RESEARCH ARTICLE

# Comparison Based on SLR and Empirical Data: Factors Affecting Sustainability Aspects in Software Crowdsourcing

**Waqas Haider**[1,*], **Adnan Khan**[1], **Samreen Ihsan**[2] and **Ihsan Adil**[2]

[1] Software Engineering Research Group (SERG-UOM), Department of Computer Science and IT, University of Malakand, Chakdara 18800, KPK, Pakistan
[2] School of Software Technology, Dalian University of Technology, Dalian 116000, China

## Abstract

**Crowdsourced software development (CSD) has gained increasing popularity in software engineering. However, empirical evidence on its sustainability remains limited. Although previous studies have suggested several sustainability-related factors, their applicability in real industrial CSD environments lacks sufficient validation. Through a systematic literature review (SLR), we identified eleven factors affecting sustainability in CSD. This study empirically validates these factors via a questionnaire survey with practitioners from various crowdsourced software development platforms. The results confirm that all eleven factors identified in the literature are also recognized by practitioners, with no additional factors reported. A comparison of factor rankings between the SLR and the survey yields a moderate positive Spearman rank correlation of 0.4909, indicating general agreement between academic and industry perspectives, albeit with some differences in prioritization. These findings provide empirical support for sustainability in CSD and offer practical insights for researchers and practitioners to better design and manage sustainable crowdsourced software development projects.**

**Keywords**: crowdsourcing, crowdsourced software development (CSD), sustainability, Empirical study, systematic literature review (SLR).

## 1 Introduction

Crowdsourcing is a term that originated from the combination of "crowd" and "outsourcing", which means outsourcing a task or function to a large, undefined group of people, typically via the Internet. Jeff Howe and Mark Robinson were the first to discover the term crowdsourcing, which they used in an article for Wired magazine. He explains that crowdsourcing can be understood as an organizational approach that leverages the collective intelligence and skills of a large external crowd to perform specific tasks or generate innovative solutions [1, 2]. People use crowdsourcing

**Submitted:** 10 January 2026
**Accepted:** 11 March 2026
**Published:** 16 April 2026

**Vol.** 1, **No.** 1, 2026.
10.62762/TAIC.2026.980583

***Corresponding author:**
✉ Waqas Haider
whaidermkd@gmail.com

**Citation**
Haider, W., Khan, A., Ihsan, S., & Adil, I. (2026). Comparison Based on SLR and Empirical Data: Factors Affecting Sustainability Aspects in Software Crowdsourcing. *ICCK Transactions on Applied Intelligence and Cybernetics*, 1(1), 36–52.

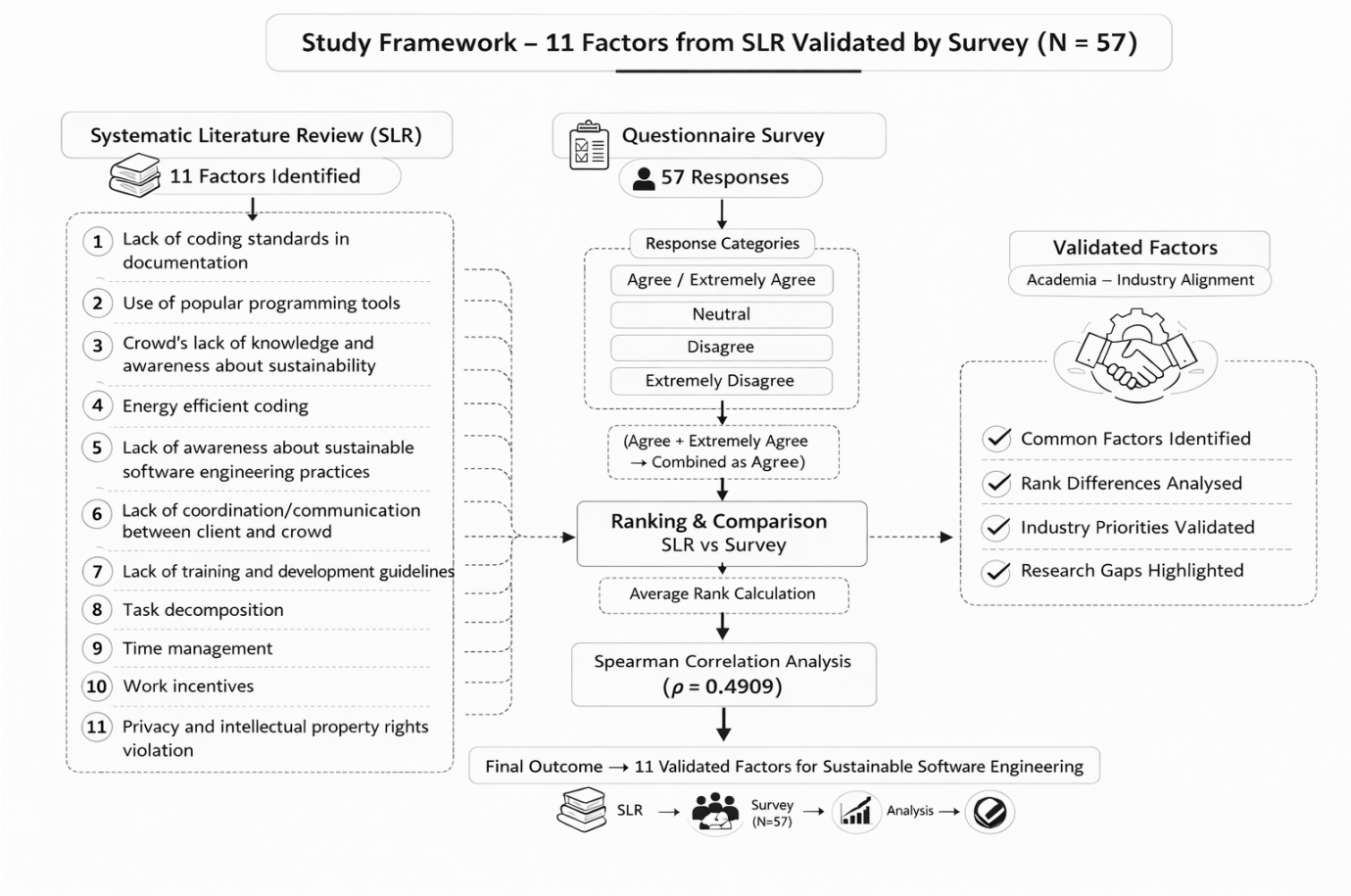


**Figure 1.** General framework of the study for identifying and validating sustainable software engineering factors.

by holding open innovation challenges, offering micro tasks, running competitions, and enabling digital platforms to collect solutions from a wide audience [3]. The concept behind crowdsourcing is that bringing together many different people can lead to better ideas or solutions than a narrower group of professionals [2].

Crowdsourcing is considered an approach to open innovation and solving problems. More and more companies are relying on crowdsourcing to encourage creativity and find solutions to problems in many fields [4]. Crowdsourcing helps in the discovery of novel solutions by using the collective intelligence of many people. Crowdsourcing initiatives that are well planned and executed have more positive results and fewer negative outcomes [5]. Numerous successful crowdsourcing platforms, such as Freelancer, TopCoder, Fiverr, AppStore, uTest, Mob4Hire, and TestFlight, rely on Crowdsourced Software Engineering (CSE). Various creative and design-oriented activities have been able to leverage the emerging distributed problem solving approach [6, 7]. CSE or SC is also becoming increasingly popular in both the industrial and academic worlds. Crowdsourcing has many benefits, but it faces several issues, such as collaboration with various staff that leads to management problems. This issue leads to wasted time because various teams generate similar strategies for the same tasks. Thus, in terms of privacy, security, and law enforcement processes, crowdsourcing is deemed inappropriate for some software projects [8, 9]. Several common problems associated with crowdsourcing include a shortage of skilled workers, communication barriers, low-quality submissions, concerns about confidentiality, difficulties in selecting the best solutions, and issues related to copyright [10].

However, many people in software engineering (SE) are already familiar with the broad definition of sustainability as the capacity to endure, and sustainable development as meeting the needs of the present without compromising the ability of future generations to meet their own needs" [11]. There is no comprehensive guide for sustainability, and there are many aspects of sustainability that can be observed from the perspective of software

engineering. The concept of sustainability in the context of software crowdsourcing may be understood in two ways: sustainable software development and software sustainability [12]. Sustainable software development involves incorporating environmental, social, economic, and technological sustainability considerations into the development process. The methods for sustainable software development encompass optimized coding practices, an energy-efficient coding approach, resource and optimization, and others [13, 14]. The main objective is to integrate sustainable software engineering standards into the software development process. Software sustainability means that the final product should be sustainable, taking into account its long-term impact and stability. It includes the overall sustainability of the software, including how long it will last. Both factors play a vital and difficult role in the longevity of software crowdsourcing [15, 16]. Recent results say 30% of projects finish on track, another 50% meet challenges, and 20% end in complete failure, which may harm software success and sustainability. With the emergence of new ideas regarding software crowdsourcing, standards are not focusing as much on ensuring the long-term sustainability of software. Consequently, the lack of criteria for long-term software crowdsourcing is recognized as a limitation in software standards. The purpose of this study is to determine key factors for sustainable software development within crowdsourcing. The generalized framework of the study is illustrated in Figure 1. In our earlier research, we used a systematic literature review approach [17], which resulted in a list of 11 factors, as presented in Table 1. A questionnaire survey was then conducted to test and validate the factors identified in this study with respondents from academia and industry. The responses collected were analyzed using ranking comparison and Spearman correlation analysis to evaluate the agreement between the findings of the literature and industry perspectives. The main aim of this article is to answer the following research questions.

RQ1: Which sustainability-related factors identified through the Systematic Literature Review are empirically validated in real-world Crowdsourced software development?

RQ2: To what extent do the empirically validated factors align with or differ from those identified through the Systematic Literature Review?

**Table 1.** List of the factors influencing sustainability aspects in CSD.

| S. No | List of Factors that Influence Sustainability Aspects in CSD |
|---|---|
| 1 | Lack of coding standards in documentation |
| 2 | Use of popular programming tools |
| 3 | Crowd lack of knowledge and awareness about sustainability |
| 4 | Energy efficient coding |
| 5 | Lack of awareness about sustainable software engineering practices |
| 6 | Lack of coordination/communication between client and crowd |
| 7 | Lack of training and development guidelines |
| 8 | Task decomposition |
| 9 | Time management |
| 10 | Work incentives |
| 11 | Privacy and intellectual property rights violation |

The remaining sections of this research paper are organized as follows. Section 2 discusses background and motivation, while Section 3 describes the research technique. Section 4 contains the results, Section 5 analyzes the discussion Finally, section 6 discusses study limitations, and Section 7 examines the conclusion and future work.

## 2 Related Work

The importance of software sustainability and software engineering for sustainability is becoming recognized as vital not just by scholars, but also by the software industry as a whole and standards groups. There has been an increase in the percentage of businesses that have incorporated sustainability into their entire business models. According to a Microsoft report and an IBM global chief executive officer (CEO) study on sustainability [18]. According to the findings of the Global Executive Study on Sustainability and Innovation, only 48% of the 4000 executives and managers from all over the world who were surveyed believe they were forced to make changes to their business models as a result of sustainability concerns [19].

König et al. [18] describes that traditional software engineering methodologies do not promote sustainability. Currently, there is no particular guidance for sustainability and the various aspects of sustainability that can be seen from the perspective of software engineering. This lack of support makes it hard to work on sustainability or makes it impossible to talk about this important idea at all. Working with

modern crowdsourcing software presents a challenge for organizations in managing work distribution, crowd selection, and collaboration effectively The sustainability of the newly created software was challenged [20–22]. In this case, the result is an infinite software development style when tasks are assigned to the crowd. The standardization of software and its sustainability is challenging and the inclusion of distributed tasks among the crowd. Currently, researchers and practitioners do not have a single source to find and understand sustainability aspects [23, 24]. According to sustainable software engineering, sustainability involves creating software products that are designed for long-lasting systems and meet the requirements of current and future generations. This involves integrating the four sustainability aspects (environmental, economic, social, and technical) to meet the requirements on time [25]. There is a lack of sufficient attention towards software sustainability aspects, which underlines the importance of using software crowdsourcing to effectively control and monitor these aspects.

Similarly, Ahmad et al. [25] stated that the IEEE and ISO standards do not pay enough attention to sustainability guidelines for the latest crowdsourcing application models, as well as for collaborative and globally distributed software development practices. This is even though the standards emphasize sustainable development practices. Consequently, this will be a lack of sustainable development criteria towards crowdsourcing applications and, therefore, will lead to software failures. There is no definitive resolution to the issues that are still unresolved in the current structure of crowdsourcing applications that have not been considered in scholarly reviews. Neglecting these issues may pose significant risks of software failure. According to Malik et al. [26], sustainability issues are in high demand, as evidenced by the United Nations' environmental goals for software development and crowdsourcing. The disregard of relevant critical phenomena during the early design phase may influence the best design of a software system, which may cause possible software failures. In addition, more research is required in this field to provide insight to software developers and project managers. Organizations can find it challenging to incorporate sustainability in software development without having standardized practices. Although the literature on sustainability-conscious software engineering and crowdsourcing is increasing, available research is mainly in conceptual debates or single issues and is not properly validated through empirical research and comparative studies on the results of research in the literature with those in the industry. Specifically, there is no systematic evidence that analyzes the correspondence of sustainability factors discovered in the literature with actual practices of Crowdsourced software development.This paper fills this gap by combining a Systematic Review of the literature with an empirical questionnaire survey to confirm and contrast sustainability factors on an academic and industrial basis.

## 3 Methodology

This research was conducted in two stages using two different methodologies: a systematic literature review (SLR) and a questionnaire survey. At the initial stage, we had an SLR. We performed an SLR to identify the factors that could affect the aspect of software sustainability in crowdsourced software development. Our second step was the empirical study, which was a questionnaire survey in the crowdsourced industry. The survey aimed to empirically confirm the findings obtained with the help of the SLR, as well as to determine whether there are other factors that we might have missed in the first stage, that is, the SLR. The same method has been applied by several authors [27, 28].

### 3.1 Systematic literature review

A Systematic Literature Review (SLR) is a research process that is designed to identify, evaluate, and integrate existing research on a specific research topic. Unlike a traditional literature review, SLR follows established and standardized processes that encourage transparency, precision, and reliability. As a rule, an SLR is performed in three primary steps: planning the review, in which research questions, scope of the review, and narrowing see criteria are established; conducting the review, during which a properly designed search string is applied to various databases, and reviewed studies are filtered and selected through systematic screening and evaluation; and reporting the review, in which the results are analyzed, synthesized, and described clearly. This systematic methodology ensures comprehensive literature coverage and produces reproducible and valid research results. A structured Boolean search string based on keywords related to factors/challenges, sustainability, crowdsourcing, and software was developed. The search initially identified 3,424 studies, which were then filtered using predefined inclusion and exclusion criteria.

("Factor" OR "challenge") AND ("sustainable" OR "sustainability" OR "green") AND ("crowdsourcing" OR "Crowdsource" OR "Crowdsourced") AND ("Software")

#### 3.1.1 Inclusion criteria

The inclusion criteria were used to determine the piece of literature (papers, technical reports, gray literature, etc.) that was found using the search term and would be utilized to extract data. Only publications that were written in English were chosen. The criterion will be enumerated as follows.

- IC1: Studies that establish factors that lead to sustainability in any of the long-term stages.
- IC2: Studies of the issues and questions of creating sustainable software in a crowdsourcing environment.
- IC3: Literature addressing the sustainability factor of crowd-sourced software development.
- IC4: literature on the criteria for sustainable software development in crowdsourced software development.

#### 3.1.2 Exclusion criteria

This section is used to exclude studies that were retrieved from search terms, and hence this section defines exclusion criteria.

- EC1: Studies Irrelevant to the objectives of the study.
- EC2: Article not about sustainability factors or troubles/issues in crowdsourced software development.
- EC3: Research conducted in environments other than crowdsourcing.

This screening process helped us identify 84 articles that were first shortlisted to undergo further screening. Through a thorough examination process, we narrowed them down to 45 papers that will be included in the study. To resolve the possible absence of relevant studies, we also used a snowballing method by examining the references of the selected articles as a way of establishing any other pertinent studies that might have been overlooked during the initial search. This strict procedure ensured that the final compilation of the articles utilized in the SLR was thorough and very applicable to the research objectives. Table 2 provides various databases that were searched in this process.

### 3.2 Empirical study

We have surveyed the crowdsourcing software development industry to validate SLR results. The use of the survey method is something that we have used in order to exploit the resources available to us and the high number of participants of different nationalities. A survey was designed to collect the information [29, 30]. To obtain information on different crowdsourcing software development developers, experts and managers, we had to fill in an online questionnaire survey. The online questionnaire survey is an efficient means of obtaining information from people on the topic of crowdsourcing to help with research. A survey of the industry has been carried out following the SLR, a comparable research method applied by other scholars [31, 32]. Our questionnaire was handled using Google Forms, which is a free survey design and delivery system. The purpose of this study was to confirm the results of the SLR and to find other factors that had not been previously obtained. In the following sections, we will discuss the steps to be followed in carrying out the survey.

#### 3.2.1 Questionnaire Design

To assess the SLR results, we conducted a questionnaire survey at the University of Malakand. We also searched for any other factors that are not on our list. The questionnaire survey uses the SLR as input. It is separated into three segments. Section 1 summarizes the research conducted to provide developers with general knowledge of the topic. Demographic details on crowdsourcing developers and employees are gathered in Section 2, whereas the factors that have been determined by SLR are described in Section 3. The use of factors was used as a basis for obtaining quantitative data from crowdsourcing developers. The value of the criteria was evaluated on a five-point Likert scale. We used open and closed-ended questions to collect data. To carry out our study, we have requested the participants to rank all factors on a five-point Likert scale. The options are as follows (Extremely Agree (EA), Agree (SA), Neutral (N), Disagree (D), and extremely Disagree (ED). To make them respond to this, we requested the crowdsourcing developers to decide whether they believed there were additional factors that needed to be reviewed on top of the ones mentioned above so that we could get their response. The questionnaire was verified by experts from the University of Malakand Software Engineering Research Group (SERG-UOM). Their questions were answered after the questionnaire survey was tested.

**Table 2.** Summary of papers retrieved from different databases for the SLR study.

| S. No | Venue name | Search results | Initial selection | Final selection |
|---|---|---|---|---|
| 1 | IEEE Xplore | 158 | 4 | 1 |
| 2 | Science Direct | 1113 | 15 | 0 |
| 3 | ACM | 983 | 11 | 3 |
| 4 | Springer Link | 744 | 10 | 1 |
| 5 | Wiley | 107 | 4 | 1 |
| 6 | Google Scholar | 319 | 40 | 12 |
| 7 | Snowballing papers | – | – | 27 |
| **Total papers** | | **3424** | **84** | **45** |

**Table 3.** Summary of professional crowdsourcing groups.

| S. No. | Group Name | Channel | Members |
|---|---|---|---|
| 1 | Crowdsourcing Sustainability | LinkedIn | 1,392 |
| 2 | Crowdsourcing.org | LinkedIn | 20,145 |
| 3 | Crowdsourcing – Freelance Recruiter (BDE) | LinkedIn | – |
| 4 | CROWDSOURCING WEEK: Global Crowdsourcing Network | LinkedIn | 1,626 |
| 5 | Iraqi Crowdsourcing Group | LinkedIn | 17,987 |
| 6 | Global EHS & Sustainability Software Partners (Moderated by Benchmark ESG Gensuite) | LinkedIn | – |
| 7 | Model-Based Software Engineering (MBSE) | LinkedIn | 12,643 |
| 8 | IEEE Technical Council on Software Engineering | LinkedIn | 623 |
| 9 | Global Software Engineering (Remote Work) | LinkedIn | 27 |

The questionnaire form is filled out and is found in Appendix B.

### 3.2.2 *Designing an online questionnaire survey*

In the case of a survey, we will have two stages of sampling, that is, searching for appropriate people and questionnaire design. The sample is an extract of a larger population whose characteristics can be analyzed so that one can know more about the entire population. Regarding individuals, it can be described as a sample of respondents who have been selected among a large number of people to take part in a survey, such as presidents, professors, or students. In order to use the questionnaire approach, participants (the sample) are required to answer a questionnaire consisting of a set of questions. All these issues are elaborated in the following sections.

### 3.2.3 *Samplings*

Sampling can be achieved by two methods: systematically and non-systematically. In small-scale surveys, it utilizes a non-systematic method as a list of the entire population is at hand, and samples are selected based on the findings of the statistical analysis [33]. Our survey was implemented by a non-systematic approach because we could not easily reach many companies, find all members of the company, and select members from the list. Other researchers and scholars have used the same methodology [34]. To select individuals for the non-systematic sample, we used a specific method. During the first stage of the questionnaire delivery plan, an invitation letter and a summary of the research were developed and sent to the following listed website which is shown in Table 3.

1. LinkedIn Groups (www.linkedin.com).
2. Whatsapp Groups (whatsapp.com).
3. The published papers contained email addresses and an invitation to participate was also sent to the authors of the industrial articles selected by the SLR.

### 3.2.4 *Executing Online Questionnaire Survey*

We sent our questionnaire survey to multiple authors of the academic publication we chose for our systematic literature review methods. Also, distribute our questionnaire survey on numerous crowdsourcing sites, which is shown in Table 4. During this poll, we

collected 63 responses.

**Table 4.** Crowdsourcing Platforms Used in the Study.

| S. No | Platforms |
|---|---|
| 1 | TopCoder |
| 2 | Upwork |
| 3 | Fiverr |
| 4 | Amazon Mechanical Turk |
| 5 | UTest |
| 6 | Freelancer |
| 7 | Appstori |

### *3.2.5 Data Screening and Quality Control*

To verify the precision of the survey findings, all submitted questionnaires were used to assist in identifying low-quality responses, which would be eliminated according to our criteria. Which are listed below.

- The inclusion criterion focuses on participants based on their CSD experience, particularly how long they have worked in crowdsourced software development and how many projects they have completed.
- A questionnaire survey that is incomplete or inadequately completed.
- Duplicate responses are not acceptable.
- More than 63 responses were sent from participants from all over the world, which is shown in Figure 2. When we applied our quality requirements, we rejected five responses. The remaining 57 responses provided information for analysis.

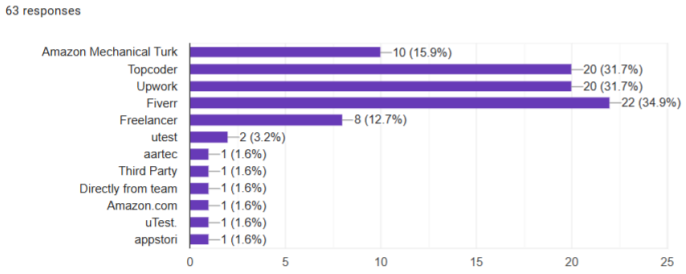


**Figure 2.** Respondent responses form QS.

### *3.2.6 Data analysis*

We used frequency analysis to evaluate data from the systematic literature review (SLR) and the questionnaire survey (QS). This is because the data are qualitative.

### *3.2.7 Frequency analysis*

Descriptive data are best studied using frequency analysis, which is also popular for organizing both quantitative and qualitative information. Working with a frequency distribution table allows you to count how often the data appear and see its overall spread. Various frequencies can help identify similarities and differences between variables within or between different groups, and can be used with either kind of data [35]. The majority of the data analysis performed in this study has been based on frequency analysis. Therefore, to examine every factor we found, we examined the presence of each factor in each response to the questionnaire. Every factor was recorded for every article using the Systematic literature review (SLR) study as the source for our sample of publications. The researchers compared the frequency of each factor with other factors to understand how important each was.

### *3.2.8 Application of frequency analysis in this research*

Researchers can first code qualitative data in empirical research, then gather quantitative data for statistical analysis [36]. In this research, all data collected from SLR and questionnaire surveys are coded and categorized to allow frequency analysis and other comparison studies of CFs related to CSD. We perform a frequency analysis at two different levels. To begin with, we counted the number of matching factors revealed during the SLR procedure. Each article mentioned the incidence of a significant factor. To determine the relative relevance of each component, the total number of occurrences in several articles was compared to the occurrences of other relevant factors in the same articles. For example, if factor F is cited in 22 out of 44 articles in the literature, it has a 50% significance level for comparison. This is the strategy that we have employed to compare the factors and rank them. In addition, we obtained the prevalence of the given factors in the survey. We recorded the incidence of factors in each research questionnaire to review the CFs derived from the questionnaire survey. The comparative importance of the different factors is determined by comparing the responses of a factor in all completed questionnaires and the responses of other identified factors in the same questionnaires. Finally, a conclusion has been reached regarding the factors that are significant in CSD studies and surveys.

## 4 Results

The results of the questionnaire survey are presented here to respond to Research Question RQ1. The

**Table 5.** Questionnaire Survey, expert feedback about Success factors.

| S. No | Factors | Positive | | | Negative | | | Neutral | |
|---|---|---|---|---|---|---|---|---|---|
| | | Ext. Agree | Agree | % | Ext. Disagree | Disagree | % | F | % |
| 1 | Lack of coding standards in documentation | 35 | 19 | 95 | 0 | 1 | 2 | 2 | 3 |
| 2 | Use of popular programming tools | 23 | 29 | 91 | 0 | 2 | 3 | 3 | 5 |
| 3 | Crowd lack of knowledge and awareness about sustainability | 28 | 19 | 82 | 1 | 1 | 4 | 8 | 14 |
| 4 | Energy-efficient coding | 18 | 33 | 89 | 0 | 2 | 4 | 4 | 7 |
| 5 | Lack of awareness about sustainable software engineering practices | 28 | 21 | 86 | 0 | 2 | 4 | 4 | 7 |
| 6 | Lack of coordination/communication between client and crowd | 24 | 14 | 67 | 2 | 6 | 14 | 11 | 19 |
| 7 | Lack of training and development guidelines | 32 | 17 | 86 | 0 | 1 | 2 | 7 | 12 |
| 8 | Task decomposition | 17 | 22 | 68 | 1 | 5 | 11 | 12 | 21 |
| 9 | Time management | 23 | 16 | 68 | 1 | 5 | 11 | 12 | 21 |
| 10 | Work incentives | 24 | 20 | 89 | 1 | 2 | 5 | 3 | 6 |
| 11 | Privacy and intellectual property rights violation | 30 | 19 | 86 | 1 | 3 | 7 | 4 | 7 |

information collected from the survey is shown in Table 5, which we analyze to address RQ1. It is important to mention that in our earlier research, we found 11 important factors overall, and 6 of them were chosen as critical factors essential to the sustainability of crowdsourced software development. We determined a factor as critical on the basis of its frequency in the literature gathered through the Systematic Literature Review (SLR) process. If a factor has a frequency percentage of 30% or more, we rated it as critical, and Other scholars have used a similar method [25, 37, 38].

**F1. Lack of coding standard in documentation:** A lack of coding standards in documentation can have a detrimental influence on software development's sustainability. The lack of proper documentation can lead to misunderstandings among developers and complicate the maintenance and updating process of the program. Consequently, there could be inefficient software development and the economy and the environment would be affected. Failure to have good documentation may make developers work longer and spend more money on correcting errors. Software bugs or errors are common when documentation is not performed to the standard, which negatively impacts the environment due to increased energy use and discarded hardware. Standardizing coding practices in the documentation of software projects helps organizations improve sustainability, ease the development process, and reduce the chance of errors [39, 40].

**F2. Using popular programming tools:** The programming tools commonly used can have a beneficial or negative effect on sustainability in the development of software. However, it is worth mentioning that these tools have large user communities, and detailed documentation will help developers adopt sustainable practices. In addition, they can also promote cooperation and knowledge flow among developers, improving the effectiveness of development efforts, and reducing time and money. However, the presence of mainstream programming tools can also pose a detriment to sustainability, in particular in terms of environmental sustainability. Many of these tools are highly computationally expensive, have high energy use, and emit carbon emissions. Furthermore, dependence on these well-known tools can inhibit innovation in the development of more ecological alternatives and limit the variety of software available for programming [41].

**F3. Lack of knowledge and awareness about sustainability:** The Lack of knowledge and awareness of sustainability is widely identified as a major factor hindering the achievement of established goals. Many researchers argue that insufficient information and limited understanding make it difficult to implement sustainability oriented strategies effectively. Developers working in the software development industry can understand the impact that software developers have, since they are always aware of the issues that arise. Lack of understanding can lead to unnecessary code investments, missed reuse opportunities, and hinder cross-coordination and progress [42]. The lack of knowledge and awareness has a significant impact on the social aspects of sustainability. It may result in misunderstanding and participation in sustainability issues, affecting progress toward sustainability goals. Increasing awareness of sustainability and its significance to software development can help overcome this barrier. Developers can help build a more sustainable future by supporting educational and training efforts focusing on sustainability [43, 44].

**F4. Energy-efficient coding:** Energy-efficient coding is a key feature of sustainable software development that contributes to improving different aspects of sustainability. Developers can lower the amount of energy and carbon emissions produced by software programs by improving their code's efficiency in saving energy. Energy-efficient software development reduces program operating cost and the need to maintain equipment, which helps the economy. Furthermore, by adopting energy-efficient coding, we can also contribute to sustainability in society, since users can enjoy smoother and more efficient application experiences. In general, the use of energy-efficient coding is generally good for the environment and the economy [45].

**F5. Lack of awareness about sustainable software engineering practices:** The lack of awareness about sustainable software engineering practices is a notable factor in implementing sustainable software development. Developers who are unfamiliar with sustainable software engineering practices may make environmentally and socially harmful software decisions. As a result, organizations can waste energy, use resources inappropriately, and fail to take advantage of opportunities for innovation [18]. Most sustainability issues affecting the social side result from lack of awareness about sustainable software engineering practices about sustainable software engineering. That means fewer people pay attention to sustainability problems, making it harder to achieve the goals. We should inform others and build additional programs that teach the method of building software responsibly. Developers who care about the planet and sustainability when developing software make products that save the environment and help everyone [57].

**F6. Lack of coordination and communication between clients and crowds:** Poor coordination and communication between clients and crowds is an important factor that influences sustainability in crowdsourced software development. This problem can significantly undermine the effectiveness and stability of such long term development models. The different objectives and the lack of organization of the communication between the client and the crowd can result in discrepancies in the project and its subsequent delays, redundancies, and degradation of the overall quality of the final product [46]. When clients and crowds do not cooperate or communicate, The role of the economy in sustainability is most affected. It may mean that resources are wasted, expenses increase, and chances for partnering and developing new ideas are ignored. Strong communication and cooperation between clients and the crowds should be put in place to solve this challenge. To help with this, people should be open with their ideas, know what is expected, and make use of technology that makes working together and giving feedback simple. If the coordination and communication between clients and crowds are improved using software, the process of software development improves, makes more sense, and is less expensive [47].

**F7. Lack of training and development guidelines:** A major problem in building software sustainably is the lack of well-defined training and development guidelines. Without clear training, developers could miss out on the knowledge needed to solve social and environmental problems. Such actions waste resources, use more energy, and prevent the opportunity to improve ideas [48]. The lack of training and development guidelines mainly influences the social part of sustainability. Sometimes people do not know enough about sustainability, which can get in the way of making progress on sustainability. To address this problem, it is important to create training guidelines that focus on sustainability. The topics of these guidelines can include energy efficiency, minimizing waste, and being socially responsible. If developers follow guidelines and get training for

sustainable development, their software will be good for both nature and society, helping to plan sustainable software [49].

**F8. Task decomposition:** Task decomposition is the process of decomposing more complex work in software development into smaller, easier-to-understand steps. The breakdown of jobs enables developers to determine what should be done and what depends on it, which is easy to manage and monitor. The work must be divided so that the appropriate members of the team are assigned the work that suits them and fits their schedule [50]. Task decomposition can help with the economic aspect of sustainability. Working in this manner allows developers to more easily detect and correct problems, resulting in the use of fewer resources and higher efficiency. It may also guaranty that work and resources are distributed, allowing managers to keep track of and control the entire process more effectively. Organizations can improve their economics and prevent unnecessary or inefficient efforts by using task decomposition throughout development [51].

**F9. Time management:** Time management is crucial for the long term growth of software services. Managing your time helps developers execute tasks without delays and generate higher-quality output. Managing your time means properly ranking your tasks properly, assigning reasonable deadlines, and checking your progress [52]. Using effective time management, we can help improve the economic side of sustainability. Increasing the way time is used allows developers to complete tasks faster with fewer resources, which results in lower expenses and higher productivity. Proper time management means work will be done on schedule, reducing the risk of delays and boosts customer satisfaction. In addition, good time management can help achieve the social and environmental goals related to sustainability. Developers can reduce the amount of energy and resources needed for development by doing tasks efficiently. Effective ways of managing time can reduce the workload of developers and make the company more socially sustainable [53].

**F10. Work incentives:** Work incentives in software development typically make staff members happier, more willing to accept new approaches, and more productive. We work to sustain the community and economy by giving the right motivation to work with rewards; This is accomplished by improving the work environment and reducing staff turnover rates; Rewards can drive people to provide new content that improves software applications. We must ensure that work benefits remain unchanged despite crises in the economy for society to remain stable. Long-term software development initiatives require inspiring incentives [54].

**F11. Privacy and intellectual property rights violation:** A violation of intellectual property rights and privacy in software development has the potential to affect both the social and economic sustainability. Software companies can lose customers and have unpredictable profits if they violate privacy laws or take intellectual property [55]. Such violations might also result in legal action, which can be costly and harmful to a company's reputation. Protecting these rights is critical for fostering sustainability in software development, especially in the social and economic dimensions. Companies that prioritize the protection of privacy and intellectual property rights may develop trust with their consumers and foster economic stability [56].

### 4.1 Analysis of Sustainability Factors from SLR and Survey

To address RQ2, we compare the results of the SLR and the questionnaire survey. This kind of comparative analysis aims to establish similarities and differences between the success factors achieved using the two sets of data. Table 6 gives a summary of the comparisons of the success factors between the two sets of data.

Here, no classification is made of the SLR data, but the survey data is divided into the categories Agree, extremely Agree, Disagree, extremely Disagree and Neutral to be compared with the results of the systematic review of the literature presented in Table 5. The survey results for Agree and Extremely Agree were combined into a single category labeled Agree.

The components have been ranked based on their frequencies in ascending or descending order; factors with the highest frequency values have been ranked starting with 1, then 2, and so on. Whenever the same frequency values occur for two or more factors, the average rank is assigned using the formula:

$$R_{\text{avg}} = \frac{R_1 + R_2 + \cdots + R_k}{k} \tag{1}$$

For example, a frequency value of 89 was obtained for the factors "Energy efficient coding" and "Work incentives" as described in Table 6, so both were assigned an average rank of 3.5 $[(3+4)/2]$. Similarly,

**Table 6.** Comparison of Factors across SLR and questionnaire survey.

| S.No | Factors | SLR (n=45) | | Questionnaire Survey (n=57) | | Average Rank |
|---|---|---|---|---|---|---|
| | | % | Rank | % | Rank | |
| 1 | Lack of coding standards in documentation | 51 | 1 | 95 | 1 | 1 |
| 2 | Use popular programming tools | 47 | 2 | 91 | 2 | 2 |
| 3 | Crowd lack of knowledge and awareness about sustainability | 40 | 3 | 82 | 8 | 5.5 |
| 4 | Energy efficient coding | 38 | 4 | 89 | 3.5 | 3.75 |
| 5 | Lack of awareness about sustainable software engineering practices | 36 | 5 | 86 | 4 | 4.5 |
| 6 | Lack of coordination/communication between client and crowd | 31 | 6 | 69 | 9 | 7.5 |
| 7 | Lack of training and development guidelines | 22 | 7 | 86 | 5 | 6 |
| 8 | Task decomposition | 20 | 8 | 68 | 10.5 | 9.25 |
| 9 | Time management | 17 | 9 | 68 | 10.5 | 9.75 |
| 10 | Work incentives | 13 | 10 | 89 | 3.5 | 6.75 |
| 11 | Privacy and intellectual property rights violation | 9 | 11 | 86 | 6 | 8.5 |

the factor "Lack of awareness about sustainable software engineering practices" was assigned rank 5. This ranking mechanism was applied to all factors in the data set.

To identify potential new factors, open-ended questions were included in the survey (Section 3). However, no new factors were found, so Table 5 shows that both the SLR and survey datasets contain the same number of factors. Additionally, the empirical results in Table 5 confirm that no factor in the survey dataset had a frequency of zero. A small difference is observed when comparing factor rankings between the two datasets; for example, "Energy efficient coding" is ranked 4 in the SLR dataset and 3.5 in the survey dataset.

### 4.2 Statistical Analysis

Statistical analysis was conducted to compare the ranking of the factors identified through the SLR and the questionnaire survey, quantifying the degree of agreement between the academic literature and perceptions of practitioners.

By analyzing the rank values of the two datasets using the Spearman rank correlation, we established a correlation coefficient of 0.4909. The Spearman rank correlation coefficient ($\rho$) was calculated using the following formula:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \tag{2}$$

where $d_i$ denotes the difference between the rank of a factor in the SLR dataset and its corresponding rank in the survey dataset ($d_i = R_{\text{SLR}} - R_{\text{Survey}}$), $d_i^2$ is the square of this difference and $n$ represents the total number of factors.

For the 11 factors in this study ($n = 11$), the sum of the squared rank differences was:

$$\sum d_i^2 = 115$$

Substituting into the formula:

$$\rho = 1 - \frac{6 \times 115}{11(11^2 - 1)}$$

$$\rho = 1 - \frac{690}{1320}$$

$$\rho \approx 0.4909$$

The moderate positive Spearman rank correlation of 0.4909 between the SLR and the survey results indicates a significant, though not entirely convincing,

agreement between the two datasets. The degree of correlation indicates that although the relative significance of the factors remains mostly similar in both academic literature and practice, there is a certain deviation. This medium degree of correlation suggests an average degree of validation strength. The factors found in the literature are widely validated by the survey data, although the differences in order (e.g., between the "Energy efficient coding" factor) suggest possible specifics in real-life implementation. The alignment between academia and industry is implied, but the dissimilarity highlights the significance of circumstances and real-world issues that industry professionals have to face. Consequently, the correlation provides a reasonable degree of confidence in the identified factors and indicates areas where additional research might be required to address minor mismatches between theoretical and empirical viewpoints.

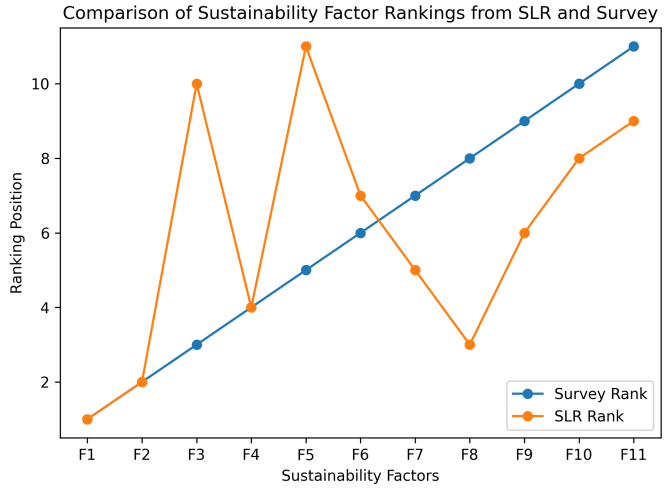


**Figure 3.** SLR and Q Survey Ranks Correlation scatter plot of Success Factors.

Figure 3 represents this relationship by the scatter plot of the SLR and the survey ranks. The findings indicate that the 11 factors are shared by both datasets (Table 6), which means that there is a significant overlap between the factors reported by the literature and the evidence that they are supported by empirical studies. The two datasets are similar rather than different, though some minor differences in their relative rankings are observed. Thus, when answering RQ2, the results indicate that the sustainability factors defined in the literature are generally in line with industry perceptions, though with minor gaps that are caused by practical and context-related factors and not deep-seated disagreement.

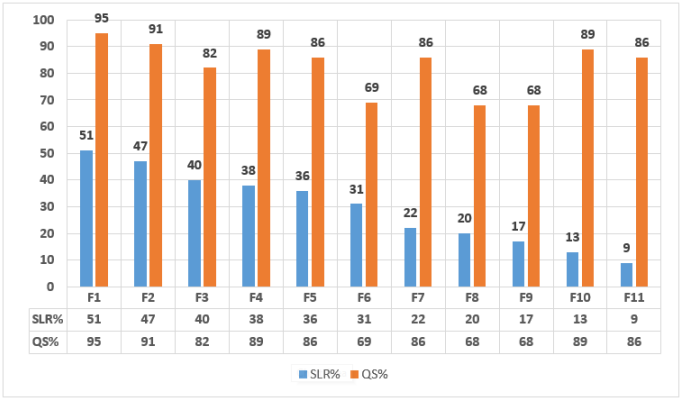


**Figure 4.** Comparisons of SLR and Questionnaire Survey (QS) results for the Factors.

The graphical comparison between the findings of the Systematic Literature Review and the questionnaire survey is presented in Figure 4. The figure demonstrates that all of the Factors listed by SLR for sustainable software development must be addressed by CSD developers, experts, testers, and managers, as confirmed by the survey findings.

Table 7 and Figure 5 show that some factors positively influence sustainability, such as popular programming tools, task decomposition, time management, and work incentives, while others have negative effects, including lack of coding standards, low awareness of sustainability, poor coordination, energy-inefficient coding, and privacy or IP issues.

**Linking the identified factors to sustainability aspects**

Table 1 is a summary of the eleven factors that were found in the SLR and the empirical analysis. To promote conceptual clarity, Figure 4 was created to show the relationship between these determined factors and the sustainability dimensions they affect. Figure 4 indicates that each of the 11 factors supports various aspects of the environmental, economic, social, and technical aspects of software development within crowdsourcing. This conceptual mapping gives a better understanding of the influence of each factor on sustainability outcomes.

## 5 Discussion

This research study adds experimental knowledge to the understanding of sustainability in crowdsourced software development (CSD) through methodically comparing sustainability factors that have been illustrated in the previous literature and those that have been reported by practitioners. The results indicate a high correlation between theory and

**Table 7.** Factors Influencing Sustainability Aspects in Crowdsourced Software Development.

| S. No. | Factors | Impact on Sustainability Aspect (+ve / –ve) |
|---|---|---|
| 1 | Lack of coding standard and documentation | –ve |
| 2 | Use of popular programming tools | +ve |
| 3 | Crowd lack of knowledge and awareness about sustainability | –ve |
| 4 | Energy-efficient coding | +ve |
| 5 | Lack of awareness about sustainable software engineering practices | –ve |
| 6 | Lack of coordination/communication between client and crowd | –ve |
| 7 | Lack of training and development guidelines | –ve |
| 8 | Task decomposition | +ve |
| 9 | Time management | +ve |
| 10 | Work incentives | +ve |
| 11 | Privacy and intellectual property rights violation | –ve |

practice, as all 11 sustainability factors revealed by the systematic literature review (SLR) were confirmed by the practitioners, and no other factors were reported by the survey. This overlap implies that the literature is comprehensive enough to define the main sustainability factors that are experienced in real life CSD settings. Therefore, the findings support the theoretical soundness and empirical significance of the sustainability factors that were claimed in earlier studies. Even with this overall agreement, the moderate positive Spearman rank correlation shows some partial divergence in the priority of these factors attributed by practitioners. Although the literature highlights the importance of factors like coding standards and sustainability awareness, there exist differences in the priorities of factors such as incentives, coordination and task decomposition among practitioners. Such differences seem to be subjective to context, organizational and operational constraints of respective individual projects. Notably, the identified divergence does not remove or eliminate the validity of the listed factors instead, it indicates the uncertainty of the CSD practice, in which practical aspects influence the manner in which the principles of sustainability are transmitted. The fact that there is theoretical compatibility and practical incompatibility underscores the need to have flexible and contextually responsive sustainability strategies in the CSD. Sustainability management cannot be developed based on prescriptive models alone, but should consider technical, human, and organizational levels. This study is a bridge between theory and practice by using secondary evidence presented in the literature and empirical verifications presented by practitioners. The results can therefore be useful for the researcher in further refining sustainability frameworks, platform developers in developing supportive mechanisms, and project managers to improve the long-term sustainability of crowdsourcing software projects.

## 6 Limitation

A survey-based questionnaire technique was used to experimentally investigate the factors that influence sustainability aspects in software crowdsourcing. The research involved 57 developers/experts from 24 different countries. Surveys are frequently criticized for having a poor response rate and a high risk of subjective bias. The survey findings represent the perspectives of the participants on the phenomenon being examined. According to the literature, the survey findings may be inaccurate and differ from the real distribution in the population. We aimed to study the perspectives and experiences of CSD experts, but we could not directly verify them. It is possible that the perceptions and observations of practitioners may not be accurate. Additionally, it is important to note that respondents who completed the questionnaire survey were not randomly selected but chose to participate of their own accord. However, the results of the pilot study demonstrate a high level of internal validity. The systematic literature review and piloting of the survey questions contributed to the comprehensive collection of variables in this research study. To ensure external validity, data were collected from a sample of 63 experts, including 57 experts from 24 different countries, which is a sufficiently diverse and representative sample. We utilized various online social and professional networking groups to contact foreign experts, employing all available means.

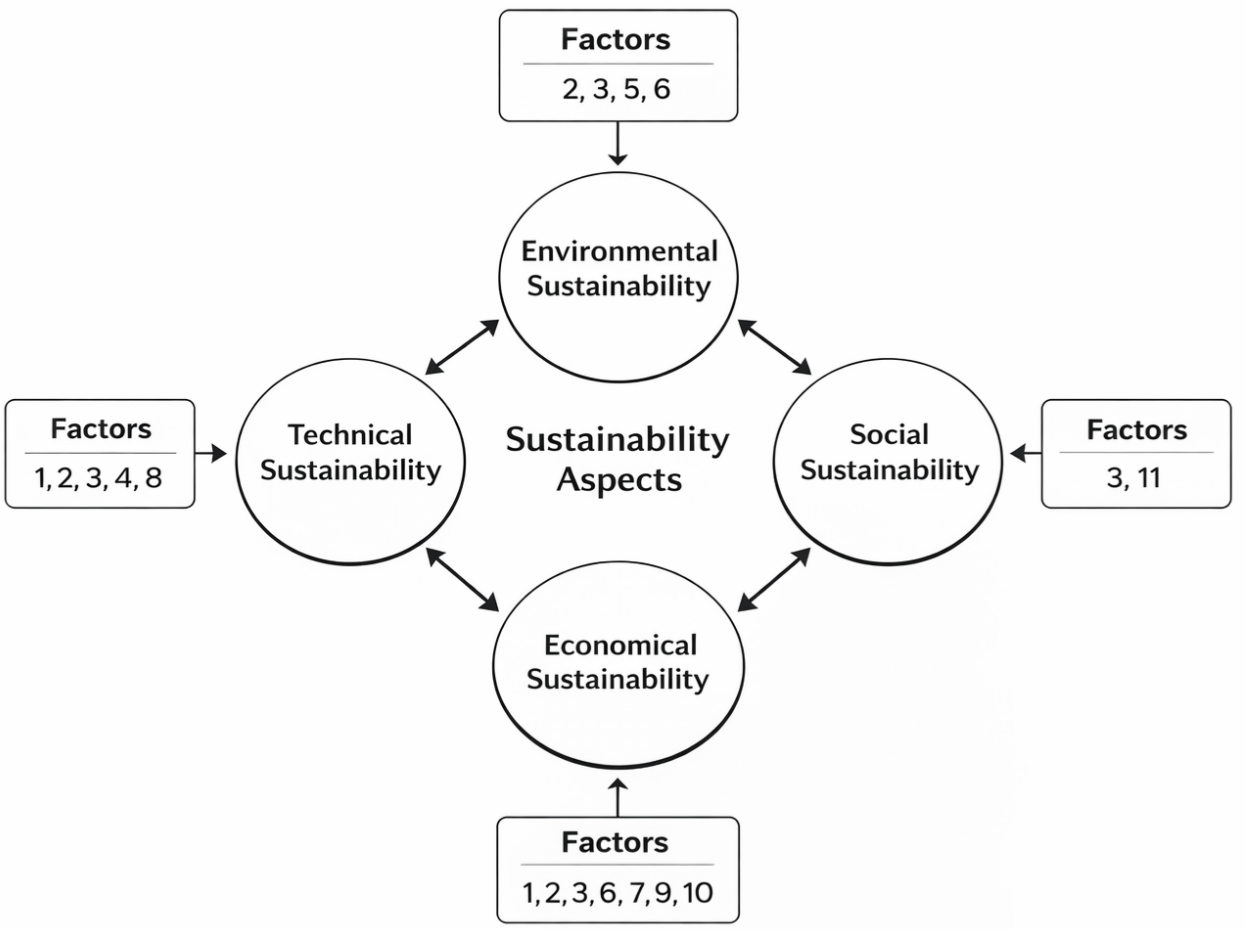


**Figure 5.** List of factors influencing a specific sustainability aspect [17].

Nevertheless, your participation in the study was entirely voluntary.

## 7 Conclusion and future work

The research was an empirical investigation of sustainability in crowdsourced software development, which validated sustainability-related factors discovered through a systematic literature review. The results verify that eleven factors mentioned in the literature are also identified by industry professionals, and there is a high correlation between research and experience, although the priorities of factors differ slightly. This correspondence indicates that the sustainability factors considered in earlier studies are mainly representative of a practice-based CSD. In practice, the valid factors offer practical advice to designers, project managers, and developers of crowdsourced software development platforms. To achieve better informed decision-making and enhance the long-term sustainability of crowdsourced software projects, it is possible to address issues related to coding practices, coordination and communication, training, incentives, and sustainability awareness. This research will involve the creation of a structured sustainability assessment model or decision support tool depending on the validated factors as future work. This model would help CSD stakeholders to assess sustainability threats in a systematic and organized way and incorporate sustainability aspects into the process planning and implementation of crowdsourced software development projects.

## Data Availability Statement

Data will be made available on request.

## Funding

This work was supported without any funding.

## Conflicts of Interest

The authors declare no conflicts of interest.

## AI Use Statement

The authors declare that no generative AI was used in the preparation of this manuscript.

## Ethical Approval and Consent to Participate

This work was conducted in accordance with the ethical principles of the Declaration of Helsinki. Ethical approval was not required for this research as it involved a voluntary online questionnaire survey with adult professionals, posed no more than minimal risk, and collected primarily non-sensitive professional opinions.

Participation in the survey was entirely voluntary. All participants were informed about the purpose of the study, the approximate time required to complete the questionnaire, and that their responses would be treated confidentially and used solely for academic research purposes. By proceeding to complete and submit the questionnaire, participants implicitly provided informed consent to participate in the study. Participants could withdraw from the survey at any time without any consequences. No personally identifiable information was used in the analysis or reporting of results.

## References


[1] Guillemin, F., Robitza, W., Wunderer, S., & Hoßfeld, T. (2022). Definitions of crowdsourced network and QoE measurements. *ACM SIGMultimedia Records, 12*(2), 1-1. [CrossRef]

[2] Howe, J. (2006). The rise of crowdsourcing. *Wired, 14*(6), 1-4.

[3] Estellés-Arolas, E., & González-Ladrón-de-Guevara, F. (2012). Towards an integrated crowdsourcing definition. *Journal of Information science, 38*(2), 189-200. [CrossRef]

[4] Hammon, L., & Hippner, H. (2012). Crowdsourcing. *Business & Information systems engineering, 4*(3), 163-166. [CrossRef]

[5] Christainas, G., Kehagias, D., Salamanis, A., Spanidis, P., Kyrkoy, M., & Tzovaras, D. (2022, August). A Crowdsourcing Framework for Reporting Available Parking Spots in Urban Areas. In *Conference on Sustainable Urban Mobility* (pp. 323-334). Cham: Springer Nature Switzerland. [CrossRef]

[6] Munir, Y., Umer, Q., Faheem, M., Akram, S., & Jaffar, A. (2025). Developer Recommendation and Team Formation in Collaborative Crowdsourcing Platforms. *IEEE Access, 13*, 63170-63185. [CrossRef]

[7] Tong, F., Yu, F., Xu, L., & Zhao, D. (2025, December). Subsystem Integration and Stability Evaluation Methods for Industrial Control Software Crowdsourcing Testing Platforms. In *2025 IEEE International Conference on Blockchain Technology and Information Security (ICBCTIS)* (pp. 1-7). IEEE. [CrossRef]

[8] Klymenko, A., Meisenbacher, S., Favaro, L., & Matthes, F. (2025). Supporting the integration of privacy-enhancing technologies into the software development life cycle. *SN Computer Science, 6*(6), 592. [CrossRef]

[9] Khojastehpour, M., Sahebi, S., & Samimi, A. (2022). Public acceptance of a crowdsourcing platform for traffic enforcement. *Case studies on transport policy, 10*(4), 2012-2024. [CrossRef]

[10] Bazaluk, O., Rahman, M. A., Zayed, N. M., Faisal-E-Alam, M., Nitsenko, V., & Kucher, L. (2024). Crowdsourcing review: The crowd workers' perspective. *Journal of Industrial and Business Economics, 51*(3), 647-666. [CrossRef]

[11] Elsawy, M., & Youssef, M. (2023). Economic sustainability: Meeting needs without compromising future generations. *International Journal of Economics and Finance, 15*(10), 23-31. [CrossRef]

[12] Zhang, Y., & Dong, C. (2024). Sustainable development of digital cultural heritage: a hybrid analysis of crowdsourcing projects using fsQCA and system dynamics. *Sustainability, 16*(17), 7577. [CrossRef]

[13] Hariram, N., Mekha, K., Suganthan, V., & Sudhakar, K. (2023). Sustainalism: An integrated socio-economic-environmental model to address sustainable development and sustainability. *Sustainability, 15*(13), 10682. [CrossRef]

[14] Moises de Souza, A. C., Soares Cruzes, D., Jaccheri, L., & Krogstie, J. (2023, December). Social sustainability approaches for software development: A systematic literature review. In *International Conference on Product-Focused Software Process Improvement* (pp. 478-494). Cham: Springer Nature Switzerland. [CrossRef]

[15] Zada, I., Shahzad, S., Ali, S., & Mehmood, R. M. (2023). OntoSuSD: Software engineering approaches integration ontology for sustainable software development. *Software: Practice and Experience, 53*(2), 283-317. [CrossRef]

[16] Khalifeh, A., Farrell, P., Alrousan, M., Alwardat, S., & Faisal, M. (2020). Incorporating sustainability into software projects: a conceptual framework. *International Journal of Managing Projects in Business, 13*(6), 1339-1361. [CrossRef]

[17] Haider, W., Ilyas, M., Khalid, S., & Ali, S. (2024). Factors influencing sustainability aspects in crowdsourced software development: A systematic literature review. *Journal of Software: Evolution and Process, 36*(6), e2630. [CrossRef]

[18] König, C., Lang, D. J., & Schaefer, I. (2025). Sustainable software engineering: Concepts, challenges, and vision. *ACM Transactions on Software Engineering and Methodology, 34*(5), 1-28. [CrossRef]

[19] Oyedeji, S., Adisa, M. O., Penzenstadler, B., & Wolf, A. (2019). Validation study of a framework for sustainable software system design and development.

*sustainable development, 3*(4), 5.

[20] Khan, H. H., Malik, M. N., Alotaibi, Y., Alsufyani, A., & Alghamdi, S. (2021). Crowdsourced Requirements Engineering Challenges and Solutions: A Software Industry Perspective. *Computer Systems Science & Engineering, 39*(2). [CrossRef]

[21] Zhen, Y., Khan, A., Nazir, S., Huiqi, Z., Alharbi, A., & Khan, S. (2021). Crowdsourcing usage, task assignment methods, and crowdsourcing platforms: A systematic literature review. *Journal of Software: Evolution and Process, 33*(8), e2368. [CrossRef]

[22] Füller, J., Hutter, K., & Kröger, N. (2021). Crowdsourcing as a service–from pilot projects to sustainable innovation routines. *International Journal of Project Management, 39*(2), 183-195. [CrossRef]

[23] Chen, Y., Pandey, M., Song, J. Y., Lasecki, W. S., & Oney, S. (2020, April). Improving crowd-supported gui testing with structural guidance. In *Proceedings of the 2020 CHI Conference on human factors in computing systems* (pp. 1-13). [CrossRef]

[24] Alyahya, S. (2020). Crowdsourced software testing: A systematic literature review. *Information and Software Technology, 127*, 106363. [CrossRef]

[25] Ahmad, A., Khan, R. A., Khan, S. U., Alwageed, H. S., Al-Atawi, A. A., & Lee, Y. (2023). Green cloud computing adoption challenges and practices: A client's perspective-based empirical investigation. *Cognition, Technology & Work, 25*(4), 427-446. [CrossRef]

[26] Malik, M. N., & Khan, H. H. (2018). Investigating software standards: A lens of sustainability for software crowdsourcing. *IEEE Access, 6*, 5139-5150. [CrossRef]

[27] Ahmad, A., Khan, S. U., Khan, H. U., Khan, G. M., & Ilyas, M. (2021). Challenges and Practices Identification via a Systematic Literature Review in the Adoption of Green Cloud Computing: Client's Side Approach. *IEEE Access, 9*, 81828-81840. [CrossRef]

[28] Li, X., Zhang, Y., Osborne, C., Zhou, M., Jin, Z., & Liu, H. (2025). Systematic literature review of commercial participation in open source software. *ACM Transactions on Software Engineering and Methodology, 34*(2), 1-31. [CrossRef]

[29] Shrestha, N. (2021). Factor analysis as a tool for survey analysis. *American journal of Applied Mathematics and statistics, 9*(1), 4-11. [CrossRef]

[30] Nyirongo, B., Jiang, Y., Niu, N., & Liu, H. (2025). An Empirical Study of Software Refactorings in Real-World Open-Source Java Projects. *IEEE Transactions on Software Engineering, 51*(11), 3013-3037. [CrossRef]

[31] Baral, M. M., & Verma, A. (2021). Cloud computing adoption for healthcare: An empirical study using SEM approach. *FIIB Business Review, 10*(3), 255-275. [CrossRef]

[32] Kitchenham, B., & Charters, S. (2007). *Guidelines for performing systematic literature reviews in software engineering* (EBSE Technical Report EBSE-2007-01). Keele University & Durham University.

[33] Li, T., Higgins, J. P., & Deeks, J. J. (2019). Collecting data. *Cochrane handbook for systematic reviews of interventions*, 109-141. [CrossRef]

[34] Lunny, C., Ramasubbu, C., Puil, L., Liu, T., Gerrish, S., Salzwedel, D. M., ... & Wright, J. M. (2021). Over half of clinical practice guidelines use non-systematic methods to inform recommendations: a methods study. *PLoS One, 16*(4), e0250356. [CrossRef]

[35] Alem, D. D. (2020). An overview of data analysis and interpretations in research. *International Journal of Academic Research in Education and Review, 8*(1), 1-27. [CrossRef]

[36] Baltes, S., & Ralph, P. (2022). Sampling in software engineering research: A critical review and guidelines. *Empirical Software Engineering, 27*(4), 94. [CrossRef]

[37] Khan, R. A., Khan, S. U., Khan, H. U., & Ilyas, M. (2022). Systematic literature review on security risks and its practices in secure software development. *IEEE Access, 10*, 5456-5481. [CrossRef]

[38] Nirmani, S., Shahin, M., Khalajzadeh, H., & Liu, X. (2025). A systematic literature review on task recommendation systems for crowdsourced software engineering. *Information and Software Technology, 184*, 107753. [CrossRef]

[39] Imran, A., & Kosar, T. (2019). Software sustainability: a systematic literature review and comprehensive analysis. *arXiv preprint arXiv:1910.06109*.

[40] Salam, M., & Khan, S. U. (2018). Challenges in the development of green and sustainable software for software multisourcing vendors: Findings from a systematic literature review and industrial survey. *Journal of Software: Evolution and Process, 30*(8), e1939. [CrossRef]

[41] Pereira, R. A. A. (2018). *Energyware engineering: techniques and tools for green software development* (Doctoral dissertation, Universidade do Minho (Portugal)).

[42] Jakobsen, A. M. S. Ø., & Kadenic, M. D. (2025). Too Little Fit, Too Much Demand: A Task–Technology Perspective on Sustainability in Engineering Management. *IEEE Transactions on Engineering Management, 73*, 987-1008. [CrossRef]

[43] Mishra, A., & Mishra, D. (2021). Sustainable software engineering: Curriculum development based on ACM/IEEE guidelines. In *Software Sustainability* (pp. 269-285). Cham: Springer International Publishing. [CrossRef]

[44] Moreira, A., Lago, P., Heldal, R., Betz, S., Brooks, I., Capilla, R., ... & Venters, C. C. (2025). A roadmap for integrating sustainability into software engineering education. *ACM Transactions on Software Engineering and Methodology, 34*(5), 1-27. [CrossRef]

[45] Marantos, C., Papadopoulos, L., Lamprakos, C. P., Salapas, K., & Soudris, D. (2022). Bringing energy efficiency closer to application developers: An extensible software analysis framework. *IEEE Transactions on Sustainable Computing, 8*(2), 180-193. [CrossRef]

[46] Mao, K., Capra, L., Harman, M., & Jia, Y. (2017). A survey of the use of crowdsourcing in software engineering. *Journal of Systems and Software, 126*, 57-84. [CrossRef]

[47] Alsayyari, M., & Alyahya, S. (2018, March). Supporting coordination in crowdsourced software testing services. In *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)* (pp. 69-75). IEEE. [CrossRef]

[48] Jayanti, E. (2012). Open sourced organizational learning: implications and challenges of crowdsourcing for human resource development (HRD) practitioners. *Human Resource Development International, 15*(3), 375-384. [CrossRef]

[49] Sarı, A., Tosun, A., & Alptekin, G. I. (2019). A systematic literature review on crowdsourcing in software engineering. *Journal of Systems and Software, 153*, 200-219. [CrossRef]

[50] de Campos, V. Q., David, J. M. N., & Braga, R. (2021, May). Coordination in crowdsourced software development: A systematic mapping study. In *2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD)* (pp. 305-310). IEEE. [CrossRef]

[51] Jiang, H., & Matsubara, S. (2014, December). Efficient task decomposition in crowdsourcing. In *International conference on principles and practice of multi-agent systems* (pp. 65-73). Cham: Springer International Publishing. [CrossRef]

[52] Uchoa, A. P., & Schneider, D. (2026). Driving the Emergence of Self-managed Enterprises of Crowdworkers: A Survey and Review of Motivating Factors. In *International Conference on Enterprise Information Systems* (pp. 63-88). Springer, Cham. [CrossRef]

[53] LaToza, T. D., & Van Der Hoek, A. (2015). Crowdsourcing in software engineering: Models, motivations, and challenges. *IEEE software, 33*(1), 74-80. [CrossRef]

[54] Maddalena, E., Ibáñez, L.-D., Reeves, N., & Simperl, E. (2023). Qrowdsmith: Enhancing paid microtask crowdsourcing with gamification and furtherance incentives. *ACM Transactions on Intelligent Systems and Technology, 14*(5), 1-26. [CrossRef]

[55] Dratler Jr, J., & McJohn, S. M. (2025). *Intellectual property law: Commercial, creative and industrial property*. Law Journal Press.

[56] Kumar, P. (2024). Intellectual property rights (IPR): nurturing creativity, fostering innovation. *Idealism Journal of Advanced Research in Progressive Spectrums (IJARPS), 3*(02), 32-38.

[57] Atadoga, A., Umoga, U. J., Lottu, O. A., & Sodiy, E. O. (2024). Tools, techniques, and trends in sustainable software engineering: A critical review of current practices and future directions. *World Journal of Advanced Engineering Technology and Sciences, 11*(1), 231-239. [CrossRef]

## Appendix

### Questionnaire Survey

The online questionnaire used in this study is available at the following link:

https://docs.google.com/forms/d/e/1FAIpQLSfd0NCPNM7zzBAUzLJ3YLIljChMTSVmHRcAWTxBZxgdGGVexg/viewform

**Waqas Haider** received the B.S. and M.S. degrees in Computer Science from the University of Malakand, Khyber Pakhtunkhwa, Pakistan, in 2016 and 2022, respectively. His research interests include empirical software engineering, software sustainability, open-source software development, software crowdsourcing, and artificial intelligence. (E-mail: whaidermkd@gmail.com)

**Adnan Khan** received the M.S. degree in Computer Science from the University of Malakand, KPK, Pakistan. His research interests include safety-critical cyber-physical systems and systematic literature review. (E-mail: afridikhan541111@gmail.com)

**Samreen Ihsan** received the B.S. degree in computer science from Ghazi Umara Khan Degree College, Samarbagh, affiliated with the University of Malakand, Pakistan. Currently, she is pursuing an M.S. degree in software engineering at Dalian University of Technology, China. Her research interests include medical image analysis and pattern recognition. (E-mail: samreenihsan09@gmail.com)

**Ihsan Adil** received the B.S. degree in computer science from Ghazi Umara Khan Degree College, Samarbagh, affiliated with the University of Malakand, Pakistan. Currently, He is pursuing a M.S. degree in Software Engineering at Dalian University of Technology, China. His research interests include medical image analysis and pattern recognition. (E-mail: ihsanadil0448@gmail.com)