Sociotechnical Dynamics in Open Source Smart Contract Repositories: An Exploratory Data Analysis of Curated High Market Value Projects

Saori Costa
State University of Ceará
Fortaleza, Ceará, Brazil
saori.costa@aluno.uece.br

Matheus Paixao
State University of Ceará
Fortaleza, Ceará, Brazil
matheus.paixao@uece.br

Igor Steinmacher
Northern Arizona University
Flagstaff, Arizona, United States
igor.steinmacher@nau.edu

Pamella Soares
State University of Ceará
Fortaleza, Ceará, Brazil
pamella.soares@aluno.uece.br

Allyson Allex Araújo
Federal University of Cariri
Juazeiro do Norte, Ceará, Brazil
allysson.araujo@ufca.edu.br

Jerffeson Souza
State University of Ceará
Fortaleza, Ceará, Brazil
jerffeson.souza@uece.br

ABSTRACT

Blockchain and Smart Contracts (SCs) have emerged as a promising avenue for organizations looking to innovate. Similar to other fields of software engineering, collaborative platforms, such as GitHub, are gaining attention in SCs development. Moreover, public blockchain platforms, such as Ethereum, commonly serve as a medium to deploy SCs. This ecosystem serves as the basis on which the sociotechnical phenomenon of SC development emerges. Despite the growth of research regarding SCs, there is a gap in understanding the sociotechnical factors involved in their development, specially the ones with high market value. To address this issue, we leveraged Sociotechnical Theory and Data Analysis to investigate the sociotechnical dynamics in open source repositories of SCs deployed on Ethereum. To ensure suitability for our analysis, we curated a list of 16 high market value SCs deployed on Ethereum. Our research yielded four primary analyses. First, we unveiled how collaboration aspects are impacted by the deployment of SCs. Second, we explored the characteristics of contributors participating in these projects. Third, we looked into commit messages to categorize commonly performed software changes. Fourth, we investigated the relationship between market metrics and SC evolution. These analyses help to deepen the understanding of sociotechnical dynamics within SC repositories, assisting organizations in designing better strategies to support their development efforts.

CCS CONCEPTS

• **Software and its engineering → Collaboration in software development**.

KEYWORDS

Smart Contracts; Sociotechnical Theory; Software Evolution; Exploratory Data Analysis.

ACM Reference Format:
Saori Costa, Matheus Paixao, Igor Steinmacher, Pamella Soares, Allyson Allex Araújo, and Jerffeson Souza. 2024. Sociotechnical Dynamics in Open Source Smart Contract Repositories: An Exploratory Data Analysis of Curated High Market Value Projects. In *Proceedings of the 20th International Conference on Predictive Models and Data Analytics in Software Engineering (PROMISE '24), July 16, 2024, Porto de Galinhas, Brazil*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3663533.3664038

1 INTRODUCTION

Blockchain technology has extended its applications beyond the context of cryptocurrencies, particularly with the emergence of the Ethereum platform and the Software Engineering (SE) of Smart Contracts (SCs), which have played a key role in facilitating the creation of decentralized applications (dApps) [4]. A dApp is an application that hosts parts of their back-end services and databases on peer-to-peer (p2p) networks, such as a blockchain, instead of typically centralized servers [16]. The market share for dApps is expected to reach USD 368.25 billion by 2027 [12].

Ethereum is one of the most important blockchain platforms for dApps, designed to function as a versatile, adaptive, and potent global shared infrastructure [24]. To this end, Ethereum offers a Turing-complete programming environment, housing the Ethereum Virtual Machine (EVM) as a runtime apparatus responsible for the execution of SC bytecode [14]. These SCs manifest themselves as self-verifying, tamper-resistant source code, delineating predefined obligations to be autonomously executed [31]. Given this distinct set of particularities, the development and maintenance of SCs entail addressing specific constraints, such the need for simplicity to mitigate operational expenses in the context of cryptocurrencies and the requirement for transparent access to the source code of the SC [5]. An important aspect of SC maintenance is data immutability. Once a SC is deployed, it cannot be modified, which changes the way software engineers handle deployment process [6].

In blockchain-based software engineering [2], an observable practice within the SC ecosystem is the adoption of open source software (OSS) development [18], with code-hosting platforms like

PROMISE '24, July 16, 2024, Porto de Galinhas, Brazil

ACM ISBN 979-8-4007-0675-2/24/07…$15.00
https://doi.org/10.1145/3663533.3664038

GitHub becoming central. This phenomenon has given rise to a thriving sociotechnical environment for SC development. The OSS's inherent public and extensive nature provides a fertile ground for gaining access to valuable data, which can be harnessed through data mining techniques to extract valuable lessons [13, 15].

As emphasized by Prikladnicki *et al.* [21], software systems are products crafted *by* and *for* individuals. In this context, the social and collaborative aspects behind the development, comprehending the collaborative aspects intrinsic to software development also assumes a critical role. When considering the substantial impact of sociotechnical factors and community engagement on the success of open source projects [27], it becomes clear the need to further understand this phenomenon within the context of SCs development. Drawing inspiration from Ghaffari *et al.* [11], our study is guided by the framework proposed by Bostrom and Heinen [3], which conceptualizes a sociotechnical phenomenon through a network of four interactive components divided into two fundamental subsystems: technology and tasks (as the *technical subsystem*) and structure and actors (as the *social subsystem*).

Hence, with this motivation, this study employs an exploratory and experimental approach to analyze a sample of 16 curated open source SC projects available on GitHub, deployed on Ethereum, and with a high market value in the Ethereum ecosystem, as per CoinMarketCap[1] Top-100 ranking. The market value refers to the total value of a particular cryptocurrency. Prioritizing high-value projects was important due to the potential advantages of understanding well-succeeded projects as a sample. Based on this curated set of high-market-value projects, this study aims to address the following research questions (RQs) from a sociotechnical standpoint:

- **[Tasks] RQ1**: How does deploying on Ethereum impact involvement of contributors, commits, and the issues in SC projects?
- **[Actors] RQ2**: What are the characteristics of the contributors in these projects?
- **[Technology] RQ3**: What types of changes are submitted to these repositories?
- **[Structure] RQ4**: What is the relationship between market value and volume metrics and software evolution (in terms of commits) in these projects?

It is noteworthy to highlight existing research endeavors [1, 17, 20, 28] that have delved into the particularities surrounding SC development. However, to the best of our knowledge, a research gap remains concerning the sociotechnical dynamics in this ecosystem, specially considering focusing on the scenario of high market value projects. By answering our RQs in the context of Sociotechnical Theory, we unveil the symbiosis between technical and social dimensions as a mutual and interdependent "co-modification" process, perceptible only when scrutinized through a holistic lens that concurrently combines the social and technical facets [10].

This study advances the field of data analysis in SE by:

- Shedding light on collaboration in SC engineering by identifying contribution patterns and the dynamics of community engagement;
- Characterizing the contributors in SC repositories, yielding valuable perspectives on team dynamics;
- Unveiling change patterns in SC project repositories, showing potential trends to guide development processes.
- Elucidating the external factors that play an important role in the success of Ethereum-based SCs by investigating the interplay between market metrics and the evolution of SCs.

[1]https://coinmarketcap.com

2 RELATED WORK

Extensive research has been conducted on different aspects of SC in the Ethereum ecosystem. However, to the best of our knowledge, no study has focused on the sociotechnical factors in this domain, specially in the context of high market value SC projects. To provide context for our research, we overview below empirical and exploratory studies focusing SCs maintenance and evolution.

Pinna *et al.* [20] studied SCs on Ethereum by focusing on aspects such as software metrics, Ethereum Virtual Machine (EVM) compiler versions, developer practices, Solidity programming language features, and blockchain environment. The research also examined Solidity's progressive attributes and SC writing methodologies. Methodologically, the focus was on understanding SC software characteristics and metrics to measure their progress and performance in the evolving Ethereum blockchain landscape. In turn, Oliva *et al.* [17] conducted an exploratory study on SCs deployed on the Ethereum blockchain. Their focus centered on three dimensions: the frequency of SC utilization, functional categorization, and code complexity. They examined all SCs on the Ethereum platform by cross-referencing data from various sources including Google BigQuery, Etherscan, DAppsNote states, and CoinMarketCap. Results showed a concentration of activity in a subset of SCs, with nearly three-quarters having undergone verification.

Ajienka *et al.* [1] analyzed the SCs from 11 Ethereum-based projects on GitHub to evaluate gas consumption during SC deployment, which measures computational effort on the Ethereum blockchain. They also assessed object-oriented metrics to correlate with gas usage. They found significant correlations between specific inheritance-based metrics and gas consumption during deployment. More recently, Chen *et al.* [7] conducted an empirical study on the maintenance of SCs after deployment. They focused on understanding the challenges developers face in corrective, adaptive, perfective, and preventive maintenance tasks.

Focusing on the motivations behind modifications made by SC developers and their impact on SC adoption and longevity, Qasse *et al.* [22] collected metadata and source code from deployed SCs to identify update patterns based on defined policies. The study identified three main reasons for contract updates: vulnerability fixes, introducing new features, and optimizing gas costs.

As we can see, different works devoted attention to various aspects within the SC context, including software maintenance. However, the exploration of sociotechnical factors, particularly within high market value SC projects, remains a research gap. Our study aims to fill this gap by investigating the interplay between technical and social dynamics. Building upon the empirical and exploratory studies outlined in this section, we contribute by focusing on the sociotechnical complexities shaping SC development of high market value projects.

3 EMPIRICAL METHODOLOGY

This research is rooted in a computational experiment founded on Mining Software Repositories (MSR) principles. Its primary objectives encompass a qualitative and quantitative exploration of four key dimensions: 1) the impact of Ethereum deployment on collaborative aspects of SC development, with specific emphasis on contributors, commits, and issues; 2) the characterization of contributors engaged in SCs' projects; 3) an assessment of categories of changes within SCs' repositories; and 4) a comprehension of the interplay between market value metrics and SCs' evolution. A data mining approach is justified by its efficacy in reaching empirical evidence from vast repository datasets [15]. This research was systematically structured into two overarching stages—Data Collection and Data Analysis—as depicted in Figure 1. All the data underpinning this study is openly available in our supporting supporting repository [8], ensuring transparency and replicability.

Stage 1 focuses on Data Collection, beginning with the manual selection of the top 100 projects on the CoinMarketCap platform, comprising the projects with the highest market value in the Ethereum ecosystem. In addition to providing this ranking, CoinMarketCap also offers access to the project's link on Etherscan[2], which functions as a block explorer, enabling access to the source code of the SCs deployed on Ethereum, as well as their unique deployment date. The SCs' project selection happened on March 18, 2022. For each project in the aforementioned ranking, we conducted a manual curation procedure based on four steps:

1) Identification of GitHub Repositories. A manual search was conducted to identify each project's repository on GitHub. For some projects, the GitHub repository URL was already available on the CoinMarketCap page. For pages where the URL was not readily available, we performed a manual search on Google and GitHub using the project name to identify the repository. Once the project's GitHub repository was identified, a triple-check was conducted: i) Does the repository provide official project data (logo, website, etc.)? ii) Does the SC use the Solidity language? iii) Is the repository not empty? This procedure was necessary to ensure that we considered the correct GitHub repository for each SC. At the end of this process, we kept 32 out of the 100 initial projects. We discarded 68 projects listed on CoinMarketCap because we could not identify their respective GitHub repositories. Indeed, it would be possible to include additional GitHub repositories if we had ignored the previously explained triple-check requirement. This a common problem in SC-related empirical studies [19]. Moreover, we opted to do not include repositories that did not host a SC in the top 100 ranking of market value. We preferred to follow a conservative methodology to ensure data consistency and empirical rigor.

2) Classification of Evolution Scenarios. This step was related to the impact of deployment in SCs development. After the identification of the repositories, our initial list of 32 projects was classified into three distinct scenarios: a) the deployment date available on Etherscan is *after* the last commit on GitHub; b) the deployment date on Etherscan falls *within* the interval on GitHub; and c) the deployment date on Etherscan is *before* the first commit on GitHub. Since one of the goals of this study is to investigate the impact of deployment in sociotechnical aspects, we need to study repositories in which we observe commits before and after deployment. Hence, after this classification, 16 projects (from the top 100) were identified as suitable for our investigation, as the deployment date on Etherscan was *within* the GitHub interval.

3) GitHub Data Collection. Following the consolidation of the sample of 16 repositories, further data collection was conducted to obtain data about commits, contributors, and issues for each repository in our sample. For this purpose, we implemented Python scripts using the GitHub REST API. Specifically, the number of commits and contributors was collected following monthly time windows before and after the repository's deployment for each project. Regarding the commits, the own commit messages (logs) were also collected. For the contributors data (the ones that contributed with at least one commit in the Git repository), we collected their public profile description, the total number of repositories they contributed to, the year of the first contribution, the number of contributions in the last month, and the affiliated organization (when existent). In the contributor's profile on GitHub, there is a section specifying which organizations the contributor is part of. About the issues, we collected the number of issues pre- and post-deployment date. Three authors manually reviewed each contributor's GitHub profile. In total, the profiles of 40 contributors were subjected to the analysis.

4) Market Data Retrieval from CoinMarketCap. Finally, we used a manual process to gather temporal market-related information for each of the 16 selected projects from CoinMarketCap: the date of the information, the market value, and volume. The date represents the day on which the market value and volume of the cryptocurrency were collected. The market value of a cryptocurrency is its total value, calculated by multiplying its current price by the total number of units in circulation. Volume represents the total value of all transactions within a period, reflecting overall interest and activity.

Stage 2 focused on Data Analysis addressing a sociotechnical arrangement to understand the subjects under investigation. During this stage, four perspectives for exploratory analyses were undertaken, aligning with the analytical framework for sociotechnical systems initially proposed by Bostrom and Heinen [3] and subsequently expanded upon by Ghaffari *et al.* [11]. Inspired by these authors, our framework, presented in Figure 2, conceptualizes the SC development as a sociotechnical phenomenon through a network of four interactive components divided into two fundamental subsystems, namely tasks and technology (as the *technical subsystem*) and structure and actors (as the *social subsystem*). This way, we can answer each of the following research questions:

[Tasks] RQ1: *How does deploying on Ethereum impact involvement of contributors, commits, and the issues in SC projects?* **Rationale**: For this analysis, a distribution of the number of commits over twelve months before and after the deployment date of the SC on Ethereum was performed. Consequently, it became possible to investigate the total quantity of commits, contributors, and issues before and after the deployment of each project. This data enables us to investigate the quantitative impact of the deployment process on the development dynamics from a collaborative task perspective, focusing on contributors, commits, and issues.

[2]https://etherscan.io

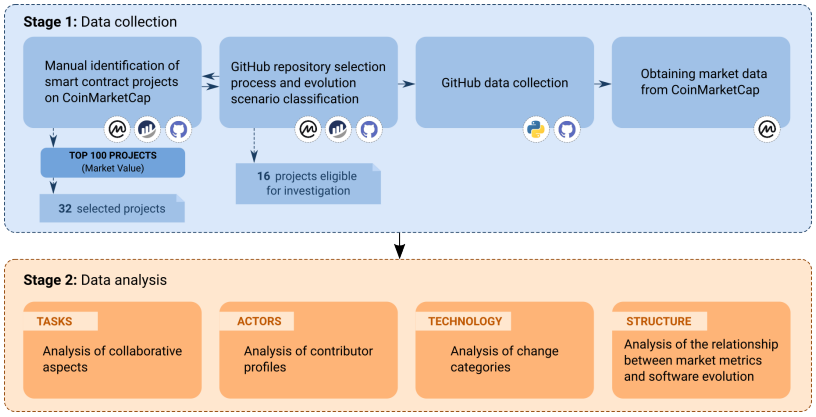

Figure 1: Methodological procedures.

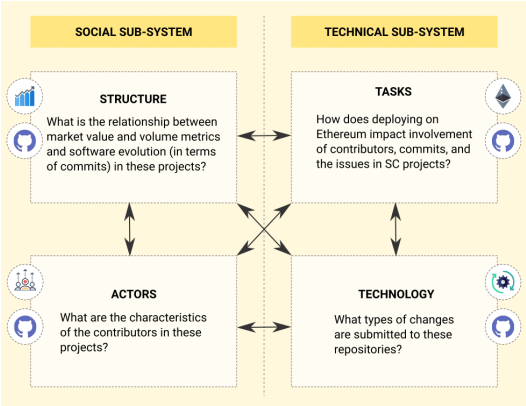

Figure 2: Adapted Sociotechnical Framework [3, 11].

[Actors] RQ2: *What are the characteristics of the contributors in these projects?* **Rationale**: From a qualitative lens, this analysis aimed to understand the profile of contributors involved in each project based on data available on GitHub. This analysis sought to examine i) the level of contributor activity through the number of repositories available in each contributor's profile; ii) recent contributions from each contributor (based on the last month of contribution); iii) the involvement of contributors in other organizations; iv) the duration contributors have been on GitHub, determined by the year of their first contribution; and v) the profile of contributors through the description available on GitHub.

[Technology] RQ3: *What types of changes are submitted to these repositories?* **Rationale**: For this question, the content of commit messages was qualitatively analyzed to identify possible patterns of changes related to the software evolution of the analyzed projects. This process involved the manual classification of commit messages following Thematic Content Analysis [9]. The first author (with three years of experience in the software industry and holding a master's degree in Computer Science) of this paper and an invited software engineer (with five years of experience and a bachelor's in Computer Science) separately analyzed the content of each of the 909 commit messages and categorized them based on the semantic context of the message, drawing inspiration from the classification defined by Chen *et al.* [7]—namely, adaptive maintenance, perfective maintenance, corrective maintenance, and preventive maintenance. A new category was suggested if a clear relationship with one of these classifications was not identified. After completing this initial classification, the author and the software engineer met in person to discuss and compare the outcomes. When they identified a discrepancy, they discussed it until reaching a consensus. Through this analysis, it became possible to identify the categories that represent the changes made during the evolution of each open source project deployed on Ethereum.

[Structure] RQ4: *What is the relationship between market value and volume metrics and software evolution (in terms of commits) in these projects?* **Rationale**: This analysis aimed to understand the relationship between market value, volume, and project commits in the context of the pre- and post-deployment periods. In other words, providing an initial understanding of the relationship between the number of commits (in terms of software evolution), volume, and market value over a specific period became possible. In this case, similar to RQ1, the time frame considered was twelve months before and twelve months after the deployment date of each project.

4 RESULTS AND ANALYSIS

4.1 [Tasks] RQ1: Impact of SC Deployment on Collaborative Aspects

RQ1's analysis was guided by the following research question: "*How does deploying on Ethereum impact involvement of contributors, commits, and the issues in SC projects?*". Figure 3 provides an overview of each SC project considered in this study, elucidating the commit distribution, total commit count, contributor count, and issue quantity. The data are derived from a twelve-month temporal span, encompassing the SC's pre- and post-deployment periods.

By analyzing the pre-deployment **commit distribution**, one can notice that 9 out of the 16 projects exhibit less than 10 commits, while the remaining seven projects demonstrate an excess of 10 commits. The projects with the most substantial commit activity in this phase include FloorDAO (330 commits), Ethereum Push Notification Service (163 commits), and Alethea Artificial

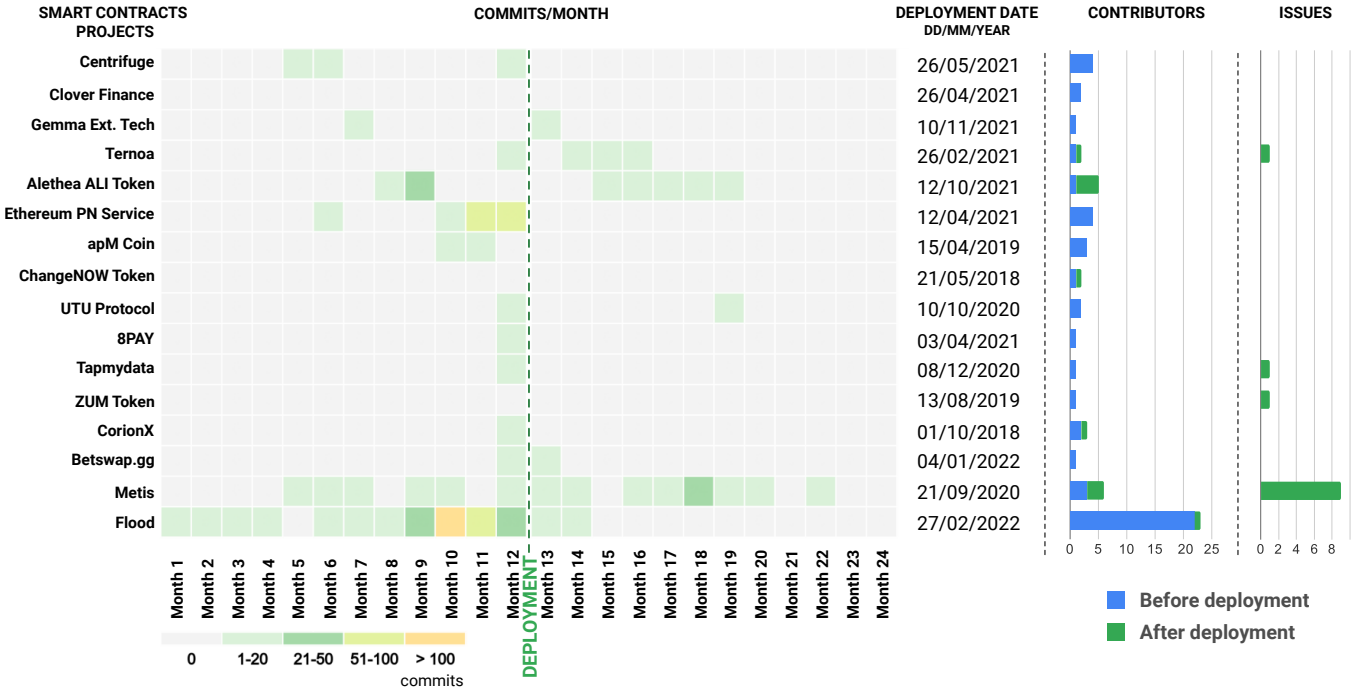

Figure 3: Distribution of commits, contributors, and issues per project (before and after deployment date).

Liquid Intelligence Token (68 commits). Upon shifting our focus to post-deployment commit statistics, we observe that 11 projects maintain fewer than ten commits, while five projects display more than ten commits. Noteworthy are the projects Metis MTS (50 commits), Alethea Artificial Liquid Intelligence Token (32 commits), and Ethereum Push Notification Service (30 commits), which report the highest post-deployment commit volumes.

Furthermore, it is interesting to note that there was continued development activity in seven projects after the deployment date. It is essential to underscore that these projects manifest a spectrum of commit intensity, ranging from lower levels of activity, as witnessed in cases like Betswap.gg (2 commits), to more consistent and sustained progress, exemplified by Alethea Artificial Liquid Intelligence Token (100 commits) and Metis (91 commits).

Drawing inference from our data, we can observe that commit activity is more pronounced in the pre-deployment phase. This assertion is substantiated when evaluating the monthly distribution of commits, where a decline in commit frequency is discerned post-deployment. For specific projects such as apM Coin, Tapmydata, and ZUM TOKEN, there is only one commit after the deployment date. Conversely, projects characterized by more pronounced pre-deployment commit activity, such as FloorDAO, Ethereum Push Notification Service, and Alethea Artificial Liquid Intelligence Token, experience a reduction in commit volume after deployment.

In our analysis of **contributor distribution** before deployment, we observed that eight repositories were characterized by only one contributor, indicating limited collaboration among different people. Additionally, eight repositories exhibited a more robust presence, with the project FloorDAO notably standing out due to its 22 contributors. Following deployment, it becomes evident that there was no substantial change in the number of contributors. New contributors were introduced in only a few projects, namely Ternoa CAPS (1), Alethea Artificial Liquid Intelligence Token (4), ChangeNOW Token (1), CorionX (1), Metis (3), and FloorDAO (1). While the overall increase in contributors post-deployment is not significant, it is worth noting that these projects continue to engage with new contributors even after the SC deployment, underscoring a certain degree of ongoing activity.

Turning our attention to **issue distribution**, none of the 16 projects under investigation reported any issues before the deployment date. After deployment, only four projects recorded the presence of issues. Specifically, Ternoa CAPS, Tapmydata, and ZUM TOKEN each reported a solitary issue, while the Metis project exhibited the highest issue count at nine. Significantly, the post-deployment phase witnessed the emergence of issues, while commit activity remained relatively high. This disparity between commit and issue activity levels unravels a peculiar contrast, wherein task creation for these repositories materialized only after deployment, despite declining commit activity following the deployment event.

Response to RQ1: Most projects exhibited high commit activity pre-deployment with decreased activity post-deployment. The presence of contributor involvement in post-deployment underscores the appeal of SC development after the initial deployment phase. These results suggest that the Ethereum deployment's impact on collaborative aspects in SC projects is multifaceted.

4.2 [Actors] RQ2: Analysis of the characterization of contributors

Guided by the question "*What are the characteristics of the contributors in these projects?*", our analysis concentrated on five dimensions: 1) the extent of contributors' engagement on GitHub; 2) the frequency of recent contributions; 3) contributors' involvement in other organizational entities; 4) the duration of contributors' tenure on GitHub; and 5) the categorization of contributors into roles.

In evaluating **contributors' engagement**, our examination centered on the aggregate number of repositories to which contributors committed, as depicted in Figure 4(a). Among the cohort of 40 profiled contributors, 18 were involved in fewer than ten repositories, 12 engaged with a range from 10 to 50 repositories, and 10 contributors participated in over 50 repositories. Notably, projects associated with contributors engaging with a broader spectrum of repositories included Alethea Artificial Liquid Intelligence Token, Ethereum Push Notification Service, and FloorDAO FLOOR. Within this context, it becomes apparent that projects characterized by a higher volume of commits, such as Alethea Artificial Liquid Intelligence Token and Ethereum Push Notification Service, also featured contributors who made a more substantial number of contributions within their respective repositories. This correlation underscores a heightened level of activity in contrast to other projects.

Regarding the **occurrence of recent contributions**, we identified the temporal proximity of these contributions by referencing the date of the last recorded contribution. As illustrated in Figure 4(b), among the 40 contributors, we found only one contributor for whom we could not determine the date of their last contribution. For another contributor, the most recent contribution dated back to 2019, while five contributors registered their last contribution in 2021. In contrast, a substantial cohort of 33 contributors (>80%) had contributions documented as recently as 2022 (the year of the data collection). The presence of contributors who exclusively contributed in 2019 and 2021 suggests a discernible pattern of reduced or discontinued activity in their current repository engagements.

We also analyzed the **contributor participation in external organizations**, checking if they are affiliated with other organizations or projects. As demonstrated in Figure 4(c), among the cohort of 40 contributors, 31 had an exclusive focus on the projects under examination, not appearing to be concurrently involved in any other organizations. In contrast, for nine contributors, we found engagement with external organizations. This subset of nine contributors featured diverse levels of participation: five were affiliated with a single external organization, one held ties with two organizations, one contributor held memberships with four organizations, one contributor demonstrated affiliations with three organizations, and, lastly, one contributor maintained connections with eleven (!!) organizations.

In the context of **contributors' GitHub registration duration**, we investigated how long each contributor was active on the GitHub platform, as exemplified in Figure 4(d). To this end, we verified the year when each contributor made their inaugural contribution on GitHub. In our sample, 14 of them had a brief tenure of up to three years, which distinctly classifies them as newcomers to SC open source development. On the other hand, 13 contributors showed an extended period of involvement, ranging from four to eight years—or a moderate level of experience on GitHub. Still, 12 individuals had their accounts on GitHub for more than eight years—and were classified as highly experienced contributors. One contributor had deactivated the public profile.

Finally, to understand the **contributors' profiles**, we examined the descriptions of the contributors' public profiles on GitHub. These descriptions were defined by the contributors themselves in their profiles. This analysis was limited to 13 contributors because, out of the 40 individuals, they were the only ones to define a role in their public GitHub profiles, as listed in Figure 4(e). As can be seen, the profile of contributors is mainly related to technology, with considerable diversity in function, ranging from developers to solution architects and CTOs. Another point to highlight is that few members formalize their organizational involvement in their profiles. One person identified themselves as a member of the contributing organization, while another contributor described themselves as a project co-founder. It is also interesting to note some particularities in expertise, such as one person identifying as an expert in quantitative finance, blockchain, and artificial intelligence and another contributor describing themselves as an aspiring "cryptodev".

Response to RQ2: Our analysis indicated varying levels of engagement. Considering all contributors, 25% were highly active (more than 50 repositories) while 40% demonstrated limited involvement (less than 10 repositories). Most contributors have a long-standing presence on GitHub, indicated by more than three years of contributions. However, 67% of contributors' profiles lacked detailed descriptions.

4.3 [Technology] RQ3: Analysis of change types in software maintenance

To answer RQ3—"*What types of changes are submitted to these repositories?*"—we qualitatively analyzed the commit messages in the repositories. This analysis is summarized in Figure 5, presenting the change categories and percentage of occurrence, and illustrative examples of commit messages for each category.

Out of 907 commit messages, we identified nine distinct change categories using a previous taxonomy [6] as a seed. **Adaptive Maintenance** emerged as the most prevalent category, encompassing 27.06% of the commit messages. This classification denotes instances where environmental shifts required project adjustments to accommodate new tool versions and Solidity upgrades. Examples of this category include commit messages such as "upgrade to solc 0.6" and "upgrade hardhat-ethers version". Following closely, the **Evolutionary Maintenance** category accounted for 23.21% of the commit messages. This category characterizes commits geared towards elevating software quality and integrating novel functionalities, as evidenced by snippets such as "clean up" and "change variable name" present within the commit messages.

Corrective Maintenance (14.63% of the commit messages) predominantly refers to content addressing bug fixes. Commit messages within this category often feature terms such as "fix," "bug", and direct references to coding anomalies, exemplified by messages such as "fix sohm init issue" and "fix transferFrom". In contrast, the

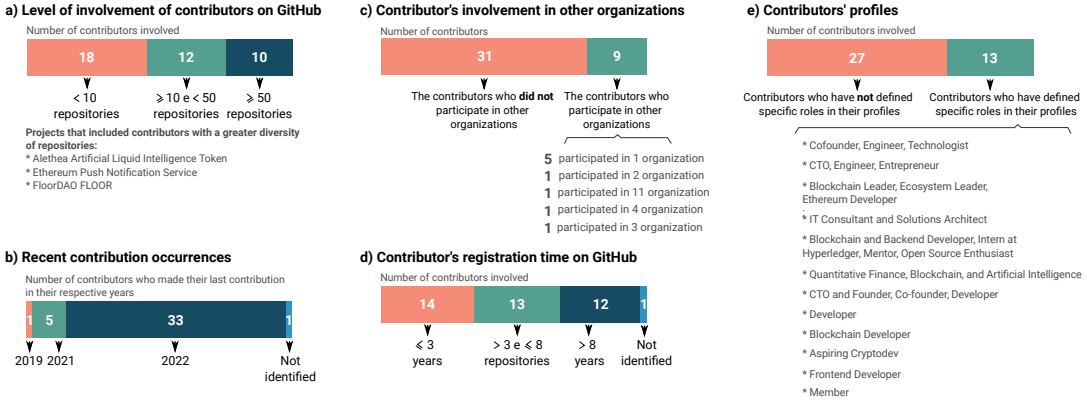

Figure 4: Contributor characterization overview.

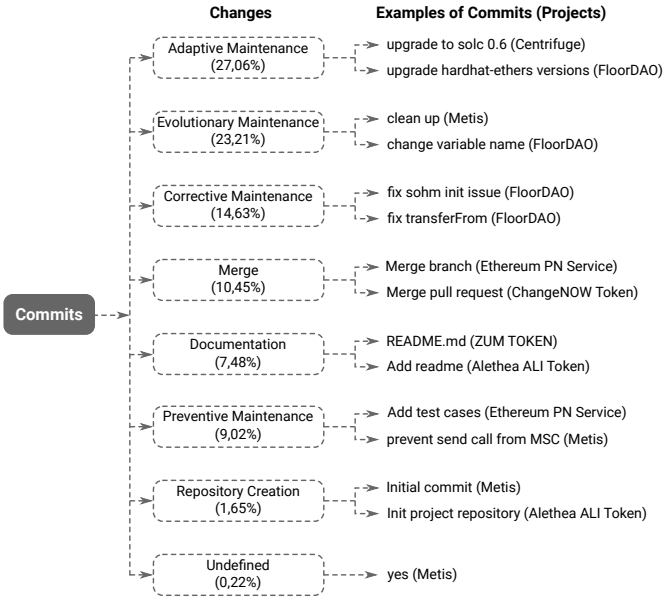

Figure 5: Categorization of commits messages.

Merge category(10.45%) comprises those messages that incorporate terminology such as "merge" and "pull request". These terms represent the integration of new features into the project repository.

The **Documentation** category (7.48%) shows the active involvement of contributors in generating essential project documentation to facilitate comprehension of the codebase and the project itself. Within this category, the commit messages commonly referenced modifying the project's README file, reflecting the endeavor to maintain and enhance project documentation for the benefit of users and developers.

The **Preventive Maintenance** category (9.02%) represents instances where actions were taken to enhance project structures in anticipation of future maintenance requirements or to forestall potential issues. Examples of commit messages within this category include "Just to avoid repetition" and "prevent send call from MSC". Notably, a prevalent practice in this category was the execution of tests for new features. Commit messages in this context frequently employed terminology such as "add tests" and "Test Cases" to articulate the deliberate testing of these features, underscoring a proactive approach to ensuring project robustness.

Finally, the **Repository Creation** (1.65% of the commit messages) encompasses expressions corresponding to the project's initial commit, such as "Initial commit" and "init project repository", indicating the creation of a new repository.

Finally, 0.22% of the analyzed commits were classified as **Undefined** category given the absence of an understandable semantic context to define an appropriate classification, such as the following commit message: "yes."

Response to RQ3: Our results identified nine distinct change categories, with "Adaptive Maintenance" being the most prevalent. Other categories, including "Evolutionary Maintenance", "Corrective Maintenance", and "Preventive Maintenance" highlighted different aspects of maintenance activities.

4.4 [Structure] RQ4: Analysis of the relationship between market metrics and SCs' evolution

In this section, we aim to answer our fourth RQ: "*What is the relationship between market value and volume metrics and software evolution (in terms of commits) in these projects?*". Figure 6 depicts an overview of the timeline of market values, trading volumes, and commit counts of seven distinct projects. These projects were chosen due to their provision of data about all three specified metrics. We did not include Betswap.gg, Metis, and FloorDAO as they did not have market value data available on CoinMarketCap. Centrifuge, apM Coin, 8PAY, Tapmydata, ZUM TOKEN, and Corion did not have commit data available during the period when market value data was accessible and were also excluded from this analysis.

For Gemma Extending Tech, the pre-deployment phase, marked by a dashed line, reveals specific commit activities. On May 12 and May 13, 2021, there were 1 and 7 commits, respectively, while another single commit occurred on June 17, 2021. The commit activity persisted post-deployment, with 2 commits documented on December 12. During the period spanning from May 13 to June 17,

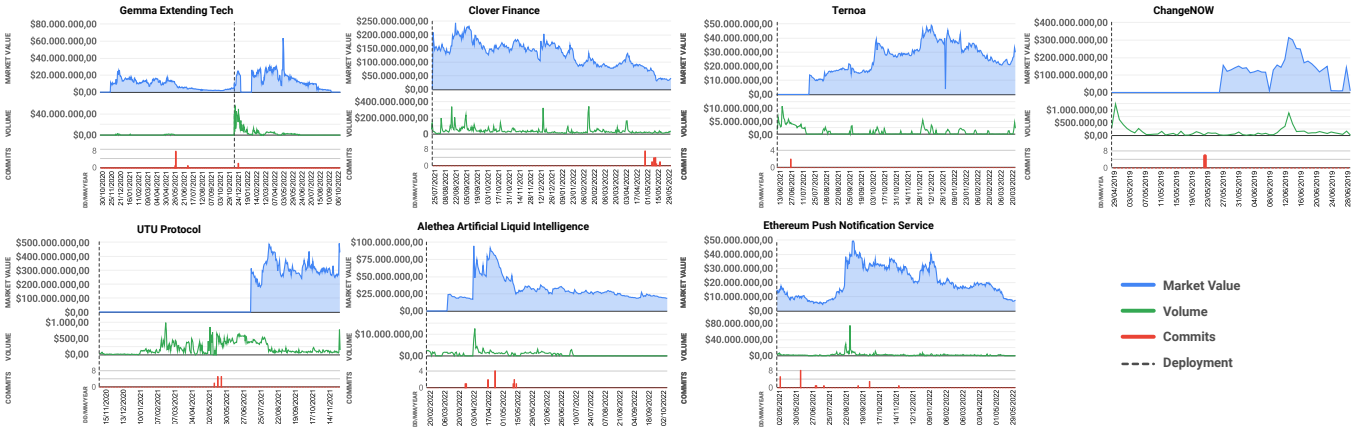

Figure 6: Results of market value, volume, and total commit count per project

a downturn in market value occurred, concomitant with an absence of commits. However, in December (post-deployment) the market value rose, regressing to zero in mid-January 2022, accompanied by two commits during this interval. In regards to trading volume, an increase was noted post-deployment, but from January 2022 onwards, a consistent decrement was observed.

For Clover Finance, no market value or trading volume data was available before the deployment date. Consequently, our analysis solely relies on post-deployment data. This drawback of limited data availability before deployment was also observed across the other projects under examination, except Gemma Extending Tech, as discussed earlier. Within the scope of available data, we identified commit activities between April and May 2022. During this period, a decreasing value trend was evident in both market value and trading volume. This consistent trend aligns with the broader patterns observed across the projects.

Regarding Ternoa, our analysis reveals the occurrence of two commits exclusively after the deployment date, specifically on June 17, 2021, coinciding with a zero market value. The market value data only began to manifest from July 2021 onwards. In contrast, the trading volume, especially in July, initially exhibited a higher magnitude than the concurrent market value. However, a noteworthy trend emerged as the trading volume gradually descended following this initial surge, with no substantial subsequent increases observed. Similarly to the Ternoa, ChangeNOW manifested six commits after its deployment on May 23, 2019. The market value remained at zero during this period, with a departure from this baseline occurring only on May 28, 2019. After this period, market value exhibited minimal fluctuations. In contrast, post-deployment trading volume commenced at a higher level compared to the concurrent market value, with a decrease observed from May 2019 onwards, punctuated by an isolated increase in June 2019.

The UTU Protocol witnessed a parallel post-deployment scenario with 12 commits in May 2021 while the market value remained at zero. A deviation from this zero point materialized only in July. Following this transition, market value exhibited relative stability without significant oscillations. The trading volume for this project presented a distinct trajectory, experiencing an increase commencing in January 2021, which was then succeeded by a declining trend from July 2021 onwards.

The Alethea Artificial Liquid Intelligence Token exhibited commit activity on the following dates: March 23 and 24, 2022; April 14 and 21, 2022; and May 9, 10, and 12, 2022. It is discernible that during March 2022, there was an upsurge in market value, albeit followed by a diminishing trajectory commencing in May. After this point, the market value remained relatively stable, without any fluctuations. A parallel trend is notable in the volume metric, mirroring the patterns observed in the market value.

The Ethereum Push Notification Service featured commits occurring after the deployment date within the following timeframes: May 5, 2021; June 8, 2021; July 3, 5, and 17, 2021; September 12, 2021; October 1, 2021; and November 19, 2021. A decrease in market value is evident from May to July 2021. However, an upward trend in market value becomes apparent in August, which persists until the initial week of September. Nevertheless, November witnessed a pronounced downturn in market value. Regarding trading volume, notable growth is observed solely between August and September 2021, while in other periods, it remains close to zero.

Extracting a relationship between market metrics such as market value and volume and software evolution characterized by commits remains challenging for most of the projects under investigation.

> **Response to RQ4:** We could not establish a clear and direct relationship between market metrics and software evolution is not straightforward.

5 DISCUSSION

Drawing upon insights from the sociotechnical paradigm delineated by Bostrom and Heinen [3] and Ghaffari *et al.* [11], our analysis approached four core socio-technical components: **Tasks** (focusing on collaborative aspects), **Actors** (centering on contributor characterization), **Technology** (concentrating on categorizations of changes), and **Structure** (emphasizing the interplay between market metrics and software evolution). Below, we examine these key interactions within the sociotechnical system under investigation.

The **Actor-Task** nexus explored the relationship between contributors (actors) and the tasks they engage in within SC projects. Our findings in RQ2 highlighted the diverse roles and levels of engagement of contributors. The Actor-Task interplay suggests that contributors with varying degrees of experience and expertise are participating in different aspects of SC project development. While some contributors exhibit high levels of engagement across multiple repositories, others engage in more specialized tasks. This diversity of actors and tasks reflects the inherent complexity of SC development and demonstrates the importance of recognizing and accommodating this diversity in SC development [6, 17].

The **Actor-Technology** nexus reflected the relationship between contributors and the technological aspects of SC projects. The findings from RQ3 are particularly relevant here. The classification of change categories in commit messages sheds light on how contributors interact with the technology when implementing modifications. It suggests that contributors engage in activities that align with the specific technological requirements and challenges posed by SC development. Understanding this nexus is important for SC maintainers to effectively match contributors with tasks that suit their technological expertise [26].

The **Task-Technology** nexus explored the relationship between the tasks involved in SC development and the underlying technological infrastructure. RQ3 addressed this relationship by categorizing changes made in SC repositories. The identified categories, such as Adaptive Maintenance and Corrective Maintenance, highlight how the technological context influences tasks. The need to adapt to new Solidity versions (Adaptive Maintenance) or address coding anomalies (Corrective Maintenance) emphasizes the interdependence between tasks and technology. This nexus reinforces the idea that SC development tasks are inherently tied to the specific technical requirements of the Ethereum platform [1, 20].

The **Structure-Technology** nexus reflected the interplay between the organizational structure of projects (*e.g.*, market metrics) and the underlying technological aspects (*e.g.*, software evolution). This issue has implications for project stakeholders and investors. Trockman *et al.* [29], for example, elucidated that the pricing dynamics of cryptocurrencies exhibit a nuanced interplay with their allure. The authors hypothesize that vibrant software development exerts a discernible influence on cryptocurrency pricing. Hence, understanding the dynamics between market metrics and software evolution could be important. However, this study evidenced the complexity and variability of these relationships, urging caution.

In summary, embracing Sociotechnical Theory and Data Analysis provided us a comprehensive lens for understanding the multifaceted nature of the sociotechnical dynamics within open-source SC projects with high market value on the Ethereum. The interplay between actors, tasks, technology, and project structure is strategic in evaluating the sociotechnical dynamics of SC development.

6 THREATS TO VALIDITY

Following Wohlin *et al.* [30], we address below the threats to the validity of this study and outline our strategies for mitigating them.

Conclusion Validity. While the absence of sophisticated statistical tests and formal hypotheses could be considered a potential threat, we have mitigated this concern by rigorously validating our findings. We contextualized our results within the existing body of knowledge and adhered to a robust research protocol. Our data analysis procedures followed an established analytical framework based on Sociotechnical Theory, a widely recognized concept in Software Engineering literature. This theoretical foundation guided our interpretation of the results and strengthened the validity of our conclusions and their implications.

Construct Validity. Our research questions were grounded in an analytical framework [3, 11] derived from Sociotechnical Theory, ensuring their relevance and coherence with our research objectives. Even though the methodological procedures and decisions were thoroughly deliberated among the research team and followed empirical standards in Software Engineering [23], there are issues that deserve attention. In particular, we acknowledge we have not addressed particularities related to forked projects, which may potentially influence the analysis.

Internal Threats. We rigorously collected and processed data to minimize internal threats, ensuring data quality and reliability. Our approach includes validation checks, thorough data cleaning, secure storage, and bias mitigation. Additionally, all study data is openly accessible via our repository [8], promoting transparency.

External Threats. Our findings are specific to Ethereum-based SC and may not be universally extrapolated to other blockchain platforms. Our focus on SC projects with high market value, which are also available on GitHub, introduces a potential bias. Nevertheless, GitHub is the largest software development platform. The study also faced challenges in identifying the respective GitHub repositories of the examined SC high-market-value projects. To enhance accuracy, a triple-check approach was employed. Additionally, the relatively small number of projects investigated may raise concerns about the generalizability. While it would be feasible to include more projects, doing so would extend beyond the top 100 projects with the highest market value and potentially diluting the unique characteristics of top-ranked projects. Moreover, this research does not aspire to achieve external validity but focuses on constructing analytical generalizations [25]. Adopting a mixed-method approach, anchored in an analytical framework, enabled us to construct a meaningful comprehension of the phenomenon.

7 CONCLUSION

The influence of blockchain technology has expanded beyond cryptocurrencies, finding applications in various industries. This widespread adoption can be attributed to platforms like Ethereum and the emergence of Smart Contracts (SCs), which are instrumental in developing decentralized applications (dApps). However, it is necessary to acknowledge that SC development presents notable challenges influenced by sociotechnical dynamics. These dynamics stem from the interaction among technological elements, task complexities, organizational structures, and actors. Consequently, these elements not only shape SC features, such as their self-verifying and immutable nature but are also shaped by them. In this context, a sociotechnical ecosystem has evolved for SC development, with platforms like GitHub serving as key hubs for collaborative and open-source SC repositories. These repositories are open and collaborative, providing a wealth of social and technical data that can be analyzed through data analysis procedures.

This research employed exploratory data analysis with the aim of investigating sociotechnical aspects within open-source repositories hosting Ethereum's high-market-value SCs. Guided by a sociotechnical framework, we delineated findings across four pivotal dimensions. Our initial focus was on examining the impact of Ethereum deployment on collaborative dynamics such as contributors, commits, and issue tracking. This investigation revealed significant differences in pre- and post-deployment dynamics, unveiling patterns of activity among contributors and their commit behaviors. Furthermore, our analysis of contributor profiles illuminated the diverse roles within the ecosystem, from developers to leaders and consultants, underscoring the importance of each role. We also categorized software evolution based on commit messages, clarifying developmental changes ranging from adaptive maintenance to creating new repositories. Additionally, we delved into the interplay between market metrics (including market value and trading volume) and software evolution, uncovering the challenging and intricate relationship dynamics between these factors.

This research contributes to both academia and practitioners in the context of data analysis in blockchain-oriented software engineering. For academia, we have expanded the understanding of sociotechnical dynamics within open-source repositories hosting high-market-value SC deployed on the Ethereum. Our findings encompassed the impact of Ethereum deployment on collaborative aspects, the characterization of contributors, the categorization of software evolution based on commit messages, and the relationship between market metrics and software development. For practitioners, this study offers a deeper comprehension of the sociotechnical landscape surrounding SC projects. The understanding gained can inform development strategies, guide collaboration practices, and enhance product management approaches within the industry.

Future research endeavors can broaden our scope by including SC projects deployed on other blockchain platforms, such as Polygon, Klaytn, and Celo. This extension holds the potential to offer a diverse understanding of additional sociotechnical factors that influence the landscape of SC development. Furthermore, another future work could involve comparing the specific characteristics identified in SC projects to those in other domains.

ACKNOWLEDGEMENTS

This work received funding from CNPq - Brazil, Universal grant 404406/2023-8.

REFERENCES

[1] Nemitari Ajienka, Peter Vangorp, and Andrea Capiluppi. 2020. An empirical analysis of source code metrics and smart contract resource consumption. *Journal of Software: Evolution and Process* 32, 10 (2020), e2267.

[2] Moritz Beller and Joseph Hejderup. 2019. Blockchain-based software engineering. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*. IEEE, 53–56.

[3] Robert P Bostrom and J Stephen Heinen. 1977. MIS problems and failures: A socio-technical perspective. Part I: The causes. *MIS quarterly* (1977), 17–32.

[4] Wei Cai, Zehua Wang, Jason B Ernst, Zhen Hong, Chen Feng, and Victor CM Leung. 2018. Decentralized applications: The blockchain-empowered software system. *IEEE Access* 6 (2018), 53019–53033.

[5] Partha Chakraborty, Rifat Shahriyar, Anindya Iqbal, and Amiangshu Bosu. 2018. Understanding the software development practices of blockchain projects: a survey. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*. ACM, 28.

[6] Jiachi Chen, Xin Xia, David Lo, John Grundy, and Xiaohu Yang. 2020. Maintaining smart contracts on Ethereum: Issues, techniques, and future challenges. *arXiv preprint arXiv:2007.00286* (2020).

[7] Jiachi Chen, Xin Xia, David Lo, John Grundy, and Xiaohu Yang. 2021. Maintenance-related concerns for post-deployed Ethereum smart contract development: issues, techniques, and future challenges. *Empirical Software Engineering* 26, 6 (2021), 1–44.

[8] Saori Costa, Matheus Paixao, Igor Steinmacher, Pamella Soares, Allysson Allex Araújo, and Jerffeson Souza. 2024. Replication Package for the paper: "Sociotechnical Dynamics in Open Source Smart Contract Repositories: An Exploratory Data Analysis of Curated High Market Value Projects". https://zenodo.org/records/11204947

[9] Daniela S Cruzes and Tore Dyba. 2011. Recommended steps for thematic synthesis in software engineering. In *2011 international symposium on empirical software engineering and measurement*. IEEE, 275–284.

[10] Henrique Luiz Cukierman, Cássio Teixeira, and Rafael Prikladnicki. 2007. Um olhar sociotécnico sobre a engenharia de software. *Revista de Informática Teórica e Aplicada* 14, 2 (2007), 199–219.

[11] Kimia Ghaffari, Mohammad Lagzian, Mostafa Kazemi, and Gholamreza Malekzadeh. 2019. A socio-technical analysis of internet of things development: an interplay of technologies, tasks, structures and actors. *foresight* (2019).

[12] Global Key Players. 2021. DApps Market Report 2020. https://www.einnews.com/prnews/554513183/dapps-market-report-2020-by-global-key-players-types-applications-countries-size-forecast-to-2027.

[13] Ahmed E Hassan. 2008. The road ahead for mining software repositories. In *2008 Frontiers of Software Maintenance*. IEEE, 48–57.

[14] Péter Hegedűs. 2019. Towards analyzing the complexity landscape of solidity based ethereum smart contracts. *Technologies* 7, 1 (2019), 6.

[15] Huzefa Kagdi, Michael L Collard, and Jonathan I Maletic. 2007. A survey and taxonomy of approaches for mining software repositories in the context of software evolution. *Journal of software maintenance and evolution: Research and practice* 19, 2 (2007), 77–131.

[16] Tian Min and Wei Cai. 2022. Portrait of decentralized application users: an overview based on large-scale Ethereum data. *CCF Transactions on Pervasive Computing and Interaction* 4, 2 (2022), 124–141.

[17] Gustavo A Oliva, Ahmed E Hassan, and Zhen Ming Jack Jiang. 2020. An exploratory study of smart contracts in the Ethereum blockchain platform. *Empirical Software Engineering* (2020), 1–41.

[18] Luisa Palechor and Cor-Paul Bezemer. 2022. How are Solidity smart contracts tested in open source projects? An exploratory study. In *Proceedings of the 3rd ACM/IEEE International Conference on Automation of Software Test*. 165–169.

[19] Giuseppe Antonio Pierro, Roberto Tonelli, and Michele Marchesi. 2020. An organized repository of ethereum smart contracts' source codes and metrics. *Future internet* 12, 11 (2020), 197.

[20] Andrea Pinna, Simona Ibba, Gavina Baralla, Roberto Tonelli, and Michele Marchesi. 2019. A massive analysis of ethereum smart contracts empirical study and code metrics. *IEEE Access* 7 (2019), 78194–78213.

[21] Rafael Prikladnicki, Yvonne Dittrich, Helen Sharp, Cleidson De Souza, Marcelo Cataldo, and Rashina Hoda. 2013. Cooperative and human aspects of software engineering: Chase 2013. *ACM SIGSOFT Software Engineering Notes* 38, 5 (2013), 34–37.

[22] Ilham Qasse, Mohammad Hamdaqa, and Björn Þór Jónsson. 2023. Smart Contract Upgradeability on the Ethereum Blockchain Platform: An Exploratory Study. *arXiv preprint arXiv:2304.06568* (2023).

[23] Paul Ralph, Rashina Hoda, and Christoph Treude. 2020. ACM SIGSOFT empirical standards. (2020).

[24] P Sajana, M Sindhu, and M Sethumadhavan. 2018. On blockchain applications: hyperledger fabric and ethereum. *International Journal of Pure and Applied Mathematics* 118, 18 (2018), 2965–2970.

[25] Robert E Stake. 2005. *Qualitative case studies*. Sage Publications.

[26] Igor Steinmacher, Tayana Uchôa Conte, and Marco Aurélio Gerosa. 2015. Understanding and supporting the choice of an appropriate task to start with in open source software communities. In *2015 48th Hawaii International Conference on System Sciences*. IEEE, 5299–5308.

[27] Chandrasekar Subramaniam, Ravi Sen, and Matthew L Nelson. 2009. Determinants of open source software project success: A longitudinal study. *Decision Support Systems* 46, 2 (2009), 576–585.

[28] Sergei Tikhomirov, Ekaterina Voskresenskaya, Ivan Ivanitskiy, Ramil Takhaviev, Evgeny Marchenko, and Yaroslav Alexandrov. 2018. Smartcheck: Static analysis of ethereum smart contracts. In *Proceedings of the 1st International Workshop on Emerging Trends in Software Engineering for Blockchain*. 9–16.

[29] Asher Trockman, Rijnard van Tonder, and Bogdan Vasilescu. 2019. Striking gold in software repositories? an econometric study of cryptocurrencies on github. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE, 181–185.

[30] Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in software engineering*. Springer Science & Business Media.

[31] Stephan Zumkeller. 2018. Using Smart Contracts For Digital Services: A Feasibility Study Based On Service Level Agreements.