More Than Code: Technical and Emotional Dynamics in Solidity's Development

Matteo Vaccargiu[1,2], Rumyana Neykova[2], Nicole Novielli[3], Marco Ortu[1], Giuseppe Destefanis[2]
[1]University of Cagliari, Italy
{matteo.vaccargiu,marco.ortu}@unica.it
[2]Brunel University of London, UK
{rumyana.neykova,giuseppe.destefanis}@brunel.ac.uk
[3]University of Bari, Italy
nicole.novielli@uniba.it

***Abstract*—Background: Solidity is the primary programming language used for developing smart contracts on Ethereum, representing a new generation of programming languages developed entirely in open environments.**
Objective: This longitudinal case study examines contribution patterns and emotional dynamics within the Solidity GitHub repository over a ten-year period (2014-2024).
Method: We developed a contribution index combining metrics from developer activities (commits, pull requests, comments, and temporal engagement) and applied emotion detection to study communication patterns in a decade-long dataset of developer interactions.
Results: The top 1% of contributors are responsible for around 85% of project contributions, yet the project exhibits dual paths to prominence: early contributors established technical foundations through code, while later contributors achieved influence through reviews and discussions. Emotional patterns show transitions from initial curiosity and confusion to eventual approval and gratitude.
Conclusion: The project's recognition of diverse contribution types and evolving emotional dynamics enables sustainable growth despite concentrated contributions, demonstrating how open-source languages can evolve while maintaining both technical rigor and community engagement.

I. Introduction

Open-source software (OSS) development relies on contributions from a diverse, global community of contributors. This collaborative approach has led to the creation and maintenance of widely used software, driven by contributors from different countries, backgrounds, and expertise [1]. However, the decentralized and often anonymous nature of OSS contributions introduces challenges in maintaining productive and respectful communication among contributors. Misunderstandings, conflicts, and unprofessional behavior can disrupt collaboration, hinder project progress, and affect the community's overall well-being [2]. A healthy development environment can promote long-term contributor retention, facilitate knowledge transfer, and attract new participants, while a toxic or unproductive environment may lead to contributor burnout, reduced participation, and ultimately threaten the project's longevity [3], [4]. A key aspect of this study is the focus on a programming language repository, rather than a repository for a traditional software system [5]. Studying the dynamics of a language like Solidity introduces challenges and opportunities for research. Unlike typical system development projects where the goal is to build a specific application or tool, contributors to a programming language are engaged in designing and evolving the language itself. This requires a higher level of expertise, as contributors must deeply understand both the technical and conceptual foundations of the language. They are tasked with shaping its syntax, functionality, and ensuring its long-term viability in a highly technical field such as blockchain. This makes language development distinct from traditional software systems, where contributions may not always demand such extensive background knowledge. Contributors in a language repository must handle complexities such as ensuring backward compatibility, managing feature requests, and addressing security considerations that are critical in the context of blockchain technologies.

The Solidity repository exemplifies a unique characteristic of emerging blockchain technologies: the intersection of technical complexity and accessibility. While core development demands expertise in both language design and blockchain technology, Solidity's JavaScript-like syntax and focused use case in smart contracts maintain a low entry barrier. This accessibility, coupled with Web3's growth and opportunities, has promoted widespread community engagement. Unlike traditional programming languages developed in centralized corporate or academic environments, Solidity represents a new generation of languages evolving in transparent, decentralized spaces. This model, combining open development with practical blockchain applications, offers insights valuable beyond Web3—particularly as domain-specific languages emerge for AI, quantum computing, and privacy-preserving computation. Solidity's community-driven development directly shapes the ecosystem of decentralized applications and smart contracts built upon it [6], [7], making it an ideal case study for understanding how contributor dynamics influence broader technological ecosystems. In this longitudinal case study [8], we examine developer communications within the *Solidity GitHub repository*[1] over a decade (2014-2024) [9], [10]. Our analysis is centered on a contribution index we define, which serves as a high-level measure of developer involvement by

[1]https://github.com/ethereum/solidity

combining multiple dimensions of participation [11].

The contribution index aggregates various metrics—such as commits, pull requests, comments, and issues—to rank contributors by their overall impact on the project. This index provides a basis for answering our two first research questions and also allows us to quantitatively assess the patterns of contribution within the Solidity community. While we demonstrate the index's effectiveness using the Solidity repository as our first case study, its design is language and domain-agnostic, making it applicable across diverse open-source projects. This approach to measuring contributions could help other OSS communities better understand their contributor dynamics, identify emerging leaders, and develop targeted strategies for sustainable project growth. The index provides a basis for answering our research questions about the Solidity community but also introduces a new framework for quantitatively assessing contribution patterns in open-source software development more broadly.

We focus on answering the following research questions:

RQ1: What is the distribution pattern of contributions in the Solidity project? We aim to understand how development efforts are distributed among contributors, which has implications for project sustainability and community dynamics. Understanding this distribution is important for assessing potential risks and opportunities in the project's governance model.

RQ2: How do the timing and patterns of initial participation influence contributors' pathways to prominence in the Solidity project? This question explores whether the project is open to new contributors rising to prominence over time or if it is dominated by early joiners.

RQ3: What distinct patterns of emotional expression emerge among contributors in the Solidity repository? Understanding the emotional tone of contributors communications provides insight into the social aspects of their interactions.

RQ4: How do emotional expressions evolve over time? By analyzing how emotions change over time, we seek to understand the long-term emotional trends within the community, as well as any significant shifts that may coincide with changes in the project.

To promote transparency and reproducibility, we have made **our complete analysis pipeline and datasets publicly available in a replication package** [link]. This package includes the raw data collected from the Solidity repository, our preprocessing scripts, the implementation of our contribution index calculation, and the emotion analysis code.

II. Related Works

A. Contribution Patterns in Open Source Software

Prior research has extensively studied how developers contribute to open source software (OSS) projects, revealing that a small group of core developers is typically responsible for the majority of contributions while a larger group makes occasional contributions [12], [13]. Core developers have traditionally been identified through commit-based metrics, specifically those who produce 80% of changes [14]. However, recognizing the limitations of raw commit counts, researchers like Joblin et al. [15] have developed more sophisticated metrics incorporating social and technical perspectives through developer networks. Nakakoji et al. [16] expanded this understanding with their "onion model" of eight different roles, while recent work has highlighted the importance of casual contributors [17] in project sustainability. Song et al. [18] further advanced contribution measurement by considering both technical contributions and social influence through collaboration networks. Their work revealed that core developers with extensive partnerships tend to collaborate with less-connected developers, while studies have shown that projects heavily dependent on few core developers face increased risks of failure [14].

B. Emotional Dynamics in Software Development

The impact of emotions in software development has gained significant research attention, as studies demonstrate how affective states influence work performance and team collaboration [19]. Research has shown that positive affect correlates with increased productivity, while negative states can impair cognitive processes crucial for programming tasks. Emotion analysis in software engineering has evolved from simple sentiment analysis to more nuanced approaches, viewing emotions as complex, dynamic events involving multiple aspects of human behavior [20]–[25]. The field has benefited from advances in affective computing, focusing on emotion detection in communication traces, understanding emotional impact on development processes [4], [26], and providing emotion-aware recommendations [27], [28]. Studies have shown that developers express diverse emotions during collaborative development [29], with emotional awareness proving crucial for team collaboration and project management.

C. Programming Language Evolution and Community Development

The evolution of programming languages is a dynamic process influenced by various factors, including technological advancements, user needs, and community-driven initiatives [30]. Community-driven development has become increasingly significant, particularly in Domain-Specific Languages (DSLs) [31]. Staples et al. [32] analyzed over 393,000 GitHub repositories to demonstrate how the R programming language has evolved through community contributions, while Almeida et al. [33] developed a framework showing how community feedback shaped Python's evolution. While these studies examined languages that grew gradually, Solidity had to evolve rapidly to serve a large blockchain user base from its early stages, providing a unique context for studying contribution dynamics under intense early adoption.

D. Research Gap and Our Contributions

Our work addresses several gaps in the existing literature. While previous studies focused on traditional application-level software projects [12], [13], we examine a domain-specific

programming language repository, finding an even higher contribution concentration (86% from top 1%). We extend Joblin et al.'s [15] network approach by incorporating temporal engagement metrics, and enhance Nakakoji et al.'s [16] model by demonstrating how contributors can rise to prominence through specialized paths. Furthermore, we complement Song et al.'s [18] work by adding emotional analysis to understand how technical discussions and social interactions interplay specifically in language development communities, providing new insights into this specialized technical context that differs from previously studied application development teams.

III. Methodology

Figure 1 shows the systematic process we followed to study activities in the Solidity repository, starting with data collection, definition of the Contribution Index, and moving through the various analytical steps.

A. Dataset Overview

We analyzed data from the Solidity GitHub repository[2] spanning from its creation in 2014 to September 2024, encompassing nearly ten years of development activity. The dataset comprises 24,617 commits, 8,887 pull requests, 6,015 issues, and 107,737 comments. This dataset captures both the technical evolution of the language and the social dynamics of its development community, providing rich material for analyzing contribution patterns and emotional expressions in communications.

The dataset was preprocessed to ensure data quality: we removed bot-generated content [34], standardized timestamps for temporal analysis, and cleaned textual data by removing code snippets and extraneous formatting. Missing values in critical fields such as issue numbers and comment bodies were handled through removal to maintain data integrity.

B. Contribution Index

We developed a novel contribution index that quantifies developer involvement through both code and non-code activities. The index combines six key metrics:

Code Contributions:

- Commits (25%): Direct code contributions to the repository
- Pull Requests Merged (20%): Successfully peer-reviewed code changes

Community Engagement:

- Comments (15%): Participation in discussions and code reviews
- Issues Opened (15%): Problem identification and feature suggestions

Temporal Engagement:

- Active Days (15%): Frequency of participation
- Time Duration (10%): Span of project involvement

The contribution index is calculated through the following steps:

[2]https://github.com/ethereum/solidity

Normalization: Temporal metrics (duration and active days) are normalized by dividing individual values by the maximum observed value, scaling them between 0 and 1.

Weighted Calculation: We compute the raw score using:

$$\begin{aligned}\text{Raw Score} = {} & 0.25 \times \text{Commits} \\ & + 0.20 \times \text{PR Merged} \\ & + 0.15 \times \text{Comments} \\ & + 0.15 \times \text{Issues Opened} \\ & + 0.15 \times \text{Norm. Active Days} \\ & + 0.10 \times \text{Norm. Duration}\end{aligned} \tag{1}$$

Final Normalization: The raw scores are normalized to a 0-100 scale using:

$$\text{Final Score} = \left(\frac{R - R_{\text{min}}}{R_{\text{max}} - R_{\text{min}}} \right) \times 100 \tag{2}$$

where R is the raw score, and R_{min} and R_{max} are the minimum and maximum raw scores across all contributors.

The weights reflect the relative importance of different contribution types while ensuring balanced recognition of both direct code contributions and community engagement. Higher weights for commits and pull requests (45% combined) emphasize the importance of code contributions, while substantial weight for community engagement (30%) and temporal factors (25%) acknowledges the value of sustained participation and non-code contributions to project sustainability. The chosen six metrics represent both technical and community engagement: commits, pull requests merged, comments, issues opened, active days, and time duration. These metrics were selected for their relevance to diverse forms of participation in open-source projects. Code contributions (commits and pull requests) reflect technical impact, while community activities (comments and issues) highlight collaborative and problem-identification roles. Temporal metrics (active days and duration) capture sustained engagement and long-term involvement. We assigned weights to emphasize the importance of code contributions while balancing community engagement and temporal factors. All metrics were computed programmatically (see replication package) using repository data, and normalization ensured comparability across contributors.

C. Research Methodology by Question

1) Distribution Analysis (RQ1): To analyze contribution patterns, we first applied the contribution index across all contributors in the dataset. The distribution analysis involved examining percentile rankings of contributors based on their contribution index, with particular focus on the proportion of total contributions from the top 1%. We further examined contribution ratios across different activity types, including commits, comments, and issues, to understand how different forms of participation were distributed among contributors.

2) Temporal Evolution Analysis (RQ2): Our investigation of how contributors rise to prominence began by identifying the top 1% of contributors based on their contribution index.

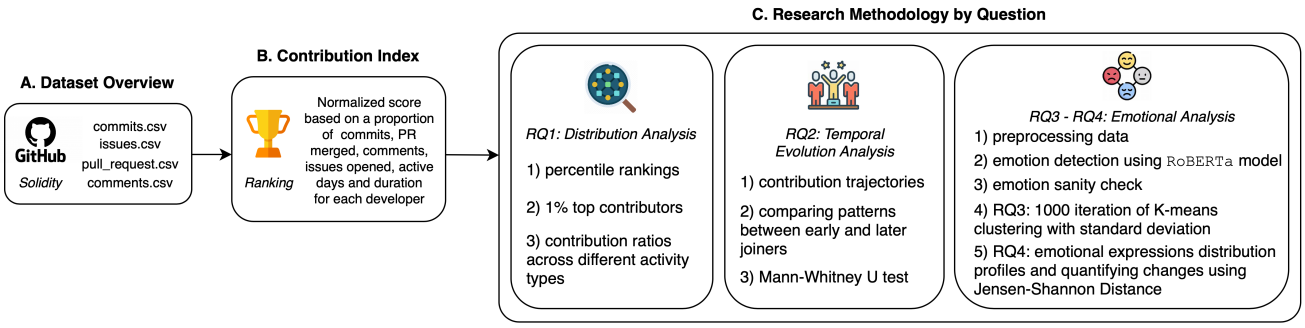

Fig. 1: Description of the methodology

We traced their contribution trajectories from their first interaction with the project, comparing patterns between early joiners (2014-2015) and those who joined later (2020 onwards). To validate our observations statistically, we employed the Mann-Whitney U test [35] to compare contribution patterns between these temporal groups, as the data did not follow a normal distribution.

3) Emotion Analysis (RQ3 & RQ4): The analysis of emotional patterns in contributor communications involved several interconnected phases. Our preprocessing stage focused on ensuring data quality by removing bot-generated content and code snippets, while standardizing text through lowercase conversion and punctuation removal. This cleaning process was essential for focusing our analysis on natural language content that genuinely reflected human communication. For emotion detection, we employed the `roberta-base-go_emotions`[3] model, which uses the RoBERTa architecture to classify text into *27 distinct emotions*. This model's ability to handle multi-label emotion detection made it particularly suitable for analyzing the complex emotional content present in developer communications. To evaluate the reliability of the `roberta-base-go_emotions` classifier, two authors independently annotated a sample dataset. We assessed agreement between Roberta's predictions and the human annotations using both simple percentage agreement and Cohen's kappa [36], with a threshold of 0.5 for binarizing emotion scores into "present" ($\geq$0.5) or "absent" ($<$0.5). Simple agreement percentages exceeded 90% across most emotions, with particularly high alignment for *gratitude*, *surprise*, and *sadness*. Cohen's kappa values provided additional insight, showing high reliability ($\kappa > 0.8$) for emotions like *love*, *curiosity*, and *caring*, but lower reliability for context-dependent emotions such as *approval* ($\kappa = 0.40$) and *anger* ($\kappa = 0.39$). These results indicate that while the classifier reliably detects clear emotional signals, it may require careful interpretation for more nuanced or context-dependent emotions.

For RQ3, we used K-means clustering [37] to identify patterns of emotional expression. The clustering was repeated 1000 times with different initializations to stabilize Z-scores across emotion dimensions, ensuring reliable and stable results. We analyzed cluster characteristics through the mean and standard deviation of standardized emotion scores and examined their relationship with contributors' participation levels. For RQ4, we conducted a temporal analysis of emotional expressions throughout issue lifecycles. This involved mapping emotional patterns from issue creation to closure and quantifying changes in emotional profiles using Jensen-Shannon Distance. We paid particular attention to how dominant emotions evolved over time and how they correlated with different stages of issue resolution.

IV. What is the distribution pattern of contributions in the Solidity project?

Understanding how contributions are distributed in open-source projects is vital for assessing project sustainability and community dynamics. In open-source software (OSS) development, contribution patterns can reveal important insights about project health, knowledge distribution, and potential risks or opportunities in the development process. In the context of open-source software (OSS), previous studies have shown that a small group of contributors tends to dominate the project's development efforts, while a larger number of participants contribute more sporadically [38], [39]. This raises important questions about sustainability, leadership, and community dynamics within OSS projects.

Our rationale for investigating this question in the Solidity project stems from the critical role that a small set of highly active contributors can play in shaping the project's trajectory. Solidity, as a programming language, is foundational to the development of decentralized applications (dApps) and smart contracts on blockchain platforms like Ethereum. As such, the efficiency, security, and evolution of the language are deeply influenced by the contributions made by its community. Identifying the type of distribution of contributions helps us understand the extent to which the project is reliant on a small group of core contributors and how that affects the overall ecosystem.

Analyzing contribution distribution patterns is important for several reasons:

Sustainability: If the Solidity project heavily depends on a small group of contributors, this could present sustainability

[3]https://huggingface.co/SamLowe/roberta-base-go_emotions

risks. These core contributors may experience burnout, or their departure could leave a significant gap in the project's leadership and development efforts. High concentration might create single points of failure, while a more distributed pattern could indicate greater resilience. Understanding these patterns helps in assessing potential risks and planning for project continuity.

Community Inclusivity: The distribution of contributions can reveal how accessible the project is to new contributors. If a small subset of contributors dominates the contributions, it can lead to barriers for new contributors trying to participate or contribute meaningfully. Investigating this distribution allows us to explore the potential for inclusivity and growth in the Solidity community.

Project Governance: A skewed contribution pattern—where contributions are heavily concentrated in a small subset of contributors rather than evenly distributed across the community—can affect decision-making and governance within the project. In open-source projects, active contributors often play a key role in influencing the direction of the project. Understanding how contributions are distributed helps evaluate the project's governance structure and its implications for future development. To answer RQ1, we calculated a contribution index for

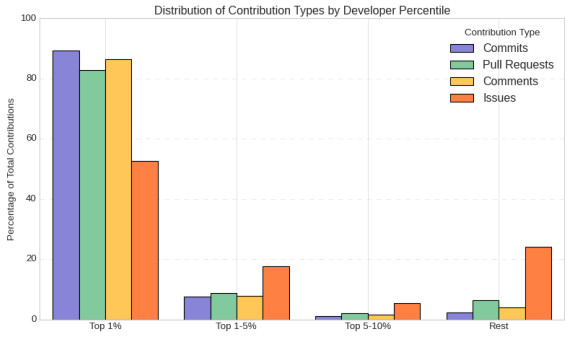

Fig. 2: Distribution of Contribution Types by Developer Percentile

each developer in the Solidity GitHub repository. This index aggregates multiple metrics, such as: the number of commits (code contributions), Pull requests merged, Comments (on issues and pull requests), Issues opened. By considering these activities, we created a normalized expertise score that allows us to rank contributors based on their overall contributions to the project. We then examined the distribution of this contribution index to see whether a small proportion of contributors are responsible for a large proportion of the contributions.

Our analysis revealed that the Solidity project exhibits a concentrated distribution, in line with the results by Yamashita et al. [40] and Xia et al. [41]. In fact, the top 1% of contributors account for nearly 86% of total commits, and 85.5% of all comments in the repository. This extreme concentration of contributions indicates that there is a high bus factor risk and that the project is overwhelmingly driven by a very small group of highly active contributors, while the majority of contributors play a much smaller role.

Figure 2 shows the distribution of different types of contributions across contributor percentiles. The plot shows four types of contributions (commits, pull requests, comments, and issues) for each percentile group of contributors. The results demonstrate a strong concentration of activity among the top contributors: the top 1% of contributors are responsible for 89.23% of all commits, 82.75% of all pull requests, and 86.49% of all comments. This concentration is less pronounced for issue creation, where the top 1% accounts for 52.73% of issues, with a more substantial contribution from other groups (17.56% from the top 1-5%, 5.52% from the top 5-10%, and 24.19% from the remaining contributors).

This pattern suggests that while code contributions and discussions are highly centralized among a small group of core developers, issue reporting shows broader community participation. This broader participation in issue reporting indicates a more diverse engagement in identifying problems and suggesting improvements to the project.

The finding that Solidity's development is dominated by such a small group of contributors has significant implications for the project's future:

High Reliance on Core contributors: The project's reliance on a small core group puts it at risk of stagnation or delays should any of these contributors reduce their involvement. A project as critical as Solidity, which underpins a large portion of the blockchain ecosystem, may face vulnerabilities if this group is not consistently supported or if new contributors are not cultivated.

Barriers to Entry for New contributors: The extreme concentration of contributions could indicate challenges for new contributors trying to break into the project. The dominance of early contributors may make it more difficult for others to rise to positions of influence, potentially hindering community growth and innovation.

Answer to RQ1: the distribution of contributions in the Solidity project is skewed, with the top 1% of contributors responsible for the vast majority of contributions. This finding highlights the importance of recognizing and addressing potential risks related to project sustainability and community inclusivity.

V. How do the timing and patterns of initial participation influence contributors' pathways to prominence in the Solidity project?

To answer RQ2, we began by analyzing the contribution patterns of the top contributors in the Solidity project. We found that the top 1% of contributors were responsible for the vast majority of contributions, forming a core group that significantly drives Solidity's development. To further confirm the results, we performed a statistical analysis. First, we checked for normality in the distribution of the contribution index for both the top 1% group and the rest of the contributors using the Shapiro-Wilk test. The results indicated that neither group followed a normal distribution, with both p-values being extremely small. Consequently, we performed a

Mann-Whitney U test, a non-parametric test more suited to comparing non-normal distributions. The p-value ($p < .001$) strongly indicates that the contribution index of the top 1% of contributors is significantly different from the rest, confirming that the top contributors are in a distinct class of their own.

To further explore how contributors achieve prominence at various stages, we visualized the relationship between the first contribution dates of top developers and their impact on the project, as measured by their contribution index.

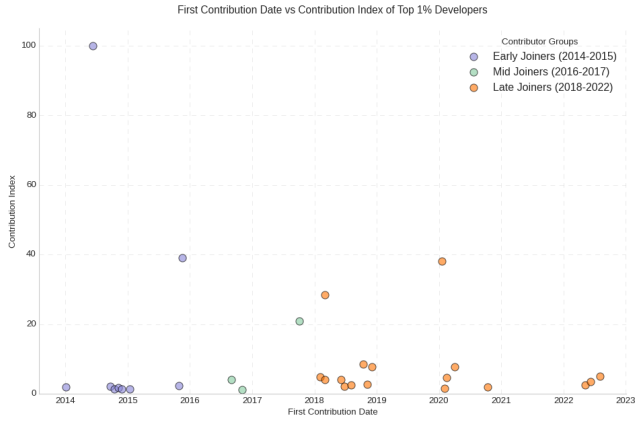

Fig. 3: First Contribution Date vs Contribution Index of Top 1% Developers

Figure 3 demonstrates that top contributors have emerged across the entire project lifecycle. Each point represents a top contributor, categorized by their joining period: early (2014-2015), mid (2016-2017), and late (2018-2022). Notably, while the highest contribution index belongs to an early joiner from 2014, several late joiners have achieved comparable impact. For instance, a 2020 joiner reached an index of 38.17, close to the second-highest index of 39.01 held by a 2015 joiner.

To understand whether contributors can become top contributors at any stage of the project lifecycle, we analyzed the contribution patterns of the top 1% of contributors. We focused on their earliest contributions and the types of contributions they made, such as commits, pull requests, issues, and comments. Our analysis included 29 contributors, and the results are presented in Figure 4.

We assessed the contributions of both early joiners (those who joined in the project's early years, such as 2014-2015) and late joiners (those who joined much later, including 2020 and beyond), focusing on how their contribution types and patterns evolved over time. For privacy reasons, we have anonymized the contributors by assigning each a letter (A, B, C, etc.) based on the order in which they made their first contributions to the project. Contributor A refers to the earliest joiner, Contributor B the second, and so on. This allows us to discuss their contributions without revealing specific identifiers.

Contributors such as **Contributor B**, who made their first contribution in 2015, and **Contributor A**, who joined in 2014, were heavily involved in traditional development activities:
Contributor B contributed **2,449 commits** and **854 pull requests merged**, opening 697 issues. This contributor also participated extensively in project discussions, contributing **11,811 comments** to various pull requests and issues. **Contributor A** made **8,998 commits** and **2,056 pull requests merged**, along with **26,904 comments**. This contributor opened **805 issues**, indicating a well-rounded involvement in both coding, PR-based contributions, and issue reporting.

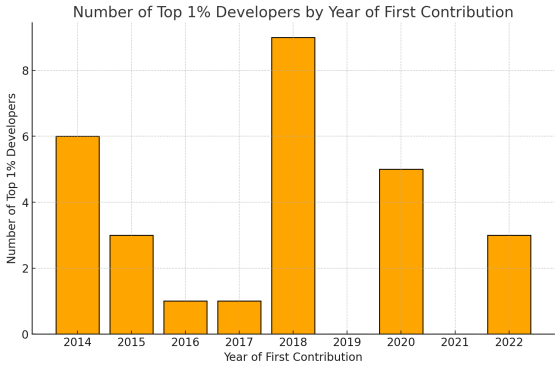

Fig. 4: Number of Top 1% developers by year of first contribution

The behavior of these early contributors demonstrates a sustained focus on core development tasks across a decade, including writing and reviewing code, as well as active participation in issue tracking and discussions. These activities were essential to the foundation and growth of the project. Their contributions included substantial **commits** and **pull requests**, along with a notable volume of **comments** in code review and discussion threads. Additionally, **Contributor A**'s significant number of issues opened reflects an ongoing role in reporting and addressing development challenges throughout this period.

Some contributors achieved top status despite joining as late as 2020. These late joiners contributed in different ways compared to their early counterparts, often focusing on **non-code activities** that supported project management, code review, and issue discussions.
Contributor C, who joined in 2020, made **1,714 commits** and has **517 pull requests merged**, indicating substantial coding activity. However, their main contribution came through **13,534 comments**, suggesting a strong focus on code review and project discussions. This contributor also opened **269 issues**, but their involvement in **comments** far outweighed other contribution types.
Contributor D, also joining in 2020, contributed **220 commits** and has **112 pull requests merged**. Similar to Contributor C, this contributor focused significantly on **comments**, with **1,563 comments** made, but opened only 14 issues.

These late joiners showed a pattern of involvement that emphasizes **non-code contributions**, particularly in areas such as reviewing code, engaging in discussions, and providing feedback. Their relatively lower number of commits and pull requests, compared to early joiners, suggests that they gained prominence by playing important roles in maintaining project quality and coordination rather than primarily writing code.

Early joiners such as **Contributor A** and **Contributor B** were primarily engaged in **code contributions** (commits and pull requests), dedicating much of their efforts to core development tasks. They also contributed significantly to project discussions, as reflected in their high comment counts, but their role in issue reporting was minimal.

Late joiners such as **Contributor C** and **Contributor D** followed a different trajectory. While they made significant contributions in terms of commits and pull requests, they focused more heavily on **comments** and **project discussions**. Their roles were more oriented toward **code reviews**, feedback, and collaborative efforts, which are essential for maintaining the integrity of the project and guiding its development. Late joiners contributed far fewer issues or commits, yet they rose to the top through consistent involvement in code review and governance tasks.

Contributors who joined early (2014-2015) were more likely to contribute extensively to **core development** (commits and pull requests), playing a foundational role in shaping the project.

Contributors who joined later (2020 onward) became top contributors by focusing on **collaborative efforts** such as reviewing code, providing feedback, and discussing project improvements. Their involvement in **non-code tasks** such as comments and discussions helped them rise to prominence despite contributing fewer commits and PRs.

The top 1% of contributors have consistently driven development within the Solidity project, though their focus has shifted over time. Early joiners (2014–2015) primarily engaged in core development tasks, contributing large numbers of commits and pull requests as they established foundational aspects of the project. In contrast, contributors joining from 2020 onward have taken on a more collaborative role, with significant involvement in code reviews and discussions. This pattern suggests that while early contributors shaped the technical framework, later contributors play a role in project governance and community engagement. The Solidity repository demonstrates a flexible contribution model, where both coding and collaborative efforts are valued across the project's lifecycle.

Answer to RQ2: Our analysis reveals that early joiners (2014-2015) rose to influence through extensive code contributions and core language development, while later contributors (2020 onwards) achieved impact through code reviews, technical discussions, and architectural oversight. This evolution of pathways reflects the project's maturing needs - from foundational development to technical governance and community leadership.

VI. What distinct patterns of emotional expression emerge among contributors in the project?

To explore how contributors' emotional expression patterns relate to their participation in the project, we analyzed a group of 50 contributors, comprising all top 1% (29 contributors who drive 86% of project contributions) and 21 systematically selected contributors with progressively lower indices (they account for over 91% of the total contributions). This balanced design ensures statistical power for comparing emotional patterns across contribution levels while controlling for potential sampling bias through stratified selection. By including both all high-impact contributors and a representative sample across other contribution levels, we can identify whether emotional patterns are unique to top contributors or present more broadly. Each contributor's emotional profile was derived from their communication based on 27 emotional dimensions, capturing a unique distribution of emotional expression. We aimed to identify patterns of emotional expression among contributors and analyze how these patterns relate to their contributions. The analysis followed these steps:

Data Preparation: The dataset consisted of 27 emotions, which we normalized to form emotion distributions for each contributor. This ensured that each row represented a valid probability distribution across emotions.

Standardization: To make the emotional features comparable, we standardized the distributions using Z-score[4] normalization, allowing for analysis across all emotional dimensions.

Clustering Analysis: To ensure robust results, we performed K-Means clustering 1000 times with different random initializations. For each cluster and emotion, we calculated both the mean Z-score and its standard deviation, providing a complete measure of clustering stability. This approach helps quantify the uncertainty in our emotional pattern identification and ensures the reliability of our findings.

Performance Comparison: We assessed contributor performance based on their "Contribution Index" and compared this index across the four clusters to understand any relationship between emotional patterns and performance.

We chose K-means clustering over more complex methods (such as hierarchical clustering, DBSCAN, or spectral clustering) due to its interpretability advantages in this context. The centroid-based approach of K-means provides intuitive mean emotion profiles that directly capture specific communication patterns, making it easier to understand and explain the different types of emotional engagement among contributors. To ensure robustness in determining the optimal number of clusters, we conducted a stability analysis using multiple methods: the Elbow method, Silhouette analysis, Gap statistic, Calinski-Harabasz index, and Davies-Bouldin index. The majority of these methods consistently suggested four clusters as the optimal number, with the Calinski-Harabasz index showing the highest score for k=4 (score=156.3) and the Davies-Bouldin index showing its lowest value (score=0.82) at the same point. This convergence across different vali-

[4]Z-scores measure how far each emotion's expression deviates from the overall mean: yellow indicates the emotion is expressed more frequently than average, while blue indicates less frequent expression. For example, a Z-score of 1.0 means the emotion is expressed one standard deviation more frequently than the mean across all contributors. Each cell displays both the mean Z-score and its standard deviation (±) across 1000 iterations

dation techniques strengthened our confidence in the four-cluster solution. Figure 5 presents the results of our clustering

Fig. 5: Cluster Centroids Heatmap for Emotion Distributions. The heatmap illustrates the average emotional profile for each of the four clusters, displaying distinct communication styles among contributors based on 27 emotional dimensions.

analysis, showing both the mean Z-scores and their associated standard deviations across 1000 iterations. Each cell displays two values: the mean Z-score (indicated by color and primary number) and the standard deviation (shown as ± value). This dual representation provides insight into both the strength and stability of emotional patterns within each cluster. The color gradient ranges from dark blue (negative Z-scores) to yellow (positive Z-scores), while the standard deviations help identify which emotional patterns are most reliable within each cluster.

The emotional profiles of each cluster reveal distinct patterns of emotional expression, with varying degrees of stability measured across 1000 iterations. Our analysis reveals the following characteristic patterns:

Cluster 0 - Balanced Emotional Expression: Shows consistent control over fundamental emotions with *surprise* (-0.07 ± 0.59), *optimism* (-0.06 ± 0.63), *caring* (0.05 ± 0.64), *gratitude* (-0.01 ± 0.67), and *approval* (-0.02 ± 0.75). More variable in complex emotions: *embarrassment* (0.16 ± 1.15), *anger* (0.15 ± 1.19), *pride* (0.16 ± 1.20), *annoyance* (0.15 ± 1.21), and *grief* (0.18 ± 1.22), suggesting measured communicators who maintain emotional balance. This cluster shows the smallest overall variance across all emotions.

Cluster 1 - Dynamic Engagement Pattern: Maintains steady basic reactions through *admiration* (-0.15 ± 0.39), *optimism* (-0.24 ± 0.51), *remorse* (-0.28 ± 0.52), *surprise* (0.14 ± 0.53), and *disapproval* (-0.10 ± 0.58). Shows dramatic range in intense emotions: *excitement* (1.28 ± 1.96), *disgust* (1.57 ± 2.03), *anger* (1.64 ± 2.21), *pride* (1.61 ± 2.22), and *grief* (1.69 ± 2.25), indicating communicators who maintain composure in routine interactions but express significant emotional depth in complex situations. This cluster exhibits the widest emotional range among all groups.

Cluster 2 - Critical Engagement Pattern: Demonstrates emotional stability in *optimism* (-0.30 ± 0.58), *surprise* (-0.01 ± 0.61), *gratitude* (-0.47 ± 0.65), *remorse* (-0.27 ± 0.68), and *disapproval* (0.01 ± 0.77). Shows heightened variability in

TABLE I: Clusters and Contributors' Performance Summary

Cluster	Cluster Characteristics	Average Contribution Index (Absolute)	Average Contribution Index (Normalized)
0	Balanced Emotional Expression with Stable Patterns (-0.75 ± 0.59)	456.01	6.69
1	Dynamic Engagement with High Emotional Variance (-2.25 ± 0.39)	166.84	2.45
2	Critical Engagement with Variable Intensity (-2.45 ± 0.58)	135.54	1.99
3	Moderate Pattern with Mixed Stability (-2.74 ± 0.63)	180.24	2.64

critical responses: *grief* (1.64 ± 2.11), *pride* (1.52 ± 2.12), *embarrassment* (1.73 ± 2.20), *annoyance* (1.88 ± 2.20), and *disappointment* (1.50 ± 2.45), characterizing communicators who maintain composure but express strong reactions to challenging situations. The cluster particularly stands out in its expression of annoyance and embarrassment.

Cluster 3 - Moderate Intensity Pattern: Exhibits stability in foundational emotions: *surprise* (-0.17 ± 0.63), *gratitude* (-0.47 ± 0.66), *optimism* (-0.31 ± 0.67), *amusement* (0.14 ± 0.78), and *remorse* (-0.18 ± 0.81). Displays significant variation in complex responses: *grief* (1.06 ± 1.81), *realization* (1.32 ± 2.00), *annoyance* (1.77 ± 2.43), *embarrassment* (1.71 ± 2.46), and *disappointment* (1.69 ± 2.74), representing communicators who maintain emotional steadiness but show substantial variation in nuanced emotional contexts. This cluster shows the highest variability in disappointment and embarrassment among all groups.

Table I presents the average performance, measured by the Contribution Index, for each cluster. The robustness of these findings is supported by our analysis across 1000 iterations.

Contributors in **Cluster 0 - Balanced Emotional Expression with Stable Patterns**, characterized by consistently low variability in emotional responses (-0.75 ± 0.59), exhibit the highest average contribution index (456.01, 6.69). This suggests that contributors with stable and balanced communication styles contribute the most extensively to the repository.

Cluster 1 - Dynamic Engagement with High Emotional Variance, marked by significant variability in emotional expressions (-2.25 ± 0.39), demonstrates a lower average contribution index (166.84, 2.45). While contributors in this group show dynamic engagement and emotional range, their contributions are less extensive compared to the balanced communication of Cluster 0.

Cluster 2 - Critical Engagement with Variable Intensity, characterized by high variability in critical emotions (-2.45 ± 0.58), exhibits the lowest average contribution index (135.54, 1.99). This suggests that predominantly critical and variable emotional patterns may limit the extent of contributors' engagement with the repository.

Cluster 3 - Moderate Pattern with Mixed Stability, showing moderate variability in emotional responses (-2.74 ± 0.63), achieves an average contribution index of (180.24, 2.64). This indicates that a combination of stability and variability can support meaningful contributions but does not reach the levels observed in Cluster 0.

These findings highlight that contributors in Cluster 0, characterized by balanced and stable emotional expressions, achieve the most substantial contribution levels, both in absolute and normalized terms. In contrast, contributors with more variable emotional patterns, whether dynamic, critical, or moderate, tend to have lower contribution indices.

Answer to RQ3: Contributors in Cluster 0, marked by balanced and stable emotional expressions, achieve the highest contribution levels, suggesting that consistent and measured communication styles are associated with greater involvement in the repository compared to dynamic, critical, or purely supportive styles.

VII. How do emotional expressions evolve over time?

The progression of emotional expressions across various stages of issue handling sheds light on how top contributors engage with the Solidity project over time. This section analyzes these emotional expressions by examining different phases in the issue lifecycle, focusing specifically on the contributions of the top 1% of contributors.

Our analysis began by examining how the top contributors distribute their comments across the different stages in the lifecycle of a closed issue.

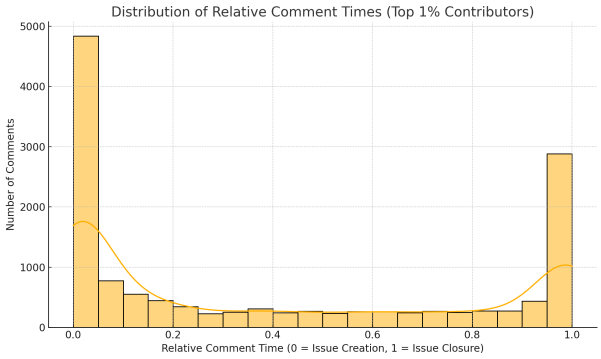

Fig. 6: Distribution of Relative Comment Times (Top 1% Contributors)

Figure 6 shows the distribution of comment activity by the top 1% of contributors throughout the lifecycle of closed issues. The x-axis represents the relative comment time, where 0 indicates the issue's creation and 1 represents its closure. The histogram reveals two distinct peaks: a significant one at the start (near 0), indicating high engagement when issues are initially opened, and another around the closure point (near 1), suggesting increased activity as issues are resolved.

The middle portion of the lifecycle sees a lower volume of comments, which may reflect a phase of active but less visible work, such as internal development, testing, or discussions occurring outside the public comment section. The pattern of heightened engagement at the beginning and end phases highlights a common workflow where contributors focus on framing the issue initially and then return to confirm or finalize its resolution.

To examine how emotional expression changes throughout an issue's lifecycle, we analyzed the distribution of emotions in the initial and final comments of closed issues.

The initial comments of issues reveal that the most common emotions are *approval*, *confusion*, and *curiosity*. This aligns with contributors' early efforts to seek clarification or understand the task as the issue is opened. By the end of the issue, there is a shift in emotional tone. While *approval* and *curiosity* remain prominent, emotions like *disappointment* and *disgust* increase, possibly indicating dissatisfaction with the issue's resolution or management. Additionally, *relief* becomes more visible, likely reflecting the resolution of challenging issues.

We further analyzed the probability distribution of emotional profiles in the top 1% of contributors' comments, focusing on the first and second peaks of comment activity. The probabilities associated with each emotion at these stages are presented in the barplot in Figure 7.

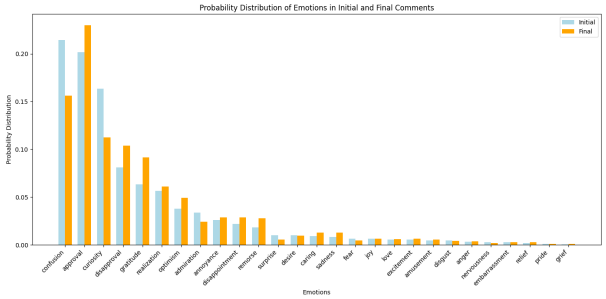

Fig. 7: Probability distribution of Emotions in Initial (light-blue) and Final (orange) Comments (Top 1% Contributors)

From the barplot, we observe that emotions such as *confusion* and *curiosity*, two of the primary emotions among the top 1% contributors, are prominent in the initial peak of comments. This suggests that early comments often reflect the contributors' initial challenges in addressing the issue, accompanied by a strong interest in finding a solution. In contrast, emotions like *disapproval*, *approval*, and *gratitude* are more common in the final comments. This shift indicates a mix of reactions: some contributors express disagreement with the outcome, while others show appreciation for the effort and results achieved.

Finally, we calculated the Jensen-Shannon Distance (JSD) between the probability distributions of emotions in the initial and final comments, finding a value of approximately 10%. JSD is particularly useful in comparing probability distributions, as it quantifies the divergence between two distributions while remaining bounded and symmetric. While a 10% JSD may not seem large in some contexts, it represents a meaningful shift when examining emotional expression. This value reflects a measurable change in contributors' emotional profiles from the beginning to the end of an issue's lifecycle, highlighting a dynamic adaptation in response to evolving challenges and resolutions. Such an emotional shift provides insight into how contributors' reactions are shaped over time as they engage more deeply with the issue.

Answer to RQ4: our findings reveal that during the early stages of an issue, emotions like *curiosity* and *confusion* are more prevalent as contributors explore and clarify the problem. Toward the issue's closure, emotions such as *approval*, and *gratitude* become more frequent, reflecting consensus and successful resolution.

VIII. Discussion

Our longitudinal case study of the Solidity repository, spanning nearly a decade (2014-2024), provides three key contributions to understanding open-source language development: *a novel contribution measurement framework*, *insights into emotional dynamics in technical communities*, and *patterns of sustainable project growth*.

A New Framework for Measuring OSS Contributions: our contribution index combines technical and social aspects of participation, balancing code contributions (45% for commits and pull requests), community engagement (30% for comments and issues), and temporal involvement (25%). This balanced approach effectively identifies diverse contributor profiles—from core developers to community managers—and reveals how late-joining contributors can rise to prominence through varied activities, offering valuable insights for other open-source projects, particularly emerging domain-specific languages.

Emotional Intelligence in Technical Communities: our emotion analysis, leveraging the state-of-the-art `roberta-base-go_emotions` model capable of detecting 27 distinct emotions, significantly advances previous studies by providing a more detailed understanding of developer interactions. Contributors with balanced emotional profiles (Cluster 0) achieved higher contribution indices, suggesting that emotional expression significantly influences collaboration effectiveness. The temporal analysis of emotions during issue lifecycles—transitioning from curiosity and confusion to approval and resolution—provides a framework for understanding and managing technical discussions. These insights could inform how project maintainers structure their communication practices and issue resolution processes.

Sustainable Growth Through Diverse Participation: while we found highly concentrated contribution patterns (top 1% responsible for 86% of commits), our temporal analysis revealed successful integration of new contributors through diverse participation paths. This suggests that language development projects can maintain technical rigor while promoting community growth by recognizing and supporting different forms of contribution. The emergence of specialized roles—from core development to community engagement—indicates natural evolution paths for sustainable project growth.

Recommendations and Future Directions: our findings emphasize the importance of structured knowledge-sharing between core and newer contributors, recognition systems for non-code contributions, and adapting issue management processes to emotional patterns. While the contribution index is a practical tool for identifying key contributors in the Solidity community, its applicability to other projects should be approached cautiously, given the unique characteristics of the Solidity repository. Future work should validate this framework across diverse repositories to assess its generalizability and identify potential domain-specific differences.

IX. Threats to Validity

Construct Validity: The primary threat lies in our operationalization of contributor involvement through the contribution index. The selection and weighting of different contribution types might not capture all aspects of meaningful participation. We mitigated this by incorporating multiple dimensions (code, communication, temporal engagement) and validating our weighting scheme through sensitivity analysis. For the `roberta-base-go_emotions` model, two authors independently annotated a sample dataset, revealing high reliability ($\kappa > 0.8$) for emotions like *curiosity* and *caring*, but lower reliability for context-dependent emotions like *approval* ($\kappa = 0.40$). We considered these variations in reliability when interpreting our emotional analysis results.

Internal Validity: Contributors' rise to prominence might be influenced by external factors not captured in our dataset, such as off-platform discussions. We minimized this by focusing on publicly available data and cross-validating findings across multiple contribution metrics.

External Validity: While Solidity is significant in the blockchain domain, it may not represent all programming language repositories. Its unique characteristics—blockchain context, financial implications, and relative youth—limit generalizability. We frame our findings within the context of emerging domain-specific languages.

Reliability: Our analysis depends on GitHub data completeness, potentially affected by deleted comments or modified pull requests. We enhanced reliability by making our analysis scripts public and validating emotion detection through manual sampling.

Time-based Validity: The blockchain space evolved significantly during our study period (2014-2024), with market conditions, technological advances, and community maturity potentially influencing contributor behavior. The rapid growth of the Ethereum ecosystem, shifts in blockchain technology, and varying levels of market interest may have affected both the volume and nature of contributions. While this evolution provides rich data for longitudinal analysis, it also means that patterns observed during different periods might reflect external factors beyond project dynamics.

X. Conclusion

By examining Solidity's development, we have documented a case of a programming language born and maintained entirely in the open, transparent environment of GitHub—a stark contrast to the traditionally closed development of programming languages. Our analysis, supported by a contri-

bution index and emotion detection approach, reveals that successful open language development requires a delicate balance between technical governance and community dynamics. Our findings demonstrate that programming languages can effectively evolve through community-driven processes while maintaining the technical rigor necessary for critical applications. The success of this open development model, despite its challenges, suggests a viable path forward for future domain-specific languages. Future studies might examine how our findings translate to other emerging programming languages and explore how different technological domains influence community dynamics and development patterns.

REFERENCES

[1] M. Gharehyazie, D. Posnett, B. Vasilescu, and V. Filkov, "Developer initiation and social interactions in oss: A case study of the apache software foundation," *Empirical Software Engineering*, vol. 20, pp. 1318–1353, 2015.

[2] I. Steinmacher, M. A. G. Silva, M. Gerosa, and D. Redmiles, "A systematic literature review on the barriers faced by newcomers to open source software projects," *Inf. Softw. Technol.*, vol. 59, pp. 67–85, 2015.

[3] G. Destefanis, M. Ortu, S. Counsell, S. Swift, M. Marchesi, and R. Tonelli, "Software development: do good manners matter?" *PeerJ Computer Science*, vol. 2, p. e73, 2016.

[4] M. Ortu, B. Adams, G. Destefanis, P. Tourani, M. Marchesi, and R. Tonelli, "Are bullies more productive? empirical study of affectiveness vs. issue fixing time," in *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*. IEEE, 2015, pp. 303–313.

[5] P. Chakraborty, R. Shahriyar, A. Iqbal, and G. Uddin, "How do developers discuss and support new programming languages in technical q&a site? an empirical study of go, swift, and rust in stack overflow," *Inf. Softw. Technol.*, vol. 137, p. 106603, 2021.

[6] S. Aufiero, G. Ibba, S. Bartolucci, G. Destefanis, R. Neykova, and M. Ortu, "Dapps ecosystems: mapping the network structure of smart contract interactions," *EPJ Data Science*, vol. 13, 2024.

[7] G. Wood *et al.*, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

[8] R. K. Yin, *Case study research: Design and methods*. sage, 2009, vol. 5.

[9] V. Cosentino, J. L. C. Izquierdo, and J. Cabot, "Findings from github: Methods, datasets and limitations," *2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR)*, pp. 137–141, 2016.

[10] E. Cohen and M. Consens, "Large-scale analysis of the co-commit patterns of the active developers in github's top repositories," *2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR)*, pp. 426–436, 2018.

[11] K. Blincoe, J. Sheoran, S. Goggins, E. Petakovic, and D. Damian, "Understanding the popular users: Following, affiliation influence and leadership on github," *Inf. Softw. Technol.*, vol. 70, pp. 30–39, 2016.

[12] A. Mockus, R. T. Fielding, and J. D. Herbsleb, "Two case studies of open source software development: Apache and mozilla," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 11, no. 3, pp. 309–346, 2002.

[13] G. Robles, J. M. Gonzalez-Barahona, and I. Herraiz, "Evolution of the core team of developers in libre software projects," *2009 6th IEEE International Working Conference on Mining Software Repositories*, pp. 167–170, 2009.

[14] T. Bock, N. Alznauer, M. Joblin, and S. Apel, "Automatic core-developer identification on github: A validation study," *ACM Transactions on Software Engineering and Methodology*, vol. 32, pp. 1 – 29, 2023.

[15] M. Joblin, S. Apel, C. Hunsen, and W. Mauerer, "Classifying developers into core and peripheral: An empirical study on count and network metrics," *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, pp. 164–174, 2016.

[16] K. Nakakoji, Y. Yamamoto, Y. Nishinaka, K. Kishida, and Y. Ye, "Evolution patterns of open-source software systems and communities," in *Proceedings of the international workshop on Principles of software evolution*, 2002, pp. 76–85.

[17] J. Coelho, M. T. Valente, L. L. Silva, and A. C. Hora, "Why we engage in floss: Answers from core developers," *2018 IEEE/ACM 11th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, pp. 114–121, 2018.

[18] Y. Song, T. Wang, Y. Shen, and J. Chang, "A new method for evaluating core developers in open source software," *2022 IEEE 13th International Conference on Software Engineering and Service Science (ICSESS)*, pp. 48–53, 2022.

[19] N. Novielli and A. Serebrenik, "Sentiment and emotion in software engineering," *IEEE Software*, vol. 36, no. 5, pp. 6–23, 2019.

[20] Z. Chen, Y. Cao, H. Yao, X. Lu, X. Peng, H. Mei, and X. Liu, "Emoji-powered sentiment and emotion detection from software developers' communication data," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 30, no. 2, pp. 1–48, 2021.

[21] R. Cowie, "Emotional life, terminological and conceptual clarifications: Technical report," *FP6 Project*, vol. 507422, 2006.

[22] D. Girardi, F. Lanubile, N. Novielli, and A. Serebrenik, "Emotions and perceived productivity of software developers at the workplace," *IEEE Transactions on Software Engineering*, vol. 48, no. 9, pp. 3326–3341, 2021.

[23] S. F. Huq, A. Z. Sadiq, and K. Sakib, "Is developer sentiment related to software bugs: An exploratory study on github commits," in *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2020, pp. 527–531.

[24] M. Ortu, S. Vacca, G. Destefanis, and C. Conversano, "Cryptocurrency ecosystems and social media environments: An empirical analysis through hawkes' models and natural language processing," *Machine Learning with Applications*, vol. 7, p. 100229, 2022.

[25] A. S. M. Venigalla and S. Chimalakonda, "Understanding emotions of developer community towards software documentation," in *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. IEEE, 2021, pp. 87–91.

[26] M. Mäntylä, B. Adams, G. Destefanis, D. Graziotin, and M. Ortu, "Mining valence, arousal, and dominance: possibilities for detecting burnout and productivity?" in *Proceedings of the 13th international conference on mining software repositories*, 2016, pp. 247–258.

[27] D. Graziotin, X. Wang, and P. Abrahamsson, "Happy software developers solve problems better: psychological measurements in empirical software engineering," *PeerJ*, vol. 2, p. e289, 2014.

[28] D. Graziotin, F. Fagerholm, X. Wang, and P. Abrahamsson, "What happens when software developers are (un) happy," *Journal of Systems and Software*, vol. 140, pp. 32–47, 2018.

[29] A. Murgia, P. Tourani, B. Adams, and M. Ortu, "Do developers feel emotions? an exploratory analysis of emotions in software artifacts," in *Proceedings of the 11th working conference on mining software repositories*, 2014, pp. 262–271.

[30] J. Kumar and S. Chimalakonda, "On the need for empirically investigating fast-growing programming languages," in *Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings*, 2024, pp. 372–373.

[31] J. L. C. Izquierdo and J. Cabot, "Community-driven language development," *2012 4th International Workshop on Modeling in Software Engineering (MISE)*, pp. 29–35, 2012.

[32] T. L. Staples, "Expansion and evolution of the r programming language," *Royal Society Open Science*, vol. 10, 2022.

[33] R. J. de Alencar Almeida, V. H. S. Durelli, I. C. Moraes, M. C. Viana, E. C. Fazzion, D. B. F. Carvalho, D. R. C. Dias, and L. C. D. da Rocha, "Combining data mining techniques for evolutionary analysis of programming languages," *2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI)*, pp. 1–8, 2019.

[34] M. Golzadeh, A. Decan, D. Legay, and T. Mens, "A ground-truth dataset and classification model for detecting bots in github issue and pr comments," *Journal of Systems and Software*, vol. 175, p. 110911, 2021.

[35] P. E. McKnight and J. Najab, "Mann-whitney u test," *The Corsini encyclopedia of psychology*, pp. 1–1, 2010.

[36] M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia medica*, vol. 22, no. 3, pp. 276–282, 2012.

[37] D. Steinley, "K-means clustering: a half-century synthesis," *British Journal of Mathematical and Statistical Psychology*, vol. 59, no. 1, pp. 1–34, 2006.

[38] M. Goeminne and T. Mens, "Evidence for the pareto principle in open source software activity," in *the Joint Porceedings of the 1st International workshop on Model Driven Software Maintenance and 5th International Workshop on Software Quality and Maintainability*. Citeseer, 2011, pp. 74–82.

[39] S. Koch and G. Schneider, "Effort, co-operation and co-ordination in an open source software project: Gnome," *Information Systems Journal*, vol. 12, no. 1, pp. 27–42, 2002.

[40] K. Yamashita, S. McIntosh, Y. Kamei, A. E. Hassan, and N. Ubayashi, "Revisiting the applicability of the pareto principle to core development teams in open source software projects," in *Proceedings of the 14th international workshop on principles of software evolution*, 2015, pp. 46–55.

[41] X. Xia, Z. Weng, W. Wang, and S. Zhao, "Exploring activity and contributors on github: Who, what, when, and where," *2022 29th Asia-Pacific Software Engineering Conference (APSEC)*, pp. 11–20, 2022.