‘‘I Quit!’’: Understanding Practitioner Abrupt Disengagement through Socio-Technical Factors in Open Source Software

Wouter Mulder[1], Massimiliano Di Penta[2], Giammaria Giordano[3], Rick Kazman[4], Fabio Palomba[5], Valeria Pontillo[6], and Damian Andrew Tamburri[2]

[1] JADS, 's-Hertogenbosch, Netherland,
[2] University of Sannio, Benevento, Italy,
[3] Pegaso University, Naples, Italy,
[4] University of Hawaii, USA,
[5] Software Engineering (SeSa) Lab, Fisciano, Italy,
[6] Gran Sasso Science Institute (GSSI), L'Aquila, Italy
Email: dipenta@unisannio.it, giammaria.giordano@unipegaso.it, kazman@hawaii.edu,fpalomba@unisa.it, valeria.pontillo@gssi.it, datamburri@unisannio.it

Abstract Open-source software (OSS) is central to modern digital infrastructure, yet research has mainly focused on technical and productivity aspects, overlooking contributors' *social* and *psychological well-being*. One underexplored threat to community sustainability is *abrupt disengagement*, *i.e.,* a contributor's negative sentiment followed by sudden withdrawal. This paper presents the first empirical investigation of its socio-technical roots. We hypothesize that *community smells* increase its likelihood, while *socio-technical proximity* mitigates it. We analyze six *Apache Software Foundation* projects, identifying disengagement events through inactivity patterns and using Kaiaulu and Oslom to derive relevant metrics. Results show that abrupt disengagement is a measurable outcome of socio-technical tensions: community smells act as risk factors, whereas socio-technical proximity plays a protective role, offering insights for more sustainable OSS ecosystems.

Keywords: Open-Source Software, Abrupt Disengagement, Socio-Technical Factors.

1 Introduction

Open-source software (OSS) is a cornerstone of modern digital infrastructure, powering systems such as operating systems, cryptographic libraries, and machine learning frameworks while enabling continuous innovation [40]. Its sustainability depends on a combination of *organizational*, *technical*, and *social* factors, including effective onboarding, maintainable infrastructures, and contributors' well-being [11,14,32,42,46,50].

While prior research has extensively addressed technical and organizational aspects of OSS [2,13,21,22,37,33,38,51], the human dimension remains less explored. Existing studies examine burnout, turnover intention, and motivation [39,16,20,46], yet limited attention has been paid to *abrupt disengagement*, *i.e.,* the sudden withdrawal of contributors following negative experiences [4]. OSS projects often rely on a relatively small group of core contributors, and prior research has shown that the departure of key developers can threaten project sustainability and increase project vulnerability. Although not all contributor departures have the same impact on a project, understanding the factors associated with abrupt disengagement may provide insights into socio-technical conditions that undermine contributor retention and community sustainability. Yet, this phenomenon remains largely anecdotal and underexplored, warranting systematic empirical investigation.

In this paper, we present the first evidence-based study of abrupt disengagement in OSS. We analyze six projects from the *Apache Software Foundation*, operationalizing disengagement as a contributor's final activity followed by prolonged inactivity (e.g., at least three months) [28]. This definition enables large-scale analysis of behavioral traces across platforms such as GitHub, JIRA, and Apache Pony Mail.

We investigate the role of two key *socio-technical factors*, reported in Table 1: **community smells** and **socio-technical proximity**. The former are organizational and communication anti-patterns that signal misalignment between social and technical structures [44], potentially increasing frustration and disengagement risk. The latter [7] captures the closeness of collaboration among developers, which is expected to mitigate such risks. Together, these constructs reflect the extent to which coordination and communication structures support or hinder contributors.

To test our hypotheses, we employ statistical models with abrupt disengagement as the dependent variable and metrics derived from Kaiaulu [36] and Oslom [30], alongside project- and contributor-level controls. Results show that community smells are associated with increased disengagement risk, both in the short and long term, while higher socio-technical proximity reduces it.

In summary, we provide: (i) the first empirical investigation of abrupt disengagement in OSS, highlighting its socio-technical determinants; and (ii) a publicly available replication package with data and supplementary materials [34].

2 Background and Related Work

Our paper builds on three pillars: (i) abrupt disengagement and social factors in OSS participation, (ii) community smells, and (iii) socio-technical proximity. We outline their theoretical background and related work, highlighting their role in understanding the socio-technical dynamics of developer disengagement.

Abrupt Disengagement and Social Factors in OSS. Abrupt disengagement refers to "*the departure from an open-source project following sustained*

friction with tasks, collaborators, or leaders, resulting in enduring anger".[7] Despite its impact [3], it remains underexplored in software engineering [19]. It can lead to sudden loss of expertise and risks to project continuity [26]. Existing evidence is largely anecdotal, leaving its underlying causes poorly understood; this paper addresses this gap through an analysis of its socio-technical determinants.

Table 1: Socio-technical Factors Considered in the Study.

Factor	Description	Potential Influence on Abrupt Disengagement
Lone Wolf	A single developer works in isolation, making decisions without collaboration or knowledge sharing.	May feel overburdened and unsupported, which can increase frustration and lead to withdrawal.
Organizational Silo	Groups within the community remain isolated, with limited communication and collaboration across boundaries.	Reduces mutual visibility and cohesion, fostering disengagement and dissatisfaction.
Radio Silence	Lack of communication or feedback, such as ignored issues, unanswered pull requests, or inactive discussions.	Contributes to feelings of neglect and devaluation, which can trigger abrupt departure.
Black Cloud	A negative atmosphere marked by conflicts, hostility, or toxic behavior among community members.	Generates emotional distress and burnout, strongly increasing the risk of abrupt disengagement.
Communication Network	The structure of message exchanges and interactions among contributors.	When dense and reciprocal, it can serve as a protective factor by fostering trust and belonging; when sparse or fragmented, it may heighten isolation and frustration.
Collaboration Network	The pattern of co-working and code-sharing relationships within the project.	When cohesive and balanced, it can strengthen engagement and mutual support; when weak or fragmented, it may reduce awareness and encourage disengagement over time.

Abrupt disengagement can be framed within *Counterproductive Work Behaviors*, i.e., actions harmful to organizations arising from frustration or anger [18]. Prior work identifies three predictors: (i) *injustice* (perceived unfairness) [27], (ii) *identity* (reinforced responses within like-minded groups) [12], and (iii) *instrumentality* (perceived effectiveness of such actions) [49]. These align with our socio-technical perspective: injustice relates to community smells, identity to subgroup dynamics, and instrumentality to socio-technical proximity, suggesting disengagement as a manifestation of underlying socio-technical tensions.

Disengagement can be detected through sentiment in contributors' final messages [28]. Prior approaches include lexicon-based (e.g., SentiStrength-SE [25]) and machine learning models (e.g., Senti4SD [10]), while transformer-based meth-

[7] `https://www.hrmorning.com/articles/rage-quitting-truths`

ods better capture semantic context. We adopt the BERT-CP model [53], given its domain-specific training and strong performance.

Prior studies highlight social factors shaping OSS participation, including motivation [48], belonging [47], trust [15], and conflict management [17], as well as burnout [39] and turnover intention [16]. These emphasize the role of community cohesion and well-being. Building on this, we examine abrupt disengagement as a visible outcome of social and emotional breakdowns in OSS projects.

Community Smells. Community smells are socio-technical anti-patterns that expose communication gaps, coordination issues, and organizational friction within software teams. While about thirty types have been identified [8], research consistently highlights four recurrent and impactful categories [35,43]: (i) *Organizational Silo:* Developers work on related code without communicating, resulting in redundant efforts; (ii) *Radio-silence:* Only a limited number of developers mediate communication between sub-teams, potentially slowing decision-making; (iii) *Lone Wolf:* An individual works in isolation, complicating coordination and integration; and (iv) *Black-cloud:* Persistent mediation between sub-teams fosters mistrust and communication breakdowns.

These categories are widely recognized as harmful to collaboration and cohesion, being linked to coordination inefficiencies and socio-technical misalignment [35]. Their definitions align with the socio-emotional dynamics of abrupt disengagement (Table 1), offering a lens to examine how organizational and communication breakdowns contribute to developer withdrawal. Research on community smells has also led to the development of automated detection tools. Examples include CodeFace [1], CodeFace4Smells [45], and Kaiaulu [36]. Among these, our study leverages Kaiaulu, given its empirical validation and support for reproducible, large-scale analysis. Detection relies on comparing collaboration and communication networks extracted from project activities. For example, a *Lone Wolf* corresponds to a contributor exhibiting limited collaboration and interaction with other community members, whereas *Organizational Silos* and *Radio Silence* capture misalignments between collaboration and communication.

Socio-Technical Proximity. Socio-technical proximity captures how closely developers are positioned within the intertwined social and technical structures of a project [7]. It reflects the alignment between communication and collaboration patterns and serves as a proxy for coordination needs: higher proximity implies lower coordination overhead.

The construct combines two dimensions: the *communication network*, representing interactions (e.g., mailing lists, issue trackers), and the *collaboration network*, capturing relationships among developers who co-modify code. These are integrated into a unified developer graph [7], where high proximity indicates alignment and low proximity signals potential coordination gaps.

Tools such as Kaiaulu [36] enable analysis by integrating repository and communication data. Prior studies show that socio-technical alignment shapes coordination needs [6], while dense interaction networks improve cohesion and productivity [5]; conversely, low proximity can lead to inefficiencies and quality issues [24].

Overall, socio-technical proximity is a key indicator of alignment between communication and development activities, making it suitable for analyzing factors influencing abrupt disengagement (Table 1).

Novelty and Positioning of our Study.

This study offers the first quantitative study of abrupt disengagement in software engineering, linking social factors, community smells, and socio-technical proximity. Unlike prior work, it models disengagement as a measurable socio-technical outcome and identifies risk and protective factors for OSS sustainability.

3 Research Method

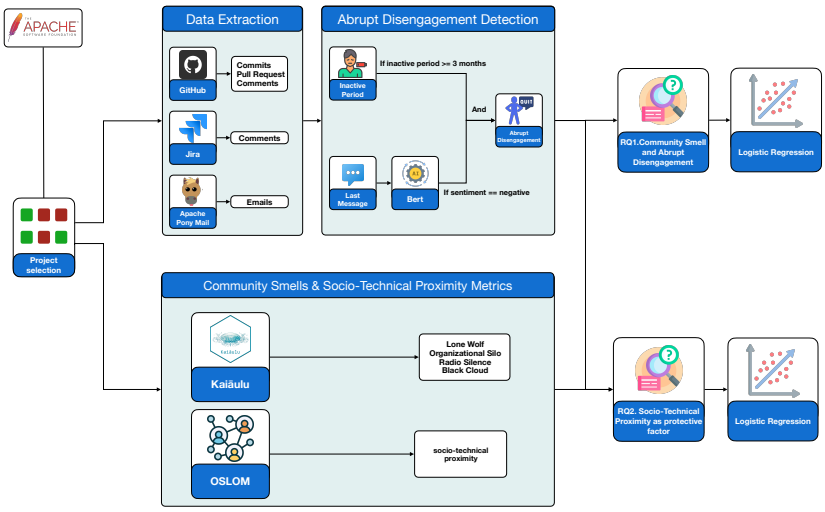

Figure 1: Research Method Overview.

Driven by the Goal Question Metric paradigm [9], our *goal* is to investigate whether community smells and socio-technical proximity can predict abrupt disengagement. The *purpose* is to assess their ability to anticipate this phenomenon. The *quality focus* is sustainability, considering well-being as a key dimension of software quality. The *perspective* includes both practitioners—who may benefit from early warning signals—and researchers, aiming to better understand these relationships. Based on this, we define two research questions:

🔍 RQ$_1$. Community smells and abrupt disengagement.

Can abrupt disengagement be predicted using community smells?

RQ$_1$ investigates whether community smells can indicate abrupt developer disengagement in OSS. As developers rarely disclose mental health issues, observable social patterns may act as proxies. We examine whether these smells explain and potentially predict risks to developer well-being.

🔍 RQ$_2$. On socio-technical proximity as a protective factor.

Can abrupt disengagement be predicted by socio-technical proximity metrics?

RQ$_2$ examines whether *socio-technical proximity* protects against developer distress. Closer ties—through communication and shared work—may enable support, knowledge exchange, and better coordination.

In the following, we describe the research method used to conduct our investigation, while Figure 1 outlines the steps of our study.

Project Selection and Data Collection. The *context* of our analysis is a set of six projects from the *Apache Software Foundation* (ASF), selected for their open access to source code and rich communication data (e.g., mailing lists, issue trackers, and pull requests). In addition, ASF projects follow well-defined contribution processes. Following prior work on community smells [45], we selected projects with long development histories, active communication channels, and heterogeneous domains.

While a larger sample could increase coverage, six projects provide a balance between representativeness and feasibility. The selected set spans multiple domains and communication styles, ensuring diversity and comparability with prior studies. Further details on the projects and collected artifacts (JIRA issues, mailing list messages, GitHub comments, and commits) are available in our replication package [34].

Data were retrieved from *Apache Software Foundation* infrastructures: JIRA for issue tracking, Apache Pony Mail for mailing lists, and GitHub for collaboration traces and commit histories. Combining these sources enables triangulation of structural and social indicators.

Data Sources and Processing Pipeline. All artifacts were analyzed using Kaiaulu [36]. Development activities were segmented into 30-day windows to capture short-term dynamics while preserving sufficient data density. For each window, we constructed (1) a *communication network*, linking developers interacting in the same discussions, and (2) a *collaboration network*, connecting developers who modified the same files. We applied the OSLOM algorithm [30] to detect community structures and computed standard network metrics (e.g., degree, betweenness, density) to characterize cohesion and influence. We also measured socio-technical congruence, defined as the proportion of collaboration

ties reflected in the communication network [29], capturing alignment between coordination and technical dependencies.

From these networks, we derived individual-level indicators of activity and embeddedness. Activity was measured via workload (LOC added/removed) and communication volume. Embeddedness was captured through socio-technical proximity, defined as the average shortest-path distance to collaborators. We also identified instances of community smells (*Lone Wolf*, *Organizational Silo*, *Radio Silence*, and *Black Cloud*), distinguishing between those that occurred within a single window and those that persisted across multiple windows.

Operationalization of Outcome. We operationalized abrupt disengagement as an exit accompanied by a negative final message. A developer was considered inactive if no activity was observed for at least three consecutive months [28]. For each case, we analyzed the most recent communication using BERT-CP. If the probability of negative sentiment exceeded 15%, the instance was labeled as abrupt disengagement. This conservative threshold accounts for the predominance of neutral messages in OSS.

Data Preprocessing. Developer identities were unified across platforms using name- and email-based matching heuristics. Textual artifacts were cleaned by removing non-English content, system-generated messages, and signatures, and were subsequently normalized through tokenization, lowercasing, and stopword removal.

Modeling Approaches. To answer our research questions, we employ statistical models focused on community smells and socio-technical proximity.

RQ_1 – Community smells and abrupt disengagement. We modeled the relationship between community smells and the probability of abrupt disengagement using logistic regression [31]. This choice of model is justified by the binary nature of the outcome and the interpretability of the regression coefficients, which enable us to estimate the strength of each community smell's effect on the dependent variable.

Dependent Variable. The binary dependent variable indicates whether an abrupt disengagement event occurred for a given developer.

Independent Variables. The predictors of interest were the four community smells, i.e., *Lone Wolf*, *Organizational Silo*, *Radio Silence*, and *Black Cloud*. We operationalized variables as binary indicators, indicating whether the developer was classified under the corresponding smell during a given time window.

Control Variables. We controlled for individual developer activity to mitigate confounding effects. Specifically, we considered the *number of messages sent*, capturing each developer's engagement in communication, and the *standardized workload*, defined as the number of lines of code added or removed, normalized within the project to account for scale differences. A model constant was also retained to capture unobserved baseline variation.

RQ_2 – Socio-technical proximity as a protective factor. Similarly to RQ_1, we applied logistic regression, but the independent variables were drawn from measures of socio-technical proximity rather than community smells.

Dependent Variable. The binary dependent variable represents the occurrence of an abrupt disengagement event for a given developer.
Independent Variables. The main predictor was *socio-technical proximity*, measured as the *average distance* from a developer to their peers within the socio-technical network, which integrates both communication (reply) and collaboration (Git) ties. Lower values indicate stronger social and technical connectedness—reflecting frequent interaction and shared work—whereas higher values represent looser, more fragmented participation across the community.
To characterize socio-technical proximity, we considered several network-based variables that capture complementary aspects of how developers interact and collaborate. Among them, *socio-technical congruence* reflects the extent to which communication patterns align with technical dependencies in the codebase, that is, whether developers who need to coordinate for technical reasons also communicate effectively. To account for individual positions within these networks, *reply degree* and *Git degree* quantify the number of direct connections each developer maintains in the communication and technical networks, respectively. Likewise, *reply betweenness* and *Git betweenness* identify developers who act as bridges between otherwise disconnected groups, highlighting their role in mediating coordination and information flow. Finally, *reply density* and *Git density* provide a project-level view of the overall cohesiveness of communication and collaboration within the development community.
Control Variables. We employed the same set of controls as in $\mathbf{RQ_1}$ *i.e., number of messages sent* and the *standardized workload*.

For both models, positive coefficients (OR$>$ 1) indicate higher disengagement risk, while negative ones (OR$<$ 1) indicate a protective effect.

4 Analysis of the Results

Table 2: Distribution of outcomes and community smells across projects.

Project	Number Dev.	Messages	Abupt Dis.	LW	OS	RS	BC
ApexCore	50	15694	10	182	99	146	56
Ambari	95	3669	6	328	222	222	141
Bookkeeper	4	773	0	80	76	76	51
Helix	44	7306	6	231	211	211	132
Geronimo	44	40886	4	560	205	205	65
Thrift	208	10283	123	292	292	28	77

Before discussing our results, we provide a statistical description of the distribution of the abrupt disengagement phenomenon and the community smells we analyzed. As shown in Table 2, the distribution of outcomes and community smells varies considerably across projects. The number of active developers ranges from very small teams (e.g., Bookkeeper with only 4 developers) to large

ones (e.g., Thrift with over 200 developers). Likewise, the volume of messages exchanged is highly uneven, with Geronimo standing out due to its more than 40,000 messages, while Bookkeeper remains at a much smaller scale. Abrupt disengagement events are also unevenly distributed, with some projects showing very few cases and others reporting substantially higher counts. Finally, the presence of community smells is consistently detected across projects, but with varying intensity, suggesting that structural issues manifest differently depending on the project size, communication activity, and time span considered.

Table 3: An overview of the numerical variables per project.

Project	Value	Avg. Dist.	Messages Std.	Workload	STC	Deg. Comm.	Bet. Comm.	Deg. Collab.	Clos. Collab.	Bet. Collab.	Den. Comm.	Den. Collab.
Ambari	min	1.000	0.000	0.000	0.000	0.000	0.000	0.000	–	0.000	0.000	0.000
	25%	2.000	0.000	0.008	0.000	0.030	0.000	0.247	–	0.170	0.001	0.086
	mean	2.723	3.590	1.000	0.062	0.089	0.004	0.444	–	0.275	0.010	0.131
	75%	3.000	5.000	0.360	0.060	0.121	0.007	0.612	–	0.379	0.016	0.178
	max	4.000	31.000	31.270	1.000	0.184	0.026	0.672	–	0.560	0.079	0.189
ApexCore	min	1.000	0.000	0.002	0.000	0.260	0.037	0.000	0.267	0.000	0.086	0.000
	25%	1.414	3.000	0.083	0.333	0.314	0.084	0.222	0.300	0.117	0.162	0.200
	mean	2.210	18.675	1.000	0.458	0.367	0.109	0.324	0.506	0.238	0.234	0.352
	75%	2.667	29.000	1.195	0.500	0.428	0.129	0.389	0.734	0.315	0.262	0.500
	max	6.000	106.000	18.590	1.000	0.518	0.521	0.509	0.889	0.833	0.464	0.730
Bookkeeper	min	1.000	0.000	0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	25%	1.000	1.000	0.063	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
	mean	1.151	4.284	1.000	0.158	0.016	0.000	0.000	0.000	0.000	0.007	0.976
	75%	1.000	6.000	0.992	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
	max	2.000	18.000	8814.000	1.000	0.150	0.000	0.000	0.000	0.000	0.100	1.000
Geronimo	min	1.000	0.000	0.000	0.161	0.170	0.049	0.111	0.188	0.009	0.040	0.158
	25%	1.528	0.000	0.014	0.276	0.335	0.069	0.286	0.354	0.133	0.110	0.333
	mean	2.260	27.007	1.000	0.405	0.369	0.107	0.342	0.454	0.220	0.166	0.451
	75%	2.800	36.750	0.671	0.519	0.396	0.133	0.400	0.495	0.284	0.199	0.545
	max	7.000	364.000	46.675	0.913	0.546	0.304	0.585	0.762	0.585	0.365	0.889
Helix	min	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	25%	1.667	0.000	0.073	0.000	0.163	0.000	0.167	0.600	0.020	0.083	0.194
	mean	2.297	10.124	1.000	0.306	0.254	0.149	0.260	0.683	0.272	0.162	0.363
	75%	3.000	12.000	0.936	0.407	0.333	0.261	0.333	0.864	0.386	0.205	0.500
	max	5.000	166.000	25.933	1.000	0.560	0.529	0.611	1.000	1.000	0.467	1.000
Thrift	min	1.000	0.000	0.001	0.000	0.046	0.002	0.000	–	0.000	0.008	0.000
	25%	1.383	2.000	0.016	0.100	0.172	0.036	0.071	–	0.000	0.021	0.028
	mean	2.867	10.203	1.000	0.431	0.269	0.125	0.179	–	0.085	0.030	0.067
	75%	4.000	8.000	0.448	0.714	0.349	0.185	0.244	–	0.129	0.038	0.090
	max	6.000	210.000	19.359	1.000	0.552	0.396	0.500	–	0.477	0.072	0.194
Total	min	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	25%	1.500	0.000	0.021	0.056	0.133	0.008	0.212	0.300	0.083	0.019	0.123
	mean	2.345	14.870	1.000	0.322	0.263	0.092	0.306	0.451	0.212	0.111	0.316
	75%	3.000	17.000	0.677	0.473	0.380	0.130	0.389	0.629	0.307	0.179	0.487
	max	7.000	364.000	46.675	1.000	0.560	0.529	0.672	1.000	1.000	0.467	1.000

Table 3 reports descriptive statistics of the numerical variables computed per project. The values are summarized over 30-day observation windows (minimum, 25th percentile, mean, 75th percentile, and maximum), rather than aggregated across the entire project history. A 30-day window was selected as a trade-off between temporal granularity and data stability. On the one hand, it is short enough to capture fluctuations in developer activity and communication. On the other hand, it is long enough to smooth out transient inactivity or short-lived peaks of activity typically associated with release cycles [52]. This design choice ensures comparability across projects of different sizes and lifespans, while capturing the temporal variability of developer activity and network structure. Consequently, apparently counterintuitive patterns, such as a minimum of zero messages combined with a strictly positive mean, simply indicate that certain monthly windows exhibited no activity, whereas others contained substantial communication, yielding a positive average. The same reasoning applies to workload, centralization, and density measures, which can assume zero values in inactive windows and higher values when activity is concentrated. These

Table 4: Results of the Logistic Regression between Community Smells and Abrupt Disengagement. Statistically significant coefficients (p <0.05) are reported in **bold**.

Variable	Next 1 month				Next 3 months				Next Year			
	Coef.	Std.Err.	z	P> |z|	Coef.	Std.Err.	z	P> |z|	Coef.	Std.Err.	z	P> |z|
Lone Wolf	**2.4610**	0.768	3.203	0.001	0.0647	0.368	0.176	0.860	-0.3136	0.202	-1.551	0.121
Organizational Silo	-0.2488	0.463	-0.537	0.591	0.3364	0.249	1.349	0.177	**0.5745**	0.133	4.328	0.000
Radio Silence	**1.1455**	0.509	2.251	0.024	0.2199	0.260	0.845	0.398	-0.1727	0.137	-1.265	0.206
Black Cloud	-1.9338	0.491	-3.940	0.000	**-1.1497**	0.285	-4.039	0.000	-0.0932	0.139	-0.670	0.503
Messages Sent	**-0.0477**	0.016	-2.913	0.004	**-0.0262**	0.005	-4.796	0.000	**-0.0209**	0.003	-7.206	**0.000**
Standardized Workload	-0.0294	0.072	-0.407	0.684	-0.0084	0.027	-0.310	0.756	**-0.0498**	0.020	-2.516	0.012
Constant	-0.2020	0.896	-0.225	0.822	**1.7429**	0.570	3.057	0.002	**0.8705**	0.284	3.062	0.002

statistics provide an overview of the typical distributional range of the variables employed in the modeling pipelines, rather than cumulative project-level totals.

4.1 RQ1 – On the Correlation between Community Smells and Abrupt Disengagement

Table 4 reports the results of the logistic regression models examining the relation between community smells and the probability of abrupt disengagement occurring across three temporal horizons (1 month, 3 months, and 1 year).

In the short term, *Lone Wolf* emerges as a significant predictor, suggesting that developers who work in isolation are more likely to abruptly disengage in the following month. This finding aligns with the intuition that a lack of social integration can amplify frustration and reduce the perceived support network necessary to cope with conflicts or setbacks. *Radio Silence* shows a similar pattern: limited communication with others increases the likelihood of short-term, abrupt disengagement, underscoring the importance of maintaining active communication channels within a project. However, for both smells, the effect dissipates over longer periods, indicating that these factors primarily trigger immediate frustration rather than long-term disengagement.

In contrast, *Organizational Silo* shows a delayed but pronounced effect, becoming significant only in the last temporal horizon. This suggests that structural separation between subgroups or teams does not lead to immediate disengagement but gradually erodes collaboration and trust, eventually increasing the probability of abrupt disengagement. *Black Cloud*, conversely, is associated with a lower likelihood of abrupt disengagement in the short and medium term, possibly reflecting the role of experienced or central developers who, despite facing negative emotions, remain engaged and resilient—though this effect fades in the long run.

Moving to the control variables, the number of *Messages Sent* consistently recreates a protective role across all horizons: developers who communicate more frequently are less likely to experience abrupt disengagement, reinforcing the idea that active participation and visibility within the community can buffer frustration. *Standardized Workload* becomes significant only in the long term,

Table 5: Logistic regression results for socio-technical proximity metrics across 1, 3, and 12-month horizons. Statistically significant coefficients ($p < 0.05$) are in bold.

Variable	Next 1 month				Next 3 months				Next year			
	Coef.	Std.Err.	z	P> |z|	Coef.	Std.Err.	z	P> |z|	Coef.	Std.Err.	z	P> |z|
Avg. distance	**-0.6765**	0.186	-3.646	0.000	**-0.2926**	0.106	-2.765	0.006	-0.0748	0.059	-1.265	0.206
Socio-technical congruence	**-3.0131**	1.471	-2.048	0.041	-0.8344	0.694	-1.202	0.229	0.0727	0.348	0.209	0.834
Reply degree	4.5713	3.113	1.468	0.142	-1.5633	1.835	-0.852	0.394	**2.3659**	0.899	2.633	0.008
Reply betweenness	3.1080	3.378	0.920	0.358	-3.7050	1.962	-1.888	0.059	**-5.1782**	1.039	-4.982	0.000
Git betweenness	0.8394	1.234	0.680	0.496	-0.5290	0.591	-0.895	0.371	-0.2561	0.324	-0.790	0.429
Reply density	2.5116	2.301	1.091	0.275	**6.2016**	1.558	3.981	0.000	0.2213	0.755	0.293	0.769
Git density	-0.8394	1.234	0.680	0.496	-0.5015	0.512	-0.979	0.328	**-0.9461**	0.271	-3.490	0.000
Messages Sent	**-0.0477**	0.016	-2.913	0.004	**-0.0262**	0.005	-4.796	0.000	**-0.0209**	0.003	-7.206	0.000
Standardized Workload	-0.0294	0.072	-0.407	0.684	-0.0084	0.027	-0.310	0.756	**-0.0498**	0.020	-2.516	0.012
Constant	-0.2020	0.896	-0.225	0.822	1.7429	0.570	3.057	0.002	0.8705	0.284	3.062	0.002

suggesting that sustained, balanced workload levels may help prevent attrition over time.

These results indicate that abrupt disengagement is influenced by both individual and structural factors that operate on different temporal scales. Short-term risks are primarily linked to social isolation and communication breakdowns, while long-term risks emerge from entrenched organizational fragmentation. This temporal differentiation suggests that abrupt disengagement is not a spontaneous act, but rather the culmination of evolving socio-technical tensions. Moreover, the observed associations provide empirical support for the theory of community smells as indicators of *social debt* [44], highlighting how unresolved organizational misalignments may accumulate and manifest as frustration and eventual withdrawal.

4.2 RQ2 – On the Correlation between Socio-technical proximity and Abrupt Disengagement

Table 5 presents logistic regression models that investigate the role of socio-technical proximity factors in predicting abrupt disengagement across different time horizons. Several consistent yet temporally distinct patterns emerge, suggesting that the interplay between communication and collaboration structures evolves over time.

In the short term (1 month), lower communication distance is significantly associated with reduced disengagement, indicating that tightly connected teams buffer immediate withdrawal. Similarly, socio-technical congruence has a protective effect, suggesting that alignment between coordination needs and communication reduces short-term disengagement. However, this effect weakens over time, indicating it prevents impulsive exits but not longer-term tensions.

Looking at the medium temporal windows, communication intensity begins to show more complex effects. *Reply density* emerges as a strong predictor, with higher density significantly increasing the odds of abrupt disengagement. This result suggests that while active communication is beneficial in the short term, excessive interaction may lead to information overload, interpersonal friction, or

burnout, eventually triggering withdrawal. At the same time, *reply betweenness* shows a negative, marginally significant association, suggesting that developers who serve as intermediaries in communication channels may play a stabilizing role by distributing information and easing coordination stress within the community.

Over the long term (1 year), the relationships between socio-technical factors and abrupt disengagement become more differentiated. *Reply degree* shows a significant positive association with abrupt disengagement, suggesting that developers who receive numerous replies may become overburdened by communication demands and expectations, increasing their risk of attrition. Conversely, *reply betweenness* becomes a strong negative predictor, indicating that developers who bridge otherwise disconnected groups are less likely to disengage, likely because of their central role and sense of purpose within the project. In parallel, *Git density* exhibits a robust negative effect, highlighting the protective influence of cohesive and distributed collaboration on code contributions, which appears to sustain engagement over time.

Regarding the control variables, the results mirror those observed for $\mathbf{RQ}_1$. The number of *Messages Sent* remains a significant protective factor across all horizons, reaffirming that maintaining communication activity is essential to developer retention. Meanwhile, *Standardized Workload* becomes significant only in the long-term window, suggesting that a balanced workload contributes to long-term stability but has little immediate impact on short-term disengagement.

Socio-technical proximity influences abrupt disengagement in time-dependent ways: communication and alignment reduce early exits, while long-term stability requires balancing workload, connectivity, and distribution. Disengagement thus emerges as an evolving phenomenon driven by both excessive interaction and disconnection.

5 Discussion and Implications

The results show that the risks of abrupt disengagement stem not only from technical issues but also from how communities organize, communicate, and evolve over time.

For developers, the findings underscore that collaboration is a double-edged sword. Being closely embedded in networks of communication and code contributions helps protect against sudden exits: lone wolves and isolated members are most at risk of abrupt disengagement. Yet, collaboration overload can itself contribute to stress and disengagement. In other words, not all collaboration is equally beneficial: meaningful, distributed contributions are protective, while constant "chatter" without structure can become a liability. Moreover, occupying bridging positions (reply betweenness) in communication networks can be stressful at first, but over time, it builds resilience by providing access to diverse information and informal support. To mitigate collaboration overload, teams can leverage Conway's Law [23] by aligning communication channels with the software architecture. This ensures structured, modular interactions mapped

to technical responsibilities, reducing cognitive fragmentation. Structural issues also stem from time-dependent effects. While *Black Cloud* dynamics initially shield developers, their persistence doubles long-term risks, and Organizational Silos silently erode resilience. Thus, leaders must monitor how structural smells evolve over time, rather than just tracking acute disengagement. Finally, linguistic traces show that frustration surfaces in words before exits, making tone and constructive dialogue the first line of defense.

For researchers, our results suggest three directions for future inquiry. First, the predictive patterns highlight that disengagement unfolds over time and across layers of collaboration. Signals that seem protective in the short run—such as intense interaction or central positions in communication networks—can later reverse as fatigue or overload accumulate. This indicates that disengagement should be studied as a dynamic, evolving process rather than a discrete event, calling for longitudinal approaches that trace how risk factors change direction as projects and relationships mature. Moreover, our findings reinforce and extend ongoing research on toxic communication and negative affect in developer interactions [41], showing how these patterns can escalate toward emotional exhaustion and abrupt withdrawal. Second, the role of socio-technical structures suggests the need for models that account for how contributors' positions within communication and collaboration networks influence their engagement trajectories. Our findings indicate that structural configurations—such as isolation, overload, or asymmetric bridging roles—reflect patterns often linked to community smells and low socio-technical proximity. These conditions can heighten coordination costs and emotional strain, gradually undermining motivation and inclusion. Future research should therefore move toward integrative models that combine behavioral, linguistic, and network perspectives—where language-aware tools operate alongside socio-technical indicators to capture both the structural and emotional dimensions of developer disengagement. Third, the growing availability of fine-grained socio-technical data opens opportunities for proactive and preventive interventions. Our findings suggest that early warning signals of abrupt disengagement, such as increasing isolation, communication breakdowns, or persistent structural misalignments, can be empirically detected through changes in network configurations and interaction patterns. Building on this evidence, future research may investigate automated alerting or feedback mechanisms that help community moderators and project maintainers recognize and mitigate these risks before they escalate. This would require interdisciplinary collaboration across software analytics, organizational psychology, and human–computer interaction to design effective and socially responsible systems.

6 Threats to Validity

In terms of construct validity, abrupt disengagement is operationalized as a contributor's final message followed by a predefined inactivity interval. This proxy may overlook psychological and social aspects of exit behavior and misclassify atypical cases (e.g., off-channel departures). To mitigate this, we adopt conser-

vative thresholds and sustained inactivity requirements, privileging sensitivity over specificity. Cross-platform identity matching can be error-prone, and tools for detecting community smells and sentiment may misclassify context. We mitigate these risks through the identity unification (e.g., names, email hashes, commit signatures across GitHub, JIRA, and mailing lists) and by using validated tools such as Kaiaulu and a fine-tuned BERT model. We also include covariates to reduce confounding effects. Finally, we included all contributors observed in the collected socio-technical traces, including developers with very limited participation (e.g., a single commit). As a result, some cases classified as abrupt disengagement may reflect the completion of a one-time contribution rather than the withdrawal of an actively engaged community member. Future work should investigate the effect of applying participation thresholds.

As for the internal validity, we acknowledge a potential threat stemming from data imbalance, as a single project (`Thrift`) accounts for approximately 82.5% of the total identified abrupt disengagement events. This high concentration may bias our logistic regression models toward the specific socio-technical dynamics of that community, meaning the identified risk and protective factors might not apply uniformly across all analyzed projects.

In terms of external validity, although we analyze six heterogeneous *Apache* projects, generalizability beyond similar OSS contexts is limited. Proprietary projects are excluded due to lack of observable socio-technical data. While project selection aims to ensure diversity, broader datasets are needed to extend these findings.

Finally, for conclusion validity, the limited number of disengagement events may reduce statistical power and increase Type II error risk. Regression assumptions may also be challenged in social-networked settings. We address these issues using standard errors, cross-validation, and cross-project triangulation. Results should be interpreted as associational rather than causal.

7 Conclusions

This paper presents an empirical study of abrupt disengagement in OSS, examining the role of community smells and socio-technical proximity across six *Apache* ecosystems. Cohesive networks act as protective factors, while isolation, poor communication, and misalignments increase risk. Some smells (e.g., *Organizational Silo*, *Black Cloud*) evolve from neutral to long-term risks. Overall, disengagement emerges as a measurable socio-technical outcome. Future work will expand the dataset, adopt longitudinal designs, and combine models with developer surveys.

Acknowledgments. This paper was partly supported by the European HORIZON-KDT-JU-2023-2-RIA research project MATISSE (grant 101140216-2, KDT232RIA 00017).

References

1. From developer networks to verified communities: A fine-grained approach. In: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering. vol. 1, pp. 563–573. IEEE (2015)
2. Afeltra, A., Cannavale, A., Pecorelli, F., Pontillo, V., Palomba, F.: A large-scale empirical investigation into cross-project flaky test prediction. IEEE Access **12**, 131255–131265 (2024)
3. Afi, N.J.: The silent disengagement: Exploring the impact of quiet quitting on organizational culture. International Journal of Research and Innovation in Social Science **9**(6), 1508–1530 (2025)
4. Alexandrovsky, D., Gerling, K., Opp, M.S., Hahn, C.B., Birk, M.V., Alsheail, M.: Disengagement from games: Characterizing the experience and process of exiting play sessions. CHI **8**, 1–27 (2024)
5. Arora, R., Goel, S., Mittal, R.K.: Supporting collaborative software development over github. Software: Practice and Experience **47**(10), 1393–1416 (2017)
6. Blincoe, K., Valetto, G., Damian, D.: Do all task dependencies require coordination? the role of task properties in identifying critical coordination needs in software projects. In: FSE. pp. 213–223 (2013)
7. Blincoe, K., Valetto, G., Goggins, S.: Proximity: a measure to quantify the need for developers' coordination. In: CSCW. pp. 1351–1360 (2012)
8. Caballero-Espinosa, E., Carver, J.C., Stowers, K.: Community smells—the sources of social debt: A systematic literature review. Information and Software Technology **153**, 107078 (2023)
9. Caldiera, V.R.B.G., Rombach, H.D.: The goal question metric approach. Encyclopedia of software engineering pp. 528–532 (1994)
10. Calefato, F., Lanubile, F., Maiorano, F., Novielli, N.: Sentiment polarity detection for software development. In: ICSE. p. 128 (2018)
11. Crowston, K., Wei, K., Howison, J., Wiggins, A.: Free/libre open-source software development: What we know and what we do not know. ACM Computing Surveys (CSUR) **44**(2), 1–35 (2008)
12. De Weerd, M., Klandermans, B.: European Journal of Social Psychology **29**(8), 1073–1095 (1999)
13. Di Meglio, S., Starace, L.L.L., Pontillo, V., Opdebeeck, R., De Roover, C., Di Martino, S.: Performance testing in open-source web projects: Adoption, maintenance, and a change taxonomy. In: 2025 IEEE International Conference on Software Maintenance and Evolution (ICSME). pp. 199–210. IEEE (2025)
14. Fagerholm, F., Guinea, A.S., Münch, J., Borenstein, J.: The role of mentoring and project characteristics for onboarding in open source software projects. In: ESEM. pp. 1–10 (2014)
15. Fang, Y., Neufeld, D.: Understanding sustained participation in open source software projects. Journal of Management Information Systems **25**(4), 9–50
16. Ferreira, F., Silva, L.L., Valente, M.T.: Turnover in open-source projects: The case of core developers. In: BSSE. pp. 447–456 (2020)
17. Filippova, A., Cho, H.: Mudslinging and manners: Unpacking conflict in free and open source software. In: CSCW. pp. 1393–1403 (2015)
18. Fox, S., Spector, P.: Counterproductive work behavior: Investigations of actors and targets. american psychological association (2005)
19. Gachechiladze, D., Lanubile, F., Novielli, N., Serebrenik, A.: Anger and its direction in collaborative software development. In: ICSE. pp. 11–14 (2017)

20. Gerosa, M., Wiese, I., Trinkenreich, B., Link, G., Robles, G., Treude, C., Steinmacher, I., Sarma, A.: The shifting sands of motivation: Revisiting what drives contributors in open source. In: ICSE. pp. 1046–1058 (2021)
21. Giordano, G., Festa, G., Catolino, G., Palomba, F., Ferrucci, F., Gravino, C.: On the adoption and effects of source code reuse on defect proneness and maintenance effort. Empirical Software Engineering **29**(1), 20 (2024)
22. Giordano, G., Sellitto, G., Sepe, A., Palomba, F., Ferrucci, F.: The yin and yang of software quality: On the relationship between design patterns and code smells. In: 2023 49th Euromicro conference on software engineering and advanced applications (SEAA). pp. 227–234. IEEE (2023)
23. Herbsleb, J.D., Grinter, R.E.: Splitting the organization and integrating the code: Conway's law revisited. In: Proceedings of the 21st international conference on Software engineering. pp. 85–95 (1999)
24. Hunsen, C., Siegmund, J., Apel, S.: On the fulfillment of coordination requirements in open-source software projects: An exploratory study. EMSE **25**(6), 4379–4426 (2020)
25. Islam, M.R., Zibran, M.F.: Sentistrength-se: Exploiting domain specificity for improved sentiment analysis in software engineering text. JSS **145**, 125–146 (2018)
26. Kazman, R.: Managing social debt in large software projects. In: SESoS-WDES. p. 1. IEEE (2019)
27. Kelloway, E.K., Francis, L., Prosser, M., Cameron, J.E.: Counterproductive work behavior as protest. Human resource management review **20**(1), 18–25 (2010)
28. Koch, A.S., Forgas, J.P., Matovic, D.: Can negative mood improve your conversation? affective influences on conforming to grice's communication norms. European Journal of Social Psychology **43**(5), 326–334 (2013)
29. Kwan, I., Schroter, A., Damian, D.: Does socio-technical congruence have an effect on software build success? a study of coordination in a software project. IEEE Transactions on Software Engineering **37**(3), 307–324 (2011)
30. Lancichinetti, A., Radicchi, F., Ramasco, J.J., Fortunato, S.: Finding statistically significant communities in networks. PloS one **6**(4), e18961 (2011)
31. LaValley, M.P.: Logistic regression. Circulation **117**(18), 2395–2399 (2008)
32. Linåker, J., Link, G., Lumbard, K.: Sustaining maintenance labor for healthy open source software projects through human infrastructure: A maintainer perspective. In: Proceedings of the 18th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. p. 37–48. ESEM '24, Association for Computing Machinery, New York, NY, USA (2024)
33. Martins, L., Pontillo, V., Costa, H., Ferrucci, F., Palomba, F., Machado, I.: Test code refactoring unveiled: where and how does it affect test code quality and effectiveness? Empirical Software Engineering **30**(1), 27 (2025)
34. Mulder, W., Penta, M.D., Giordano, G., Kazman, R., Palomba, F., Pontillo, V., Tamburri, D.A.: Online appendix for "i quit!": Understanding practitioner abrupt disengagement through socio-technical factors in open source software. `https://github.com/giammariagiordano/i_quit.github.io` (2025), online website: `https://giammariagiordano.github.io/i_quit.github.io/`
35. Palomba, F., Tamburri, D.A., Fontana, F.A., Oliveto, R., Zaidman, A., Serebrenik, A.: Beyond technical aspects: How do community smells influence the intensity of code smells? TSE **47**(1), 108–129 (2018)
36. Paradis, C., Kazman, R.: Building the msr tool kaiaulu: Design principles and experiences. In: ECSA (2021)

37. Pontillo, V., Martins, L., Machado, I., Palomba, F., Ferrucci, F.: An empirical investigation into the capabilities of anomaly detection approaches for test smell detection. Journal of Systems and Software **222**, 112320 (2025)
38. Recupito, G., Giordano, G., Ferrucci, F., Di Nucci, D., Palomba, F.: When code smells meet ml: on the lifecycle of ml-specific code smells in ml-enabled systems. Empirical Software Engineering **30**(5), 139 (2025)
39. Ribeiro, D.M.: Understanding the relationships between the perceptions of burnout and instability in software engineering. In: BSSE. pp. 58–67 (2022)
40. Robles, G., Gamalielsson, J., Lundell, B.: Setting up government 3.0 solutions based on open source software: the case of x-road. In: ICEG. pp. 69–81 (2019)
41. Sarker, J.: Identification and mitigation of toxic communications among open source software developers. In: ASE. pp. 1–5 (2022)
42. Stol, K.J., Ali Babar, M.: Challenges in using open source software in product development: a review of the literature. In: Proceedings of the 3rd international workshop on emerging trends in free/libre/open source software research and development. pp. 17–22 (2010)
43. Tamburri, D.A., Kazman, R., Fahimi, H.: The architect's role in community shepherding. IEEE Software **33**(6), 70–79 (2016)
44. Tamburri, D.A., Kruchten, P., Lago, P., Vliet, H.v.: Social debt in software engineering: insights from industry. Journal of Internet Services and Applications **6**(1), 10 (2015)
45. Tamburri, D.A., Palomba, F., Kazman, R.: Exploring community smells in open-source: An automated approach. IEEE Transactions on software Engineering **47**(3), 630–652 (2019)
46. Tamburri, D.A., Zhang, H., Blincoe, K., Kazman, R., Giordano, G., Pontillo, V., Palomba, F.: "the candle is burning out on its own..": Modeling fatigue and empathy among chinese developers. In: Euromicro Conference on Software Engineering and Advanced Applications. pp. 155–173. Springer (2025)
47. Trinkenreich, B., Stol, K.J., Sarma, A., German, D.M., Gerosa, M.A., Steinmacher, I.: Do i belong? modeling sense of virtual community among linux kernel contributors. In: ICSE. pp. 319–331 (2023)
48. Trinkenreich, B., Wiese, I., Sarma, A., Gerosa, M., Steinmacher, I.: Women's participation in open source software: A survey of the literature. ACM Transactions on Software Engineering and Methodology (TOSEM) **31**(4), 1–37 (2022)
49. Van Zomeren, M., Postmes, T., Spears, R.: Toward an integrative social identity model of collective action: A quantitative research synthesis of three socio-psychological perspectives. Psychological bulletin **134**(4), 504 (2008)
50. Vasilescu, B., Yu, Y., Wang, H., Devanbu, P., Filkov, V.: Quality and productivity outcomes relating to continuous integration in github. In: Proceedings of the 2015 10th joint meeting on foundations of software engineering. pp. 805–816 (2015)
51. Voria, G., Scala, B., Todisco, L., Venditto, C., Giordano, G., Catolino, G., Palomba, F.: Fair and square? evaluating fairness of llm-generated synthetic datasets. Information and Software Technology p. 107980 (2025)
52. Wiggins, A., Howison, J., Crowston, K.: Social dynamics of floss team communication across channels. In: ICOSS. pp. 131–142 (2008)
53. Zhang, T., Xu, B., Thung, F., Haryono, S.A., Lo, D., Jiang, L.: Sentiment analysis for software engineering: How far can pre-trained transformer models go? In: ICSME. pp. 70–80 (2020)