2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE)

An Empirical Study on Task Documentation in Software Crowdsourcing on TopCoder

Luis Vaz
MunDDos Research Group
School of Technology
PUCRS, Brazil
luis.vaz@acad.pucrs.br

Igor Steinmacher
School of Informatics, Computing, and Cyber Systems
Northern Arizona University, USA
igor.steinmacher@nau.edu

Sabrina Marczak
MunDDos Research Group
School of Technology
PUCRS, Brazil
sabrina.marczak@pucrs.br

***Abstract*—In the Software Crowdsourcingcompetitive model, crowd members seek for tasks in a platform and submit their solutions seeking rewards. In this model, the task description is important to support the choice and the execution of a task. Despite its importance, little is known about the role of task description as support for these processes. To fill this gap, this paper presents a study that explores the role of documentation on TopCoder platform, focusing on the task selection and execution. We conducted a two-phased study with professionals that had no prior contact with TopCoder. Based on data collected with questionnaires, diaries, and a retrospective session, we could understand how people choose and perform the tasks, and the role of documentation in the platform. We could find that poorly specified or incomplete tasks lead developers to look for supplementary material or invest more time and effort than initially estimated. To better support the crowd members, we proposed a model on how to structure the documentation that composes the task description in competitive software crowdsourcing. We evaluated the model with another set of professionals, again relying on questionnaires, reports, and a retrospective session. Results showed that although the documentation available covered the elements of the proposed model, the participants had issues to find the necessary information, suggesting the need for a reorganization. Participants agreed that the proposed model would help them understand the task description. Therefore, our study provides a better understanding of the importance of task documentation in software crowdsourcing and points out what information is important to the crowd.**

***Index Terms*—Software crowdsourcing, Task documentation, TopCoder, Empirical study**

I. INTRODUCTION

Software Crowdsourcing, defined as the act of outsourcing software development to an undefined and large group of people—the crowd—through an open call, is a phenomenon that has been gaining attention [1], [2]. Crowdsourcing is mediated by a platform and is based on tasks that are executed by the crowd. The tasks can be distributed on-demand following a recruitment model performed by the platform based on the analysis of crowd members' profiles; or in a competitive model, in which the members register themselves for tasks and submit a solution aiming to receive a reward (e.g., financial).

In the competitive model, the crowd members can choose the tasks they will work on according to their own interest. For instance, they can filter the tasks according to the programming language, application domain, or simply because they feel challenged by the task description. Thus, in this context, the description (or documentation) of the task presented by the platform becomes an important factor for the crowd members—who rely on it to choose and execute the tasks—and for the clients—who aim to receive complete and correct solutions to their problems. However, little is known about how the documentation influences the selection of the tasks and their subsequent development in the crowdsourcing model. Seeking to contribute to filling this gap in the literature, this paper presents an empirical study aiming to understand the role of documentation in the selection and development of tasks in software crowdsourcing.

To conduct this study, we focused on the TopCoder platform [3], which implements the competitive model. We conducted a two-phased research. In Phase 1 we conducted a case study with professionals attending to a graduate course on Collaborative Software Development. The participants received an assignment that consisted of selecting, developing, and submitting two tasks available on TopCoder. The data collection was carried out through questionnaires, diaries, and a retrospective session. The results of Phase 1 showed that the quality of the task documentation, in particular its general description, influences the selection of the task by the members of the crowd. Tasks that do not present clear and objective description—without technology requirements, or environment setup instructions—demotivate developers, leading them to abandon the task's solution attempt. By analyzing the challenges and suggestions reported by the participants, we proposed a way to structure the information that composes the task description in competitive software crowdsourcing.

The goal of Phase 2 was to evaluate the structure we proposed in Phase 1, in addition to gain more insights. To this end, we conducted another case study with professionals of a second session of the same course. We worked again with an assignment in which the 10 professionals had to choose and submit a solution to a task in TopCoder. At this time, we surveyed them about the challenges with documentation. We also asked them to write a report about their experience and conducted a retrospective session as means to triangulate data. The results of Phase 2 indicate that although the information available in the tasks covered the elements of our proposed model, the participants had issues to find the information,

978-1-5386-9196-0/19/$31.00 ©2019 IEEE
DOI 10.1109/ICGSE.2019.00041

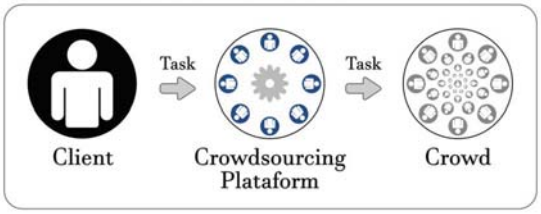

Fig. 1. Crowdsourcing model components.

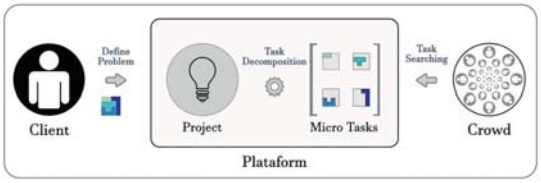

Fig. 2. Representation of task decomposition [7].

suggesting the need for a reorganization. Also, most of the participants agreed that the proposed structure would help them identify and better understand different aspects of the task, reinforcing the need for improvement.

Our results provide a better understanding of the importance of task documentation in software crowdsourcing and identify a set of information that may be provided to improve the experience of crowd members. Thus, this paper contributes as follows: (i) listing the factors that influence the task choice and development; (ii) listing the quality characteristics of the task documentation that are relevant to support the process of task choice and development; (iii) proposing a structured way to present the information when documenting crowdsourcing tasks; and (iv) evaluating how the description provided currently adhere to the proposed structured model and how the structure would benefit the users.

II. Background

Howe [4] explains that the crowdsourcing phenomenon has its origins in the Open Source Software movement, evidencing that a motivated group with a common goal is able to create a high-quality product. It involves outsourcing an activity so that a client reaches its business goals by leveraging a third-party entity capable of solving a particular problem [4].

With a similar idea behind it, Software Crowdsourcing is specialized in software development activities, and involves the client, who proposes the task; the crowd, composed of professionals willing to perform tasks; and the platform, which mediates the relationship between the other two elements [2], [5]. Among these elements is the task, which represents the activities proposed by the client, as illustrated in Figure 1. The tasks are decomposed and managed by the platform, being carried out by the crowd [6].

A. Tasks in Software Crowdsourcing

A task in software crowdsourcing can represent a high complexity problem to be solved in the long term, in several steps—called Challenges; can propose an innovation model; or can propose software development activities, such as the graphic design of an interface or the coding of a specification [2], in the form of Competitions. The task can also take on different formats: high-level description, leaving the crowd member free to define how to develop the solution; a detailed description with technical documents (e.g., UML models); or a set of technical specifications followed by detailed instructions on how to organize and submit the solution.

The task represents the problem or part of the problem defined by the client and is generally defined by the platform. This process of definition and decomposition into micro-tasks, illustrated in Figure 2, is considered one of the major challenges of this development model [7]. This decomposition process needs to safeguard that the tasks made available on the platform do not lose their characteristics and their interdependence with the other parties that represent the problem [7].

When fragmenting the task into micro-tasks, the platform needs to ensure that each micro-task has sufficient information to enable its development. The resulting task documentation cannot be too specific that loses focus nor too broad that challenges its comprehension. Tasks decomposition turns out to be a crucial factor for the tasks solution given that the quality of the resulting documentation is likely to affect the crowd members' performance [7].

B. TopCoder Platform

Some crowdsourcing platforms are specialized in a certain software activity, like uTest[1] and TestBirds[2], which focus on testing; and others provide support the the entire development cycle, such as GetACoder[3] and TopCoder[4]. The latter stands out among the commercial platforms as the pioneer and for having thousands of active members [8].

TopCoder was created in 2001 when its founder, Jack Hughes, proposed an innovative way of dealing with challenges in recruiting talents and high turnover of professionals [3]. Hughes aimed to reduce the costs and effort of his customers by reusing software components instead of building a complete system from scratch. Thus, TopCoder defines a model that prioritizes the reuse of components to solve customer problems and, when necessary, crowdsourced tasks to get solutions for the development of new components. As a way to enable this strategy, the platform becomes a mechanism to attract new members and form a community of skilled programmers—the Top Coders—willing to compete to build the best solution.

In this model, a task must follow a specific flow from creating the announcement, selection, completion, delivery, and validation [2]. Tasks in the Competition category are

[1]URL: https://www.utest.com/
[2]URL:https://www.testbirds.com/
[3]URL:http://www.getacoder.com
[4]URL:https://www.topcoder.com/

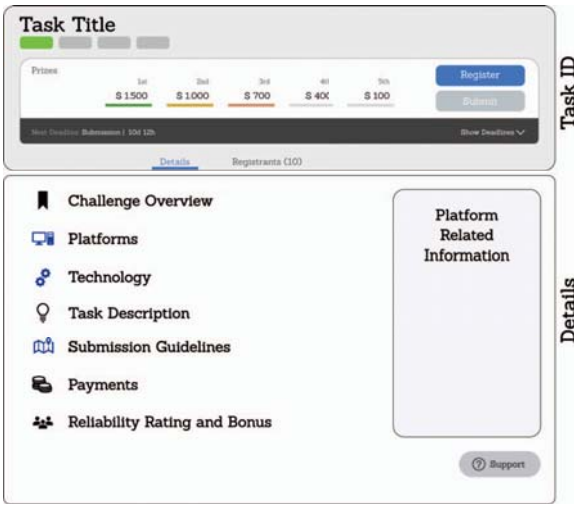

Fig. 3. Current model of the documentation structure of a task in TopCoder.

presented to the crowd by means of a short general description, open to all users, and a detailed description associated (or not) to supplementary documentation (e.g., models, prototype screens), restricted to those who subscribe to the task competition. Figure 3 illustrates the typical structure of a general task description, consisting of: (i) task header (e.g., name, deadlines, prize value); (ii) fixed information highlighted by black icons; and (iii) complementary information (which may vary), represented by blue icons.

III. RESEARCH CONTEXT

As mentioned, to explore the role of documentation during task selection and development in software crowdsourcing, we carried out a study using TopCoder. TopCoder was chosen because it is currently the largest software crowdsourcing platform, with more than one million registered developers, and covers different software development activities [8].

The research was conducted in two phases, as depicted in Figure 4. Phase 1 consisted of a case study with 20 professionals attending to a graduate course, as a course assignment in which the participants had to select tasks and develop a solution for them. We made use of diaries, questionnaires, and a retrospective session with the participants to collect data.

The outcomes of Phase 1 include the role and quality of the documentation in task selection and execution; and a proposed structure to organize the documentation.

For Phase 2, a second case study was conducted with the aim to preliminarily evaluate the structure proposed during Phase 1. The study was conducted with 10 professionals who attended yet another session, a year later, of the same graduate course. Similarly to Phase 1, the participants contributed to TopCoder, submitting a solution to a chosen task. We collected data by means of a survey at the end of the course, asking them to evaluate how the description of the tasks they chose adhere to the proposed structure, and how they perceive the items of

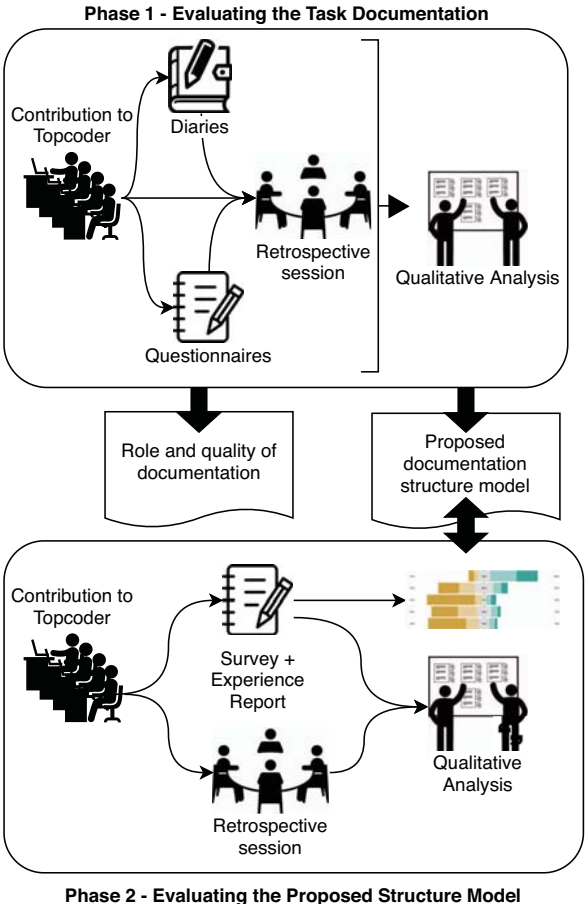

Fig. 4. Research design

the proposed structure. Additionally, we asked the participants to write a report on their TopCoder experience and conducted a retrospective session to triangulate data.

To facilitate the reading, the details about the research method for each phase are presented next to the results of each of the steps (Sections IV and V).

IV. PHASE 1 - EVALUATING THE DOCUMENTATION

In this Section, we present the method and results for the case study conducted to assess how the documentation influences the selection and development of the task in software crowdsourcing. We analyzed the perspective of crowdworkers (with previous industry experience, but no experience in crowdsourcing) while selecting and working on a task.

A. Course Description and Participants' Profile

We conducted the study with the 20 professionals attending to a graduate course on Collaborative Software Development, as an assignment of the course. In addition to Software Crowdsourcing, the course syllabus also included the topics of Web 2.0, Global Software Engineering, Open Sourcing, Agile and Large-Scale Agile Development, and Continuous

Software Engineering. The assignment lasted 16 weeks, the period comprising the course duration. An introductory one-hour long lecture on software crowdsourcing was given on the first week by the course instructor followed by a 30 minutes TopCoder presentation by one of the co-authors, with 15 years of experience in the industry. Participants have, on average, 12 years of working experience in software development, with expertise mostly in Java, C, C++, HTML, JavaScript, CSS, .NET, among other technologies. None of the participants reported having previous experience with software crowdsourcing.

B. Research Method

We organized this phase into two iterations of 8 weeks each to enable the participants to get familiar with TopCoder and understand the competition model. We highlighted that the participants were not obligated to submit task solutions in this first iteration. However, the submission was expected in iteration 2, so that the participants could experience the whole process, including the feedback and reward steps after the submission. Two participants won the first prize (one of the prizes was about $ 1,500.00) and two others had their solutions financially rewarded given their recognized quality.

The process of choosing a task to work on was free of rules, except by the request of picking among the Competition tasks only to allow for a complete experience, including task browsing and selection, completion, delivery, and validation, within the assignment time frame. For each of the two 8 weeks long iteration, the participants received 2 weeks to indicate their task selection (Iteration 1: Weeks 1 and 2, Iteration 2: Weeks 9 and 10) and 6 weeks to complete and submit it (Iteration 1: Weeks 3 to 8, Iteration 2: Weeks 11 to 16).

We made use of questionnaires and a work diary to collect data in both iterations. The use of diaries enables the researcher to follow the participant's continuous behavior with minimal intrusion [9]. This type of collection has been routinely used in longitudinal studies in other areas and more recently in Software Engineering (e.g., [10], [11]). We chose to conduct a diary study because, as reported by Davidson et al. [10], we could not observe our participants. They were able to work whenever they wanted, conducting the work at their chosen time and place. Therefore, it was not possible to be with them every time they were working on the assignment, then we decide to use diaries to follow their journey. We chose to use unstructured diaries, in which the participants could write anything they wanted about the process.

A shared-empty diary was initially created and could be accessed by the participant and two of the co-authors. Participants were motivated to freely write throughout the assignment duration. These co-authors commented and discussed online the diaries at least twice a week. This interaction through the diary entries is an important mechanism for researchers to obtain the expected level of detail in a study [12]. Interactions mostly involved clarification requests, decision justifications, and information addition requests.

In regards to the questionnaires, they were applied in three moments, and previously validated by two researchers and piloted with 3 former participants of a previous course session. The three applied questionnaires are described below:

(1) Participant's profile: applied in the first week to identify the participants profile with regard to their skills and experience with software crowdsourcing.

(2) Task selection: completed by the end of the second week of each iteration (Weeks 2 and 10) to record the participants' understanding of the task description before completion started; selection process, including information on task type (e.g., first to finish, development, design); and reasons for task selection (and disregard of candidate tasks, if applicable).

(3) Task development: completed by the end of the eighth week of each iteration (Weeks 8 and 16) and aimed to collect the participants' perceptions regarding the task completion and solution submission. The following information was collected: task title; completion status (delivered/not delivered) and, in case it was not delivered, reasons for not doing so; and overall experience. As for the documentation, we asked the participants to indicate their level of agreement using a 5-point Likert-scale with the quality of the task's documentation (general and detailed description, including supplementary documents) based on a set of predefined criteria. We defined the criteria (atomic, complete, precise, cohesive, feasible, concise, unambiguous, testable, and correct) based on BABOK's [13], [14] and on Wiegers' lists [15]. Finally, we asked for improvement recommendations.

Finally, we conducted a structured retrospective session in the last week (Week 16) upon the completion of the second task. The session lasted 90 minutes and aimed at providing the chance to the professionals comment on the overall experience and reported data. We handed out a printed copy of each participant's third questionnaire responses, more specifically those related to the quality criteria, and asked the participants to revisit the task documentation, also printed out, and highlight documentation excerpts that led them to indicate such level of agreement to each of the criteria. For instance, if a participant indicated that a task was ambiguous, she was asked to indicate which parts of the task documentation influenced her to consider ambiguity.

The data were analyzed using qualitative content analysis, following the steps proposed by Krippendorff [16]: organization and pre-analysis, reading and categorization, and recording the results. We applied it to the answers to the open questions of the questionnaires, the diaries and the transcripts and documents from the retrospective session. The analysis was conducted primarily by the first author, with weekly discussion meetings with the other co-authors, iterating until reaching consensus on the themes. Thus, we identified themes among the collected data, as reported in the following section.

C. Results

All the participants selected tasks to work on in both iterations, but only 10 of them submitted their solution during Iteration 1; this number increase to 11 in Iteration 2. Among the reasons for the non-submission in Iteration 1, we found the following: lack of technical knowledge to achieve the objective

of the task (P1 - Participant 1, P3, P11, P19); underestimating the time needed to finish the task (P5, P9, P10, P12, P17); and difficulty in setting up the local workspace (P14). In Iteration 2, we identified the following reasons: lack of information in the description of the task (P3, P5, P16); difficulty in setting up the local workspace (P6), lack of time to investigate a solution (P11, P13, P14, P19), and lack of technical knowledge (P20).

Despite the difficulties, the participants mentioned their interest in learning new technologies (P1, P5, P6, P9, P10, P20), seeking feedback for their solutions (P8), participating in the competitions (P1), and even seeking for a financial reward (P3, P4, P10, P12, P14). Although most of them mentioned beforehand that they were regular or little satisfied with their experience, 16 (80%) mentioned that would contribute to TopCoder if they had time available.

Selected Tasks. In Iteration 1, the participants selected tasks of the following types: programming (11 participants), code design (3), interface design (2), interface prototyping (2), bug hunt (1), and generation of ideas to solve a specific problem (1). The 20 participants selected 13 distinct tasks. In Iteration 2, the programming tasks remained the most selected (by 10 participants), followed by code design tasks (5), interface prototyping (3), and interface design (2). In this iteration, the participants selected 12 distinct tasks.

Task Selection. In Iteration 1, the participants (14 out of 20) looked for tasks they had prior knowledge on. Some of them indicated that they searched directly for tasks associated with the programming languages they were familiar with: "*My main criterion was finding something close to what I do at work. This task indicated the tags '.Net Core' and 'SQL Server' in the general description, technologies I know*" (P20). Other participants avoided certain software development activities – "*I searched for tasks related to software or business analysis since this is my background; I ran away from coding*" (P14).

The delivery deadline was another considered reason (7 out of 20), either because of the participants' availability—"*I wanted a task with a reasonable timeline. I ended up choosing one I think I will have time to finish*" (P5)—or because they believed that the TopCoder estimation was adequate – "*I think it is feasible within the provided timeframe*" (P6).

Feeling motivated or challenged to solve the problem was another factor that influenced the task selection (2 participants) – "*Curiosity to know if I can resolve the task*" (P18). Others wanted to have the opportunity to learn something new – "*I'm interested in AI and especially in IBM's tools for the topic. I'll take the opportunity to find out how these tools work*" (P7); or to be rewarded – "*I foresee the possibility to be among the winners and maybe receive a reward for my effort*" (P1).

Understanding the task documentation also appeared as a reason (4 out of 20) – "*Among all open tasks that I felt qualified to work on, this was the one that seemed to me to have the best presentation and a description easy to follow*" (P9) and "*What called my attention was the simplicity of the description. It was very easy to understand it*" (P18).

In Iteration 2, the concern related to having prior knowledge to execute the task, including mastering the required programming language/technology, also prevailed (11 out of 20 participants) — "*I know well bootstrap, HTML and CSS. Among the options, this was the task that seemed the best fit to my profile*" (P12). However, more participants took the opportunity to learn something new (7 participants mentioned this reason) – "*As I invest in Bitcoins, but without knowing for sure how digital coins and blockchain work, I decided to try this task to learn about how they work*" (P8).

The worry with the delivery deadline was of lesser importance (2 participants) – "*I believe I'll be able to conclude on time*" (P4) and the expectation of being rewarded, although one participant received a reward in Iteration 1, was not mentioned now. The need to understand the task description was mentioned by 4 participants – "*I work with data visualization. When reading the title of the task, followed by detailed and rich supplementary documentation, I promptly became interested*" (P1) and "*The API is in JavaScript and is clearly presented in the task documentation*" (P7).

An interesting finding is that the factors that motivated the selection of a task were also influential for **not selecting other tasks**. For example, in Iteration 1, the concern with the lack of knowledge was the most mentioned factor among the participants (cited by 18 out of the 20) – "*The fact that the tasks require programming languages that I am not proficient was demotivating. I discarded these tasks*" (P8) and "*Despite working with graphic design, some [design] tasks required specific knowledge, like Adobe Illustrator*" (P14). This factor was followed by the concern with the delivery deadline (mentioned by 7 participants) – "*Several coding tasks that I inspected seem to have a very short completion time, which makes me think that TopCoder is geared towards specialists with the local workspace ready to go*" (P5). The concern with the understanding and quality of the documentation also appeared here (3) – "*It is impossible to work on a task with lack of clarity of what is expected. I could not find the missing information even in the supplementary material*" (P19).

Similarly, in Iteration 2, tasks were also disregarded because of participants' lack of knowledge (16 participants) – "*From the description I am inferring that I need to be an expert in this programming language*" (P14). The delivery deadline and the time to be invested (7 participants) were also concerns – "*Available tasks are offering 2 or 3 days for resolution only*" (P9). The lack of motivation to solve a task was cited by 2 participants: "*I honestly do not have an interest in this [task] subject*" (P8), and the expectation of being remunerated was also mentioned (1) – "*I wanted to consider this challenge but it was just for fun, it does not offer any reward*" (P7).

The understanding and quality of the task documentation becomes a more prominent factor (7) – "*I couldn't grasp the goal of the task, it was poorly written*" (P10) and "*I definitely didn't understand the business rule. I even asked a colleague for help but he didn't understand either*" (P11).

Task Quality Criteria. Regarding the overall quality of the tasks documentation in Iteration 1, 18 out of the 20 participants

indicated that they were "excellent, very good or good"; while two indicated that it was of "low quality". When considering the listed quality criteria, most of the task descriptions were considered "atomic" and with "complete, accurate, cohesive, and feasible descriptions", and "concise, unambiguous, and correct" (16 or more participants totally or partially agreed). Also, among those who disagreed that the documentation is "accurate", one mentioned that "*sometimes the information is very summarized and in other cases, very extensive. The ideal was to have only what is needed*" (P2). P1 disagreed that the information is "correct" – "*The description mentioned that an additional material describing metrics to be used was available in the forum, but what was indeed available was a note pointing out to an external link*". For the testable criterion, 10 participants agreed and 10 remained neutral about it – "*I don't think it is possible to test these scenarios, I'll see*" (P6).

In Iteration 2, there was a decrease in the perception of the overall quality of the documentation. Thirteen participants indicated that the documentation was "very good" or "good" and 7 that it was "of low quality" or "poorly described". This perception was reflected in the evaluation of the quality criteria of the task documentation. Apart from the "feasible" and "concise" criteria (17 participants totally agreed/partially agreed), for the other criteria ("atomic, complete, precise, cohesive, unambiguous, testable, and correct"), 10 participants neither agreed nor disagreed, and partially or totally disagreed. Part of the detailed comments on these criteria is reported as part of the improvement recommendations.

Improvements to the Tasks Documentation. Although some participants pointed out that "*the documentation was very clear*" (P6) and "*complete*" (P18), a set of improvement recommendations were suggested. In Iteration 1, one participant suggested to add detailed information in the open call description – "*In my opinion, it lacked essential information*" (P9). Another participant believes that "*a brief summary of the task"* should be included next to the task title "*with two or three lines because it took a lot of patience and good faith to infer from the task title only*" (P3).

As for the supplementary documentation, accessible only by those who subscribe to a certain task, participants (3 out of 20) suggested that it should "*include the target audience of the resulting solution, so that I can use proper wording for this audience*" (P1) and that this additional documentation should be made available immediately "*at the time of enrollment in the task*" (P16). It was also noted that "*...explanatory videos or diagrams to complement the textual explanation could improve the understanding of what should be done*" (P11) and that "*mockups and the description of the behavior expected by stakeholders would have helped to understand the context of use and the actual need for the request*" (P7).

The participants suggested that information about the prerequisites (or task requirements) for working on the task should be explicitly added to the documentation – "*They should clarify the tools necessary for task completion, for example*" (P21). Also, the acceptance criteria – "*It would be important to know the quality and tests criteria for the solution*" (P4) and "*a checklist to indicate that the task was terminated correctly could help a lot, like the agile acceptance criteria*" (P10). The way of evaluating the delivered results was also a nice-to-have item – "*I would like to know how the choice for the best solution will be made*" (P4). Furthermore, the submission process needs to be explicit – "*The submission process was not clear... the form of submission could have been better explained*" (P15) and which artifacts should be delivered – "*it lacked an explanation of what was precisely necessary to be in the deliverable since the task required the product design explanation*" (P17).

It was also suggested that the task documentation should "*better describe the [software] requirements*" (P4) so that "*it reduces the need for reading several messages in the forum with explanations that were supposed to be part of the documentation in the first place*" (P2), or still, "*having to ask [in the forum] and receive back unsatisfactory responses*" (P9).

The suggestions for improvement were repeated almost entirely in Iteration 2. Some participants mentioned that it would be important, as for the general description of the documentation, "*to present descriptions that clearly separate the technical aspects of the Instructions on TopCoder*" (P1) and "*seek to structure a bit the description of the task, like, indicating the goal, mentioning what must be done and delivered, so anyone will be able to understand it*" (P6). The standardization of the structure of the tasks has been recommended by 3 participants – "*Each task has a different format. It's so hard to grasp the different structures. It would be nice if a standard would stand out*" (P14) and "*standardization could come in the flexible format as long as it was made easier to identify any missing information*" (P19).

Participants also added, as for the supplementary documentation, that instructions for setting up the environment should be provided – "*It would have been helpful to know more about the IDE assembly and the configuration of the libraries*" (P11) and "*improve the technical details of the setup environment. Maybe create a separate session for that?*" (P13). And they reinforced the importance of stating the prerequisites of the task – "*The task description did not state that I needed to download the Ethereum wallet. It took me hours to find out that this was a prerequisite" (P8) and "mention the necessary tools*" (P17). Expliciting the acceptance criteria was also suggested – "*The acceptance criteria should be better described, showing the main points that need to be met*" (P12) and details of the submission guidelines – "*The greatest difficulty is to identify what is being requested as a deliverable. In several tasks, there is no information about what is requested (if a file of a specific program, such as Sketch, Photoshop, Microsoft SQL) or if it's a generic image... So having a box that clearly indicates which artifacts need to be delivered... would help not only to identify what is desired but also to filter the tasks I consider to be able to accomplish*" (P3).

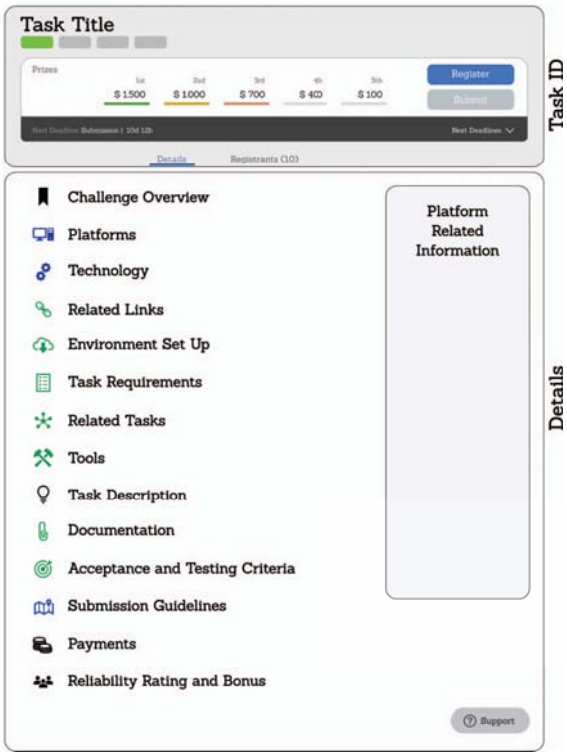

Fig. 5. Proposed documentation structure model.

D. Proposed Documentation Structure Model

In this study, through a qualitative analysis of work diary and questionnaire data, we observed that the process of choosing tasks in software crowdsourcing is heavily based on the information provided in the task description. The participants made recommendations for improving the documentation based on their experiences with the study. These recommendations are expressed through a set of information that, if presented as part of task description, could facilitate the process of task selection and development. This identified information is highlighted in Figure 5.

More specifically, the information regarding the task summary (challenge overview) and the financial rewards details (payments), presented as standard items in all TopCoder tasks (black icons) remain important. Information about the required platforms and technologies, and the submission guidelines (blue icons) were suggested to become mandatory items. The presentation of this information would enable a quick evaluation of whether a crowd member has previous knowledge, if she has the required resources in place, as well as to identify what is expected in terms of artifacts to be delivered.

We also identified that a set of information that is presented in some tasks, as part of the supplementary documentation description, should be explicitly cited in all tasks to facilitate the identification of prerequisites and comprehension of what is requested (green icons). This information is as follows: related links, environment setup, task requirements (or prerequisites), related tasks, tools, documentation, and acceptance and testing criteria. The organization of the suggested documentation items is illustrated in Figure 5. The item Reliability rating and bonus, used to rank the developer in TopCoder, is not a piece of task-related information but is kept as part of the task description model given that it is intrinsic to the platform itself.

It is important to reinforce that we propose a documentation model for the tasks in the form of a logical structure for information aimed at standardizing the documentation and facilitating its reading and understanding. However, we do not define or enforce the way information should be described. Thus, it is essential that the documentation follows a set of rules and standards (either textually or with the use of graphic elements), to make the information clear and objective to the crowd members. The change in the structure implies that the client or the crowdsourcing platform needs to be more careful when providing or defining the task documentation in order to maintain an organized information structure.

V. Phase 2 - Evaluating the Proposed Model

In this phase, our goal was to preliminarily evaluate to what extent the existing documentation adhere to our proposed model, and the perception of users about this model.

A. Participants

This study was conducted with 10 professionals who attended to another edition of the Collaborative Software Development graduate course. The participants had, on average, 10 years of experience in industry, and no previous experience related to TopCoder. All professionals have previous experience as developers although one is currently working as a UX designer, another as a product manager, and a third one on quality assurance. Eight of them work in small to mid-size companies and two in large IT multinationals. Projects they work on vary from finance and education to HR and media systems. Similarly to the Phase 1, the study was conducted as part of an assignment that had an 8 weeks-long period to be handed in, and again an introductory lecture on software crowdsourcing and TopCoder were offered in Week 1. Once again, one professional received a reward for her solution (3rd place) and two others were asked to resubmit their solutions with improvements during the review and feedback phase.

B. Research Method

Differently from Phase 1, this phase of the study was designed to happen in a single iteration. We requested the participants to spend some time lurking in the TopCoder in order to familiarize themselves with the platform and the competition model. After this initial activity, we asked the participants to select a task. The selection of the task was free, so the participants would choose a task that they were able to develop and submit within the assignment period.

At the end of the 8-week period, each participant had selected a task and submitted a solution. In terms of deliverables to the research, as a means of providing us data about their participation, the participants were requested to provide answers to a survey questionnaire. The questionnaire was provided after they concluded the tasks. We requested them to review the task they had worked on, reflecting about the process of task selection and execution, focusing on how they would evaluate the documentation, as an open-ended question (Q1). This was followed by a close-ended question asking them to evaluate to what extent they could identify the elements of the proposed structured in the documentation provided in the task that they worked on (Q2). In addition, we asked them to evaluate how the information provided in each item of the proposed model would help and motivate them to select and execute the tasks (Q3). We asked Q2 and Q3 by means of 5-point Likert-scale items, followed by an open-ended box asking them for further information about their answers.

The participants also wrote a 4 pages-long experience report and participated in a retrospective session in Week 8 that aimed to debrief the participants on their experience. The session lasted 90 minutes, and the participants had the opportunity to discuss the process, the issues with the documentation, and how they feel about the model we were proposing. These were additional data used to supplement the survey results.

C. Results

The analysis that we present refer to the questions about the task description presented and the adherence of the description with the structure proposed in our model. The results are presented in Figure 6 in which is possible to observe the agreement level regarding the mapping between the information made available in the task, and each element of the model.

As it is possible to observe, the participants could find most of the content that match the model elements. Some of them had an agreement level of 100% ("challenge overview" and "submission guidelines") or really close to it ("acceptance and testing criteria" and "task description"). This is justified by the current structure of the tasks, which has some mandatory fields. On the other hand, items like "related links," "environment setup," and "related tasks" had a really low acceptance rate. This can be easily explained since some tasks did not have any kind of related task or link. Following up on that, one thing that needs to be observed in the graphic is the high number of neutral answers. It happened because in many cases the tasks did not require a specific kind of information, so the participants informed a neutral answer (meaning it does not apply).

Although the results seem to show that the information available in the task covered the elements of our proposed model, by analyzing the answers provided to the open-question that came before this one (Q1), we found that the participants had issues to find the information. For example, P21 mentioned that "*the description of the task was well created, however, there was a lack of organization without a clear separation between sections... the information organization was messy*." A similar problem was reported by P28: "*...a good amount of information was available, however, some information was missing and the task became ambiguous*."

In addition, the participants mentioned some potential suggestions, that are in-line with the structure proposed. For example, P30 suggested to "*concentrate all the prototype screens in a single place along with the required skills*", while P25 mentioned that it would be a good idea to "*standardize the description with some mandatory fields*" (P35).

Thus, we can conclude that, although the information is provided (in the task documentation, forum, and sometimes outside the platform), it is not trivial for the crowdworkers to find the right piece of information. Therefore, it becomes necessary to reorganize the information, so it can be found more easily. The information provided as justification for the answers to the Likert-scale shows that the participants believe that the structure proposed can be beneficial. They mentioned that "*all the elements are rather important [to organize the documentation]*" (P25), and "*the elements are really important and necessary for the understanding of the task*." (P27).

In Q3 our goal was to investigate how the proposed model would support crowdworkers addressing their concerns and motivating them to work on a task. To do so, we built upon the findings of Phase 1 about motivation and reasons to select a task. When we asked the participants to reflect about how

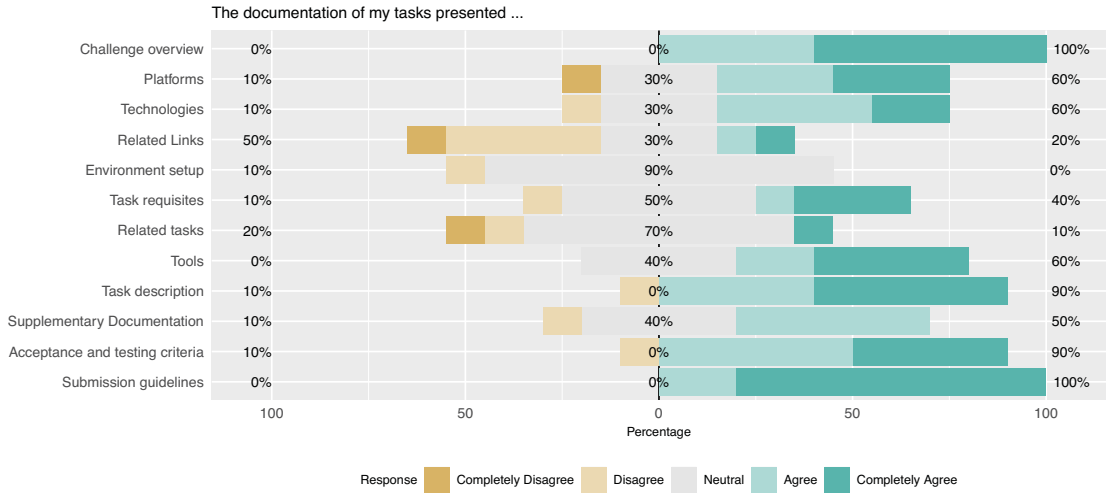

Fig. 6. Adherence of the task description with the elements of the proposed model.

the model would support them with a set of items, we got the answers presented in Figure 7.

As it is possible to observe, most of the participants agreed that the proposed structure would help them identify and better understand different aspects of the task. We would like to highlight the item that mention "better understand the task description," which represents the main goal of the proposed model. For this item specifically, the agreement level was 100% (3 participants agreed, and 7 completely agreed). This shows the potential of a reorganization and better information presentation, as mentioned by P26: "*I believe that the suggested elements to standardize the documentation may support the understanding of the tasks, their goals, pre-requisites, etc.*"

In addition, we would like to highlight that there was no disagreement with two other items, namely "identify if I have the knowledge to work on the task" and "find and understand the supplementary documentation." For the first one, the information is actually already provided, but, having it structured as technologies and platforms may have a difference. For the latter, from the tasks, we could see that sometimes this kind of information was provided in the forums (in some cases just when requested), and in some cases, it was placed outside of the platform. Thus, having this information in a place reserved for it may benefit the crowdworkers.

It is also important to mention that some items had some level of disagreement. Two of them received two negative answers: "identify if I can learn something new" and "identify who are the final users of the solution." Although the first one seems to be related to the subjectivity of the learning experience, the second item clearly represents something that is not explicit from the structure proposed.

Finally, two participants mentioned that, although the proposed structure seems promising, it may have pitfalls, mainly if it is not well used by the requesters. This is clearly reported by P24, who said that: "*the elements structure itself does not guarantee that the task description will be improved. I mean that, if the quality of the information provided is not good enough, the structure will not bring benefits.*"

VI. Limitations

The participant selection (based on a convenience sample) may pose a threat. We would like to highlight that the participants had previous experience in software industry, most of them (16 out of 20 in Phase 1 and 8 out of 10 in Phase 2) were working (full or part-time) in industry during the course. Also related to the sampling, we understand that the regional context and the small number of participants may not enable generalizing our results [17]. In addition to it, the reduced time to perform some tasks may have restricted the participants to a limited amount of tasks, since the submission of the solution was expected within the course period. In addition, the TopCoder had been unavailable for 5 consecutive days close to the Iteration 2's deadline (Weeks 14 and 15) in Phase 1, which may have hindered some participants with availability in this period. The participants also noticed that in the same period the available tasks had 2 to 3 days short submission deadlines, which is not a common practice in the platform. A strategy used to minimize the impact of the tasks selection was to give the participants freedom to choose among the available ones. Thus, each participant, regardless of her previous experience, could select a task to which she had interest and required skills. In this sense, there was a reproduction of the actual process of tasks selection in Topcoder. This freedom should improve the validity of our study as recommended by Creswell [18].

VII. Conclusion

The crowdsourcing model has been widely adopted and studied in software development. In the competitive model, the documentation (task description and supporting materials)

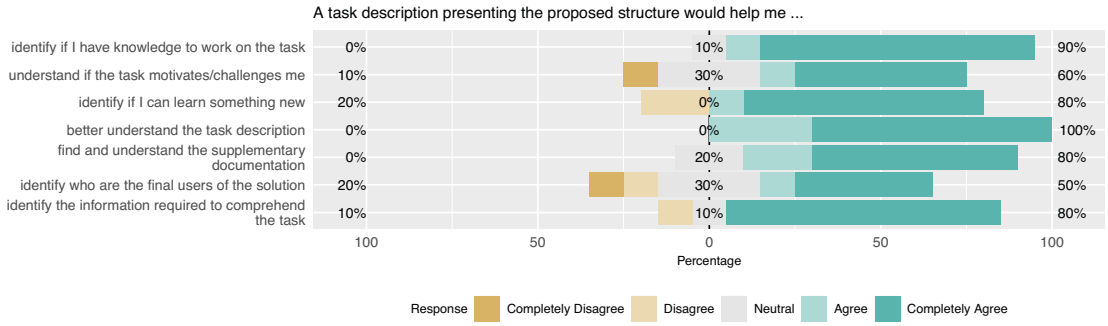

Fig. 7. How the structure would support the crowdworkers while selecting and executing their tasks

offered by the platform serves as a basis for members to choose the tasks they consider appropriate and to build the respective solutions. In this paper, we presented an empirical study aiming to understand how this documentation influences the task choice and execution in a competitive crowdsourcing software environment. Factors that influence the selection and development of a task were identified, based on the documentation available, as well as which criteria define the quality of this documentation. Finally, we could identify a series of suggestions for improving the documentation of tasks and proposed a model based on these suggestions. The preliminary evaluation of the model shed some light on the need for software crowdsourcing requesters improve task description and structure in order to facilitate the crowdworkers' working experience. Further studies with a larger sample and other software crowdsourcing platforms are welcome.

REFERENCES

[1] N. H. Thuan, P. Antunes, and D. Johnstone, "Factors influencing the decision to crowdsource: A systematic literature review," *Information Systems Frontiers*, vol. 18, no. 1, pp. 45–68, jun 2015.

[2] K. Mao, L. Capra, M. Harman, and Y. Jia, "A survey of the use of crowdsourcing in software engineering," *Journal of Systems and Software*, vol. 126, pp. 55–87, apr 2017.

[3] K. Lakhani, D. Garvin, and E. Lonstein, "Topcoder: Developing software through crowdsourcing," *Harvard*, vol. 610, no. 32, 2010.

[4] J. Howe, *Crowdsourcing: Why the Power of the Crowd is Driving the Future of Business*. New York, USA: Random House Business, 2008.

[5] R. Prikladnicki, L. Machado, E. Carmel, and C. de Souza, "Brazil software crowdsourcing: a first step in a multi-year study," in *International Workshop on CrowdSourcing in Software Engineering*. Hyderabad, India: ACM, 2014, pp. 1–4.

[6] M. Hosseini, K. Phalp, J. Taylor, and R. Ali, "The four pillars of crowdsourcing: A reference model," in *International Conference on Research Challenges in Information Science*. IEEE, 2014, pp. 1–12.

[7] K. Stol and B. Fitzgerald, "Two's company, three's a crowd: a case study of crowdsourcing software development," in *International Conference on Software Engineering*. Hyderabad, India: ACM, 2014, pp. 187–198.

[8] A. L. Zanatta, L. Machado, G. Pereira, R. Prikladnicki, and E. Carmel, "Software crowdsourcing platforms," *IEEE Software*, vol. 33, no. 6, pp. 112–116, nov 2016.

[9] G. Symon, *Qualitative research diaries. Essential Guide to Qualitative Methods in Organizational Research*. London, UK: Sage, 2004.

[10] J. L. Davidson, U. A. Mannan, R. Naik, I. Dua, and C. Jensen, "Older adults and free/open source software: A diary study of first-time contributors," in *OpenSym '14*. New York, NY, USA: ACM, 2014.

[11] I. Steinmacher, T. Conte, C. Treude, and M. A. Gerosa, "Overcoming open source project entry barriers with a portal for newcomers," in *Int'l Conf. on Software Eng.* Austin, USA: ACM, 2016, pp. 273–284.

[12] J. Rieman, "The diary study: A workplace-oriented research tool to guide laboratory efforts," in *Conference on Human Factors in Computing Systems*. Amsterdam, The Netherlands: ACM, 1993, pp. 321–326. [Online]. Available: https://dl.acm.org/citation.cfm?id=169255

[13] I. I. of Business Analysis, *A Guide to the Business Analysis Body of Knowledge (BABOK) V2*. Toronto, Canada: IIBA, 2009.

[14] ——, *Guide to the Business Analysis Body of Knowledge (BABOK) V3*. Toronto, Canada: IIBA, 2015.

[15] K. Wiegers, *Software Requirements*. Redmond, USA: Microsoft, 2013.

[16] K. Krippendorff, *Content Analysis: An Introduction to Its Methodology*. New York, USA: Sage, 2018.

[17] J. E. Mcgrath, "Methodology matters: Doing research in the behavioral and social sciences," *Readings in HumanComputer Interaction*, vol. 2, pp. 152–169, 1995.

[18] J. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, 5th ed. Los Angeles, USA: Sage, 2018.