users the opportunity to say thanks to developers whose code they rely on, along with a deeper study of how an appreciation system in the open source ecosystem should be built. We recruited participants to try out our extension over the course of a two-week period, and we found that users started to notice their reliance on and gratitude for other developers' code they used, in addition to patterns for how they tended to say thanks and how they felt about saying thanks. We then surveyed the developers whose code these users sent messages of thanks to and found how they felt upon receiving these thanks and what they thought about these messages of thanks, finding that developers felt positively about receiving these messages of thanks but sought more detail and scoping in them. Based on our experience from developing and findings from deploying *Hug Reports*, we presented design features and challenges we believe should be considered when building an appreciation system in the open source ecosystem.

Bibliography

- [1] Anonymous. Leaving toxic open source communities, July 2014. URL <https://modelviewculture.com/pieces/leaving-toxic-open-source-communities>. 1
- [2] Christina N Armenta, Megan M Fritz, Lisa C Walsh, and Sonja Lyubomirsky. Satisfied yet striving: Gratitude fosters life satisfaction and improvement motivation in youth. *Emotion*, 22(5):1004, 2022. 2.2
- [3] Wendy Conaway and Sonja Bethune. Implicit bias and first name stereotypes: What are the implications for online instruction?. *Online Learning*, 19(3):162–178, 2015. 3.2.3
- [4] Tech Crunch. Yammer adds badges and in-line videos to enterprise communications app, May 2011. URL <https://techcrunch.com/2011/05/31/yammer-adds-badges-and-in-line-videos-to-enterprise-communications-app>. 2.3
- [5] Nadia Eghbal. *Roads and bridges: The unseen labor behind our digital infrastructure*. Ford Foundation, 2016. 3.2.1
- [6] Fabio Ferreira, Luciana Lourdes Silva, and Marco Tulio Valente. Turnover in open-source projects: The case of core developers. In *Proceedings of the XXXIV Brazilian Symposium on Software Engineering*, pages 447–456, 2020. 2.1
- [7] Javier Luis Cánovas Izquierdo and Jordi Cabot. Enabling the definition and enforcement of governance rules in open source systems. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 2, pages 505–514. IEEE, 2015. 1
- [8] Carlos Jensen, Scott King, and Victor Kuechler. Joining free/open source software communities: An analysis of newbies’ first interactions on project mailing lists. In *2011 44th Hawaii international conference on system sciences*, pages 1–10. IEEE, 2011. 1
- [9] Joseph Kasof. Sex bias in the naming of stimulus persons. *Psychological bulletin*, 113(1): 140, 1993. 3.2.3
- [10] Amit Kumar. Some things aren’t better left unsaid: Interpersonal barriers to gratitude expression and prosocial engagement. *Current Opinion in Psychology*, 43:156–160, 2022. 2.2
- [11] Amit Kumar and Nicholas Epley. Undervaluing gratitude: Expressers misunderstand the consequences of showing appreciation. *Psychological science*, 29(9):1423–1435, 2018. 2.2
- [12] Z Lahey. The art of appreciation: Top-tier employee recognition. Technical report, Techni-

cal report, Aberdeen Group, 2015. 2.3

- [13] Nolan Lawson. What it feels like to be an open-source maintainer, March 2017. URL <https://nolanlawson.com/2017/03/05/>. 1
- [14] Kristin Layous, Kate Sweeny, Christina Armenta, Soojung Na, Incheol Choi, and Sonja Lyubomirsky. The proximal experience of gratitude. *PloS one*, 12(7):e0179123, 2017. 2.2
- [15] Abby C Mayes. Maintaining balance for open source maintainers. URL <https://opensource.guide/maintaining-balance-for-open-source-maintainers/>. 1
- [16] Courtney Miller, David Gray Widder, Christian Kästner, and Bogdan Vasilescu. Why do people give up flossing? a study of contributor disengagement in open source. In *Open Source Systems: 15th IFIP WG 2.13 International Conference, OSS 2019, Montreal, QC, Canada, May 26–27, 2019, Proceedings 15*, pages 116–129. Springer, 2019. 2.1
- [17] Courtney Miller, Sophie Cohen, Daniel Klug, Bogdan Vasilescu, and Christian Kästner. ” did you miss my comment or what?” understanding toxicity in open source discussions. In *Proceedings of the 44th International Conference on Software Engineering*, pages 710–722, 2022. 1
- [18] Alessandro Narduzzo and Alessandro Rossi. The role of modularity in free/open source software development. In *Free/Open source software development*, pages 84–102. Igi Global, 2005. 1
- [19] Aditya Pal and Scott Counts. What’s in a@ name? how name value biases judgment of microblog authors. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 5, pages 257–264, 2011. 3.2.3
- [20] Marc Palyart, Gail C Murphy, and Vaden Masrani. A study of social interactions in open source component use. *IEEE Transactions on Software Engineering*, 44(12):1132–1145, 2017. 1
- [21] Huilian Sophie Qiu, Bogdan Vasilescu, Christian Kästner, Carolyn Egelman, Ciera Jaspan, and Emerson Murphy-Hill. Detecting interpersonal conflict in issues and code review: cross pollinating open-and closed-source approaches. In *Proceedings of the 2022 ACM/IEEE 44th International Conference on Software Engineering: Software Engineering in Society*, pages 41–55, 2022. 2.1
- [22] Huilian Sophie Qiu, Anna Lieb, Jennifer Chou, Megan Carneal, Jasmine Mok, Emily Am-spoker, Bogdan Vasilescu, and Laura Dabbish. Climate coach: A dashboard for open-source maintainers to overview community dynamics. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–18, 2023. 2.1
- [23] Naveen Raman, Minxuan Cao, Yulia Tsvetkov, Christian Kästner, and Bogdan Vasilescu. Stress and burnout in open source: Toward finding, understanding, and mitigating unhealthy interactions. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: New Ideas and Emerging Results*, pages 57–60, 2020. 2.1
- [24] Annie Regan, Lisa C Walsh, and Sonja Lyubomirsky. Are some ways of expressing gratitude more beneficial than others? results from a randomized controlled experiment. *Affec-*

tive Science, 4(1):72–81, 2023. 2.2, 6.1

- [25] Emma Spiro, J Nathan Matias, and Andrés Monroy-Hernández. Networks of gratitude: Structures of thanks and user expectations in workplace appreciation systems. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 10, pages 358–367, 2016. 2.3
- [26] Los Angeles Times. Bonus.ly looks to motivate workers with peer-to-peer bonuses, May 2013. URL <https://www.latimes.com/business/technology/la-fi-tn-bonus-ly-startup-motivates-unmotivated-20130510-story.html>. 2.3
- [27] Parastou Tourani, Bram Adams, and Alexander Serebrenik. Code of conduct in open source projects. In *2017 IEEE 24th international conference on software analysis, evolution and reengineering (SANER)*, pages 24–33. IEEE, 2017. 1
- [28] Marat Valiev, Bogdan Vasilescu, and James Herbsleb. Ecosystem-level determinants of sustained activity in open-source projects: A case study of the pypi ecosystem. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 644–655, 2018. 3.2.1
- [29] Lisa C Walsh, Annie Regan, and Sonja Lyubomirsky. The role of actors, targets, and witnesses: Examining gratitude exchanges in a social context. *The Journal of Positive Psychology*, 17(2):233–249, 2022. 2.2
- [30] David Gray Widder and Dawn Nafus. Dislocated accountabilities in the “ai supply chain”: Modularity and developers’ notions of responsibility. *Big Data & Society*, 10(1): 20539517231177620, 2023. 1
- [31] Sawyer X. I am stepping down from psc and core, effective immediately, April 2021. URL <https://perl.topicbox.com/groups/perl-core/T7a4f1bf9e069641f>. 1
- [32] Yiqing Yu, Alexander Benlian, and Thomas Hess. An empirical study of volunteer members’ perceived turnover in open source software projects. In *2012 45th Hawaii International Conference on System Sciences*, pages 3396–3405. IEEE, 2012. 1