WHY DO DEVELOPERS PARTICIPATE IN OPEN SOURCE SOFTWARE PROJECTS?

Version 1.01/2024.1.1

Lappeenranta–Lahti University of Technology LUT

Degree Programme in Software Product Management and Business, Master's thesis

2024

Duc Thinh Tran

Examiners: Title First name and Last name (optional degree)

Title First name and Last name (optional degree)

Abstract

Lappeenranta–Lahti University of Technology LUT
LUT School of Engineering Science
Degree Programme in Software Product Management and Business

Duc Thinh Tran

Why do developers participate in open source software projects?
Version 1.01/2024.1.1

Master's thesis
2024
42 pages, 9 figures, 5 tables and XX appendices
Examiners: Title First name and Last name (optional degree) and Title First name and Last name (optional degree)
Keywords: open source software motivation, systematic literature review, software development, barriers, social impact

Acknowledgements

I would like to express my deeply thanks to my thesis supervisor, Associate Professor Antti Knutas, for his invaluable guidance and support throughout this project. His expertise in civic technology software engineering was instrumental in shaping my research and keeping me focused. I especially appreciate his insightful feedback.

I am particularly grateful to my classmates, espicially Nan Yang and Roshan Devullapalli, for their invaluable co-operation and encouragement throughout this master program. Their willingness to brainstorm ideas and provide encouragement proved to be a tremendous help, especially during the thesis development stage. Our experience as teammates in previous courses fostered a strong foundation for collaboration, making this journey all the more rewarding.

My heartfelt thanks to my friends in Lahti and Helsinki for their incredible support during my time in Finland.

Last I would also like to express my sincere gratitude to my family, my relatives and especially my mother for her unwavering love. Her constant encouragement, especially during challenging moments, motivated me to persevere and finish my thesis. I am truly fortunate to have her in my life.

Despite our commitment, I'm aware that this project is still inadequate and contains inevitable errors. I would like to receive feedback from lecturers in order to improve furthermore.

Duc Thinh Tran
Lahti, May 2024

Acronyms

ACM Association for Computing Machinery

GPL General Public License

IEEE Institute of Electrical and Electronics Engineers

OSS Open-source software

PR Pull Requests

SLR Systematic Literature Review

Table of contents

List of Figures

List of Tables

1 Introduction

In today's rapidly evolving landscape of software development, open-source software has become a transformative force [11]. Its collaborative model fuels innovation by making code accessible, fostering a global community of developers, and lowering barriers to entry.

The widespread adoption of the internet created a natural environment for open-source collaboration and distribution [32]. Additionally, the emergence of cloud computing and agile methodologies has further streamlined the development process, empowering developers to build and deploy open-source solutions with unprecedented speed and flexibility [30].

Moreover, the exponential growth of data highlights the need for solutions capable of handling vast amounts of information [3]. This need drives the development of fields such as big data and data science, which often rely heavily on open-source tools and libraries.

The accessibility and adaptability of open source software have democratized the software landscape. It offers a valuable alternative to proprietary solutions, particularly for individuals, start-ups, and organizations with limited resources. The collaborative nature of open-source development often leads to rapid innovation and the creation of robust, secure, and continuously improving software solutions.

While the benefits of open-source software are widely recognized - from its cost-effectiveness and adaptability to enhanced security and strong community support – the motivations that drive developers to contribute their time and expertise to these projects remain less understood. This gap in our understanding highlights an important question: why do companies and individuals dedicate themselves to creating and maintaining the open-source software that powers so much of our digital world?

Investigating this question is crucial for several reasons. Understanding developer motivations offers insights into how we can ensure the long-term sustainability of open-source projects. Furthermore, companies and organizations hoping to attract top talent within the open-source space would benefit from knowing what drives developer participation within these communities. Uncovering these motivations can also shed light on how open-source projects function, how collaboration occurs, and how knowledge is shared and developed.

1.1 Objectives

This research project ventures beyond the surface of open-source software development, delving into the intricate web of motivations that drives developers within this collaborative

community. By employing a multifaceted approach that combines empirical analysis with in-depth qualitative inquiries, the study seeks to illuminate the underlying forces that propel developers to contribute. These forces encompass not only the tangible incentives but also the value systems and socio-technical factors that influence their engagement. Ultimately, this investigation aims to provide a more nuanced understanding of the dynamics that shape the ever-evolving landscape of modern software development, with a specific focus on the crucial role played by open-source communities.

1.2 Motivation

As a former software engineer engrossed in the dynamic world of software development, my daily experience was enriched by the constant use of open-source software. From leveraging fundamental frameworks like ReactJS from Facebook to seamlessly integrating libraries and packages sourced from NPM, open-source solutions became an indispensable aspect of my workflow. Their ubiquitous presence became so ingrained in my routine that their influence often went unnoticed, seamlessly woven into the fabric of my daily tasks. However, as I transition into the realm of program software product management, I find myself drawn to exploring a new frontier – one that encompasses not only the technical aspects but also delves into the world of human psychology. It is this intersection of technology and human behavior that piques my curiosity and drives my desire to embark on this research endeavor.

Through this thesis, I aspire to investigate the nuanced interplay between open-source software adoption and the human factors influencing developer behavior and decision-making processes. By unraveling these complexities, I aim to contribute to a deeper understanding of how human elements shape the adoption, utilization, and evolution of open-source software, ultimately informing more effective software product management strategies.

2 Research questions

This thesis seeks to explain the factors influencing developer participation in open-source projects. To achieve this, the study will delve into three key areas. First, it will explore the primary motivations driving developers to contribute their time and expertise to collaborative open-source endeavors. Second, the research will investigate the impact of social dynamics within these communities. This includes examining how interactions with other developers and potential networking opportunities influence a developer's decision to participate and their ongoing engagement. I would like to approach these two questions using a Systematic Literature Review methodology, which will be discussed in detail in chapter 4.1.

Finally, the study will identify the barriers and challenges that developers encounter when working on open-source projects. It will then explore the strategies and solutions employed by developers to navigate these obstacles and ensure continued participation. By addressing these multifaceted aspects, this thesis aims to provide a comprehensive understanding of the dynamics shaping developer involvement in the open-source landscape.

2.1 What are the primary motivations driving developers to participate in open-source software projects?

This question seeks to understand the underlying reasons why developers choose to contribute their time, skills, and expertise to collaborative software development efforts that are open and freely accessible to the public. By exploring this question, the study aims to uncover the diverse range of factors that incentivize developers to engage in open-source projects. These motivations may vary widely among individuals and can include a combination of intrinsic and extrinsic factors.

2.2 To what extent do social dynamics, such as community interactions and networking opportunities, impact developer participation in open-source software projects?

Open-source software thrives on the contributions of volunteer developers. While individual motivations to participate have been explored, a gap exists in understanding how the social environment itself fosters engagement. The second question investigates the impact of social dynamics within open-source projects. Specifically, it examines how interactions within the community and opportunities to build professional networks influence the level of developer participation. By studying these dynamics, the study aims to illuminate how open-source communities can be nurtured to maximize developer engagement and project success.

2.3 What barriers or challenges do developers encounter when participating in open-source software projects, and how do they navigate these obstacles?

Last, a crucial aspect of this research involves understanding the roadblocks developers encounter when contributing to open-source projects. This includes technical hurdles like complex codebases, time constraints, and communication challenges within geographically dispersed communities. Additionally, factors like unclear project direction and unfamiliarity with open-source etiquette can hinder participation. However, the research will also explore how developers navigate these obstacles, examining how mentorship, flexible contribution models, and strong community building practices can foster a more inclusive and engaging open-source development environment.

3 Background

Open-source software (OSS) has become a cornerstone of modern software development. It refers to software that is freely available for use, modification, and distribution. The open-source movement has gained significant traction in recent years, driven by a collaborative development philosophy and the benefits it offers to both developers and users.

3.1 Origins and Early Practices

The origins of open source software lie in the early days of computing, where a culture of sharing and collaboration thrived. Programmers in academic and research settings freely exchanged and modified code, driven by a desire to improve the technology they worked with. However, as software began to be commercialized, this openness clashed with the rise of proprietary software models. Restrictions on use and a lack of transparency led to frustration among many programmers, sparking the desire for an alternative.

In 1983, Richard Stallman's launch of the GNU Project became a catalyst for the modern open source movement [8]. This ambitious initiative aimed to create a completely free operating system built on the principles of software freedom. Stallman's advocacy for free redistribution, user modification rights, and transparent development laid the philosophical foundation for the open source movement.

Early open source projects relied heavily on collaborative development through online communities of programmers. This distributed model allowed for efficient bug fixing, rapid development, and fostered knowledge exchange. Moreover, the creation of licenses like the GNU General Public License (GPL) was crucial. These licenses protected open source freedoms, ensuring that software and any modifications made to it would remain accessible to all [24].

Several notable projects defined the early era of open source software. Linus Torvalds' creation of the Linux kernel in 1991 became the poster child for successful open source development, eventually forming the heart of countless operating systems. Additionally, the GNU Project provided essential tools, the Apache Web Server became the dominant force powering the internet, and BIND played a critical role in the internet's infrastructure. These landmark projects proved the viability and potential of the open source model.

3.2 Defining Open Source

The definition of open source software is contested. It sparks debate between those who see it as synonymous with "free software" and those who view "open source" as a distinct

approach [12].

“Free software”, a term stemming from the GNU project, prioritizes user freedoms over cost. At its core, free software grants users the right to use, share, study, modify, and even improve software without restrictions [37]. Access to the source code is vital, as it is what empowers users to exercise these freedoms.

The term “open source” was introduced later than “free software.” While both aim to describe software with accessible source code, there are key philosophical differences. The “open source” term was partly motivated by a desire for a less politically charged, business-friendly label, though its official definition aligns closely with “free software”. Stallman argues that the everyday interpretation of “open source” merely denotes visibility of the source code, not the full freedoms of free software [35]. This ambiguity allows the term to be applied to semi-free or even proprietary software, diluting its meaning.

While open source initiatives appear in many forms, there are two primary types: open source software and open source content [28]. Open source software refers to code that is freely available for anyone to inspect, modify, and distribute. This collaborative approach fosters innovation and rapid development. Open source content, on the other hand, encompasses a wider range of creative materials, such as educational resources, scientific data, and artistic works. Like software, open source content licenses permit users to access, share, and modify the materials under specific guidelines.

3.3 Open Source Project

An open-source project is a collaborative endeavor where the source code of a software program or application is made freely available to the public. This means that anyone can view, modify, and distribute the code according to the terms of the project’s license. Open-source projects typically encourage transparency, community-driven development, and collaboration among developers from diverse backgrounds.

One of the most widely used definitions of open source comes from the Open Source Initiative (OSI), which defines open source software as having a license that meets the following criteria in the figure 1.

Some of the most popular and influential open-source projects:

- Linux: An extremely popular and versatile operating system kernel. Linux powers a vast range of devices from servers and supercomputers to smartphones and embedded systems. It is known for its stability, security, and customization options [10].

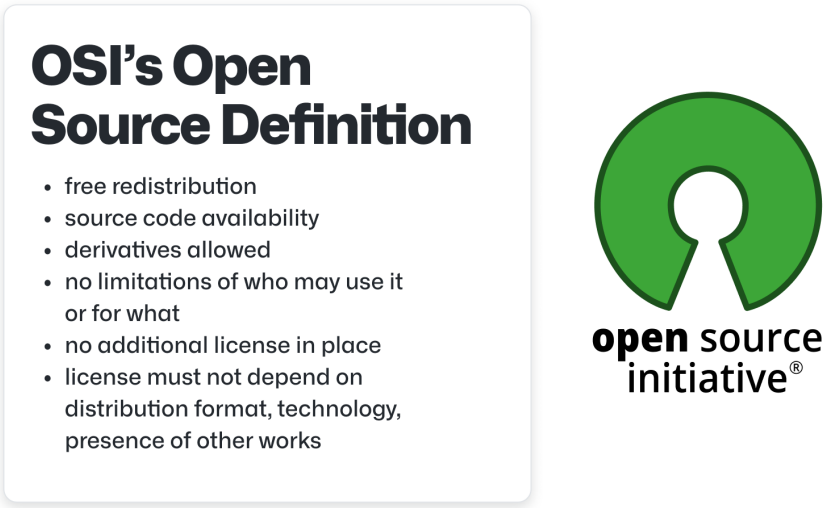

Figure 1: OSI's Open Source Definition [17]

- Git: A distributed version control system that has become the de facto standard for software development. Git allows developers to track changes to code, collaborate easily, and experiment with different branches of development [25].
- Apache HTTP Server: One of the most widely used web server software in the world. It is responsible for serving a significant portion of websites on the internet. Apache is known for its reliability, flexibility, and extensive features [9].
- Python: A high-level, general-purpose programming language that emphasizes readability and ease of use. Python is extremely popular in fields like data science, web development, machine learning, and system administration [34].
- TensorFlow: A free, open-source toolset designed to make machine learning accessible. It offers a wide range of tools, libraries, and a supportive community, so both researchers and developers can create and use powerful ML applications [7].

3.4 Ownership and Licensing

The concept of ownership within the open-source software landscape deviates from the traditional models of individual or corporate proprietorship. While thousands of developers might contribute to a single open-source project, the idea of ownership is more accurately framed in terms of rights, intellectual property, and copyright [15]. In this context, it's vital to understand the pivotal role of open-source licenses.

In the world of software, open-source licenses reign supreme. The MIT License offers broad permissions for use, modification, and distribution, making it incredibly developer-friendly [31]. Similarly, the Apache License 2.0 is permissive and emphasizes providing clear copy-

right and patent notices [33]. For those seeking “copyleft” protection, ensuring that derived works stay open-source, the GNU GPL is a common choice [24].

Licenses establish the boundaries and freedoms granted to users and contributors in utilizing and modifying open-source software [22]. These licenses come in a variety of forms, each with its own set of rules and restrictions. Some licenses, for example, expressly prohibit the sale of the original software or its derivative versions [26]. This is done to safeguard the open-source ethos and prevent commercial exploitation that may stifle community-driven development.

In sum, understanding the intricate interplay of ownership and licensing is essential for navigating the open-source ecosystem. While traditional notions of ownership take a backseat, the principles of intellectual property, copyright, and the specific terms of open-source licenses dictate the rights and responsibilities of all those who interact with this collaborative software model.

3.5 Morden adoption and future

OSS has become a firmly established pillar of the software development landscape, and its influence is poised to grow further in the coming years. As the open-source community continues to mature and address its challenges, we can anticipate the emergence of even more innovative and powerful solutions driven by this collaborative model.

While the origins of open source can be traced back to the 1960s, with companies like IBM providing free software with their early mainframe systems [27], its widespread adoption has accelerated significantly in recent decades. Despite the dominance of tech giants like Google, Apple, and Microsoft in the commercial sphere [18], these companies also play a pivotal role in the open-source ecosystem, particularly through contributions on platforms like GitHub.

The substantial impact of open source is exemplified by EU-based companies, which invested an estimated €1 billion in OSS in 2018. This investment yielded a significant return for the European economy, with an estimated contribution ranging from €65 to €95 billion [5].

Figure 2 underscores the active participation of tech leaders in the open-source movement. In 2020 alone, 5,709 Google employees submitted over ten commits to GitHub’s public code repository. This was closely followed by contributions from Microsoft, Red Hat, IBM, and others...

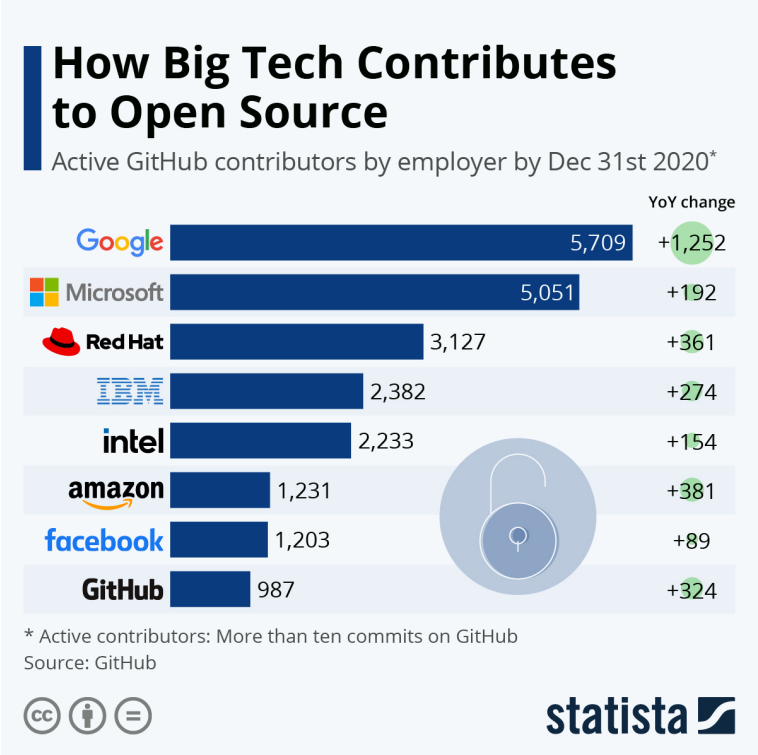

Figure 2: How How Big Tech Contributes to Open Source [36]

The open-source movement's trajectory promises transformative impacts across industries, fueled by increasing collaboration, transparency, and knowledge sharing within its expanding community. This trajectory indicates the emergence of groundbreaking technologies, tools, and solutions that will serve the interests of both users and developers, underscoring the potential of open source to revolutionize sectors and drive innovation on a global scale.

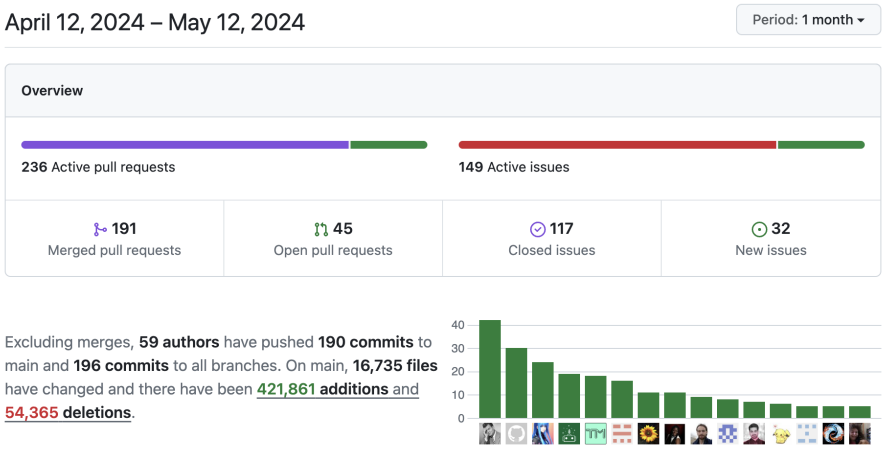

Figure 3: FreeCodeCamp's Open Source Contribution

4 Research methods

This thesis delves into the complex motivations behind developer participation in open-source projects by combining a Systematic Literature Review (SLR) and an in-depth case study. The literature review establishes a robust foundation of knowledge, while the case study illuminates the nuanced social interactions and experiential factors shaping individual developers' decisions to contribute.

4.1 Systematic Literature Review

SLR offers a disciplined research method that promotes a clear, targeted investigation through a well-defined question. This study employs the SLR methodology as outlined by Kitchenham, ensuring a reliable and unbiased review [20].

4.1.1 Systematic Literature Review definition

SLR is a meticulously planned research approach aiming to identify, evaluate, and synthesize all available evidence related to a clearly defined research question [20]. It employs a transparent, reproducible protocol to minimize bias, emphasizing comprehensiveness and critical appraisal of included studies.

The methodology consists of three main phases. First, in the planning phase, researchers must clearly define the research questions that guide the entire review. A detailed review protocol should then be developed, outlining the search strategy (databases, search terms), study selection criteria (e.g., publication types and dates), quality assessment methods, data extraction plans, and the strategy for synthesizing the findings.

The second phase involves conducting the review. This includes identifying research using the search strategy, selecting studies based on the eligibility criteria, critically evaluating study quality, extracting the relevant data, and finally, synthesizing that data using techniques like meta-analysis or thematic analysis.

Finally, the reporting phase focuses on transparently documenting the entire SLR process for reproducibility and critical assessment. Researchers must present the results in a clear manner, addressing the initial research questions and their implications, while also acknowledging any limitations within the review process.

4.1.2 Systematic Literature Review approach

Open source software development presents a unique model of collaboration where developers voluntarily contribute their time and expertise. Unlike traditional software engineering environments, motivations in open source extend beyond direct financial compensation. Researchers have investigated these motivations from various perspectives, including psychological, economic, and social factors. This complex landscape can lead to potentially varied interpretations of what drives participation.

The SLR will lay the groundwork for my thesis by providing a thorough understanding of current research on open source developer motivations. Based on the SLR findings, I will be able to identify under-investigated areas or potential gaps in the theoretical frameworks used to explain developer behavior.

5 Systematic Literature Review

Guided by Kitchenham's SLR framework, this study carefully addressed review planning (defining research questions, developing a protocol), conducting (literature search, study selection, data extraction and quality assessment), and reporting [20].

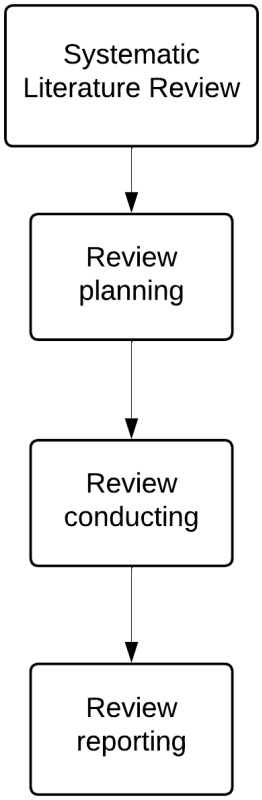

Figure 4: Systematic Literature Review process [20]

The figure 4 shows an overview of my work to execute a systematic literature review. I meticulously outlined a protocol and defined my overarching research questions 1 and 2. A comprehensive search strategy was then established to pinpoint pertinent articles within electronic databases relevant to the field of open source software motivation. Stringent inclusion and exclusion criteria ensured the selection of methodically sound studies. Following the search, I carefully screened titles and abstracts of retrieved database articles. Studies initially deemed relevant underwent a thorough full-text analysis. This systematic review will culminate in a detailed report encompassing search strategies, selection criteria, the number of articles evaluated at each stage, and an overarching synthesis of the findings.

5.1 Review planning

In chapter 2, I explored the process of defining a research question. Here, I will concentrate on developing a protocol for the initial review planning phase of a SLR. While numerous scientific databases exist online, selecting the most suitable ones for evidence synthesis can be challenging [16]. This section will offer guidance on database selection and explain the rationale behind different choices.

5.1.1 Search criteria

Building on a previous study from Chua and Zhang [2], I established inclusion criteria to ensure the selected articles addressed my research questions and utilized strong methodologies:

- Research articles and conference papers, peer-reviewed and available through major academic search platforms like Google Scholar, Springer, Institute of Electrical and Electronics Engineers (IEEE), Association for Computing Machinery (ACM), Science Direct.
- Restriction to English-language sources.
- Disciplines in Computer Science, Software Engineering
- Publication date: from 2000 to 2024.
- Specific search terms (“open source” OR “open source software”) AND (“motivation” OR “contribution challenges” OR “social dynamics impact” OR “interest” OR “contribution barriers”) within titles or descriptions.
- Prioritization of publications from leading information systems conferences and journals. Ex: ACM/IEEE International Conference on Software Engineering, IEEE Transactions on Software Engineering, Journal of Systems and Software,...

These search terms were employed to address the three research questions which are categorized in the table 1.

Category	Search terms
1	OSS motivation and all synonyms
2	Social dynamics impact on OSS and all synonyms
3	Contribution barriers of OSS and all synonyms

Table 1: Specific categories of search terms

5.1.2 Data source

To identify relevant literature, I executed our search strings across multiple databases. Each database was queried with the complete set of search strings, yielding varying results. In some instances, identical search strings produced an overabundance of results, while in others, they returned no results. Databases with no results were excluded from further analysis. The selected databases and their corresponding result counts are presented in the following table 2.

Database	Category 1	Category 2	Category 3
Google Scholar	244 000 results	17 800 results	175 100 results
Springer	623 results	161 results	327 results
IEEE	396 results	68 results	48 results
Science Direct	4 200 results	97 results	15 results

Table 2: Initial search results from scientific databases

5.1.3 Inclusion and exclusion criteria

To maintain a focused and methodologically review, I used inclusion and exclusion criteria to pinpoint studies aligned with my research questions. These criteria served as a consistent filter for all potential sources. The specific criteria are outlined below, and they were applied to every study retrieved from our selected databases.

Inclusion:

- Accessible: The research paper must be accessible and downloadable either through LUT Academic Library or online database.
- Peer-Reviewed: The paper should be published in a reputable, peer-reviewed journal or conference proceedings. This ensures the quality and credibility of the research.
- Relevant to research questions: The paper must directly address the specific research questions at hand.
- Methodology transparency: The reasearch and data collection methodologies must be metioned.

Exclusion:

- Irrelevant content: Exclude papers that deviate significantly from my research questions or lack a clear connection to the area of study.
- Older papers (released before 2000) ought to be disregarded.

- Papers having fewer than four pages were removed.
- Papers cannot be accessed
- Papers were not written in English
- Papers topic were about open source but in hardware, economy, environment,... but not software

5.1.4 Study selection

A further procedures were taken for the final paper selection after applying the inclusion/exclusion criteria to each of the resultant papers.

- Review the title, keyword, abstract, description for filtering
- Excluding duplicated studies.
- Ranking research papers by the number of times they've been cited and relevant to search term by search engine.

The initial search across various databases yielded approximately 500,000 potentially relevant papers. Due to resource constraints, the review process was limited to 50 papers per database. Following the application of inclusion and exclusion criteria, along with further selection procedures, a final set of 20 relevant and suitable articles was identified for this SLR. These papers are listed in table 3, with full details provided in the Appendix 1. The paper selection process is displayed in the figure 5

The figure 6 depicts the collaboration network between authors of 20 selected papers for a systematic literature review on open-source software motivation, social impact, and challenges. The circles represent authors, and the connecting lines indicate co-authorship on a paper. The size of a circle corresponds to the number of papers an author has co-authored within the dataset. Steinmacher and Igor appears to be the most prolific author in this dataset, having co-authored papers with several other researchers.

5.2 Review conducting

5.2.1 Extract the data

Data extraction is a crucial step in a SLR. It involves the methodical gathering of specific and relevant information from the research papers I've chosen for the review. This process entails pinpointing key data points aligned with the research questions, and subsequently organizing this data systematically to facilitate analysis and synthesis of the findings. Here is the data which will be extract from each paper:

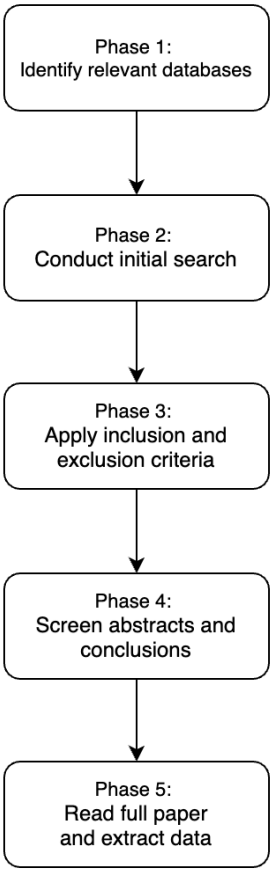

Figure 5: Process of selecting papers for SLR

- Title
- Abstract
- Authors
- Publication date
- Database
- Keyword
- Research method
- Data collection method
- Answers to research questions

5.2.2 Synthesis

To systematically analyze the extracted data, we formed two distinct groups (see Table 4) based on their focus:

Publisher	Number of articles	Authors
Elsevier	6	Bitzer, Schrettl & Schröder (2007); Choi & Pruett (2015); Li, Tan & Teo (2012); Oreg & Nov (2008); Steinmacher, Silva, Gerosa & Redmiles (2015); Wu, Gerlach & Young (2007)
AISNET	1	Ke & Zhang, P. (2008)
IEEE	3	Ye & Kishida (2003); Gerosa, Wiese, Trinkenreich, Link, Robles, Treude & Sarma (2021); Zhang, Yuxia (2024)
ResearchGate	1	Zhao, Shengyu (2024)
Springer	3	Steinmacher, Conte, Gerosa & Redmiles (2019); Fershtman & Gandal (2007); Hannemann & Klamma (2013) ; Hannemann & Klamma (2013)
ACM	3	Steinmacher, Conte, Gerosa & Redmiles (2015); Guizani, Chatterjee, Trinkenreich, May, Noa-Guevara, Russell & Sarma (2021) ; Hannebauer & Gruhn (2017)
Informs	1	Roberts, Hann & Slaughter (2006)
Taylor & Francis Online	1	Alexander Hars (2002)
EASST	1	Freeman (2007)

Table 3: Selected papers for SLR

- Group 1: addressing interrelated research questions: This group delves into both research questions 1 and 2. Interestingly, a significant portion of papers addressing question 1 also provide answers to question 2. This overlap suggests a potential relationship between these two research questions. Group 1 comprises a total of 12 carefully selected papers.
- Group 2: exclusive focus on research question 3: This group offers a concentrated exploration of research question 3, without addressing the other research areas. Group 2 contains a total of 5 papers, ensuring a targeted examination of this specific question.

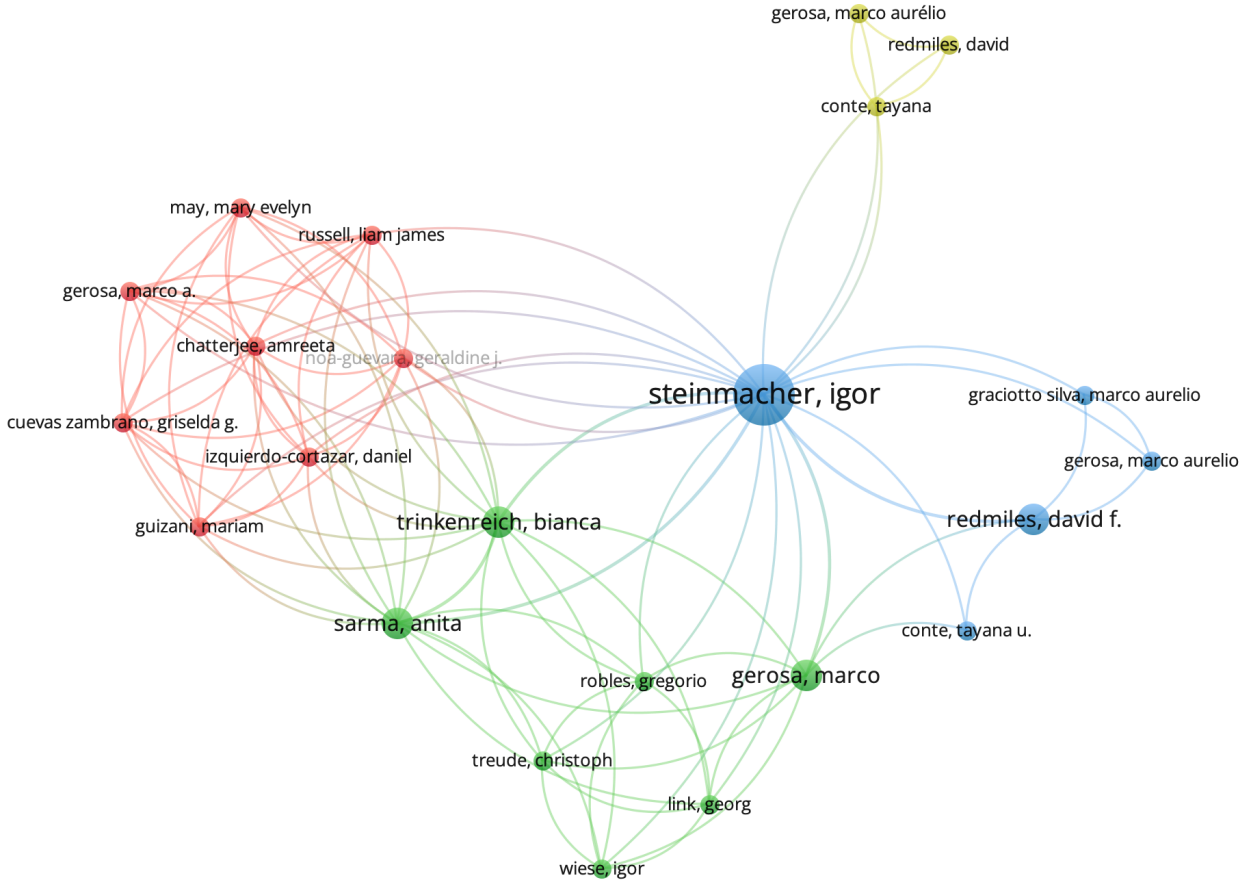

Figure 6: Collaboration among open-source software researchers

Research question	Article ID
1	A05, A06 , A08 , A09, A10, A11, A12, A13, A15, A16, A17, A18
2	A05, A06, A07, A09, A10, A12, A13, A16, A17, A18, A19, A20
3	A01, A02, A03, A04, A14

Table 4: Data synthesis

6 Review reporting

6.1 Primary motivation for developers contributing in open source

The main purpose of this study is to find the movitvation of developers contributing in OSS project. For each selected study, I analyzed any motivation reported that was empirically identified or evaluated.

6.1.1 Intrinsic motivations

Our exploration of developer motivation in open-source software (OSS) projects begins with intrinsic motivations, the internal drivers that fuel participation for personal satisfaction rather than external rewards. This encompasses a broad spectrum of factors, including: play value, the inherent enjoyment derived from the coding process; community engagement, the sense of belonging and collaboration found within OSS projects; learning, the opportunity to develop new skills and expand technical knowledge; personal interest, the desire to work on projects that align with individual passions; altruism and ideology, the belief in contributing to a greater good and supporting the open-source philosophy; need for autonomy, the freedom to work independently and creatively; and reciprocity/introjected regulation, the desire to contribute back to the community and maintain a sense of personal responsibility for the project's success. I will delve deeper into each of these intrinsic motivations in the following sections, examining their unique influence on developer behavior within the OSS landscape.

1. Play value

In contrast to traditional software development, which often prioritizes external rewards like monetary compensation and career advancement, OSS projects offer a unique space where play value emerges as a central driving force. Play value, in this context, encapsulates the inherent enjoyment, intellectual stimulation, and creative fulfillment developers experience through the act of programming and problem-solving [4, 39, 40, 21, 13, 6, 23, 19, 1, 29]. Let's examine why this is such a powerful motivator.

For many developers, OSS represents a playground for experimentation and innovation. Unburdened by strict commercial deadlines or rigid specifications, they are free to explore novel ideas, test unconventional approaches, and engage in the iterative process of building software purely for the intrinsic satisfaction it provides . The act of turning concepts into functional code can be deeply rewarding.

OSS communities often tackle complex technical problems that demand creative solutions. Developers who are drawn to intrinsically motivating challenges revel in the opportunity to

dissect intricate issues, devise elegant workarounds, and optimize code performance. This continuous learning process creates a sense of mastery and accomplishment that fuels further engagement.

Commercial software development typically necessitates compromises – feature trade-offs, adherence to proprietary standards, and prioritization of market demands over pure technical curiosity. In contrast, OSS projects offer developers a liberating space to exercise their technical creativity without external pressures. This autonomy nourishes problem-solving and innovation for its own sake.

The collaborative aspect of OSS can itself be a form of play. Engaging with fellow developers, brainstorming solutions, exchanging knowledge, and contributing to a shared creation can be intellectually stimulating and enjoyable. This camaraderie fosters a playful sense of experimentation and discovery within the community.

2. Community engagement

The act of conceptualizing the open-source community as a metaphorical family, united in pursuit of shared objectives, can be a powerful catalyst for developer participation [4, 41, 40, 21, 23, 19, 1]. This collaborative environment fosters contributions aimed at communal advancement, even when they may not yield immediate personal gain for the individual developer. Participation in open-source projects can be fueled by a profound sense of belonging and an alignment of personal values with those of project teams and the open-source movement at large.

The chart 7 underscores the complex interplay of motivations that drive developer participation in open-source projects. While social factors are paramount, career considerations and political beliefs also play a significant role. This diversity of motivations highlights the need for a nuanced understanding of the open-source community and tailored strategies to attract and retain developers.

Additionally, a conviction regarding the inherent value of open-source code, paired with a perceived responsibility to contribute to the free and OSS ecosystem, serves as a significant motivator for developers. Cultivating a sense of belonging and fostering a shared purpose within the community or project team can thus be instrumental in empowering individuals to become active participants and contributors in the open-source software development landscape.

3. Learning

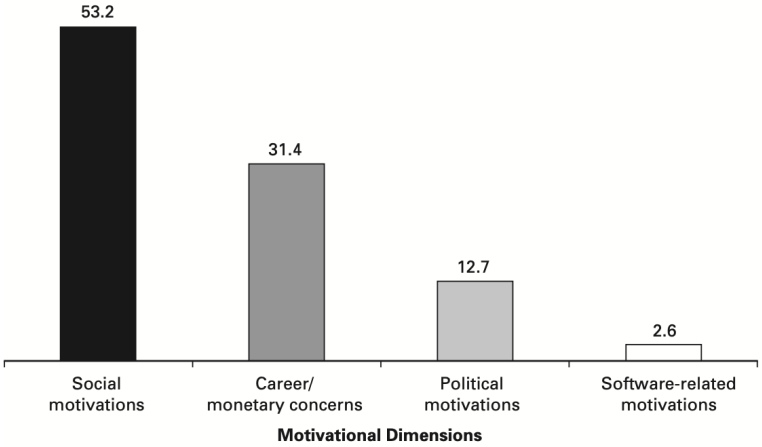

Figure 7: As a proportion of all contributors to OSS, developers by motivation class [14]

Research consistently identifies learning as a primary impetus for individuals to actively engage in OSS communities [39, 41, 40, 21, 38, 13, 6, 1, 29]. OSS projects present multifaceted learning environments; developers are drawn to the inherent opportunities to acquire knowledge from the systems themselves, to collaborate and gain insights from fellow community members, and to reciprocally disseminate their own expertise.

Participation within OSS communities extends beyond the purely technical exchange of knowledge, encompassing a rich social dimension. By directly engaging in open-source projects, developers immerse themselves in a collaborative network of peers, often spanning skill levels and expertise. This fosters a dynamic learning ecosystem where individuals benefit from informal mentorship opportunities, observing problem-solving approaches employed by more experienced contributors, and receiving constructive feedback that accelerates their professional growth.

Moreover, the act of contributing to a shared knowledge base empowers developers and reinforces self-efficacy. The potential for continuous self-development, the pursuit of mastery, and the ability to give back to a community dedicated to knowledge-sharing serve as profound and enduring sources of motivation for many software contributors.

4. Personal interest

The concept of the “personal itch,” as articulated by Eric S. Raymond, illuminates a key motivator for developer participation in OSS projects. Individuals often engage in OSS development to address a specific problem or augment functionality that directly aligns with their personal or professional needs. The desire to create a solution that may not otherwise exist, driven by this personal necessity, serves as a potent catalyst for engagement.

Furthermore, the inherent intellectual challenge of solving complex programming problems stands as a significant motivator for developers seeking to contribute to open-source initiatives. The opportunity to grapple with intricate coding puzzles, apply problem-solving strategies, and ultimately contribute to the solution can be deeply fulfilling for those driven by a passion for programming.

The sense of creativity fostered within the OSS landscape is another powerful draw. Developers are empowered to express their ingenuity, explore innovative solutions, and continuously hone their skills through the development of tools or solutions that serve their own requirements or those related to their work. This blend of personal utility, creative expression, and continuous learning establishes a compelling environment that attracts and sustains developer involvement.

5. Altruism and ideology

OSS development thrives in part due to the contributions of individuals motivated by altruism and ideological convictions. This section explores these factors and their influence on developer participation. A significant driver for many developers is the inherent satisfaction derived from assisting others. Contributing to open-source projects allows them to directly improve software used by a wider community. This collaborative environment fosters a sense of purpose, as developers witness the positive impact of their work on others

Many developers are drawn to the core principles of open-source software, including transparency, collaboration, and the democratization of technology. Participation allows them to contribute to a development model that emphasizes open access and fosters a sense of community. Additionally, developers can be motivated by a desire to create software that benefits the greater good by being freely available and readily modifiable. This aligns with their altruistic desire to contribute to society and maintain strong social bonds.

Altruism and ideological alignment with open-source principles play a vital role in propelling developer participation. Both the satisfaction of helping others and the commitment to open-source ideals create a compelling environment that attracts and retains developers within the OSS ecosystem.

6. Autonomy

The open-source software environment provides a platform where developers can exercise a high degree of autonomy, making it particularly attractive to those valuing self-determination. The ability to select projects of interest, dictate their involvement, and contribute indepen-

dently fulfills the intrinsic need for autonomy. This freedom to innovate and pursue solutions without rigid constraints becomes a compelling motivator, drawing developers who seek a sense of control and ownership over their contributions.

Unlike traditional software development environments that might be constrained by rigid hierarchies or top-down management styles, the open-source model empowers developers to chart their own path. They can choose to focus on areas that align with their passions, explore new technologies, or experiment with novel approaches without the need for constant external approval. This sense of agency and self-direction is deeply fulfilling for those who thrive in environments where their initiative and creativity are valued.

7. Reciprocity and introjected regulation

Open-source communities thrive on a powerful sense of reciprocity. Developers who have directly benefited from freely available open-source software often feel a deep-seated obligation to give back, fueling their participation and ensuring the continued growth of the ecosystem. This desire to repay the community for the invaluable resources they've received becomes a motivating force.

Additionally, introjected regulation plays a role in influencing developer behavior. The internalization of expectations can lead to feelings of pride, guilt, or shame regarding contributions to open-source projects. This desire to maintain a positive self-image, live up to personal standards, and avoid negative emotions can significantly drive participation as developers strive to meet both their own expectations and those they perceive the community holds.

6.1.2 Extrinsic motivations

Beyond the intrinsic factors explored in the previous chapter, extrinsic motivations also play a significant role in driving developer participation in open-source projects. This chapter delves into these external factors, including the potential for signaling skills and experience to potential employers, garnering recognition and building reputation within the open-source community, and potentially obtaining external rewards such as monetary compensation or job opportunities.

I will also examine how extrinsic motivators can intersect with a developer's desire to improve software quality. Contributions to high-profile projects can serve as a powerful signal of competence, while active participation may lead to opportunities to collaborate with skilled developers and gain valuable experience. Furthermore, I will explore the concept of role transformation: how continued involvement in the open-source landscape can elevate a

developer's standing, potentially opening doors to leadership roles, consulting positions, or job offers within companies heavily invested in open-source technologies.

1. Signaling and recognition

Participating in OSS projects allows developers to publicly showcase their abilities and commitment. Within the highly competitive software development field, OSS contributions provide concrete evidence of a developer's abilities, enhancing their reputation and potentially unlocking new opportunities. Open-source involvement demonstrates not only technical skills but also a dedication to the broader community and a drive for innovation.

Open-source projects offer developers a platform to display their talents to potential employers, boosting their professional standing. Unlike traditional resumes or interviews that provide a more limited view, OSS contributions offer real-world proof of a developer's capabilities. Employers often see active participation as indicative of both technical skill and the ability to collaborate effectively in a team setting.

The recognition garnered from fellow developers within the open-source community serves as a powerful motivator. The open-source model promotes collaboration, transparency, and continuous improvement, resulting in a space where contributions are acknowledged and celebrated. This validation from peers acts as a potent incentive for developers to further advance their skills and continue making meaningful contributions to projects they're passionate about.

Active engagement in open-source projects supports developers in building strong reputations within their field, positioning them as experts in their niche. Through consistent, high-quality contributions, thoughtful insights, and constructive participation, developers gain respect within the community. This recognition elevates their professional stature and fosters new possibilities for collaboration, networking, and career progression. Ultimately, leveraging their open-source work as a showcase is beneficial not only to the individual developer but also contributes to the advancement of the entire developer community.

2. Improving software quality

Some developers engage in open-source projects to create high-quality software that is accessible to a wider audience and can benefit the community. Open-source development fosters a collaborative environment where developers from diverse backgrounds come together to share their expertise and work towards common goals. By leveraging the collective intelligence and resources of the community, developers can create software that is not only robust

and reliable but also tailored to address the evolving needs of users across different industries and domains. This democratization of software development ensures that innovative solutions are not confined to proprietary ecosystems but are freely available for anyone to use, modify, and redistribute.

By participating in open-source projects, developers can access and contribute to software that meets their specific needs and preferences, often surpassing proprietary alternatives. Unlike closed-source software, which may be limited by proprietary restrictions and licensing fees, open-source projects offer greater flexibility and transparency. Developers have the freedom to inspect, modify, and enhance the code according to their requirements, empowering them to create customized solutions that are more efficient, secure, and adaptable. This collaborative and iterative approach to software development not only fosters innovation but also fosters a sense of ownership and pride among contributors, who are motivated by the collective impact of their efforts on the broader community.

3. External rewards

While publicly discussions often prioritize the significance of intrinsic motivations, extrinsic rewards such as promotions, financial incentives, increased compensation, and professional advancement remain potent drivers of developer participation in open-source projects. Tangible rewards hold substantial appeal, particularly for those who utilize open-source involvement as a strategic tool for career development and financial gain. Within a highly competitive labor market emphasizing demonstrable skills and practical experience, active open-source contributions tangibly augment a developer's professional credentials and enhance their overall marketability.

Developers may be drawn by the potential financial returns derived from open-source participation, such as new job opportunities or consulting contracts. By establishing a visible record of expertise and successful contributions, developers attract the attention of companies or clients who value their skills, potentially leading to lucrative positions. Beyond traditional employment, open-source involvement can serve as a foundation for supplementary income streams, such as consulting services, training workshops, or speaking engagements, which cultivate both financial rewards and professional recognition.

Furthermore, the pursuit of career advancement and professional distinction strongly motivates developers to engage with open-source projects. Establishing oneself as a thought leader or subject matter expert within the community cultivates opportunities for leadership positions, mentorship roles, or invitations to esteemed conferences and industry events. The visibility and reputation fostered through such contributions heighten a developer's stand-

ing and create new pathways for professional growth and development. Ultimately, while intrinsic motivations undeniably fuel enthusiasm and dedication, extrinsic rewards remain indispensable in attracting and sustaining long-term participation in open-source initiatives.

A study examining the dynamics of paid and volunteer open-source developers within the Rust project has revealed significant disparities in their contribution behaviors [40]. Notably, core developers who receive compensation demonstrate a higher frequency of contributions compared to volunteers 8. This suggests that financial incentives may play a role in driving sustained engagement.

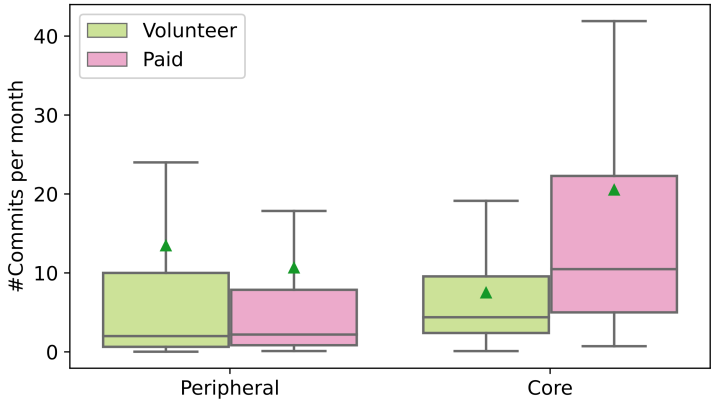

Figure 8: How often paid and volunteer developers contribute to Rust project [40]

Moreover, commits from one-time paid developers tend to be larger in scope, potentially encompassing more impactful code changes than those of one-time volunteers. This highlights a possible correlation between compensation and the magnitude of contributions. Peripheral paid developers exhibit a higher inclination towards implementing new features compared to unpaid contributors. This trend underscores how financial incentives might influence not only the quantity but also the innovative nature of contributions within the open-source ecosystem.

Collectively, these findings illustrate the complex dynamics of mixed-motivation OSS projects. Understanding these distinctions is crucial for project maintainers seeking to effectively leverage the collaborative potential of both paid and volunteer contributors, ultimately strengthening the sustainability of OSS projects.

4. Role transformation

One of the defining characteristics of OSS projects is the transformation of roles. Unlike traditional software development models, where users and developers occupy distinct positions, OSS communities blur these lines. Anyone, from seasoned developers to individuals

with technical curiosity, can become a contributor. This inclusive nature fosters a sense of ownership and empowers users to actively shape the project's evolution. The potential to transition from user to developer offers a compelling incentive for participation, fostering a community where everyone's voice is valued, and diverse perspectives are encouraged.

Furthermore, the ability to directly address user needs serves as a significant motivator for developers to contribute to open-source projects. Whether these needs arise from professional or personal endeavors, developers within the OSS community are driven by the desire to create software that solves real-world problems and enhances the user experience. This direct connection between developers and users fosters a collaborative environment where both parties benefit from the shared knowledge and dedication to continuous improvement.

6.2 Impact of social dynamics

In addition to investigating the motivations of developers, this research also examined the significant influence of social dynamics on the participation of developers in open-source projects. Through a comprehensive analysis of 13 pertinent studies selected from a pool of over 20 papers, this research has yielded several key findings. These findings highlight the multifaceted nature of developer engagement in open-source initiatives and underscore the importance of social interactions in shaping participation patterns. The subsequent sections will elaborate on these findings, providing a nuanced understanding of the interplay between individual motivations and social forces within the open-source software development ecosystem.

6.2.1 Community interaction

The level of interaction within an open-source community is a significant determinant of developer participation. Active communication channels, encompassing forums, mailing lists, and chat platforms, provide essential avenues for collaboration, knowledge sharing, and mutual assistance. These interactions foster a sense of community and belonging, encouraging developers to actively engage with the project and contribute their expertise. Conversely, projects with limited or ineffective communication channels may struggle to attract and retain contributors, as developers may feel isolated or lack the necessary support to make meaningful contributions.

Empirical research consistently demonstrates that developers derive substantial satisfaction from collaborative endeavors and the opportunity to assist others within the open-source ecosystem. Collaboration and teamwork are not merely instrumental means to achieve project goals but are also intrinsically rewarding for developers. The sense of community engendered by open-source projects, along with the opportunity to interact with peers and

contribute to a shared endeavor, are integral to the ethos of open-source software development.

The establishment of robust communication infrastructure and feedback mechanisms is paramount for sustaining active developer participation. Clear, transparent, and efficient communication facilitates the resolution of technical issues, the exchange of innovative ideas, and the coordination of efforts among team members. Moreover, constructive feedback loops enable developers to learn from each other, refine their skills, and enhance the quality of their contributions. Cultivating a supportive and communicative environment fosters a sense of camaraderie and shared purpose, thereby augmenting developer engagement and productivity.

The integration of social features within open-source platforms, such as mechanisms for connecting individuals seeking assistance with those willing to provide it, can substantially enhance community interactions and support. The open-source ethos is intrinsically predicated on collaboration and the open exchange of knowledge, and social platforms facilitate these interactions by creating virtual spaces for developers to connect, communicate, and collaborate. The sense of belonging to a community of like-minded individuals fosters camaraderie, mutual assistance, and a shared sense of purpose, all of which contribute to sustained engagement and project success.

The presence of experienced developers who are willing to mentor newcomers is a crucial catalyst for promoting participation and retention within open-source communities. Mentorship programs provide novice developers with invaluable guidance, support, and encouragement, empowering them to overcome challenges, acquire new skills, and integrate seamlessly into the community. This intergenerational transfer of knowledge is essential for the long-term sustainability and growth of open-source projects. By fostering a welcoming and inclusive environment that values mentorship and knowledge sharing, open-source communities can attract and retain a diverse range of contributors, ensuring the continued vitality and innovation of the open-source software ecosystem.

6.2.2 Networking opportunities

Novice contributors often transition their initial motivations towards career-oriented goals, leveraging open-source projects as a portfolio to showcase their skills to potential employers [4, 13]. Participation in these projects offers invaluable networking opportunities, fostering connections with industry professionals and paving the way for career advancement [38, 13, 23]. By demonstrating their expertise and building a reputation within the open-source community, developers can attract job offers, consulting opportunities, and further professional

development. Moreover, the open-source environment allows developers to gain experience with diverse technologies, tools, and methodologies, broadening their skillset and making them more adaptable to the evolving demands of the tech industry.

Open-source projects serve as a platform for developers to connect with industry peers, experts, and potential employers [38, 13, 23]. These connections can lead to collaborations on new projects, expanding professional networks and opening doors to career growth opportunities. The collaborative nature of open-source projects allows developers to establish relationships with like-minded individuals, fostering a supportive community that encourages knowledge sharing and mutual growth. Additionally, engaging with established open-source communities can provide developers with exposure to industry best practices, coding standards, and project management methodologies, further enhancing their professional capabilities.

Open-source projects are inherently collaborative environments, providing developers with ample opportunities to share knowledge, learn from others, and enhance their skills. The exchange of ideas, feedback on code, and exposure to diverse perspectives within the community foster continuous learning and skill improvement. Through interactions with other developers, mentorship, and exposure to new ideas and technologies, developers are motivated to stay engaged and contribute to the project's ongoing success [4, 39, 21, 23]. This culture of continuous learning and knowledge sharing also helps developers stay abreast of the latest trends and innovations in the tech industry, ensuring their skills remain relevant and in demand.

6.2.3 Community culture and support

Engaged and supportive open-source communities act as a catalyst for developer contributions by offering assistance, constructive feedback, and a sense of belonging. This collaborative atmosphere nurtures knowledge sharing, mutual support, and a strong sense of community among developers. Ultimately, positive social interactions and a supportive environment within these communities drive increased motivation and sustained engagement among developers. [38, 6, 23, 19].

Social coding platforms have revolutionized the open-source landscape, shifting the culture from its traditional hacker-centric roots to a more inclusive, collaborative community. By lowering barriers to entry, these platforms have made open-source projects more accessible and welcoming to newcomers, regardless of their technical expertise [39, 13]. They foster a sense of belonging and encourage participation through features like issue tracking, discussion forums, and code review tools, promoting knowledge sharing and collaborative

problem-solving. This cultural shift has not only broadened the pool of contributors but has also led to more diverse perspectives and innovative solutions within the open-source ecosystem.

Roles within OSS communities are dynamic and fluid, allowing members to assume greater responsibilities by contributing meaningfully to projects. As individuals transition between roles, they actively influence the social dynamics and structure of the community, ultimately driving its evolution [39]. This flexibility enables OSS communities to adapt and thrive in response to the evolving needs of projects and the diverse contributions of their members.

Open-source platforms that promote collaboration and offer diverse avenues for appreciation, ranging from formal accolades to informal gestures like awarding stars to projects, significantly enhance the sense of belonging and recognition among community members. Active engagement in these communities allows developers to gain recognition for their contributions, cultivate a positive reputation, and establish a personal brand within the wider developer community [13, 23].

The paper "OpenRank Leaderboard: Motivating Open Source Collaborations Through Social Network Evaluation in Alibaba" presents a study conducted to explore the impact of the OpenRank Leaderboard on open source collaborations within Alibaba's projects [41]. The research methodology involved a mixed-methods approach, including case studies, surveys, analysis of project metrics data, semi-structured interviews, and thematic coding. The study focused on seven open source projects initiated by Alibaba, aiming to investigate how gamified leaderboards can motivate collaboration and drive innovation in software development.

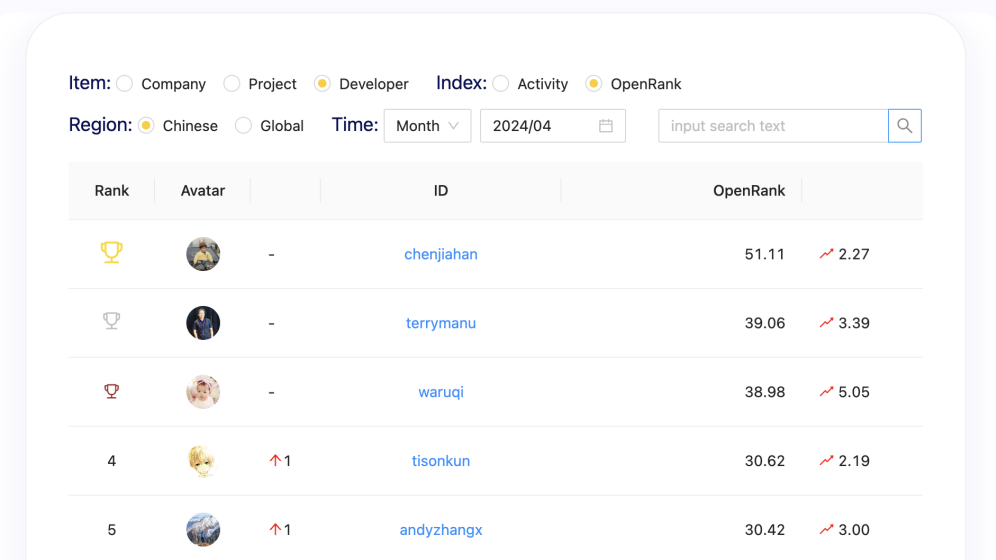

Figure 9: A screenshot of the interface of the OpenRank Leaderboard in May, 2024

Through the implementation of the OpenRank Leaderboard, the study found that developers were motivated to engage in more transparent communication, leading to improved collaboration behavior and a better community atmosphere. The leaderboard incentivized developers to make smaller, independent Pull Requests (PR) and avoid direct commits to the repository, ultimately enhancing the quality of code contributions and fostering continuous improvement within the projects. The research also highlighted the role of the leaderboard in promoting healthy competition among developers, encouraging sustained engagement, and driving innovation within the open source projects.

The findings of the study indicated that the OpenRank Leaderboard effectively evaluates and steers developers' contributions, leading to positive behavioral changes and enhanced collaboration habits. Developers expressed a favorable perception of using graph network algorithms for contribution evaluation, with many acknowledging the alignment of rankings with their community perceptions and the value of combining results with community incentive operations. Overall, the study contributes valuable insights into the impacts and perceptions of using leaderboards as a gamification mechanism in company-led open source projects, emphasizing the importance of social network evaluation in motivating open source collaborations and driving innovation in software development.

6.3 Contribution barriers

References

[1] Shaosong Ou Alexander Hars. Working for free? Motivations for participating in open-source projects. In: *International journal of electronic commerce* 6.(3) (2002), pp. 25–39.

[2] Applying a Systematic Literature Review and Content Analysis Method to Analyse Open Source Developers' Forking Motivation Interpretation, Categories and Consequences. In: 24 (June 2020). DOI: 10.3127/ajis.v24i0.1714. URL: https://journal.acs.org.au/index.php/ajis/article/view/1714.

[3] Jules J Berman. *Principles of big data: preparing, sharing, and analyzing complex information*. Newnes, 2013.

[4] Jürgen Bitzer, Wolfram Schrettl & Philipp JH Schröder. Intrinsic motivation in open source software development. In: *Journal of comparative economics* 35.(1) (2007), pp. 160–169.

[5] Knut Blind et al. The impact of Open Source Software and Hardware on technological independence, competitiveness and innovation in the EU economy. In: *Final Study Report. European Commission, Brussels, doi* 10 (2021), p. 430161.

[6] Namjoo Choi & Joseph A Pruett. The characteristics and motivations of library open source software developers: An empirical study. In: *Library & Information Science Research* 37.(2) (2015), pp. 109–117.

[7] TensorFlow Developers. TensorFlow. In: *Zenodo* (2022).

[8] Chris DiBona & Sam Ockman. *Open sources: Voices from the open source revolution*. " O'Reilly Media, Inc.", 1999.

[9] Roy T. Fielding & Gail Kaiser. The Apache HTTP server project. In: *IEEE Internet Computing* 1.(4) (1997), pp. 88–90.

[10] Martin Fink. *The business and economics of Linux and open source*. Prentice Hall Professional, 2003.

[11] Brian Fitzgerald. The transformation of open source software. In: *MIS quarterly* (2006), pp. 587–598.

[12] Alfonso Fuggetta. Open source software—an evaluation. eng. In: *The Journal of systems and software* 66.(1) (2003), pp. 77–90. ISSN: 0164-1212.

[13] Marco Gerosa et al. The shifting sands of motivation: Revisiting what drives contributors in open source. In: *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE. 2021, pp. 1046–1058.

[14] Rishab A Ghosh et al. Free/Libre and Open Source Software: Survey and Study FLOSS. Final Report. In: *International Institute of Infonomics, University of Maastricht, Maastricht, The Netherlands* (2002).

[15] Michaela Greiler, Kim Herzig & Jacek Czerwonka. Code ownership and software quality: a replication study. In: *Proceedings of the 12th Working Conference on Mining Software Repositories*. MSR '15. Florence, Italy: IEEE Press, 2015, 2–12. ISBN: 9780769555942.

[16] Michael Gusenbauer & Neal R. Haddaway. Which academic search systems are suitable for systematic reviews or meta-analyses? Evaluating retrieval qualities of Google Scholar, PubMed, and 26 other resources. In: *Research Synthesis Methods* 11.(2) (2020), pp. 181–217. DOI: https://doi.org/10.1002/jrsm.1378. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jrsm.1378. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jrsm.1378.

[17] Gina Häußge. *A Dev's Guide to Open Source Software Licensing*. https://github.com/readme/guides/open-source-licensing. n.d.

[18] Michael G Jacobides. Regulating Big Tech in Europe: why, so what, and how understanding their business models and ecosystems can make a difference. In: *Available at SSRN 3765324* (2020).

[19] Weiling Ke & Ping Zhang. Motivations for participating in open source software communities: Roles of psychological needs and altruism. In: *PACIS 2008 Proceedings* (2008), p. 76.

[20] Barbara Kitchenham. Procedures for Performing Systematic Reviews. In: *Keele, UK, Keele Univ.* 33 (Aug. 2004).

[21] Karim R Lakhani & Robert G Wolf. Why hackers do what they do: Understanding motivation and effort in free/open source software projects. In: (2005).

[22] Andrew M St Laurent. *Understanding open source and free software licensing: guide to navigating licensing issues in existing & new software*. " O'Reilly Media, Inc.", 2004.

[23] Yan Li, Chuan-Hoo Tan & Hock-Hai Teo. Leadership characteristics and developers' motivation in open source software development. In: *Information & Management* 49.(5) (2012), pp. 257–267.

[24] GNU General Public License. Gnu general public license. In: *Retrieved December* 25 (1989), p. 2014.

[25] Jon Loeliger & Matthew McCullough. *Version Control with Git: Powerful tools and techniques for collaborative software development*. " O'Reilly Media, Inc.", 2012.

[26] Michael J Madison. Reconstructing the software license. In: *Loy. U. Chi. Lj* 35 (2003), p. 275.

[27] Ernesto Moreno & Moreno Muffatto. *Open Source: A Multidisciplinary Approach*. World Scientific Publishing Company, 2006. URL: https://ebookcentral.proquest.com/lib/lut/detail.action?docID=1679623.

[28] Shaul Oreg & Oded Nov. Exploring motivations for contributing to open source initiatives: The roles of contribution context and personal values. eng. In: *Computers in human behavior* 24.(5) (2008), pp. 2055–2073. ISSN: 0747-5632.

[29] Shaul Oreg & Oded Nov. Exploring motivations for contributing to open source initiatives: The roles of contribution context and personal values. In: *Computers in human behavior* 24.(5) (2008), pp. 2055–2073.

[30] Pethuru Raj, Veeramuthu Venkatesh & Rengarajan Amirtharajan. Envisioning the cloud-induced transformations in the software engineering discipline. In: *Software Engineering Frameworks for the Cloud Computing Paradigm* (2013), pp. 25–53.

[31] Jerome H Saltzer. The origin of the "MIT license". In: *IEEE Annals of the History of Computing* 42.(4) (2020), pp. 94–98.

[32] Charles M Schweik & Robert C English. *Internet success: a study of open-source software commons*. MIT Press, 2012.

[33] Andrew Sinclair. License profile: Apache license, version 2.0. In: *IFOSS L. Rev.* 2 (2010), p. 107.

[34] KR Srinath. Python–the fastest growing programming language. In: *International Research Journal of Engineering and Technology* 4.(12) (2017), pp. 354–357.

[35] Richard Stallman. *Why Open Source Misses the Point of Free Software*. https://www.gnu.org/philosophy/open-source-misses-the-point.html. GNU Project - Free Software Foundation, n.d.

[36] Statista. *Infographic: How Big Tech Contributes to Open Source*. Sept. 2021. URL: https://www.statista.com/chart/25795/active-github-contributors-byemployer/.

[37] *What is free software?* https://www.gnu.org/philosophy/free-sw.html. GNU Project - Free Software Foundation, n.d.

[38] Chorng-Guang Wu, James H Gerlach & Clifford E Young. An empirical analysis of open source software developers' motivations and continuance intentions. In: *Information & Management* 44.(3) (2007), pp. 253–262.

[39] Yunwen Ye & Kouichi Kishida. Toward an understanding of the motivation of open source software developers. In: *25th International Conference on Software Engineering, 2003. Proceedings.* IEEE. 2003, pp. 419–429.

[40] Yuxia Zhang et al. How Are Paid and Volunteer Open Source Developers Different? A Study of the Rust Project. In: *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 2024, pp. 1–13.

[41] Shengyu Zhao et al. OpenRank Leaderboard: Motivating Open Source Collaborations Through Social Network Evaluation in Alibaba. In: (2024).

Appendix 1 Selected Articles and Papers for SLR

Table 5: Articles and Papers selected for SLR.

Article ID	Title	Authors	Y.O.P
A01	A systematic literature review on the barriers faced by newcomers to open source software projects	Steinmacher, I., Silva, M. A. G., Gerosa, M. A., & Redmiles, D. F.	2015
A02	Social barriers faced by newcomers placing their first contribution in open source software projects	Steinmacher, I., Conte, T., Gerosa, M. A., & Redmiles, D.	2015
A03	Overcoming social barriers when contributing to open source software projects	Steinmacher, I., Gerosa, M., Conte, T. U., & Redmiles, D. F.	2019
A04	The long road ahead: Ongoing challenges in contributing to large oss organizations and what to do	Guizani, M., Chatterjee, A., Trinkenreich, B., May, M. E., Noa-Guevara, G. J., Russell, L. J., ... & Sarma, A.	2021
A05	Intrinsic motivation in open source software development	Bitzer, J., Schrettl, W., & Schröder, P. J.	2007
A06	Toward an understanding of the motivation of open source software developers	Ye, Y., & Kishida, K.	2003
A07	OpenRank Leaderboard: Motivating Open Source Collaborations Through Social Network Evaluation in Alibaba	Zhao, Shengyu	2024

A08	How Are Paid and Volunteer Open Source Developers Different? A Study of the Rust Project	Zhang, Yuxia	2024
A09	Why hackers do what they do: Understanding motivation and effort in free/open source software projects	Lakhani, K. R., & Wolf, R. G.	2005
A10	An empirical analysis of open source software developers' motivations and continuance intentions	Wu, C. G., Gerlach, J. H., & Young, C. E.	2007
A11	The shifting sands of motivation: Revisiting what drives contributors in open source	Gerosa, M., Wiese, I., Trinkenreich, B., Link, G., Robles, G., Treude, C., ... & Sarma, A.	2021
A12	The characteristics and motivations of library open source software developers: An empirical study	Choi, N., & Pruett, J. A.	2015
A13	Leadership characteristics and developers' motivation in open source software development	Li, Y., Tan, C. H., & Teo, H. H.	2012
A14	On the relationship between newcomer motivations and contribution barriers in open source projects	Hannebauer, C., & Gruhn, V.	2017
A15	Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects	Roberts, J. A., Hann, I. H., & Slaughter, S. A.	2006
A16	Motivations for participating in open source software communities: Roles of psychological needs and altruism	Ke, W., & Zhang, P.	2008

A17	Working for free? Motivations for participating in open-source projects	Alexander Hars, S. O.	2002
A18	Exploring motivations for contributing to open source initiatives: The roles of contribution context and personal values	Oreg, S., & Nov, O.	2008
A19	Community dynamics in open source software projects: Aging and social reshaping	Hannemann, A., & Klamma, R.	2013
A20	The material and social dynamics of motivation: Contributions to Open Source language technology development	Freeman, S.	2007