Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

RESEARCH PAPER

A Novel, Tool-Supported Catalog of Community Smell Symptoms

Antonio Della Porta[1] | **Stefano Lambiase**[1] | **Gemma Catolino**[1] | **Filomena Ferrucci**[1] | **Fabio Palomba**[1]

[1]Software Engineering (SeSa) Lab, Department of Computer Science, University of Salerno, Italy

Correspondence
Corresponding author Antonio Della Porta
Email: adellaporta@unisa.it

Abstract

Software development is a multifaceted endeavor, requiring a profound grasp of both social dynamics and technical intricacies. Poor collaboration often leads to the accumulation of *social debt*, manifesting as unforeseen project costs due to sub-optimal team interactions. *Community smells* have emerged as indicators of these socio-technical inefficiencies and potential social debt. While previous research has focused on automated detection of community smells through analyzing developer communication patterns, our study offers a complementary approach. We emphasize the critical role of project managers in assessing socio-technical dynamics and propose a novel, tool-supported catalog of symptoms. This catalog can be used for manual inspections to identify early signs of community smells at the individual level, allowing managers to address issues before they escalate. Using a mixed-method design that leveraged an existing literature review and a user survey, we cataloged symptoms related to four community smell types. Additionally, we developed `TOAST`, a tool that operationalizes this catalog, and assessed its usability. The paper concludes by shedding light on the potential impact of our work and its contribution to advancing the detection and analysis of community smells.

KEYWORDS
Social Debt; Community Smells; Mixed-Method Research; Recommendation System.

1 | INTRODUCTION

Software development is a socio-technical activity in which social phenomena, e.g., collaboration and communication between team members, and technical aspects, like the software product or technology adopted, are profoundly connected[1,2,3,4]. Because of this interconnected nature, practitioners need ad hoc artifacts and constructs to navigate the complexity of the development process and achieve the best product. This is even more critical during software maintenance and evolution activities, which often require maintainers to communicate and collaborate (the social) to prevent the inevitable deterioration of a software product (the technical)[5,6]. As teams navigate the complex tasks of updating, refining, and extending software systems, the quality of their interactions significantly impacts the efficiency and effectiveness of their work.

Recalling the famous quote from Tom DeMarco, *"You can't control what you can't measure"*[7], the research community started proposing a plethora of socio-technical metrics—e.g., socio-technical congruence and turnover—resulting in the creation of the so-called *"Social Debt"*, i.e., the unforeseen project costs connected to the presence of poor collaboration and communication conditions within a software community[8]. Sooner after, *Community Smells*—inspired by the well-known Code Smell[9]—arose to characterize sub-optimal socio-technical phenomena in managing a software community that are precursors of Social Debt[10,11]. Since the definition of Community Smells, the research community started investigating their role in terms of impact[12,13,14,15], the way to predict them[16,17,18,19], and the development of tools for their detection and management[20,21].

While significant progress has been made in understanding and detecting community smells, most existing research has focused on developing automated tools that rely on mining software repositories to identify these socio-technical inefficiencies[12,13,14,15,16,17,18,19,20,21]. However, these tools are primarily designed to analyze patterns in communication and code, which means they can sometimes overlook the subtle, individual behaviors, e.g., reluctance to collaborate or ineffective communication,

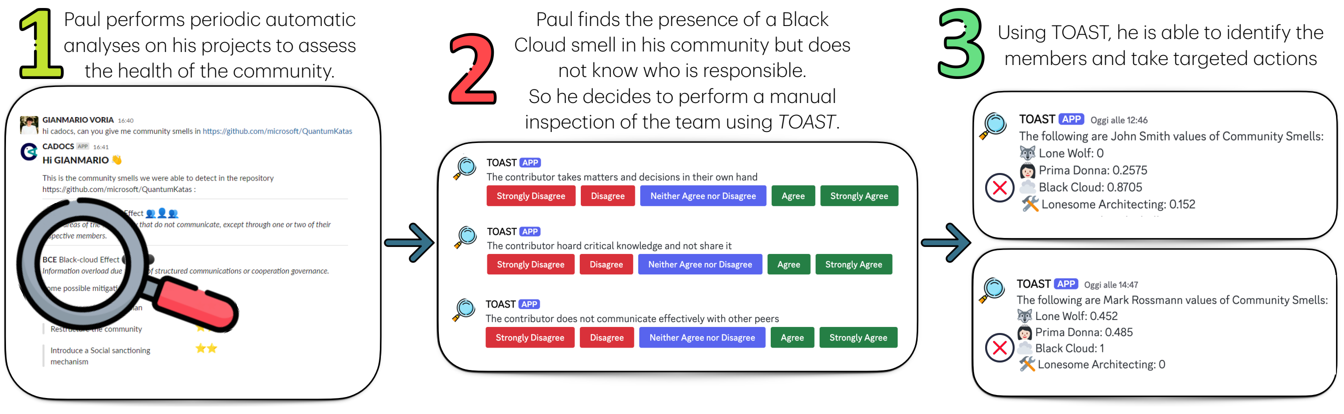

FIGURE 1 TOAST - Visionary scenario.

that contribute to community smells. As a consequence, we argue that by complementing automated tools with a human-centric approach, project managers can achieve a more comprehensive understanding of the socio-technical environment, enabling the early detection of community smells that might otherwise go unnoticed.

This paper introduces a complementary approach to community smell detection, emphasizing the critical role of project managers in assessing the socio-technical dynamics within a software community. Specifically, we propose a novel, tool-supported catalog of symptoms that indicate the emergence of community smells. This catalog is designed to be used by project managers to monitor and evaluate individual developers' behaviors, enabling them to identify and address early signs of community smells before they escalate into more significant issues. To build the catalog, we conducted a mixed-method research, combining a literature review with empirical data gathered from a survey of experienced practitioners. In addition, we developed a preliminary recommendation system, TOAST, in the form of a Discord bot, which allows users to apply the knowledge from our research to effectively manage social debt within their communities.

> ◎ **Research Objective**
>
> *The objective of this work is to enhance community smell detection by identifying and validating symptoms, enabling managers to recognize issues early. Additionally, a Discord bot-based recommendation system is developed to help practitioners apply this knowledge in managing social debt within their communities.*

This work makes the following contributions to the current state of the art:

- A detailed analysis of the symptoms associated with four specific Community Smells, accompanied by a catalog and an assessment of the importance of each symptom.
- A recommendation system[†] designed to be user-friendly for practitioners, making the research findings accessible and applicable for managing social debt within communities.
- Made all tests and data publicly available in an online appendix [22] to ensure reliability and promote open-source collaboration.

1.1 | Visionary Scenario

To better illustrate how our approach can be applied, let's consider a visionary scenario that demonstrates the integration of automatic and manual methods for the analysis of the community. This scenario is visually represented in Figure 1. Imagine Paul, a seasoned Project Manager, overseeing multiple software projects and teams. His responsibilities include conducting semi-annual team evaluations, managing project timelines, and ensuring the financial health of each project. To maintain efficiency and prevent unforeseen costs, Paul typically relies on automated tools to analyze project repositories and monitor team interactions,

[†] Tool repository link: https://github.com/atdepo/toast-tool

identifying potential community smells that could disrupt progress. However, Paul understands that automated tools, while powerful, can sometimes miss the subtle, human-centric behaviors that contribute to socio-technical inefficiencies. For this reason, Paul turns to `TOAST`, a tool we propose in this study, to assess individual contributors manually. `TOAST` allows Paul to identify the nuanced behaviors—such as ineffective communication or reluctance to collaborate—that might be overlooked by automated systems. For instance, when `TOAST` flags a potential *Black Cloud* smell, indicating that critical knowledge is being hoarded, Paul can directly pinpoint the team members who might not be sharing information effectively. The tool provides him with a detailed analysis of communication patterns, recurring behaviors, and individual tendencies that contribute to the smell. Conversely, Paul could begin his analysis by using `TOAST` during his routine team evaluations. By manually identifying early signs of community smells through the behaviors of individual team members, Paul gains a deeper understanding of the socio-technical dynamics at play. Using these insights, he can then apply automated tools to validate these findings at a broader scale, ensuring that the issues identified are not only isolated incidents but are reflective of deeper, systemic problems within the team or project. This dual approach allows Paul to cross-verify and refine his strategies, addressing issues with a level of precision that significantly reduces the risk of social debt.

By integrating manual assessments with automated detection—or vice versa—Paul can take targeted actions to address issues before they escalate. Whether it is arranging a one-on-one meeting with a specific contributor to discuss their communication habits or setting up a team workshop to enhance knowledge-sharing practices, Paul's proactive and informed approach ensures that his teams remain collaborative, efficient, and aligned with the project's goals. In Figure 1, it is shown how `TOAST` can help Paul in his activities. In essence, `TOAST`—and the catalog of symptoms—can empower a manager to move beyond surface-level insights, enabling him to manage the socio-technical dynamics of his teams with unprecedented precision. This integration of manual and automated analysis not only has the potential to enhance practitioner's ability to maintain a healthy project environment but also sets a new standard for community smell detection and management in software development.

The paper is structured as follows: in Section 2, we describe the state of the art of community smells and their detection. Section 3 outlines our research design and methodology. Section 4 presents our research findings, while Section 5 discusses a recommendation system built on the main reserach work findings. Section 6 examines the impact, implications, and validity threats of the research. Finally, Section 7 concludes our study by summarizing findings and proposing future work.

2 | BACKGROUND AND RELATED WORK

Software development and its engineering are socio-technical activities. To measure the impact of social phenomena on software development, researchers—inspired by the well-known concept of Technical Debt [9]—defined *Social Debt*, i.e., the unforeseen project costs derived from sub-optimal choices in the collaboration and communication aspects—so, management—of a software development team [8]. Moreover, aiming to provide a way to characterize better and identify the source of Social Debt, Tamburri et al. defined *Community Smell*, i.e., socio-technical anti-patterns whose existence could lead to Social Debt [10].

The research investigated how community smell relates to different aspects of software development. Interestingly, Palomba et al. [12] studied the relationship between community smells and code smells, their product-oriented counterpart, demonstrating that the first are among the top factors influencing the emergence of the last. Furthermore, other researchers focused on establishing the impact of community smells on other dimensions of software engineering, e.g., architecture debt [13] and organizational structure types [14]. Tamburri et al. [15] conducted a large-scale investigation on 60 open-source ecosystems to evaluate (1) community smell diffusion and (2) the perceived impact by developers, showing that not only are smells largely present but they are also perceived to have an impact in the evolution and sustainability of software communities. Such a result shows that socio-technical antipatterns could influence maintenance and evolution in two parallel ways: directly increasing costs (i.e., social debt) and impacting product factors, thus increasing technical-related costs (i.e., technical debt).

More related to their detection, Palomba and Tamburri [18] provided a machine-learning approach to predict community smells considering socio-technical metrics, obtaining promising results (i.e., F-measure of 78%). Moreover, Almarimi et al. [19] proposed a multi-label learning model based on genetic algorithms to detect ten community smells. The work of Almarimi resulted then in the publication of csDetector [21], a tool implementing the detection strategy described in previous work. Based on the work of Almarimi et al. [21], Voria et al. [20] developed CADOCS, a conversational agent able to detect, given a software repository, ten community smells and propose strategies to refactor some of them. As a final note, Catolino et al. [16] and Lambiase et al. [17] conducted two mining studies, revealing that a correlation seems to exist between gender (in the former) and cultural (in the latter) heterogeneity and the emergence of community smells.

TABLE 1 An overview of the community smells defined in literature, according to Caballero-Espinosa et al.[11].

Community Smells				
1. Architecture by osmosis	7. Cookbook development	13. Informality excess	19. Newbie free-riding	25. Prima donnas
2. Architecture hood	8. DevOps clash	14. Institutional isomorphism	20. Obfuscated architecting	26. Radio silence
3. Black cloud	9. Disengagement	15. Invisible architecting	21. Organizational silo	27. Sharing villainy
4. Class cognition	10. Dispersion	16. Leftover techie	22. Organizational skirmish	28. Solution defiance
5. Code red	11. Dissensus	17. Lone wolf	23. Power distance	29. Time warp
6. Cognitive distance	12. Hyper community	18. Lonesome architecting	24. Priggish members	30. Unlearning

To provide a comprehensive catalog of the community smells defined in literature, Caballero-Espinosa et al. conducted a literature review[11] to identify and catalog the various community smells reported in software engineering research. Table 1 lists the 30 distinct community smells identified in the literature. These findings have served as the foundation for our research, providing the basis for our empirical study.

When comparing our work with previous research efforts in the field, our study introduces a novel perspective by focusing on the proactive role of project managers in detecting and mitigating community smells. Unlike previous studies that primarily rely on automated detection methods and machine-learning techniques to identify community smells based on socio-technical metrics, our research emphasizes the development of a comprehensive, tool-supported catalog of symptoms specifically tailored for manual inspection by project managers. This human-centric approach not only supplements automated methods but also enables early identification of potential issues by leveraging the insights and observations of project leaders, thereby bridging the gap between technical detection and managerial intervention. Moreover, by incorporating a mixed-method research design that includes both literature review and user surveys, our study provides a more holistic understanding of the factors leading to the emergence of community smells, offering actionable guidance that is both technically informed and contextually relevant for practitioners. On the basis of these considerations, the scientific novelty of our work lies in shifting the focus from automated detection to empowering project managers with actionable insights for early intervention, while the technical novelty lies in the development of TOAST, a specialized tool designed to operationalize the symptom catalog and facilitate the hands-on detection and management of community smells in real-time project settings.

3 | OBJECTIVE AND RESEARCH DESIGN

The *goal* of this study was to introduce a novel approach for the identification and characterization of community smells through the manual recognition of their symptoms. The *purpose* was to equip practitioners—particularly managers and team leaders—with structured knowledge to make more informed decisions, thereby enhancing the likelihood of software project success. The *perspective* encompasses both practitioners and researchers: practitioners are interested in innovative strategies and tools to effectively address communication and collaboration challenges throughout the software development lifecycle, while researchers are focused on advancing the understanding of the symptoms behind community smells, which can inform future studies and lead to improvements in existing community smell detection techniques.

3.1 | Research Questions

To achieve the study's goal, we conducted a mixed-method research investigation, comprising both qualitative data extraction and a survey study. We began by breaking down our primary goal into two specific sub-goals, each associated with a research question. Below, we introduce each research question along with its motivation.

Our first objective was to develop a new approach for recognizing community smells that could be used both in conjunction with state-of-the-art automated detection methods and independently in a manual community inspection process. To this end, we focused on identifying a comprehensive catalog of symptoms (e.g., unusual behaviors, recurring errors, performance drops, communication breakdowns) that are indicative of these smells, leading to the formulation of the following research question:

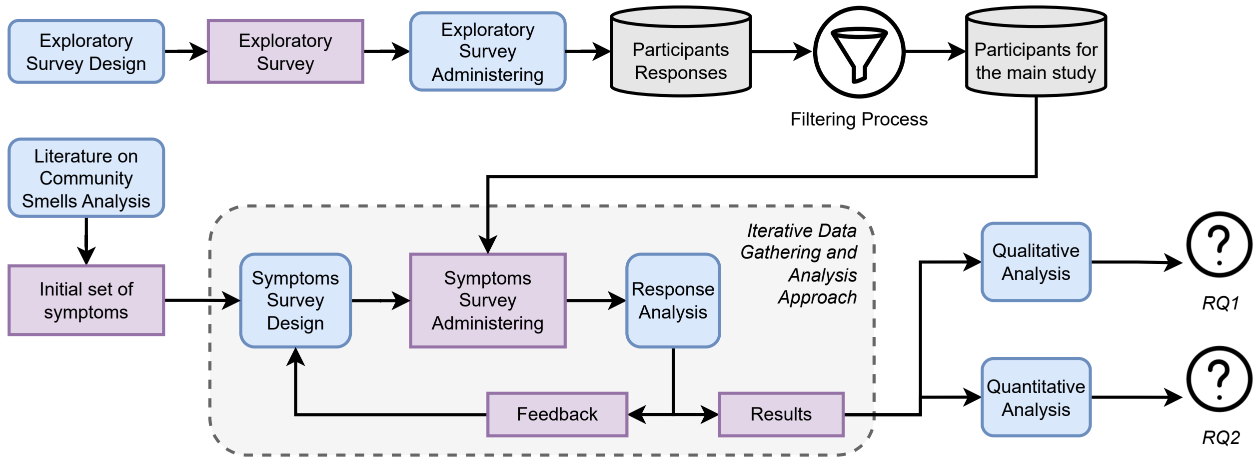

FIGURE 2 Research Method Overview.

> ⓘ **RQ$_1$** — *What are the most commonly reported symptoms of community smells?*

After identifying the key behaviors that practitioners associate with community smells, we sought to further refine and characterize these symptoms to provide more practical guidance for practitioners. This led us to our second research question, which aims to assess the relevance of these symptoms by ranking them according to practitioners' experiences and perceptions.

> ⓘ **RQ$_2$** — *How indicative are the identified symptoms of the presence of community smells?*

Altogether, our study results in a catalog of symptoms associated with community smells, which forms the foundation for a practical diagnostic tool and enhances automated detection methods. The first research question identifies the key symptoms, while the second evaluates their relevance, ensuring the catalog is both comprehensive and practical for real-world application.

3.2 | Research Method Overview

To address our research questions, we employed a three-step methodological approach illustrated in Figure 2:

1. **Participant Selection through Exploratory Survey.** We first conducted an exploratory survey aimed at identifying practitioners with substantial expertise in software development and project management. This preliminary step was required to recruit participants who may have provided credible and insightful data.
2. **Literature Analysis for Initial Symptom Compilation.** Next, we performed a comprehensive analysis of existing literature on community smells, building upon the systematic review by Caballero-Espinosa et al.[11]. This step enabled us to extract an initial set of symptoms associated with various community smells, providing a well-founded starting point for our investigation. Additionally, this analysis allowed us to contextualize our research within the existing body of knowledge, identifying a reasonable set of community smells to focus on.
3. **Iterative Survey Study for Symptom Refinement and Ranking.** Finally, we conducted an iterative survey study designed with both open-ended and close-ended questions. This dual-format approach qualitatively addressed **RQ$_1$** by capturing insights and experiences from practitioners, while also allowing for quantitative analysis to address **RQ$_2$** through the ranking and evaluation of identified symptoms. The iterative nature of the survey ensured refinement and validation of the data collected, leading to robust and actionable results.

The combination of the three methodological steps is directly aligned with the objectives of our work. The symptoms behind community smells can be accurately identified only through the firsthand experiences of project managers, who possess the deep understanding and practical insights necessary to recognize these complex socio-technical issues within their teams. As such, a survey study represented the most valid research method to employ. Unlike interviews, which are time-consuming and limit the number of participants, surveys allow us to reach a broader, more diverse group of practitioners, enabling the collection of a wide range of experiences and perceptions. Additional quantitative research methods, like statistical analysis or data mining, are not suitable in this context because they rely on existing datasets, which often focus solely on developer interactions within software repositories. These methods may overlook the symptoms of community smells that emerge from the broader social and organizational dynamics best captured through the firsthand experiences of project managers.

Regarding the specific steps taken, the exploratory survey enabled us to engage with knowledgeable practitioners who could provide valuable insights into community smells. The literature analysis established a theoretical foundation for identifying an initial catalog of symptoms. The iterative survey was then instrumental in not only prioritizing these symptoms but also in uncovering additional symptoms based on the practical, real-world experiences of these experts.

For both surveys, we relied on PROLIFIC,[‡] a web-based platform designed for researchers to efficiently sample participants and administer questionnaires. PROLIFIC allowed us to precisely target participants who met our selection criteria by customizing survey preferences and constraints (Section 3.2). Additionally, the platform's reliability metric enabled us to select participants with a proven track record of providing valuable responses, thereby enhancing data quality. It is worth noting that PROLIFIC employs an *opt-in* strategy, which may introduce self-selection or voluntary response bias. To mitigate this potential bias, we adhered to the guidelines proposed by Reid et al.[23], which outline best practices for conducting surveys in the software engineering domain using this platform.

3.3 | Participants Selection—Exploratory Survey

Our first step consisted into the identification of our participant sample; we conducted an exploratory survey aimed at identify a reliable sample of participants for the next step of the research. To achieve this, we developed an exploratory survey to gauge the manager's perception of the community smells. Using the data gathered by this survey, we also aimed to gain insight into the technology transfers' state on this matter. It is important to note that since this questionnaire, participants have been informed—thus asked to express agreement—of the possibility of participating in an ulterior survey study.

To design the first draft of the exploratory survey, we relied on the well-known guidelines developed by Kitchenham and Pfleeger[24], along with Andrews et al.[25], which are widely recognized in software engineering research. After, we conducted a pilot study with three practitioners and two researchers in the computer science domain to (1) collect feedback and consequentially improve the survey and (2) estimate the time to completion.[§] Then, we identified the target population of our survey and administered it to them; for the exploratory survey, we put as the only criteria of being actively involved in software development as a team member or manager (and similar leading figures). Nevertheless, we asked in the questionnaire for the working role to perform a post-execution evaluation. According to Flanigan et al.[26] advice, we ensured the survey remained anonymous to avoid influencing participants' responses. Furthermore, according to similar studies, we included two attention-check questions to ensure data reliability. The survey was created using a Google Form and was designed to be completed within 10 to 15 minutes.

The exploratory survey questionnaire was designed as a mix of open- and close-ended questions. It was composed of four sections. In the first one, we provided participants with general information on the study, data policy, and requests for agreement. The second section asked for general information on the participants, such as nationality, for a demographic analysis of the sample. It is important to note that the platform used for administering the questionnaire (i.e., PROLIFIC) already provided us with such knowledge; we asked them for reliability and attention-check reasons. We also collected the gender of participants for demographic reasons; we did not ask participants directly but used the information they provided in the public participant data (this data is sent to Prolific at the registration stage). The third section asked for working data, e.g., company size, working role position, and self-assessed experience in software development or management. Moreover, we asked participants if they had certification in project management (e.g., PMI[¶]) as a way to assess their skills. Last, in the fourth section, we asked participants about their experiences with community smells (after providing them with the definition) and asked them to provide us with a

[‡] PROLIFIC website: https://www.prolific.com/

[§] The pilot highlighted an estimated completion time of 2 minutes. Feedback from the pilot participants led us to change the question on community smell knowledge level, i.e. *S1-9*, and add an open question to add personal experiences with the smells, i.e. *S1-10*, to gauge (1) their perception of the topic and (2) the form on which they experienced them.

[¶] Project Management Institute website: https://www.pmi.org/ (8 August 2024)

TABLE 2 Esploratory Survey Questions.

ID	Question
S1-1	Nationality
S1-2	What role best describes your current job?
S1-3	What is your company size?
S1-4	Are you familiar with managing distributed teams (where team members are spread across different parts of the globe)?
S1-5	For how many years have you covered a position of team manager or similar?
S1-6	How do you evaluate your experience in Team Management?
S1-7	What is your team size?
S1-8	Have you ever achieved PMI certification? Alternatively, indicate which other certification you have earned.
S1-9	Are you familiar with the concept of Community Smell, i.e., social anti-pattern characterizing communication and collaboration patterns in a development community?
S1-10	If yes, tell us briefly about your experience in that matter

TABLE 3 Inclusion Criteria

ID	Question	Criteria
S1-5	For how many years have you covered a position of team manager or similar?	$\geq$ 3 years
S1-6	How do you evaluate your experience in Team Management?	$\geq$ 4
S1-8	Have you ever achieved PMI certification? Alternatively, indicate which other certification you have earned.	Yes or Other Certifications

textual description of these characteristics. Then, we thanked the study participants and provided an opportunity to leave feedback on the questionnaire. Table 2 reports the complete list of questions. All demographic information was collected primarily to characterize our sample and provide more reliability to our results, as well as provide some statistical data to support our results.

After disseminating the exploratory questionnaire, we selected only the managerial figures from the sample using the responses given to the *S1-2* question that defines the job role. The answers provided by the managers were analyzed to help us comprehend the demographics of our sample, their field expertise, and their awareness and knowledge of community smell, which is essential for assessing the status of technology transfer. In particular, we analyzed the responses to the question *S1-9* to estimate how many managers had never experienced community smells (value 0 on the Likert Scale) and the ones that somehow experienced them (value greater than 0). From the managers who experienced community smells, we analyzed the responses given to the question *S1-4* to estimate if there was a correlation between a distributed community and the knowledge of community smells, and the responses given to the open-question *S1-10* to understand what were their experiences on the matter and also assess how they perceive them. Through such a mixed analysis, we were able to cluster each participant based on community smell knowledge, thus providing a picture of the overall knowledge of the matter and experience in the software development field.

The answers provided for this questionnaire were mainly used to select the participants' sample for the main survey study (described in Section 3.5) aimed at answering our research question; we selected from the managers of the exploratory survey only the ones that satisfied our criteria, reported in Table 3. Specifically, we evaluated the managers in terms of (1) years of experience, (2) self-assessed evaluation of managerial skills, and (3) obtained certifications.

3.4 | Analysis of the Literature on Community Smells

Since our study involved identifying a set of symptoms of community smells, we started by extracting what was already available in the literature. By doing so, we wanted to (1) be sure to capture already identified patterns to start asking with and (2) identify which community smells best suited our goals.

3.4.1 | Literature and Smells Identification and Selection

We did not perform a literature review ourselves since Caballero-Espinosa et al.[11] published a recent one in *Information and Software Technology*. This review provided a thorough overview of community smells, making it highly relevant to our study's objectives. Thus, instead of duplicating efforts, we analyzed their work to extract the specific information needed for our research.

We then selected the set of community smells that best aligned with our goals by focusing on the type of analysis that would be most beneficial to project managers in the intended use case: diagnosing issues at the level of individual contributors. Since our research aims to develop a tool that project managers can use to identify and address community smells, we concentrated on those originating from individual behaviors rather than at the organizational level. This focus is particularly useful because it allows managers to detect early signs of community smells at the contributor level, enabling more precise and timely interventions.

The procedure we envision is especially valuable for this type of analysis because individual-level smells are more directly observable and actionable within a team setting. Managers can more easily recognize and address these symptoms, which often manifest as specific behaviors or patterns within their teams. This targeted approach not only makes the tool more practical for day-to-day use but also ensures that interventions can be made before these issues escalate into larger, more systemic problems. Based on this reasoning, we focused on four community smells[15,27,28], whose definition is provided in the following:

Black Cloud. This smell occurs when organizations do not provide the conditions for social interactions and effective communication between teammates, thus not supporting the exchange of knowledge during software development processes.

Lone Wolf. This smell occurs when defiant teammates carry out their work irrespective or regardless of their peers, reflecting poor communication addressing project needs. The effects are, for instance, unsanctioned architecture decisions across the development process, code smells, and project delays.

Lonesome Architecting. Non-architect teammates see the need to make architecture decisions because the current architects are too few and far apart. From a social perspective, developers are unaware of what they are doing. Also, this scenario leads to incompatibility problems and faster decision-making.

Prima Donnas. This smell indicates the presence of teammates working in isolation. They are unwilling to welcome the change of legacy products and support from other teammates. These teammates prevent the organization from innovative solutions or processes and effective communication and collaboration.

3.4.2 | Analysis of the Literature

After identifying the smells, we analyzed the findings of the literature review on them to extract the ones that seem related to human behaviors between the reported causes. To do so, we conducted one round of *deductive coding*[29] analysis on the parts of the paper reporting each smell. More precisely, we conducted *structural coding*[29], i.e., a categorization process of text according to a specific structure, with a structure of two elements: the causes of smell and the smell itself. Moreover, relying on the literature review set of identified papers, we also analyzed the content of several papers on the selected smells in the same way. Figure 3 provides an example of such a coding process.

The first author began by thoroughly reviewing all identified literature and systematically extracting the necessary information. This data extraction process was straightforward, with the first author compiling all relevant details—such as the paper, key text, and associated smells—into a structured sheet. The extraction of causes from the identified text was a collaborative effort between the first two authors, allowing for a comprehensive analysis of the data. Given the manageable amount of data, the authors were able to review and discuss the findings together thoroughly. Similar causes were then iteratively consolidated into singular concepts until no further merging was possible. The outcome of this process, presented in our online appendix[22], was a refined set of symptoms for the selected smells, which served as the foundation for developing the questionnaire.

3.5 | Symptoms Survey

We performed an iterative survey study approach to answer our two research questions. In the survey—better described in the following paragraphs—we asked practitioners to report their experiences when encountering community smells, focusing on the human behaviors that most helped to recognize them. Such a study was explicitly designed with both open- and close-ended questions in order to allow both a qualitative and a quantitative analysis. The former allowed us to conduct a coding process

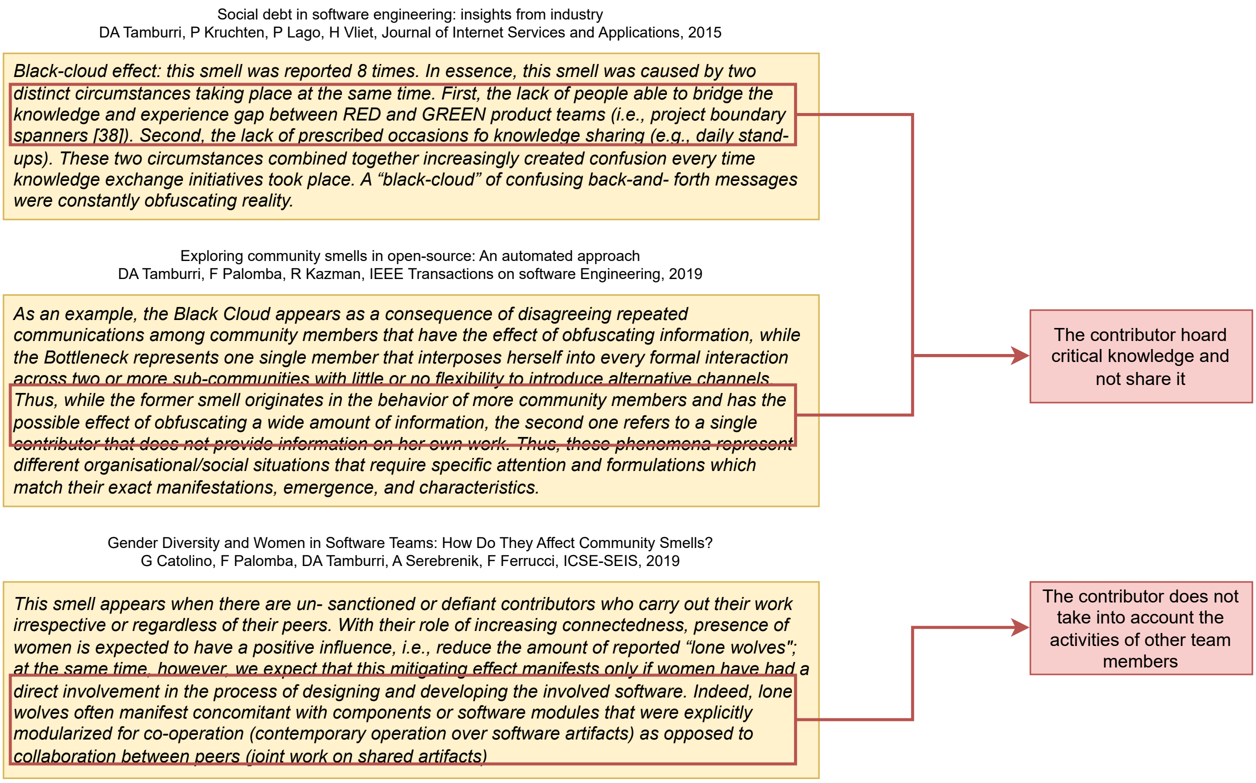

FIGURE 3 Example of the Coding Process.

to identify the symptoms—thus answering our first research question—while the latter was used to assess their importance in evincing associated smells—thus answering the second research question.

As with the exploratory survey, we relied on the well-known guidelines for this type of study in software engineering[24,25,26]. Moreover, we improved the first draft by conducting a pilot study[#] and included some attention-check questions in the survey. We adopted an iterative approach; the questionnaire was administered in multiple instances to a different set of participants, each time after data analysis and consequential modification to the questionnaire were done[‖]. The modifications regard adapting the survey to new findings obtained after each step, aiming to obtain more profound knowledge and reach theoretical saturation.

The questionnaire was made up of seven sections. The first section provided an introduction for participants and included details of the study, data policy information, and a request for agreement. The second section contained the general definition of community smells to ensure that participants had the same understanding of the topic. The next four sections were based on the four community smells identified during the literature analysis step (Section 3.4). Each section began with the definition of the smell and followed with questions asking participants how many times they had experienced the smell and if it had a tangible impact on communication and collaboration in their team. Additionally, for each smell, participants were asked to indicate (using a Likert Scale from 1 to 5) their agreement with the relationship between the symptoms identified for the smell and the smell itself. Furthermore, for each smell, an open-ended question was included to gather any additional symptoms experienced by the participant. Lastly, to gain more insight, participants were asked to provide feedback on other anti-patterns not included in the questionnaire and to express any concerns or feedback on the questionnaire itself. The analysis of the open-ended questions was made similarly to the one described in Section 3.4.

To answer the first research question, the answers to the open and close-ended questions, combined with the literature review analysis, were analyzed to identify reliable data and define (by means of coding) the set of symptoms. Regarding the second research question, whose aim was to provide a ranking of the symptoms, we statistically analyzed the participants' answers

[#] The pilot highlighted an estimated completion time of 15 minutes. No modifications were made to the draft.
[‖] We conducted three iterations of questionnaire administering and data analysis.

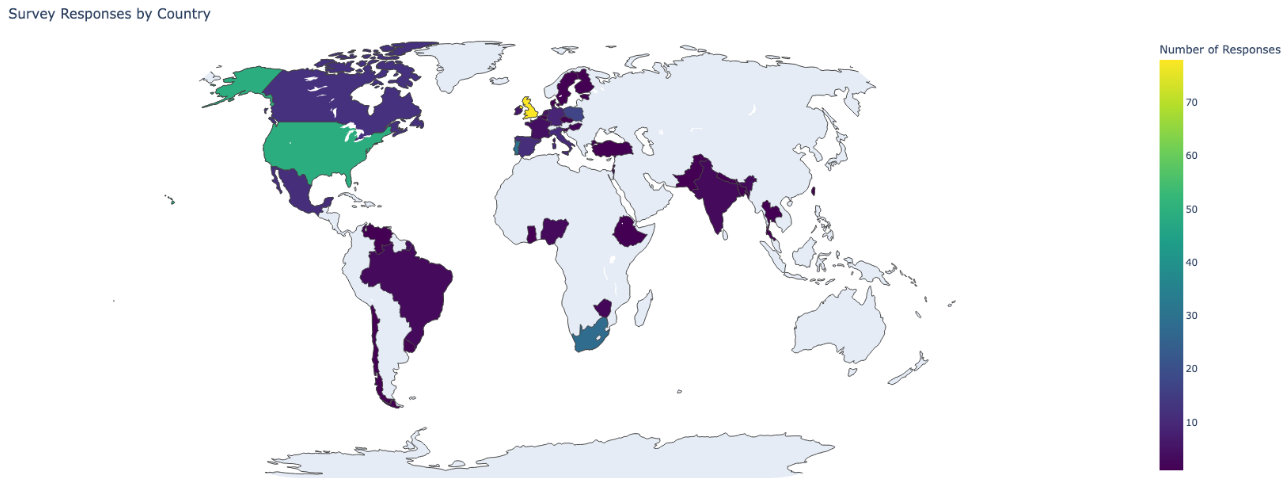

FIGURE 4 Geographical Distribution of Participants in the Exploratory Study.

for each behavior in an aggregated manner. Specifically, for each candidate behavior, we considered the median and standard deviation of the set of all participants' answers. We used this metric to compute how much the participants agreed on the perceived relevance of the behavior. Moreover, we also included in our analysis the answer to the question asking the encounter frequency of each smell; we intended to add a "weight" to the opinion of each manager according to his/her experience.

3.6 | Ethical Consideration

Studies involving human participants in our country do not require approval from an Ethical Review Board yet. Nevertheless, the survey design considered numerous ethical and privacy concerns. Below, we outline the precautions we have taken to ensure full compliance with ethical considerations:

- All activities related to the surveys were entirely anonymous. We recorded no identifying information of the involved participants.
- We made it clear to participants that they could withdraw their survey submission at any time and that no information entered up to that point would be tracked.
- We followed the indications of PROLIFIC to ensure alignment between privacy policies.

These measures were all agreed upon and explicitly communicated by the authors of the paper. All of them were made clear to the participants before any surveys for our research were conducted.

4 | ANALYSIS OF THE RESULTS

In this section, we highlight the study's main findings and answer the research question posed in the previous chapter.

4.1 | Participant Selection—Exploratory Survey

The first step of our research involved administering the exploratory questionnaire (described in section 3.2) to a set of software engineers using PROLIFIC. With this questionnaire, we primarily aimed to identify reliable managerial figures to collect data for our research questions. We also wanted to find out if managers know about these problems, how often they see them, and what they think about them since this helps us connect our research with what actually happens in software teams. We received 304 responses from participants across various regions worldwide, as can be seen in Figure 4. A demographic analysis shows that among them, 78 participants (25.7%) are from the United Kingdom, and 49 (16.2%) are from the United States. Additionally,

FIGURE 5 Word Cloud Of The Answers Of *S1-10*

there is representation from non-Anglophone nations, with 30 (9.90%) from Portugal and 28 (9.24%) from South Africa, among others. The sample exhibits diversity not only geographically but also in participant roles. The most prominent role in the sample is the Project Manager with 96 participants (31.7%), followed by Development Team Member with 76 participants (24.1%), and Software Architect with 30 participants.

In order to understand whether the sample of participants was familiar with the concept of community smells and to measure the state of technology transfer, we leveraged the answers given to the closed-question *S1-9* (Table 2). From the entire set of participants, we found that 245 (80.86%) were not familiar with it, 26 (8.58%) were familiar, and 32 (10.56%) were not sure. This information exposes the fact that from the entire set of participants, the majority of them had never heard of the concept of community smell. Even by selecting only managerial figures, using the answers from question *S1-2* (Table 2) and thus obtaining a subset of 140 managers, the percentages do not vary: 111 of the managers were not familiar with the community smells (79.28%), 16 of them were familiar (11.42%), and 13 were not sure (9.28%).

To broaden the scope of the discussion, it is worth reporting that, while analyzing the responses to the open-ended question *S1-10* (Table 2), we realized that the participants predominantly perceive community smells as communication issues. To further explore this perception, we generated a word cloud, shown in Figure 5, from the open-ended responses provided by participants. As shown in the figure, the most prominent terms, such as "communication", "community", "team", and "patterns", indicate a strong association between community smells and communication-related challenges. This visual representation underscores that, according to participants, ineffective communication is a key factor in the emergence of community smells. The frequency of terms like "collaboration", "development", and "interaction" further emphasizes the critical role of effective communication and teamwork in mitigating these socio-technical issues. This finding has two significant implications. Firstly, our results diverge slightly from the existing body of knowledge on community smells, as they suggest a stronger emphasis on communication problems than previously documented, which traditionally focuses on broader socio-technical factors, like collaboration among team members. More importantly, it emphasizes the need for tools that complement automated approaches, as communication issues are inherently tied to behavioral dynamics that influence how developers interact. This reinforces the motivation behind our study, which aims to develop an instrument that enables managers to diagnose individual behavioral symptoms of community smells, like those rooted in communication issues.

Following our analysis of the selection survey responses, we developed and applied criteria to identify a subset of expert managers from the exploratory survey participants. This refined group was then targeted for our main survey distribution. The set of inclusion criteria is defined in Table 3 and maps the questions chosen to be the filter of our sample to the inclusion criteria to match in order to be selected. After applying our inclusion criteria to the 303 survey respondents, we identified 31 expert managers who qualified to receive the main questionnaire focused on community smell symptoms that will help us answer the study research questions.

4.2 | RQ_1: What are the most commonly reported symptoms of community smells?

The focus of the first research question was the identification of the symptoms associated with each community smell. We first analyzed the already published literature [11] to identify an initial set of symptoms for the four community smells considered in the study. We adopted deductive coding, as defined in Section 3.4. For the analysis of the symptoms of the Lone Wolf smell were analyzed the works by Tamburri et al. [15] and Catolino et al. [16], for Prima Donna were analyzed the works by Tamburri et al. [27],

TABLE 4 Catalog of the symptoms elicited from our work, along with their impact on the emergence of community smells.

ID	Community Smell	Symptom Identified	Source	Weighted Mean	Standard Deviation	Score
1	Lone Wolf	The contributor has insufficient communication with the team	Literature	3.712	0.984	3.774
2	Lone Wolf	The contributor does not take into account the activities of other team members	Literature	4.110	0.899	4.571
3	Prima Donna	The contributor have an unwillingness to accept help or support from peers	Literature	4.205	0.894	4.702
4	Prima Donna	The contributor refuses to listen to the ideas or opinions of peers	Survey	3.963	0.895	4.427
5	Black Cloud	The contributor takes matters and decisions in their own hand	Literature	3.444	1.295	2.659
6	Black Cloud	The contributor hoard critical knowledge and not share it	Survey	3.857	1.079	3.576
7	Black Cloud	The contributor does not communicate effectively with other peers	Survey	4.036	1.000	4.033
8	Lonesome Architecting	The contributor complained of a lack of knowledge of the product requirements	Literature	3.597	1.099	3.273
9	Lonesome Architecting	The contributor complained of a loss of general vision of the product	Literature	3.389	1.240	2.733
10	Lonesome Architecting	The contributor was called upon to make architectural decisions that were not his responsibility	Literature	3.705	1.248	2.969

for Black Cloud were analyzed the works by Tamburri et al.[27] and by Palomba et al.[12], and for Lonesome Architecting was analyzed the works by Damian A. Tamburri[28].

Afterwards, we conducted a survey study with managers selected in the exploratory survey (Section 3.3). We aimed at (1) evaluating the symptoms and (2) collecting insight for new ones. As exposed in the previous section, we selected 31 experts from the exploratory survey. Such 31 experts were then divided into subsets to participate in consecutive data collection and analysis iterations. This iterative process (detailed in Section 3) aimed to confirm the identified symptoms and to collect insights from managers, potentially uncovering additional behaviors for validation in subsequent iterations.

In the first iteration, we disseminated the questionnaire to 15 participants randomly chosen from the set of 31. The survey was available for four days; all participants responded. After the iteration, a qualitative analysis of the open-ended question at the end of each section was performed, based on the participants' experience, to determine if some other behaviors or factors can indicate the presence of the community smell discussed in that section. The responses were analyzed to identify whether participants provided novel symptoms not previously captured.

The analysis revealed three other potential symptoms for Black Cloud and Prima Donna. For example, one participant answered the open-ended question on the Black Cloud: *"When someone is hoarding critical knowledge and not sharing it or when there is not an effective communication between team members"*. Based on this answer, we can observe that this community smell is seen not only as having autonomous decision-making authority as expressed in the literature but also as exhibiting reluctance in sharing crucial knowledge. Given the pieces of information gained, we added to the catalog the novel symptom *"The contributor hoard critical knowledge and not share it"*. Following the same process (detailed in the online appendix[22]), we uncovered new symptoms that were added to the new version of the questionnaire.

In the second iteration, the augmented questionnaire was distributed to 16 participants who were not included in the first iteration. The questionnaire was available for seven days; 11 managers answered it, and one was excluded because of an attention check question failure. The responses of the second iteration confirmed the previously identified symptoms and did not offer new insights into others, so we decided to stop our iterative process.

At the end of the process, we successfully identified and validated 10 symptoms. This comprised 2 pertaining to the Lone Wolf community smell, both drawn from existing literature; 2 concerning the Prima Donna smell, with one sourced from literature and the other uncovered through questionnaire responses; 3 associated with the Black Cloud, with 1 extracted from literature and 2 uncovered through questionnaire responses; and finally, 3 linked to Lonesome Architecting, all of which were derived from existing literature. All the symptoms validated with the source of provenience are reported in Table 4.

4.3 | RQ$_2$: How indicative are the identified symptoms of the presence of community smells?

Our second question aimed to evaluate the weight of each symptom associated with smell. To achieve this, we quantified the importance of each behavior by means of a summary measure.

First, we leveraged the responses to the questions of the perceived importance of a symptom identified, whose answers were rated on a Likert Scale ranging from 1 to 5. Initially, we conducted a weighted mean computation for each question, integrating the familiarity rating provided in the initial question of each survey section as the weighting factor—this methodological choice aimed to accommodate participants' varying degrees of familiarity with the community smell under scrutiny. The weights range between 0.2, which corresponds to the answer "*Never*" to 1, which corresponds to the answer "*Always*", with incremental steps of 0.2. Then, we computed the weighted means and assayed the standard deviation for each question's responses. This statistical metric enlightened the dispersion or variability characterizing participants' opinions concerning the importance of each symptom. Lastly, we derived a distinct score for each question by computing the weighted mean and standard deviation ratio. This scoring framework enabled the quantification of the symptoms that garnered high scores of perceived importance and manifested relatively low variability in participant responses. So, the higher the score is, the more the community smell to which it refers can be present (according to the managers' perception). This metric, along with all others produced by this statistical analysis, are listed in Table 4.

The analysis revealed key insights. Symptoms with particularly high scores, such as *"The contributor does not take into account the activities of other team members"* and *"The contributor refuses to listen to the ideas or opinions of peers"*, are strong indicators of the presence of community smells like Lone Wolf and Prima Donna. These findings suggest that project managers should prioritize monitoring these behaviors as early warning signs of potential socio-technical issues.

At the same time, symptoms with lower scores, such as *"The contributor takes matters and decisions into their own hands"* and *"The contributor was called upon to make architectural decisions that were not their responsibility"*, while still relevant, may be more context-dependent and less universally recognized as indicators of community smells. This implies that these symptoms might require additional context or corroborating evidence before being flagged as significant concerns.

In conclusion, the metrics produced by this statistical analysis provide a practical diagnostic tool for project managers. By focusing on the highest-scoring symptoms, managers can proactively address potential issues within their teams, thereby contributing to a healthier and more collaborative development environment.

5 | TOAST: A COMMUNITY SMELL SYMPTOMS TRACKING TOOL

To make our research results practical and actionable for practitioners, we developed a simple recommendation system as a proof of concept for leveraging the catalog of symptoms we identified and validated. This tool, named `TOAST` (**T**eam **O**bservation **A**nd **S**mells **T**racking tool), is built on the Discord** platform. `TOAST` empowers managers to conduct a manual analysis using the catalog of symptoms, drawing on their management expertise and perspective to identify and mitigate community smells early. The complex social dynamics underlying community smells[30,16,28] suggest that relying solely on automated detection tools may not fully capture these phenomena within team structures. The development of `TOAST` is driven by two key objectives: (1) Enhance practitioners' awareness of potential issues, which is critical for effective problem resolution; and (2) Integrate seamlessly with the widespread use of recommendation systems in software development environments.

The source code of the tool can be found in the online appendix[22], complete with the installation steps and all the needed procedures to generate and retrieve the token needed to start the application correctly.

It is worth emphasizing that `TOAST` is intended to complement, not replace, existing evaluation processes. The final interpretation of its scores should be integrated with other information sources to have a comprehensive understanding of the situation. In this section, we outline the key features of `TOAST` and report practical use cases that illustrate how the tool can be employed.

5.1 | Tool Functionalities

`TOAST` employs a survey-based methodology to identify community smells where a manager can fill out a questionnaire based on the catalog of symptoms, aiming at recognizing the presence of symptoms of community smells within the team. Each

** https://discord.com

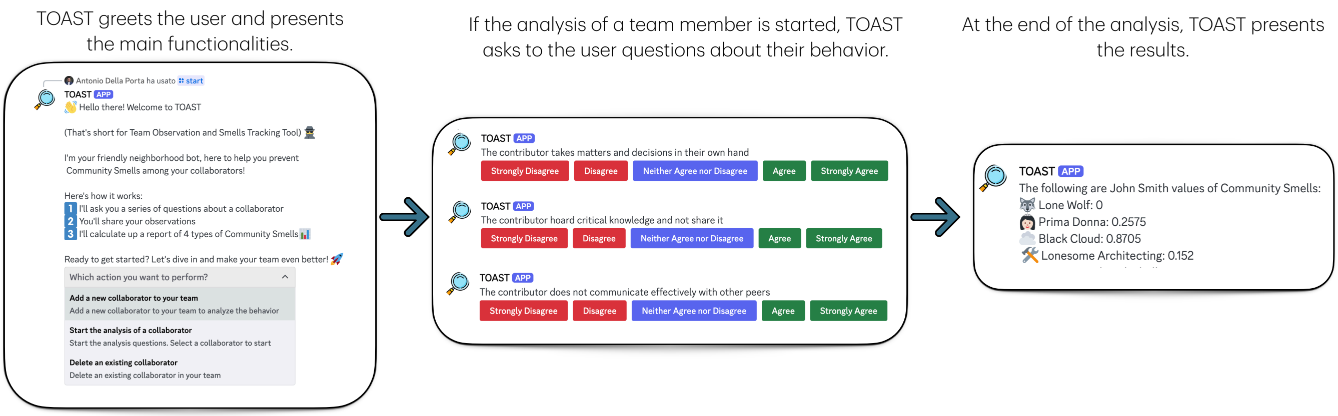

FIGURE 6 TOAST flow of actions to identify potential community smells.

behavior is presented as a specific statement, and managers are asked for each team member to indicate their level of agreement or disagreement regarding the individual's adherence to these behaviors on a Likert Scale of 5 values ranging from *Strongly Disagree* to *Strongly Agree*. User interaction with TOAST to analyze a single contributor follows these steps:

Step 1. Interaction is initiated through the `/start` command in the Discord chat interface. Upon activation, the bot presents itself, as can be seen in step 1 of Figure 6, issues a greeting message and presents to the user the three primary functionalities: the option to **start an analysis** of a team contributor or manage team composition through **member addition or removal**.

Step 2. When the user selects the analysis process, the bot systematically presents survey questions one by one, each accompanied by interactive response buttons to facilitate manager input, as can be seen in step 2 of Figure 6. These questions encompass all validated symptoms associated with community smells. As the manager progresses through the survey, responding to each question, the bot records and processes the inputs.

Step 3. Upon answering all the 10 questions TOAST proposes, the analysis process terminates, and a final summary containing all the scores calculated for each community smell relative to the contributor in question is shown as can be seen in step 3 of Figure 6. After the analysis, TOAST prompts the user to another selection box where the main functionalities are presented again, and the bot is ready for another command.

The final score for each contributor is computed using a weighted system based on the values in the score column of Table 4. This system incorporates a scaling factor determined by the managers' responses to each survey question. Specifically:

- For a *strongly agree* response, a factor of 1 is applied, allocating the entire score to the relative community smell.
- An *agree* response results in a factor of 0.5, allocating half of the potential score.
- *Disagree* and *strongly disagree* responses are weighted similarly, with factors of -0.5 and -1, respectively.

For example, if the user given the question *"The contributor has insufficient communication with the team"* (Table 4 ID 1) selects *disagree*, then a score of -1.856 is added to the final score, if they answer *strongly agree* the score added will be 3.712 and so on. The weighted scores for each question are then summed to produce the final score, standardized in a range of [0, 1] for each community smell. The score analysis serves as a quantitative measure of the contributor's susceptibility to various community smells. Moreover, the outcome of the analysis process produces final scores disaggregated by each analyzed, expressed in numerical form rather than as binary classifications. This approach eschews a simplistic susceptible/non-susceptible dichotomy for specific community smells. Using numerical scores empowered managers to establish context-specific thresholds for identifying contributors exhibiting particular smells. This flexibility is crucial given the heterogeneity of organizational contexts and situations, as establishing universally applicable thresholds presents a significant challenge within the constraints of currently available data. The numerical scoring system thus provides a flexible approach that accommodates the diverse environments in which community smells may manifest, allowing for nuanced interpretation and application of the results across varied team dynamics and organizational cultures.

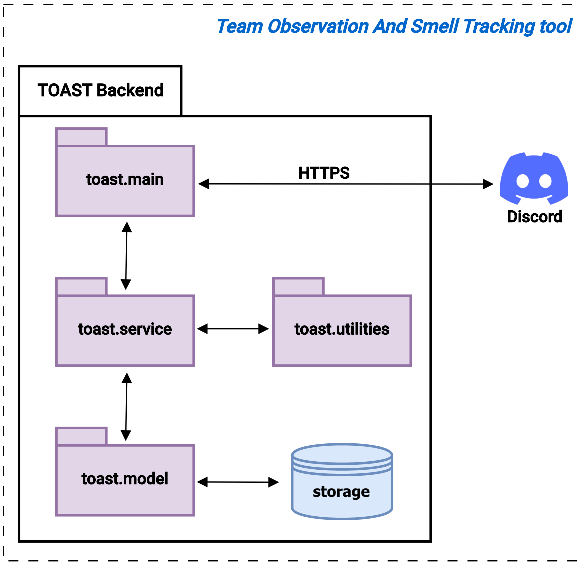

FIGURE 7 TOAST Architecture.

5.2 | Tool Architecture and Technical Choices

As we already said, TOAST was developed as a Discord bot. Discord's widespread adoption within the software development community, where it has emerged as a robust alternative to enterprise communication tools like Slack and Microsoft Teams, has led to its selection as the primary interface. Discord's popularity among developers, rich API, and extensible bot ecosystem provides an ideal environment for deploying and scaling the tool.

For the development, we used `discord.js`,[††] a JavaScript library that wraps the Discord API to streamline integration with JavaScript applications. This library benefits from robust open-source community support and offers extensive documentation and comprehensive guides that significantly facilitate development.

The architectural foundation of TOAST is built upon the Three-Tier pattern. This well-established software design paradigm stratifies the application into distinct functional layers, enhancing system maintainability and scalability. This architectural choice is particularly suited given the usage of the Discord platform. In this configuration, the Discord infrastructure manages the presentation layer, allowing the application to focus on core functionalities. Specifically, they are streamlined into three primary tasks: structuring message content, retrieving analytical results, and computing final scores. The module composition of the application is detailed in Figure 7.

5.3 | Tool Evaluation

To evaluate TOAST and gather insights on its practical application, we applied *iterative usability testing*[31] to assess the overall usability of the graphical interface and the interaction patterns that were produced. This strategy is based on an iterative process in which, at each step, users give feedback about the tool's user interface—after executing a series of tasks—and developers modify the tool accordingly. We recruited five graduate students who attended and achieved a Human-Computer Interaction course during their degree. We asked them to conduct three tasks during each process iteration:

1. Start the TOAST bot in a Discord server and initiate the assessment process for a team member.
2. Add a new contributor to the team and then remove an existing contributor using the tool's team management features.
3. Answer the questions proposed for a team member using the provided interface to select responses.

[††] https://discord.js.org/

After each iteration, we directly interviewed participants to measure the tool's usability using well-known instruments. Specifically, we adopted the System Usability Scale[32,33], the Questionnaire for User Interaction Satisfaction[34], and the NASA Task Load Index (NASA-TLX)[35]. We kept iterating the usability evaluation until reaching saturation[31].

Based on the feedback provided in the 4 iteration of the process, we improved the tool as follows:

- **Redesigned The Response Selection Mechanism:** The original dropdown menu for selecting responses was replaced with clickable buttons. This change aimed to streamline the response process and reduce the cognitive load on users.
- **Dynamic Question Display:** We implemented a feature to remove the question text once answered. This modification was designed to reduce visual clutter and help users focus on the current question at hand.
- **Color-Coded Response Buttons:** To enhance the visual distinction between different response options, we implemented a color-coding system. "Disagree" and "Strongly Disagree" buttons were colored red, while "Agree" and "Strongly Agree" buttons were colored green. This change aimed to provide visual cues about the nature of each response and prevent the perception that all options held equal weight.
- **Conversation Cleanup:** A feature was added to automatically delete the conversation containing the responses once the assessment was complete. This enhancement was intended to maintain confidentiality and reduce potential bias in future assessments.

Thanks to the feedback provided, we reached the final user interface, shown in Figure 6.

6 | DISCUSSION, IMPLICATIONS, AND LIMITATIONS OF THE STUDY

In the following sections, we will discuss our research findings, their impact, and, ultimately, the threats to validity.

6.1 | Discussion

The analysis performed on the sample exposed in section 4 revealed that, despite the significance of community smells in effective team management, 80.86% of the whole sample and 78.95% only in the subset of managers were not familiar with them. Among all the managers familiar with community smells, 87.5% had experience managing distributed teams, thus indicating a potential correlation between these smells and distributed team management. We also found that from all the characteristics of the community smells, the managers perceived them mainly as a matter of communication issues but also as a lack of transparency and reluctance to share knowledge. It suggests that the technology transfer of this matter is in a very early stage, and thus, its recognition and management are difficult practices. Consequently, maintaining a healthy community becomes difficult, leading to a heightened risk of financial loss.

> **Key Finding 1** — *From the analysis performed, the majority of managers interviewed were not familiar with community smells. They perceive community Smells mainly as a matter of communication issues but also as a lack of transparency and reluctance to share knowledge.*

The outcomes of **RQ$_1$** illustrate that the current literature does not completely represent all behaviors contributors exhibit in the presence of community smells within software projects. This observation stems from the fact that specific actions and behaviors indicative of community smells, identified through insights provided by managers, are often not documented in existing literature. Out of the total ten symptoms identified, three were developed based on feedback from managers who regularly engage with their teams. These managerial insights have been crucial in refining the set of symptoms, thereby enhancing their utility in practical settings. For instance, the behavior *"The contributor hoards critical knowledge and does not share it"* was included based on suggestions from managers, addressing a common but frequently unreported issue that automated tools are likely to miss.

> **Key Finding 2** — *As an answer for* ***RQ_1****, we developed a comprehensive catalog of the symptoms behind the considered community smells (Table 4). Our study identified additional symptoms not previously documented in the literature, thereby advancing the current understanding and state of the art in community smell detection.*

The results and findings of **RQ_2** revealed how much each of the symptoms is considered relevant and indicative by managers to indicate the presence of a community smell. The results of this research question were the calculation of metrics based on the answers given in the multiple iterations of the questionnaire and a final score that serves as (1) a measure to understand how much each symptom is perceived as indicative of the smell presence and (2) to make actionable the results provided by the research. Analyzing Table 4 that contains all the data gathered, we can observe that the scores from Lonesome Architecting are the lowest. This finding can be motivated by the fact that from all the smells, this specific behavior of the contributor might be less present in teams and thus was not encountered often. From the table, we can also observe that all the scores of the symptoms uncovered by the survey depict them as relevant as the one elicited by literature, further assessing the validity of the feedback given by the managers.

> **Key Finding 3** — *In response to* ***RQ_2****, we evaluated the relevance of each identified symptom by calculating metrics based on manager feedback across multiple survey iterations. The results revealed how strongly each symptom is perceived as indicative of community smells, with the analysis confirming that symptoms identified through managerial insights are just as significant as those documented in the literature. This validation not only reinforces the practical applicability of our findings but also underscores the importance of integrating managerial perspectives into community smell detection.*

6.2 | Implications of the Study

While current research mainly focuses on detecting community smells through team-level interaction patterns, our study adds a new layer by identifying the symptoms of four specific community smells. This approach emphasizes issues that may stem from the behaviors or characteristics of individual team members, which has important implications for both research and practical applications.

Implications for Practitioners. Our research insights have been applied to develop `TOAST`, a proof-of-concept recommendation system that makes our findings accessible to practitioners, thus enhancing the technology transfer. Both the symptom catalog and `TOAST` have been developed to be complementary to the automatic analysis tool as described in Figure 1 since we envision that the combination of approaches can depict a clearer picture of the health status of a community, thus reducing the risk of unforeseen cost connected to bad community issues. Moreover, managers may also use our symptom catalog as a diagnostic checklist during performance reviews, team meetings, or project retrospectives without the need for a tool like `TOAST`, making our symptoms catalog also useful standalone. It is important to remember that *prevention is better than cure*, and prevention starts with education and awareness. According to Lehman's *"Conservation of Familiarity"* law[4], educating team members about the importance of healthy communication and collaboration is essential for the long-term success of software projects. Recognizing community smells through their symptoms not only helps in the early identification of issues but also serves as a critical learning opportunity. It raises awareness among team members about the impact of their interactions on project outcomes.

Implications for Researchers. We argue that *more behavioral literature needs to be conducted in the context of Community Smells*, and our study provides a foundation for further exploration into this topic. The motivation is that despite the presence of automatic tools based on mining strategies, some smells cannot be fully recognized without observing people's behaviors. The symptom catalog we developed offers new avenues for expanding the scope of community smell detection beyond the current literature. Future research could focus on identifying additional symptoms across various project types or team structures or on refining the metrics and methods used to quantify these symptoms' relevance. Moreover, the open-source nature of our tool, `TOAST`, provides researchers with a practical platform for testing and validating new theories or models in diverse contexts. The ability to replicate and extend our study using the materials provided in our online appendix further contributes to the development of more sophisticated and context-aware community smell detection tools, advancing the field of software engineering and offering richer, more actionable insights for both practitioners and scholars.

6.3 | Limitations and Threats to Validity

This section reports the limitations and threats of the study, as well as the strategies adopted to face them. From a qualitative perspective, we mainly relied on qualitative research's validity and reliability threat[36,37]. Moreover, we also faced our work against common threats in software engineering research[38].

Threats to Validity of the Research In terms of validity of the research[36], since we conducted a survey study and mainly qualitative research, we had to face researcher bias and respondent bias.

Regarding the possibility of negatively influencing the study by our own bias, i.e., research bias, we planned to adopt both triangulation and peer briefing. In terms of triangulation, our survey was based on and evaluated against the already published literature on community smells; this allowed us to confront our findings with already peer-reviewed literature continuously. No issues were encountered in this sense. Moreover, we continuously adopted peer briefing; the paper's first two authors confronted each other's findings and reported them to the research team after each iteration and before writing the final report. Such an approach provided helpful feedback and allowed us to avoid misconceptions of the data.

Respondent bias is a lasting friend of qualitative investigation. As a first note, we mitigated this threat by using the strategies mentioned above (i.e., triangulation and peer briefing). Moreover, we also adopted an iterative and prolonged approach (i.e., prolonged involvement) to ensure that our results were consistent among a plethora of participants and on time. Furthermore, as discussed in section 3, we designed the questionnaires with multiple attention-checking questions to capture unreliable data. Moreover, the combination of close- and open-ended questions allowed us to recognize situations in which the respondent behaved unethically or used other tools (e.g., Generative AI) to answer the questionnaire.

Threats to Reliability of the Research Reliability is intended as how much credible and reliable the work done and the report conducted are in terms of the process and data collected[37]. In order to ensure the reliability of our research, we adopted different approaches[37,38], most of which are common in other qualitative studies in software engineering. First, the process is described in detail in Section 3, and the rationale behind our choices is presented. Then, in the results Section, we included information on study participants and supported our claim with data (both graphical and textual), including participants' answers to open-ended questions. Last but not least, we included all the used material and results in a public online appendix[22].

Threats to Transferability of the Research Regarding the transferability of the results, i.e., the degree to which the study results can be transferred to other contexts or settings with other respondents[36], we tried to support it by selecting a representative sample—as described in Section 3—and involving a plethora of practitioners. As mentioned before, since our study involved studying the phenomena of community smells, we first searched for general practitioners and then for managers. This was because the sample study was conducted before starting with the research questions to depict a picture of the general state of practice. In contrast, the research questions were intended to deepen the origin of community smells, which were more related to a managerial perspective. Furthermore, we also took advantage of PROLIFIC to select a reliable and representative sample.

7 | CONCLUSIONS AND FUTURE WORKS

Our study introduces a complementary approach to detecting community smells, focusing on symptom recognition at the contributor level rather than the more common team-level interactions. We employed a mixed-method approach, combining a literature review with a practitioner survey, to identify and validate a catalog of symptoms associated with four specific community smells. The catalog provides managers with a practical resource for early intervention, complementing existing detection methods and enhancing overall community health management. To support this approach, we developed TOAST, a tool designed to assist managers in monitoring and addressing these issues.

Future research aims to expand the set of symptoms to cover a broader range of community smells and develop refined detection metrics based on feedback from managers. These advancements may also be integrated into tools like CSDETECTOR[21]. Additionally, enhancing TOAST by extending its functionality to platforms like Teams or Slack and incorporating dashboards for historical analysis would increase its utility. Finally, integrating the symptoms catalog into proactive management practices could help prevent community smells from emerging, thereby improving software development and maintenance processes.

References

1. Institute PM. *A Guide to the Project Management Body of Knowledge*. 7 ed., 2021.
2. Brooks Jr FP. *The mythical man-month: essays on software engineering*. Pearson Education, 1995.
3. Ralph P, Chiasson M, Kelley H. Social Theory for Software Engineering Research. In: EASE '16. Association for Computing Machinery 2016; New York, NY, USA
4. Lehman MM. On understanding laws, evolution, and conservation in the large-program life cycle. *Journal of Systems and Software.* 1979;1:213–221.
5. Gupta A, Sharma S. Software maintenance: Challenges and issues. *Issues.* 2015;1(1):23–25.
6. Ulziit B, Warraich ZA, Gencel C, Petersen K. A conceptual framework of challenges and solutions for managing global software maintenance. *Journal of Software: Evolution and Process.* 2015;27(10):763–792.
7. DeMarco T. *Controlling software projects: Management, measurement, and estimates*. Prentice Hall PTR, 1986.
8. Tamburri DA, Kruchten P, Lago P, Vliet Hv. What is social debt in software engineering?. *2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE).* 2013:93-96.
9. Fowler M. *Refactoring: improving the design of existing code*. Addison-Wesley Professional, 2018.
10. Tamburri D, Kruchten P, Lago P, Vliet Hv. What is social debt in software engineering?. *2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE).* 2013.
11. Caballero-Espinosa E, Carver C. J, Stowers K. Community smells—The sources of social debt: A systematic literature review. In:
12. Palomba F, Andrew Tamburri D, Arcelli Fontana F, Oliveto R, Zaidman A, Serebrenik A. Beyond Technical Aspects: How Do Community Smells Influence the Intensity of Code Smells?. *IEEE Transactions on Software Engineering.* 2021;47(1):108-129. doi: 10.1109/TSE.2018.2883603
13. Martini A, Bosch J. Revealing social debt with the CAFFEA framework: An antidote to architectural debt. In: IEEE. 2017:179–181.
14. Tamburri DA, Palomba F, Serebrenik A, Zaidman A. Discovering community patterns in open-source: a systematic approach and its evaluation. *Empirical Software Engineering.* 2019;24(3):1369–1417.
15. Tamburri DA, Palomba F, Kazman R. Exploring Community Smells in Open-Source: An Automated Approach. *IEEE Transactions on Software Engineering.* 2019;47(3):630-652. doi: 10.1109/TSE.2019.2901490
16. Catolino G, Palomba F, Tamburri DA, Serebrenik A, Ferrucci F. Gender Diversity and Women in Software Teams: How Do They Affect Community Smells?. *IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Society.* 2019.
17. Lambiase S, Catolino G, Tamburri DA, Serebrenik A, Palomba F, Ferrucci F. Good Fences Make Good Neighbours? On the Impact of Cultural and Geographical Dispersion on Community Smells. 2022.
18. Palomba F, Tamburri DA. Predicting the emergence of community smells using socio-technical metrics: a machine-learning approach. *Journal of Systems and Software.* 2021;171:110847.
19. Almarimi N, Ouni A, Chouchen M, Saidani I, Mkaouer MW. On the detection of community smells using genetic programming-based ensemble classifier chain. In: 2020:43–54.
20. Voria G, Pentangelo V, Della Porta A, et al. Community Smell Detection and Refactoring in SLACK: The CADOCS Project. *Information and Software Technology.* 2022;146:106853.
21. Almarimi N, Ouni A, Chouchen M, Mkaouer MW. csDetector: an open source tool for community smells detection. *ESEC/FSE 2021.* 2021. doi: 10.1145/3468264.3473121
22. Della Porta A, Lambiase S, Catolino G, Ferrucci F, Palomba F. Online Appendix - A Novel, Tool-Supported Catalog of Community Smell Symptoms.
23. Reid B, Wagner M, d'Amorim M, Treude C. Software Engineering User Study Recruitment on Prolific: An Experience Report. *arXiv preprint arXiv:2201.05348.* 2022.
24. Kitchenham BA, Pfleeger SL. Personal Opinion Surveys. In: , , Springer, 2008:63–92.
25. Andrews D, Nonnecke B, Preece J. Conducting research on the internet:: Online survey design, development and implementation guidelines. 2007.

26. Flanigan TS, McFarlane E, Cook S. Conducting survey research among physicians and other medical professionals: a review of current literature. In: . 1. 2008:4136–47.

27. Tamburri DA, Kruchten P, Lago P, Vliet H. Social Debt in Software Engineering: Insights from Industry. *Journal Loof Internet Services and Applications.* 2015. doi: 10.1186/s13174-015-0024-6.

28. Tamburri DA. Software Architecture Social Debt: Managing the Incommunicability Factor. *IEEE Transactions on Computational Social Systems.* 2019;6.

29. Wicks D. The coding manual for qualitative researchers. *Qualitative research in organizations and management: an international journal.* 2017;12(2):169–170.

30. Palomba F, Tamburri DA. Predicting the emergence of community smells using socio-technical metrics: a machine-learning approach. *Journal of Systems and Software.* 2021.

31. Genov A. Iterative usability testing as continuous feedback: A control systems perspective. *Journal of Usability Studies.* 2005;1(1):18–27.

32. Klug B. An overview of the system usability scale in library website and system usability testing. *Weave: Journal of Library User Experience.* 2017;1(6).

33. Grier RA, Bangor A, Kortum P, Peres SC. The system usability scale: Beyond standard usability testing. In: . 57. SAGE Publications Sage CA: Los Angeles, CA. 2013:187–191.

34. Maryland HCILUO. “QUIS: The Questionnaire for User Interaction Satisfaction.”. .

35. Hart SG, Staveland LE. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In: , , . 52. Elsevier, 1988:139–183.

36. Lincoln YS, Guba EG. *Naturalistic inquiry*. sage, 1985.

37. Robson C. Real world research: A resource for social scientists and practitioner-researchers. *Oxford, UK: Blackwell Publishers..* 2002.

38. Seaman CB. Qualitative methods in empirical studies of software engineering. *IEEE Transactions on software engineering.* 1999;25(4):557–572.