An Empirical Study on Task Documentation in Software Crowdsourcing: The Case of the TopCoder Platform

Luis Vaz
Grupo de Pesquisa MunDDos
Escola Politécnica - PUCRS
luis.vaz@acad.pucrs.br

Sabrina Marczak
Grupo de Pesquisa MunDDos
Escola Politécnica - PUCRS
sabrina.marczak@pucrs.br

Igor Steinmacher
DACOM – UTFPR-CM – Brasil
SICCS – NAU – EUA
igorfs@utfpr.edu.br

ABSTRACT

Software Crowdsourcing, the act of outsourcing software development tasks to a crowd in the form of an open call, happens mediated by a platform and is based on tasks. In the competitive model, the members of the crowd seek for tasks and submit solutions attempting to receive financial rewards In this context, task description plays a relevant role since its understanding supports the choice and development of a task. Little is known about the role of task description as support for these processes. In order to contribute to fill this gap, this paper presents an empirical study exploring the role of documentation when developers select and develop tasks in software crowdsourcing. The TopCoder platform was studied in two stages: a case study with newcomers to crowdsourcing (in the classroom); and a study based on interviews with industry professionals. We identified that the documentation quality influences task selection. Tasks with unclear objective description, without specifying required technologies or environment setup instructions, discourage developers from selecting the task. We also found that poorly specified or incomplete tasks lead developers to look for supplementary material or invest more time and effort than initially estimated. The results provide a better understanding about the importance of task documentation in software crowdsourcing and point out what information is important to the crowd.

CCS CONCEPTS

• **Software engineering**; • **Documentation**; • **Requirements analysis**; • **Open source model**;

KEYWORDS

Software crowdsourcing, Documentação, TopCoder

ACM Reference Format:
Luis Vaz, Sabrina Marczak, and Igor Steinmacher. 2018. An Empirical Study on Task Documentation in Software Crowdsourcing: The Case of the TopCoder Platform. In *Proceedings of SBES - Simpósio Brasileiro de Engenharia de Software (SBES'18).* ACM, New York, NY, USA, Article 4, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

SBES'18, 2018, São Carlos, SP, Brazil

ACM ISBN 978-1-4503-5326-7…$00.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUÇÃO

O *Software Crowdsourcing*, modelo definido como o ato de terceirizar o desenvolvimento de software para um grupo indefinido e grande de pessoas–a multidão (ou *crowd*)–por meio de uma chamada aberta, é um fenômeno que vem ganhando cada vez mais espaço [7, 16]. O *crowdsourcing* ocorre mediado por uma plataforma e é executado baseado em tarefas. As tarefas podem ser distribuídas por um modelo de recrutamento realizado pela plataforma, baseado no perfil pré-indicado do membro da *crowd*; ou por meio de um modelo de competição, em que o próprio membro se registra em uma tarefa de seu interesse e busca submeter uma solução almejando uma premiação (*e.g.*, financeira, prestígio) por ganhar a competição.

Nesta abordagem de *software crowdsourcing* competitivo, o membro da *crowd* tem à sua disposição a gama de tarefas disponíveis e pode usar critérios diversos para selecionar a que lhe for de interesse (*e.g.*, linguagens de programação que conhece, domínio da aplicação, ou simplesmente o fato de se sentir desafiado pela descrição da tarefa). Neste contexto, assume-se que a descrição (ou documentação) da tarefa apresentada pela plataforma passa a ser importante. Espera-se que o membro da *crowd* use esta documentação para decidir qual tarefa deseja selecionar e também como base para solucionar a tarefa. Entretanto, sabe-se pouco sobre como a documentação influencia a seleção de uma tarefa e o subsequente desenvolvimento da mesma neste modelo. Buscando contribuir para preencher este *gap* da literatura, este artigo apresenta um estudo empírico para explorar o papel da documentação na seleção e desenvolvimento de tarefas em *software crowdsourcing*.

Estudou-se o caso da plataforma comercial TopCoder [6], que implementa o modelo competitivo de desenvolvimento de tarefas. O estudo ocorreu em duas etapas concomitantes. Uma das etapas consistiu de um estudo de caso com novatos–alunos de Pós-Graduação que estavam tendo seu primeiro contato com o modelo de *crowdsourcing*, porém com experiência prévia na indústria de desenvolvimento de software. A atividade solicitada aos estudantes consistia em selecionar, desenvolver e submeter duas tarefas na plataforma Topcoder. A coleta de dados deu-se por meio de questionários e diário de bordo durante 16 semanas. A segunda etapa foi um estudo baseado em entrevistas com profissionais da indústria. Nesta etapa os profissionais receberam um lista de tarefas previamente selecionadas e deveriam escolher e justificar a escolha das tarefas.

Identificou-se que a qualidade da documentação de uma tarefa, em especial a sua descrição geral, influencia a seleção da tarefa pelos membros da *crowd*. Tarefas com descrição do objetivo não claro, sem especificação de tecnologia a ser usada ou instruções de configuração de ambiente desmotivam os desenvolvedores, levando-os a abandonar a tentativa de solução da tarefa. A documentação é também de relevância na realização da tarefa em si. Tarefas mal

especificadas ou incompletas em sua descrição e material de apoio levam os desenvolvedores a ter que consultar material complementar ou mesmo a perderem o prazo de submissão da solução. Ainda, certas informações que são eventualmente apresentadas de forma desestruturada no corpo da descrição da documentação complementar (*e.g.*, pré-requisitos, *links*, critérios de aceite e regras de submissão) seriam mais facilmente compreendidas se organizadas de outra forma, *i.e.,* se explicitadas na chamada aberta.

Os resultados oferecem um melhor entendimento sobre a importância da documentação de tarefas no *software crowdsourcing* e identificam um conjunto de informações a serem providas para os membros da *crowd*. Assim, este artigo contribui da seguinte forma: (i) lista os fatores que influenciam a seleção e o desenvolvimento de uma tarefa; (ii) lista as características de qualidade da documentação da tarefa relevantes para apoiar os processos de seleção e desenvolvimento da tarefa; (iii) identifica informações a serem consideradas na documentação de uma tarefa; e (iv) relata um estudo com novatos e profissionais da indústria.

2 FUNDAMENTAÇÃO TEÓRICA

Segundo Howe [5], o termo *crowdsourcing* originou-se a partir do movimento de *Open Source Software*, evidenciando que um grupo ou comunidade motivada e com um objetivo comum é capaz de criar um produto de alta qualidade. Envolve a terceirização de uma atividade visando que o contratante alcance suas metas de negócio através de uma entidade externa–que recebe ou não pelo serviço prestado–capaz de solucionar um determinado problema [5]. Essa entidade externa, diferentemente do *outsourcing*, não possui uma identificação clara. É representada por uma multidão de pessoas que representam um conjunto de potenciais profissionais qualificados para a tarefa. Desta forma, os conceitos fundamentais relacionados a este modelo são [4]: a existência de um problema estabelecido ao qual se procura obter uma solução; a realização de uma chamada aberta para que os interessados saibam da existência da atividade; e a multidão, capaz e disposta a realizar uma tarefa.

De estrutura análoga, o *Software Crowdsourcing*, especializado em atividades de desenvolvimento de software, envolve o Contratante, que propõe a tarefa; a Multidão, composta profissionais dispostos a realizar tarefas; e a Plataforma, que media a relação entre os outros dois elementos [7, 11]. Entre estes elementos existe a Tarefa, que representa as atividades propostas pelo contratante, conforme ilustrado na Figura 1. As tarefas são decompostas e gerenciadas pela plataforma, sendo realizadas pela multidão [4].

2.1 Tarefas em Software Crowdsourcing

Um tarefa em *software crowdsourcing* pode representar um problema de alta complexidade para ser resolvido a longo prazo, em diversas etapas (denominadas desafios); pode propor um modelo de inovação; ou ainda, atividades corriqueiras de desenvolvimento de software, como o *design* gráfico de uma interface ou a codificação de uma especificação [7]. A tarefa pode ainda assumir diferentes formatos: descrição em alto nível, deixando o membro da *crowd* livre para definir como representar a solução; uma descrição detalhada acompanhada de documentos técnicos como modelos em UML; ou um conjunto de especificações técnicas seguidas de instruções detalhadas de como organizar e submeter a solução.

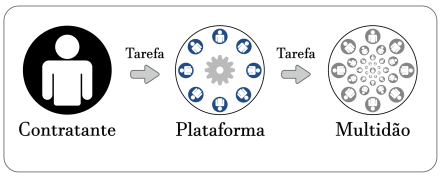

Figura 1: Componentes do modelo de *Crowdsourcing*

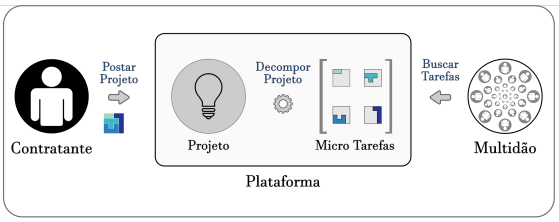

Figura 2: Representação da decomposição de tarefas ([14])

A tarefa representa o problema ou parte do problema definido pelo Contratante e é, em geral, definida pela Plataforma. Esse processo de definição e decomposição em micro-tarefas, ilustrado na Figura 2, é considerado um dos grandes desafios deste modelo de desenvolvimento [14]. Este processo de decomposição precisa resguardar que a tarefa disponibilizada na plataforma não perdeu suas características específicas e sua interdependência com as outras partes que representam o problema em questão [14].

Ao fragmentar a tarefa em micro-tarefas, deve-se resguardar que cada micro-tarefa possua informação suficiente para possibilitar que a mesma seja realizada. É preciso que a documentação seja readequada para que a micro-tarefa não seja específica demais ou ampla demais. A decomposição das tarefas acaba por tornar-se um desafio na eficiência da resolução das tarefas, uma vez que através da decomposição e da qualidade da documentação das mesmas é possível melhorar o desempenho da multidão [14].

2.2 Plataforma TopCoder

Existem plataformas especializadas em um único tipo de atividade, como por exemplo a uTest [1] ou TestBirds[2], que oferecem tarefas de teste de software; ou àquelas que servem a todo o ciclo de desenvolvimento de software, como a GetACoder[3] ou a TopCoder[4]. A última se destaca entre as plataformas comerciais por ser uma das pioneiras e por apresentar milhares de colaboradores [18].

A TopCoder foi criada em 2001 quando seu fundador, Jack Hughes, propôs uma forma inovadora de trabalho a partir de problemas identificados em suas experiências profissionais, tais como o alto custo e esforço no recrutamento de talentos, alta rotatividade dos profissionais e obsolescência na qualificação dos mesmos [6]. Hughes vislumbrava a oportunidade de reduzir os custos e o esforço de seus clientes utilizando componentes de software já existentes ao invés de construir um sistema completo a partir do zero.

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

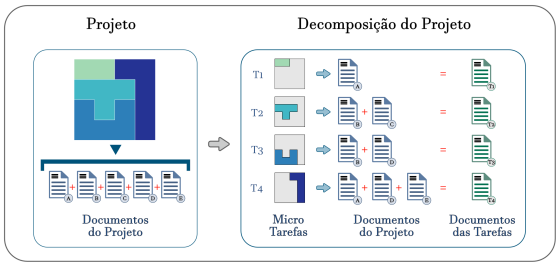

Figura 3: Processo de adequação da documentação às tarefas

Figura 4: Modelo atual da estrutura da documentação de uma tarefa na TopCoder

Assim, a TopCoder define um modelo que prioriza a reutilização de componentes para solucionar o problema dos clientes e, quando necessário, realiza uma competição para a construção de novos componentes. Como forma de viabilizar essa estratégia, foi necessário organizar uma comunidade programadores capacitados–os Top Coders–dispostos a competir na construção de novos componentes.

O processo de desenvolvimento de software proposto sugere uma abordagem similar ao modelo cascata, em que as tarefas devem seguir um fluxo determinado através da competição até a sua finalização, entrega e validação [7]. As tarefas são apresentadas à multidão através de uma descrição geral, aberta a todos os usuários da plataforma, e uma descrição detalhada associada ou não à uma documentação complementar (*e.g.*, modelos, protótipos de telas), restrita àqueles que se inscrevem na competição. A Figura 4 ilustra a estrutura típica da descrição geral de uma tarefa, composta por: (i) cabeçalho de identificação da tarefa (*e.g.*, nome, prazos, valor da premiação); (ii) informações fixas–destacadas pelos ícones pretos; e (iii) informações variáveis complementares–com ícones azuis.

3 CONTEXTO DE PESQUISA

Visando explorar o papel da documentação na seleção e desenvolvimento de tarefas em *software crowdsourcing*, realizou-se um estudo empírico sobre a plataforma TopCoder. Escolheu-se a TopCoder por ser a maior plataforma de *software crowdsourcing* da atualidade, com mais de um milhão de desenvolvedores registrados [18].

O estudo foi organizado em duas etapas. A primeira etapa foi composta de um estudo de caso com 20 novatos no modelo de *Crowdsoucing*, realizado como parte do projeto da disciplina de Desenvolvimento Colaborativo de Software do Programa de Pós-Graduação em Ciência da Computação da PUCRS. Utilizou-se um diário de bordo, acompanhado durante as 16 semanas do projeto, questionários para coleta de dados, bem como uma sessão de retrospectiva com os participantes para coletar os dados. A segunda etapa consistiu de um estudo baseado em entrevistas com 7 profissionais da indústria e utilizou-se de um questionário sobre uma listagem de tarefas pré-selecionadas pelos pesquisadores.

Ambas etapas foram inicialmente planejadas, as coletas de dados das duas etapas foram realizadas quase que concomitantemente–a segunda etapa (com profissionais da indústria) se iniciou junto com a segunda tarefa da primeira etapa (estudo em sala de aula)–e a análise dos dois conjuntos de dados foi realizada em paralelo. Este *design* de pesquisa foi intencional, para atender à restrição de tempo para realização do estudo. Não houve intenção de usar-se o segundo como forma de validar os resultados do primeiro. Acredita-se que ambos estudos são complementares visto os diferentes públicos-alvo. Os instrumentos de coleta das duas etapas foram definidos, validados e pilotados. A análise de dados seguiu uma abordagem qualitativa, em que se conduziu uma análise de conteúdo [1] e triangularizou-se os dados quando possível (por exemplo, entre os dados apontados no questionário de realização da tarefa e o diário de bordo da mesma). Detalhes sobre os métodos de pesquisa são apresentados junto ao relato de cada uma das etapas (Seções 4 e 5).

4 ETAPA 1 - ESTUDO DE CASO

O estudo de caso visou explorar como a documentação de uma tarefa influencia a seleção e o desenvolvimento da mesma por um novato em *software crowdsourcing*. Neste contexto, um novato foi definido como um desenvolvedor que estava tendo a sua primeira experiência contribuindo para uma plataforma desta natureza.

4.1 Público-Alvo

Como mencionado, o estudo foi conduzido com os 20 alunos da disciplina de Desenvolvimento Colaborativo de Software, como parte do projeto da disciplina. Dos 20 alunos 7 eram formados em Ciência da Computação, 6 em Sistemas de Informação, 5 em outras áreas (*e.g.*, Administração) e 2 Tecnólogos em TI. Os alunos possuem, em média, 12 anos de experiência em TI. Apesar de 14 dos alunos atuarem na indústria, pelo menos metade deles indicaram "não saber utilizar" grande parte das linguagens listadas (.NET, Angular, Android, Bootstrap, CSS, Docker, Groovy, Ionic, JavaScript, JSON, JSP, Node, Python e ReactJS) e metade ou mais indicaram "conhecer pouco" ou "saber utilizar" as demais linguagens (C, C++, HTML5 e MySQL). Java foi indicada por 8 alunos como "tenho domínio avançado" ou "conheço bem". Ainda, 19 dos participantes responderam que não possuíam experiência prévia com *software crowdsourcing*. O estudante que mencionou que possuía conhecimento prévio relatou "Eu tinha ouvido meus colegas falarem sobre isto" (P1), mas quando solicitado pessoalmente o mesmo confirmou que nunca tinha experimentado, qualificando-se também como novato.

A escolha dessa disciplina deu-se em função do alinhamento do objetivo da pesquisa com o conteúdo programático proposto para a disciplina. A duração do projeto foi de 16 semanas (período que compreende todo o semestre da disciplina), incluindo uma aula introdutória sobre o tópico de *software crowdsourcing* pela professora e uma apresentação sobre a TopCoder por um dos co-autores, com 15 anos de experiência na indústria.

4.2 Método de Pesquisa

Este estudo foi organizado em duas iterações visando possibilitar que os novatos se familiarizassem com a TopCoder e entendessem a lógica do seu modelo de competição. Esclareceu-se que não se esperava que o novato submetesse sua solução da tarefa na Iteração 1, visto o objetivo de familiarização. Entretanto, a submissão era esperada na Iteração 2, para que o novato pudesse experimentar a etapa de *feedback* da plataforma após a submissão da solução e soubesse se a sua havia sido vencedora (duas soluções de duas tarefas foram eleitas as ganhadoras e outros dois alunos foram compensados financeiramente pela qualidade de suas soluções).

O processo de seleção das tarefas pelos novatos foi completamente livre, apesar da sugestão dada de tentarem evitar as tarefas da categoria desafio pela restrição de prazo do projeto. Os participantes foram instruídos a selecionar, dentre as tarefas disponíveis, aquela que fossem capazes de desenvolver e submeter uma solução.

Para cada iteração, o participante precisava selecionar uma tarefa e desenvolver/submeter sua solução dentro dos prazos limites da iteração (e da própria tarefa), conforme o cronograma de 16 semanas do projeto: Iteração 1–seleção até semana 2 e submissão até semana 8; e Iteração 2–seleção até semana 10 e submissão até semana 16.

Quanto à coleta de dados, para cada uma das iterações, utilizou-se questionários e um diário de bordo. O uso de diário de bordo possibilita que o pesquisador acompanhe o comportamento periódico do participante com a mínima intrusão no seu comportamento [15]. Este tipo de coleta tem sido costumeiramente usada em estudos longitudinais em outras áreas e mais recentemente na Engenharia de Software (*e.g.*, [3, 13]). A escolha pelo uso de um diário de bordo foi pela ausência da possibilidade de se observar os participantes visto que os mesmos poderiam trabalhar quando desejassem, independentemente de local ou horário. O diário foi de formato livre. Criou-se inicialmente um diário privado entre o aluno e dois dos pesquisadores para cada iteração por aluno. Os alunos foram motivados a escrever livremente durante a realização do projeto e comentários e discussões com estes pesquisadores eram conduzidas no mínimo duas vezes por semana. Essa interação é importante pois é um mecanismo para os pesquisadores obterem o nível esperado de detalhe no estudo [12]. Interações, em geral, envolveram o esclarecimento de detalhes sobre como os alunos realizaram algo, a razão de uma certa tomada de decisão, ou a indicação de como e aonde localizaram alguma informação.

Utilizou-se três questionários, administrados via a ferramenta Qualtrics [5] em diferentes momentos do projeto, conforme detalhado a seguir. Ainda, os questionários foram validados com dois pesquisadores da área de *Software Crowdsourcing* e pilotados com 3 ex-alunos de edições anteriores da disciplina.

[5]URL: https://qualtrics.com/

(1) Perfil do Participante: Questionário aplicado na primeira semana com o objetivo de identificar o perfil dos participantes no que diz respeito à sua formação (*e.g.*, formação, tempo de experiência em TI, conhecimento em linguagens de programação) e experiência prévia com *software crowdsourcing* (*e.g.*, se possuía experiência prévia e detalhes sobre a mesma).

(2) Seleção da Tarefa: Questionário preenchido até o término da segunda semana de cada uma das iterações (semana 2 e semana 10 do projeto) com o objetivo de registrar o entendimento do participante antes de iniciar o desenvolvimento da tarefa selecionada bem como registrar a consideração de escolha por outras tarefas, conforme segue: tipo de tarefa selecionada (desafio ou codificação e seu subtipo, *e.g., bug hunt, UI design*, programação), título da tarefa, objetivo da tarefa segundo o entendimento do novato, razões por ter selecionado a tarefa e por ter descartado outras, caso se aplicasse.

(3) Realização da Tarefa: Questionário preenchido até o término da sexta semana de cada uma das iterações (semana 8 e semana 16 do projeto) e teve como objetivo coletar as percepções dos participantes quanto à realização e submissão da tarefa. Coletou-se as seguintes informações: título da tarefa, *status* da conclusão (entregue/não entregue) e, no caso de não entregue, as razões da não entrega. Também se perguntou os aspectos positivos e negativos da experiência e o nível de satisfação com a experiência. Quanto à documentação, pediu-se que o participante indicasse seu nível de concordância sobre a qualidade da documentação da tarefa baseado em um conjunto de critérios pré-definidos. Estes critérios (atômica, completa, precisa, coesa, viável, concisa, não ambígua, testável e correta) foram extraídos do BABOK [9, 10] e de Wiegers [17]. Também se coletou a percepção sobre a qualidade da descrição da tarefa, se alguma documentação complementar (*e.g.*, diagramas, padrões de desenvolvimento, critérios de aceitação) havia sido disponibilizada e a classificação geral da qualidade da documentação complementar. Por fim, o participante indicou sugestões de como melhorar a documentação (descrição geral e complementar) da tarefa.

Houve ainda uma sessão de retrospectiva na última semana (semana 16) após a entrega da segunda tarefa. A sessão de 90 minutos, realizada em sala de aula, visou a corroboração dos dados fornecidos no questionário de realização da tarefa e no diário de bordo. Essa sessão foi planejada espontaneamente durante a Iteração 2 com a intenção de resguardar que eventuais diários de bordo que não estivessem tão detalhados quanto esperado pudessem ter seu conteúdo suplementado. Entregou-se, para cada um dos participantes, uma cópia impressa da questão sobre a qualidade da documentação da tarefa baseado no conjunto de critérios pré-definidos, solicitada anteriormente no questionário, acompanhada das respostas do aluno para cada um dos critérios. Solicitou-se que o novato revisitasse a documentação de sua tarefa da Iteração 2, também entregue impressa, bem como suas respostas da questão mencionada, e então destacasse na própria documentação trechos que o tivessem levado a indicar tal nível de concordância com cada um dos critérios. Pediu-se também que fossem apontadas oportunidades de melhoria na documentação das tarefas que havia sido selecionadas e realizadas.

Quanto à análise dos dados, realizou-se análise de conteúdo seguindo as etapas propostas por Bardin [1]–organização e pré-análise, leitura e categorização,e registro dos resultados–para as questões abertas dos questionários, dos diários de bordo e do instrumento da sessão de inspeção. Assim, identificou-se temas entre

os dados coletados, conforme relatado a seguir.

4.3 Resultados

De forma geral, para a Iteração 1, 10 dos participantes sentiram-se muito satisfeitos/satisfeitos com sua participação no projeto, enquanto os outros 10 consideraram que seu envolvimento foi regular/pouco satisfatório. A satisfação reduziu na Iteração 2, passando a ter-se 8 dos novatos satisfeitos e 12 regular/pouco satisfeitos.

Todos trabalharam nas tarefas selecionadas em ambas iterações, mas apenas 10 novatos submeteram sua solução na plataforma na Iteração 1 e 11 na Iteração 2. Dentre as razões para a não submissão de uma solução na Iteração 1 estão a falta de conhecimento técnico identificado com a melhor compreensão do objetivo da tarefa (P1, P3, P11, P19), ter subestimado o tempo de desenvolvimento necessário (P5, P9, P10, P12, P17) e complicações na configuração do ambiente (P14). Já na Iteração 2, essas razões foram a falta de informação na descrição da tarefa (P3, P5, P16), dificuldade na configuração do ambiente (P6), falta de tempo para investigar uma solução (P11, P13, P14, P19) e falta de conhecimento técnico (P20).

Apesar das dificuldades ou restrições, os novatos apontaram que ficaram interessados em aprender novas tecnologias (P1, P5, P6, P9, P10, P20), buscar *feedback* para suas soluções (P8), e que se sentiram desafiados ao participar das competições (P1), e mesmo buscar uma recompensa financeira (P3, P4, P10, P12, P14). Dezesseis deles contribuiriam com a TopCoder se tivessem tempo disponível.

Tarefas Selecionadas. Na Iteração 1, foram selecionadas tarefas de programação (11 participantes), *design* de código (3), *design* de interface (2), prototipação de interface (2), *bug hunt* (1), e geração de ideias para solucionar um problema específico (1). Destas 20 seleções, 13 foram tarefas distintas. Na Iteração 2, as tarefas de programação seguiram sendo as mais selecionadas (10 participantes), seguidas de tarefas de *design* de código (5), *design* de interface (2) e prototipação de interface (3). Foram 12 as tarefas distintas.

Seleção das Tarefas. Na Iteração 1, os participantes (14 dos 20 novatos) buscaram, majoritariamente, tarefas em que tivessem conhecimento prévio para desenvolver. Alguns deles indicaram que procuraram diretamente por tarefas associadas às linguagens de programação que conheciam– *"Meu principal critério foi encontrar algo próximo do que faço. Essa tarefa tinha as tags .Net Core e SQL Server, tecnologias que conheço, então achei a melhor opção para mim"* (P20). Outros evitaram certas atividades de desenvolvimento de software– *"Busquei tarefas relacionadas à análise de produtos/negócios já que este é meu background, mas eu queria mesmo era evitar ter que codificar visto que esta não é minha atividade profissional"* (P14).

O prazo de entrega foi outra preocupação (7 dos 20 participantes), seja pela disponibilidade do participante– *"Queria uma tarefa com uma timeline confortável. Acabei por escolher uma que acredito eu terei tempo de terminar"* (P5), ou por acreditar que a estimativa apresentada pela TopCoder estava adequada– *"Acredito que o escopo desta tarefa é exequível no prazo estipulado"* (P6).

Sentir-se motivado ou desafiado a resolver o problema foi outro fator que influenciou a seleção da tarefas (citado por 2 participantes)– *"Pura curiosidade de saber se conseguirei resolver a questão"* (P18). Outros ainda queriam ter a oportunidade de aprender algo novo– *"Tenho interesse em AI e principalmente nas ferramentas da IBM para o tópico. Vou aproveitar para descobrir como elas funcionam"* (P7); ou de ser remunerado– *"... além disso [sentir-se motivada], vi potencial de ficar entre as primeiras colocadas no desafio, se não tirar o primeiro lugar, e receber uma bonificação pelo esforço realizado"*[6] (P1).

O entendimento da documentação da tarefa também aparece entre os fatores citados (4 dos 20 participantes)– *"Dentre todas atividades abertas que eu me senti capacitado para resolver, esta foi a que me pareceu com a melhor apresentação, o enunciado estava bem acessível"* (P9) e *"O que me cativou foi a simplicidade do enunciado. Foi muito fácil de entendê-lo"* (P18).

Na Iteração 2, a preocupação em ter conhecimento prévio para desenvolver, incluindo dominar as linguagens de programação/ tecnologias necessárias, também prevaleceu (11 do 20 participantes)– *"Ser uma tarefa de design gráfico, o que faço no meu dia-a-dia, foi o principal quesito"* (P9) e *"Conheço bem bootstrap, HTML e CSS. Dentre as opções que vi, me pareceu a tarefa que mais se encaixava ao meu perfil"* (P12). Entretanto, neste ciclo, mais novatos vislumbraram a oportunidade de aprender algo novo ao realizar a tarefa (7 novatos citam este fator)– *"Admiro as plataformas blockchains. Como invisto em bitcoins mas sem saber ao certo como moedas digitais e blockchains se comportam, resolvi tentar esta tarefa para aprender sobre como elas trabalham por baixo dos panos"* (P8).

A preocupação com o prazo de entrega reduziu-se (apenas 2 dos participantes)– *"Acredito que conseguirei concluir no prazo"* (P4) e a expectativa de ser remunerado pela submissão, apesar de um dos participantes ter sido bonificado anteriormente, desaparece.

A necessidade de entender o enunciado a partir da documentação apresentada é igualmente mencionada (4 dos participantes)– *"Trabalho com visualização de dados. Ao ler o título da tarefa, seguido da detalhada e rica documentação complementar, fiquei interessada"* (P1) e *"A API é feita em JavaScript e tem um escopo bem compreensível. Foi bem descrita na [documentação da] tarefa"* (P7).

Um achado interessante é que os fatores que motivaram a seleção de um tarefa também foram influentes na **desconsideração de outras tarefas**. Por exemplo, na Iteração 1, a preocupação com a falta de conhecimento foi o fator primordial entre os novatos (citado por 18 dos 20 participantes)– *"O fato delas [as tarefas] esperarem programação em linguagens que não domino foi desmotivante. Descartei de cara estas tarefas"* (P8) e *"Apesar de trabalhar com design gráfico, algumas tarefas exigiam conhecimento específicos tipo o Adobe Illustrator"* (P14). Este fator foi seguido da preocupação com o prazo de entrega da tarefa (mencionado por 7 participantes)– *"As tarefas de codificação parecem ter um tempo muito curto, o que me faz pensar que elas realmente sejam voltados para especialistas com a infraestrutura e ambiente de desenvolvimento prontos"* (P5). A preocupação com o entendimento e a qualidade da documentação também apareceu entre os participantes (citado por 3 deles)– *"Impossível realizar uma tarefa com falta de clareza no que é esperado. Precisei mais explicações e não consegui achar nem no material complementar"* (P19).

Igualmente, na Iteração 2, as tarefas foram majoritariamente desconsideradas pela falta de conhecimento (16 participantes)– *"Pelo enunciado, inferi que precisava um conhecimento vasto e eu apenas tenho uma noção desta linguagem"* (P14). E pela preocupação com o prazo de entrega e o tempo a ser investido (7 participantes)– *"Muito estranho, nessa etapa as tarefas estão tendo 2 ou 3 dias para resolução*

[6]Esta aluna de fato ganhou primeiro lugar nesta sua tarefa.

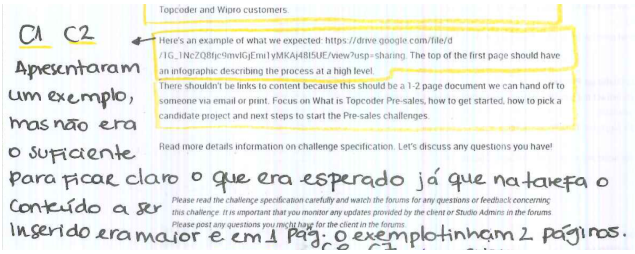

Figura 5: Exemplo de comentário sobre a qualidade da documentação

apenas" (P9). A falta de motivação em resolver a tarefa foi citada como motivo por 2 participantes: *"Mais pela questão da falta de interesse no assunto mesmo"* (P8), bem como a expectativa de ser remunerado (1 citação)–*"Queria considerar este desafio mas ia ser só para brincar, não tem premiação"* (P7).

O entendimento e qualidade da documentação das tarefas passa a ser mais destacado (7 participantes)–*"Não consegui entender a descrição do problema, estava muito mal escrito"* (P10) e *"Definitivamente não entendi a regra de negócio associada. Cheguei até a pedir ajuda a um colega, que também não entendeu"* (P11).

Critérios de Qualidade das Tarefas. No que diz respeito à qualidade da documentação das tarefas selecionadas na Iteração 1, 18 dos 20 participantes indicaram que a mesma era "excelente/muito boa/boa" e dois indicaram que era "de baixa qualidade". Quanto aos critérios específicos de qualidade, as tarefas foram consideradas majoritariamente (16 ou mais participantes indicaram concordo totalmente/concordo parcialmente) como atômicas e com descrições completas, precisas, coesas, viáveis (de serem desenvolvidas), concisas, não ambíguas e corretas. Apesar disto, ilustra-se na Figura 5 um exemplo de comentário que indica a razão pela qual uma participante acredita que a descrição da tarefa não era atômica nem mesmo completa. Ainda, dentre os que discordaram que a documentação é precisa, tem-se *"Algumas vezes a informação é muito sucinta e em outros casos, muito farta. O ideal era ter apenas o que se precisa"* (P2). E um argumento discordando que a informação é correta: *"...na descrição da tarefa é mencionado que tem um material adicional no fórum que especifica as métricas a serem mensuradas mas eu não consegui encontrar o material e depois achei uma nota no fórum indicando outro local"* (P1).

O critério testável teve 10 participantes concordando e 10 sem opinião formada (nem concordo/nem discordo)–*"Eu não acho que seja possível testar estes cenários, mas vou tentar"* (P6).

Na Iteração 2, houve um decréscimo quanto à percepção da qualidade geral da documentação. Treze participantes indicaram ser "muito boa/boa" e 7 opinaram ser "de baixa qualidade/pobremente descrita". Esta percepção foi refletiva na avaliação dos critérios de qualidade da documentação das tarefas. Fora os critérios viável e concisa (17 participantes concordaram totalmente/concordaram parcialmente), os demais critérios todos–atômica, completa, precisa, coesa, não ambígua, testável e correta, foram indicados por 10 participantes como nem concordo, nem discordo/discordo parcialmente/discordo totalmente. Parte dos comentários detalhados sobre estes critérios é relatado como aspecto de melhoria a seguir.

Melhorias na Documentação das Tarefas. Apesar de alguns participantes terem apontado que *"a documentação estava bem clara"* (P6) e *"completa"* (P18), com suas experiências e olhar de *novatos*, os participantes apontaram um conjunto de melhorias na documentação das tarefas na TopCoder na Iteração 1. Um participante sugeriu informações mais detalhadas no enunciado apresentado em chamada aberta–*"Na minha opinião faltaram informações primordiais"* (P9). Outro acredita que junto ao título da tarefa deveria ser incluído *"um breve resumo da tarefa, em duas ou três linhas, pois foi preciso muito boa vontade e quase mágica para inferir do que algumas tarefas se tratavam"* (P3).

Quanto à documentação complementar, acessível pelas pessoas que se inscrevem na tarefa, sugeriu-se que a mesma *"incluísse quem seria o público-alvo que iria receber o relatório entregue como parte da atividade, para que pudesse usar um linguajar para este público"* (P1) e ficasse disponível de imediato *"já no momento da inscrição na tarefa"* (P16). Também se observou que *"... vídeos explicativos ou diagramas para complementar a explicação textual poderiam melhorar a compreensão do que deve ser feito"* (P11) e que *"mockups e a descrição do comportamento esperado pelos stakeholders teria ajudado a entender o contexto de uso e a real necessidade da solicitação"* (P7).

Os participantes sugeriram que fosse adicionado explicitamente à documentação informações sobre os pré-requisitos para realização da tarefa–*"deveriam deixar mais claras as ferramentas necessárias para a execução, por exemplo "* (P21); os critérios de aceitação–*"seria importante saber os critérios e testes da qualidade da solução"* (P4) e *"acredito que um checklist para indicar que a tarefa foi terminada corretamente poderia ajudar muito, tipo os critérios de aceite do ágil"* (P10); e a *forma de avaliação dos resultados entregues*–*"eu gostaria de saber como vai ser feita a escolha da melhor solução"* (P4). Ainda, que seja explicitada a forma de submissão da solução–*"faltou clareza na forma de submissão das tarefas. A tarefa em si estava clara, mas a forma de submissão na plataforma poderia ter sido melhor explicada"* (P15) e quais artefatos devem ser entregues–*"faltou explicar o que precisamente era necessário conter nos arquivos entregáveis, já que era necessário explicar sobre o design do produto"* (P17).

Recomendaram também que os autores da documentação das tarefas busquem sempre *"melhor descrever os requisitos"* (P4) para que *"se reduza a necessidade de ficar vasculhando o fórum da tarefa e tendo que ler páginas e mais páginas de esclarecimentos que deveriam ser parte da documentação da tarefa"* (P2), ou ainda, *"ter que ficar perguntando e receber de volta respostas não satisfatórias"* (P9).

As sugestões de melhoria se repetiram quase que na íntegra na Iteração 2. Alguns participantes mencionaram que seria importante, quanto à descrição geral da documentação, *"apresentar descrições que separem claramente os aspectos técnicas das instruções sobre a TopCoder"* (P1) e *"Buscar estruturar um pouco a descrição da tarefa, tipo, indica o objetivo, o que deve ser feito e entregue, assim qualquer um vai poder entender"* (P6). A padronização da estrutura das tarefas chegou a ser recomendada–*"Cada tarefa tem um formato diferente. É difícil assim. Seria legal se houvesse um padrão"* (P14) e *"A padronização poderia vir no formato de um checklist, assim facilmente se identificaria se falta alguma informação"* (P19).

Adicionaram ainda, quanto à documentação complementar, que deveria ser indicadas instruções para a configuração do ambiente–*"teria sido útil saber mais sobre a montagem da IDE e a configuração das bibliotecas"* (P11) e *"melhorar os detalhes técnicos do setup de*

ambiente. Talvez criar uma sessão separada para isso?" (P13). E reforçaram a importância de se indicar os pré-requisitos da tarefa– *"ficou faltando eles me dizerem que eu precisava fazer o download da carteira da Ethereum. Levou horas para descobrir que isto era um pré-requisito"* (P8) e *"citar as ferramentas necessárias"* (P17); bem como os critérios de aceitaçáo– *"pode ser melhor descritos os termos de aceite para a tarefa com os principais pontos que devem ser atendidos"* (P12) e detalhes da forma de submissão da solução– *"a maior dificuldade é identificar o que está sendo solicitado como entrega. Em vários locais não é informado o que é desejado (se um arquivo de um programa específico, como Sketch, Photoshop, Microsoft SQL) ou se é uma imagem genérica... Então ter uma caixa que indicasse claramente quais são os artefatos a serem entregues ... ajudaria não só a identificar o que é desejado mas também a filtrar as tarefas as quais me considero apto a realizar"* (P3).

4.4 Limitações

Tendo em vista a natureza empírica deste estudo, o mesmo apresenta algumas limitações. O número de participantes foi limitado, o que não possibilita a generalização dos resultados. Como forma de mitigar esse fator, o grupo de participantes possui experiência diversificada de mercado e acadêmica o que proporcionou um espectro mais amplo de perspectivas. Outra limitação é relacionada às tecnologias (linguagens, frameworks, etc.) necessárias à realização das tarefas, o que tornou a escolha de tarefas pelos participantes mais complexa. O tempo reduzido de realização de algumas tarefas pode ter imposto aos participantes uma certa restrição na seleção de tarefas, visto que a submissão da solução era esperada. Ainda, a plataforma TopCoder ficou fora do ar por 5 dias consecutivos próximo ao prazo limite da Iteração 2 (parte das semanas 14 e 15), o que pode ter prejudicado alguns novatos com disponibilidade neste período. Os alunos também notaram que neste mesmo período as tarefas disponíveis possuíam de 2 a 3 dias de prazo para submissão da solução, o que não é comum na plataforma. Uma estratégia utilizada para minimizar o impacto da seleção das tarefas realizadas foi a liberdade que os participantes possuíam para pesquisar tarefas e selecioná-las. Com isso, cada participante, independente de sua experiência anterior, poderia selecionar uma tarefa a qual possuísse interesse sem nenhuma intervenção dos pesquisadores. Neste sentido, houve uma reprodução do processo real de seleção das tarefas. Estas são estratégias consideradas adequadas para melhorar a validade dos resultados de um estudo de caso [2] e foram usadas extensivamente neste estudo.

5 ETAPA 2 - ESTUDO BASEADO EM ENTREVISTAS

O estudo baseado em entrevistas visou explorar o papel da documentação no processo de seleção de uma tarefa por um profissional da indústria. Definiu-se como profissional um desenvolvedor com experiência em projetos reais. Estes profissionais não precisavam ter experiência prévia com *software crowdsourcing*, visto o *design* do estudo (veja Seção 5.2). O estudo limitou-se a explorar o processo de seleção e não o de submissão de solução de uma tarefa visto o custo em horas para tal.

5.1 Público-Alvo

Por conveniência, selecionou-se 10 profissionais que se conhecia *à priori*, com conhecimento nas linguagens de programação Angular e/ou Java (as linguagens com maior número de tarefas na TopCoder nos seis meses anteriores ao estudo). Os profissionais trabalham no departamento de TI de um órgão público, vínculo profissional de um dos co-autores. Dos 10 profissionais apenas 7 deles de fato entregaram suas considerações. Estes 7 participantes possuem, em média, 11 anos de experiência em TI. Todos se auto avaliaram com o perfil sênior em desenvolvimento Java e de conhecimento variado em Angular (2 sênior, 3 pleno e 2 júnior).

O estudo realizou-se no período de duas semanas, organizado em três grandes atividades, conforme detalhado a seguir.

5.2 Método de Pesquisa

Para atingir o objetivo do estudo e facilitar a participação dos profissionais selecionados, um conjunto de tarefas foi pré-selecionado pelos pesquisadores para ser disponibilizado aos participantes. Focou-se nas tarefas que mencionavam o uso das linguagens Angular e/ou Java. De um total de 150 tarefas disponíveis, 15 tarefas (6 em Angular e 9 em Java) foram selecionadas. A seleção destas tarefas foi baseada nas seguintes premissas: (i) possuir o acesso ao fórum de discussão, uma vez que ali são fornecidas informações complementares que auxiliam na compreensão das tarefas, e (ii) buscar linguagens conhecidas pelos profissionais (Angular e Java). As tarefas foram selecionadas aleatoriamente para que os pesquisadores não adicionassem viés à seleção. De posse destas tarefas, conduziu-se o estudo em três passos: apresentação do estudo e da atividade a ser realizada; seleção e análise das tarefas pelos participantes; e uma conversa *follow-up* com os participantes na entrega dos resultados.

Em uma sessão de 60 minutos, um dos pesquisadores fez uma breve apresentação sobre *software crowdsourcing* e a TopCoder, e sobre o estudo e a razão dos participantes terem sido convidados. Em seguida, foram detalhados os procedimentos envolvidos, incluindo uma discussão sobre o instrumento de coleta de dados. Ao término da apresentação, os participantes receberam uma cópia impressa das tarefas e do questionário de resposta, e foram instruídos a retornarem suas considerações em uma semana. Também foram instruídos a não conversarem entre si. Distribuiu-se um usuário e senha genéricos criados para o estudo, de forma que os participantes pudessem consultar a documentação complementar da tarefa.

Conforme os participantes foram concluindo suas escolhas, os mesmos agendavam a conversa (*follow-up*) com o pesquisador. Esta conversa informal teve como objetivo oferecer ao pesquisador a oportunidade de revisar o material entregue e do participante expressar qualquer consideração sobre a sua experiência.

O questionário foi elaborado baseado nos instrumentos aplicados no estudo de caso, e organizado em três seções: perfil do participante (*e.g.,* tempo de experiência em TI e conhecimento em Angular e Java); listagem e indicação de tarefas selecionadas; e análise das tarefas. Na listagem das tarefas, indicou-se os títulos das 15 tarefas organizadas por linguagem de programação e instruiu-se o participante a consultar o material das tarefas e indicar quais tarefas o mesmo tinha se interessado em ler por completo e quais ele selecionaria caso tivesse a oportunidade de desenvolver uma solução. Nota-se que não se impôs que o participante lesse a documentação

das 15 tarefas para se poder explorar quais documentações chamariam a sua atenção. Também se deixou claro que o participante poderia optar por não selecionar nenhuma tarefa ou selecionar todas, conforme julgasse relevante. Na última seção, solicitou-se que o participante relatasse sua percepção quanto à documentação das tarefas selecionadas. Perguntou-se sobre os motivos para a seleção das tarefas, as razões por desconsiderar as tarefas não selecionadas, e indicar sua opinião quanto à documentação das tarefas selecionadas. Esta análise se deu de forma similar ao questionário de Avaliação da Tarefa do estudo de caso, em que o participante indicou sua opinião sobre os critérios de qualidade da documentação, detalhou os trechos da documentação que lhe levou a avaliar os critérios como indicado, apontou aspectos de melhoria na documentação, e indicou sua motivação para de fato contribuir com *software crowdsourcing* a partir desta experiência.

O questionário foi avaliado pelos mesmos 2 pesquisadores da etapa anterior e pilotado com 2 profissionais com experiência em Java e em Angular, com 12 anos de experiência em TI cada. Um dos profissionais foi quem sugeriu a disponibilização do usuário e senha genéricos, e o outro acabou por sugerir que o pesquisador exemplificasse durante a sessão de apresentação como avaliar a qualidade da tarefa para resguardar a qualidade das contribuições.

5.3 Resultados

Baseados na experiência com este estudo, três dos participantes relataram sentirem-se muito motivados ou motivados a tentarem contribuir com a TopCoder de fato. Os demais (4 deles) não tiveram opinião formada, *i.e.,* não se sentiram nem motivados nem desmotivados. Oportunidade de recompensa (P21, P23, P24, P25, P27), ganho de experiência e conhecimento (P21, P22, P24, P25), oportunidade de aprender novas tecnologias (P22, P23), e flexibilidade de local e horário de trabalho (P23) foram alguns dos fatores citados que levariam os profissionais a contribuir com a TopCoder.

Por sua vez, fatores como complexidade das tarefas (P26) para o curto prazo de entrega determinado (P23, P26, P27), bem como uma documentação *"de certa forma deficitária"* (P23) seriam razões para os profissionais não se aventurarem no *software crowdsourcing*. Destaca-se a ressalva feita por um dos participantes: *"Vislumbro que quem tem tempo para se envolver nessas tarefas, consegue adquirir uma boa base de código que será possível reaproveitar em outras tarefas. Dessa forma, é provável que consiga terminar no prazo estipulado e com mais qualidade de código."* (P26)

Neste contexto, apresenta-se nesta seção os resultados do estudo baseado em entrevistas em relação às tarefas lidas e selecionadas, os critérios para tal seleção (ou desconsideração de certas tarefas), o entendimento sobre os critérios de qualidade da documentação disponibilizada, e as sugestões de melhoria nesta documentação.

Tarefas Lidas e Selecionadas. Do total de 105 possíveis combinações de tarefas (=15 tarefas * 7 participantes), 58 foram lidas e 33 foram selecionadas (Tarefa Java 1–J1– foi selecionada por 4 participantes, J2–3 vezes, J3–1, J4–2, J6–4, J7–2, J8–2, J9–2, Tarefa Angular 1–A1– foi selecionada 1 vez, A2–4, A3–2, A4–2, A5-1, e A6-3 vezes). A Tarefa Java 5 foi a única não selecionada.

Seleção das Tarefas. Dentre os fatores que influenciaram a decisão de seleção de uma determinada tarefa, os participantes indicaram a possibilidade de utilizar tecnologias recentes com o intuito de aprender algo novo– *"Fiquei com interesse de conhecer estes frameworks"* (P26) e *"Seria trabalhoso, mas eu aprenderia"* (P25). Também sentiram-se motivados a fazer algo útil– *"A tarefa vai facilitar a inclusão de uma política de segurança. É algo útil em um app"* (P25) ou foram atraídos pela possibilidade de ser remunerado– *"Considerei o esforço necessário em relação à recompensa"* (P21).

Entretanto, dentre os fatores mais citados (5 dos 7 participantes) tem-se o conhecimento prévio para resolver a tarefa– *"Possuía as habilidades necessárias para resolver a tarefa"* (P23) e *"Ter domínio da linguagem de programação associada à familiaridade com o tipo de trabalho prestado "* (P24). O entendimento da documentação foi considerado, mais especificamente a clareza do que foi solicitado, e o nível de detalhe técnico apresentado nesta documentação– *"Pela clareza da documentação como um todo"* (P23), *"Escolhi primariamente quando a tarefa possuía um melhor nível de detalhamento técnico* (P22), e *"O nível de detalhamento das documentações foi essencial para as minhas escolhas"* (P27).

Alguns fatores foram similarmente usados na **desconsideração de uma tarefa**. Por exemplo, por não identificar a utilidade de *"uma app para motos"*, a tarefa foi descartada pelo participante P25. Ou, por não conseguir *"estimar o esforço em relação à recompensa oferecida usando a documentação impressa [descrição geral] ou mesmo a documentação complementar [acessada na plataforma]"* (P21).

A falta de conhecimento prévio foi um fator largamente citado (5 dos participantes) no descarte de uma tarefa– *"sescartei as que envolviam tecnologias que eu possuo pouco ou nenhum conhecimento"* (P23) e *"pelo título da tarefa, consegui descartar aquelas que eu não conhecia a tecnologia ou os frameworks a serem utilizados"* (P26). Tarefas julgadas como de alta complexidade (P1, P25), em especial *"pelo pouco tempo disponibilizado para realização"* (P26), ou de escopo abrangente (P22) também foram eliminadas após lidas.

Entretanto, a falta de detalhes na apresentação da documentação foi o fator mais relatado, implicando em informações incompletas (P21, P22), na falta de clareza dos requisitos solicitados (P21, P26, P27) e na não compreensão como um todo do que estava sendo solicitado pela tarefa (P25, P27).

Critérios de Qualidade das Tarefas. Considerando a documentação das tarefas selecionadas de forma geral, 6 participantes indicaram que a mesma era "muito boa/boa" e um indicou que era "de baixa qualidade". Quanto aos critérios específicos de qualidade, as tarefas foram consideradas majoritariamente (5 ou mais participantes indicaram "concordo totalmente" ou "concordo parcialmente") como atômicas e com descrições coesas, concisas, não ambíguas e corretas. Mais especificadamente, os participantes que consideraram que a documentação não expressava uma tarefa atômica indicaram que *"Na minha visão, cada componente no diagrama poderia ser uma tarefa específica"* (P22) e *"Precisei consultar outras tarefas para poder compreender esta aqui, que acabou não sendo uma tarefa independente das demais"* (P24). Outra tarefa foi considerada ambígua por *"A visão [descrição] geral da tarefa e os requisitos específicos na documentação complementar dão margem para duas interpretações, não saberia qual usar"* (P25).

Por sua vez, 5 dos participantes entenderam que as documentações eram problemáticas no que diz respeito à precisão da descrição da tarefa e em identificarem se a tarefa era viável. Quanto à precisão da descrição, um participante argumentou que *"simplesmente falou clareza na especificação"* (P21) e outro detalhou *"achei a documentação meio confusa. Tinha muito detalhe específico mas não havia uma visão mais ampla do objetivo geral da tarefa"* (P22). Este mesmo profissional comentou sobre à precisão de outra tarefa: *"a tarefa pareceu interessante, a documentação relativamente bem escrita, porém, está em muito alto nível. Está mais em nível arquitetural do que em nível pronto para ser desenvolvido. Acabo não sabendo exatamente o que deve ser feito, tem margem para interpretação."* (P22). Já quanto à tarefa ser viável, um participante destacou a possível inviabilidade técnica: *"acredito que a tarefa, por sua complexidade, deve gerar diversos defeitos, principalmente na parte de configuração do ambiente. Não sei se seria possível gerenciar isto"* (P27), enquanto outros relataram a inviabilidade em função do curto prazo estabelecido *"a tarefa necessita um sistema similar para adaptar a solução; precisaria aumentar o prazo de entrega"* (P26) e *"achei ambas tarefas complexas para o curto prazo. Acho que não seria possível concluí-las"* (P23).

Por fim, não houve consenso sobre o entendimento da documentação das tarefas serem completas e testáveis (3 concordaram e 4 discordaram, respectivamente). Por exemplo, alguns participantes argumentaram positivamente quanto à completude da documentação *"Talvez esta tenha sido a melhor documentação [dentre todas lidas]"* (P22), *"O documento possui os passos das atividades a serem realizadas de forma bem detalhada"* (P23) ou *"A documentação possui um diagrama que auxilia no entendimento do escopo"* (P23), *"... não poderia estar mais completa, possui inclusive links para um material auxiliar"* (P25) e *"Ainda por cima, [a documentação] também especifica as ferramentas a serem utilizadas e os padrões de codificação a serem seguidos"* (P27). Enquanto outros postularam contrariamente *"A documentação está muito rasa"* (P22), *"Senti falta da descrição das tecnologias a serem usadas"* (P21) e *"A tarefa pede para baixar um arquivo com a documentação complementar, mas pelo resumo este requisito não é parte do sistema [tarefa]"* (P26).

Melhorias na Documentação das Tarefas. Baseados na descrição da tarefas selecionadas, os participantes sugeriram a melhoria de alguns aspectos, como prazos menos exíguos– *"Uma ideia seria ter prazos maior e, sendo o caso, oferecer um bônus a quem entregar antecipadamente"* (P26) e tarefas de caráter mais atômico– *"Algumas tarefas vão contra a ideia de tarefas pequenas, deveriam ser quebradas em tarefas menores para dizerem respeito a um único item"* (P22).

Mencionaram também que seria importante resguardar a clareza dos requisitos– *"Gostaria de mais clareza e objetividade nos requisitos de negócio e melhoria na descrição dos requisitos não funcionais"* (P21). Incluir, sempre que possível, uma documentação complementar–seja a mesma técnica ou de outra natureza, para o entendimento da tarefa a ser realizada- *"Acho que poderia ter mais diagramas para auxiliar no entendimento, prints de telas ou wireframes"* (P23), *"Os links para baixar documentação complementar ajudaram no entendimento da tarefa e deixaram a mesma mais completa"* (P25), e *"Seria importante saber a estrutura [de código] do projeto como um todo a fim de orientar o desenvolvimento a ser feito"* (P26).

Uma padronização na estrutura das tarefas, incluindo a descrição e apresentação da documentação das tarefas também foi solicitada– *"Notei uma grande variedade de formato de descrição das tarefas. Poderia ter um padrão no site [da plataforma] para facilitar"* (P22), e *"Acho que a definição das ferramentas a serem utilizadas e os padrões de código a serem seguidos são pontos fundamentais e deveriam ser sempre apresentados"* (P27).

5.4 Limitações

Este estudo também teve limitações. A seleção dos participantes (baseada em conveniência) e o número restrito de envolvidos não permite a generalização dos resultados [8]. Entretanto, efeitos da seleção foram minimizados pelo fato dos profissionais opinarem sobre tarefas em tecnologias distintas (Java e Angular) e ter uma amostra para seleção conforme desejo do próprio profissional. A escolha dessas duas tecnologias foi para propiciar aos participantes um conjunto maior de tarefas a sua disposição uma vez que estes possuem conhecimento sobre as mesmas. Ainda, por terem uma experiência profissional recente similar (na mesma empresa), permitindo que efeitos de experiência prévia fossem considerados. Teria sido oportuno que os profissionais pudessem ter desenvolvido e submetido a solução de suas tarefas. Apesar de não disporem de tempo para tal contribuição, todos os profissionais consultaram as documentações complementares disponíveis na plataforma, dando aos mesmos um senso de uso real da TopCoder e aproximando do realismo [8] de realização da tarefa.

6 DISCUSSÃO

Neste estudo, por meio de análise qualitativa de dados coletados por meio de diários e questionários com estudantes, e entrevistas com profissionais, pode-se observar que o processo de seleção das tarefas se baseia fortemente nas informações providas na descrição da tarefa. A maioria dos novatos mencionou que a escolha das tarefas se dá baseada no conhecimento prévio sobre linguagens e conhecimentos requeridos. Tal fator também foi relatado por profissionais. Além disso, entendimento da documentação da tarefa foi mencionado como um fator que influencia a escolha das tarefas. De forma análoga, a preocupação com o entendimento e qualidade da documentação foi relatado como razão para desconsiderar algumas tarefas. Este último foi o fator mais relatado pelos profissionais, que se preocuparam com a falta de compreensão do que era solicitado.

Os participantes fizeram ainda recomendações de melhoria da documentação baseados em suas experiências com o estudo. Estas recomendações são expressadas através de um conjunto de informações que, se apresentadas como parte da documentação de uma tarefa devem facilitar o processo de seleção e desenvolvimento da mesma. Estas informações são destacadas na Figura 6.

Mais especificamente, as informações referentes ao resumo da tarefa, o detalhamento e compensação financeira (pagamentos), apresentadas como itens padrão em todas as tarefas pela TopCoder (ícones pretos) continuam sendo importantes. Informações sobre as plataformas e tecnologias utilizadas, e as regras de submissão, apresentados variavelmente conforme a tarefa (ícones azuis), foram sugeridas tornarem-se obrigatórias pelos participantes. A apresentação destas informações possibilitaria a rápida avaliação se a pessoa possui conhecimento prévio, se recursos estão disponíveis para desenvolver a tarefa, bem como tomar conhecimento sobre do que

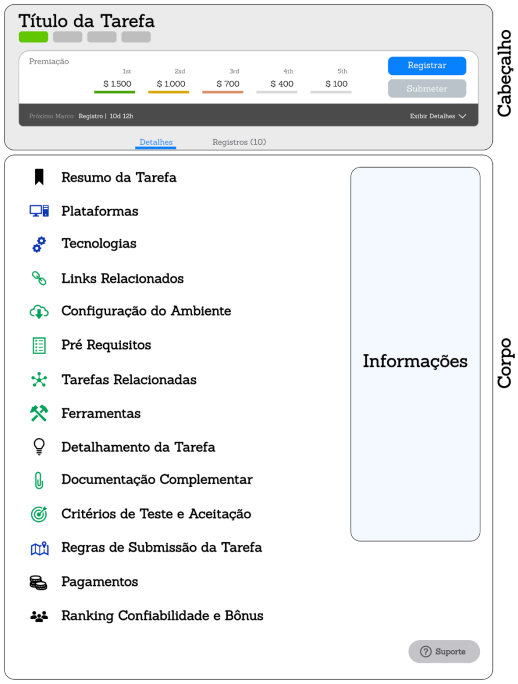

Figura 6: Resumo de informações sugeridas para a melhoria da documentação de uma tarefa

é esperado em termos de artefatos a serem submetidos.

Ainda, identificou-se que um conjunto de informações que são esporadicamente apresentadas como parte do corpo da descrição da documentação complementar e deveriam ser explicitamente citadas em todas as tarefas para facilitar a identificação de pré-requisitos e o entendimento do que é solicitado (ícones verdes). Estas informações são: links relacionados, configuração do ambiente pré-requisitos, tarefas relacionadas, ferramentas, documentação complementar e critérios de aceite. A disposição das mesmas é ilustrada na Figura 6. O item *Ranking*, Confiabilidade e Bônus, para regras sobre o posicionamento da pessoa junto à TopCoder, não diz respeito à tarefa mas é destacado visto que é intrínseco à plataforma em si.

É importante reforçar que este trabalho propõe um modelo de documentação para as tarefas na forma de uma estrutura lógica para as informações visando padronizar o documento e facilitar sua leitura e entendimento. Entretanto, esta pesquisa não define ou impõe a maneira como a informação deve ser descrita. Assim, é fundamental que a documentação siga um conjunto de regras e padrões na sua forma de registrar os fatos (textualmente ou com o uso de elementos gráficos), de modo a tornar a informação clara e objetiva a todos que se propuserem a ler. A mudança no processo de escrita implica que o contratante tenha maior cuidado na sua confecção a fim de manter a estrutura lógica das informações em cada uma das seções.

7 CONCLUSÃO

O modelo de *crowdsourcing* tem sido amplamente adotado e estudado na área de desenvolvimento de software. No modelo competitivo, a documentação (descrição e materiais de apoio) oferecida pela plataforma serve de base para que os membros selecionem as tarefas que consideram adequadas e submetam soluções. Este artigo apresentou um estudo empírico sobre o papel dessa documentação em um ambiente de *software crowdsourcing* competitivo na escolha e execução das tarefas pelos membros da *crowd*. Identificou-se fatores que influenciam a seleção e desenvolvimento de uma tarefa tendo como base a documentação disponibilizada bem como quais critérios definem a qualidade desta documentação. Por fim, pode-se identificar uma série de sugestões para melhoria na documentação das tarefas, resumidas na Figura 6. Acredita-se que apresentando documentação mais focada, e como detalhes suficientes seja possível melhorar o processo de escolha e execução das tarefas, resultando em uma seleção de pessoas mais apropriadas e melhores soluções.

8 AGRADECIMENTOS

Agradecemos aos participantes envolvidos neste estudos. Esta pesquisa foi parcialmente apoiada pelo Convênio Dell/PUCRS, financiado pela Lei de Informática nro. 8.248/91. Também agradecemos o apoio financeiro do CNPq (processo 430642/2016-4)

REFERÊNCIAS

[1] Laurence Bardin. 2016. *Análise de Conteúdo.* Edições 70, São Paulo, Brasil.
[2] John Creswell. 2018. *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches* (5nd ed.). Sage Publications, Los Angeles, USA.
[3] Jennifer L. Davidson, Umme Ayda Mannan, Rithika Naik, Ishneet Dua, and Carlos Jensen. 2014. "Older Adults and Free/Open Source Software: A Diary Study of First-Time Contributors". In *OpenSym '14.* ACM, New York, NY, USA.
[4] Mahmood Hosseini, Keith Phalp, Jacqui Taylor, and Raian Ali. 2014. "The four pillars of crowdsourcing: A reference model". In *RCIS 14.* IEEE, 1–12.
[5] Jeff Howe. 2008. *Crowdsourcing: Why the Power of the Crowd is Driving the Future of Business.* Random House Business, New York, NY, USA.
[6] Karim Lakhani, David Garvin, and Eric Lonstein. 2010. "Topcoder (A): Developing software through crowdsourcing". *Harvard Business Sc.* 610, 32 (2010).
[7] Ke Mao, Licia Capra, Mark Harman, and Yue Jia. 2017. "A survey of the use of crowdsourcing in software engineering". *Journal of Systems and Software* 126 (apr 2017), 55–87.
[8] Joseph E. Mcgrath. 1995. Methodology matters: Doing research in the behavioral and social sciences. *Readings in Human–Computer Interaction* 2 (1995), 152–169.
[9] International Institute of Business Analysis IIBA. 2009. *A Guide to the Business Analysis Body of Knowledge (BABOK), Version 2.0.* International Institute of Business Analysis, Toronto, ON, Canada.
[10] International Institute of Business Analysis IIBA. 2015. *Guide to the Business Analysis Body of Knowledge (BABOK) V3.* International Institute of Business Analysis, Toronto, ON, Canada.
[11] Rafael Prikladnicki, Leticia Machado, Erran Carmel, and Cleidson de Souza. 2014. "Brazil software crowdsourcing: a first step in a multi-year study". In *Int'l Workshop on CrowdSourcing in Software Engineering.* ACM, New York, USA, 1–4.
[12] J Rieman. 1993. "The Diary Study: A Workplace-oriented Research Tool to Guide Laboratory Efforts". In *Conference on Human Factors in Computing Systems (CHI 93).* ACM, New York, NY, USA, 321–326. https://doi.org/10.1145/169059.169255
[13] Igor Steinmacher, Tayana Conte, Christoph Treude, and Marco A. Gerosa. 2016. Overcoming Open Source Project Entry Barriers with a Portal for Newcomers. In *ICSE '16.* ACM, New York, NY, USA, 273–284.
[14] Klaas-Jan Stol and Brian Fitzgerald. 2014. "Two's company, three's a crowd: a case study of crowdsourcing software development". In *International Conference on Software Engineering (ICSE 2014).* ACM, New York, NY, USA, 187–198.
[15] Gillian Symon. 2004. *Qualitative research diaries. Essential Guide to Qualitative Methods in Organizational Research.* Sage Publications Ltd, London, UK.
[16] Nguyen Hoang Thuan, Pedro Antunes, and David Johnstone. 2015. "Factors influencing the decision to crowdsource: A systematic literature review". *Information Systems Frontiers* 18, 1 (jun 2015), 45–68.
[17] Karl Wiegers. 2013. *Software Requirements – Third Edition.* Microsoft Press, Redmond, WA, USA.
[18] Alexandre Lazaretti Zanatta, Leticia Machado, Graziela Pereira, Rafael Prikladnicki, and Erran Carmel. 2016. "Software Crowdsourcing Platforms". *IEEE Software* 33, 6 (nov 2016), 112–116.