# Um Modelo para o Gerenciamento do Crowdsourcing em Projetos de Software

**William Simão de Deus**[1], **Renata Marques Barros**[1], **Alexandre L'Erario**[1]

[1] Programa de Pós Graduação em Informática (PPGI)
Universidade Tecnologica Federal do Parana (UTFPR)
Cornélio Procópio – Paraná – Brasil

{williamsimao,renata_mbarros}@outlook.com, alerario@utfpr.edu.br

***Abstract.*** *Crowdsourcing is an emerging business model applied to software development projects due to the cost reduction and involvement of experts. However, there are challenges on how to manage it in such projects due to the dispersion of participants and the activities of parallelization. In this study, we conducted a literature review and verified practices applied to the management of crowdsourcing in software projects. We have compiled the results of our analysis on a management model that has been validated in a quasi-experiment crowdsourcing. The application of the model demonstrated capacity and efficiency to realize the management of dispersed participants and activities.*

***Resumo.*** *O crowdsourcing é um emergente modelo de negócios aplicado ao desenvolvimento de projetos de software em virtude da redução de custos e envolvimento de especialistas. Entretanto, existem desafios sobre como gerenciá-lo em tais projetos devido à dispersão de participantes e a paralelização de atividades. Neste estudo, nós realizamos uma análise da literatura e verificamos práticas aplicadas para o gerenciamento do crowdsourcing em projetos de software. Nós compilamos o resultado da nossa análise em um modelo de gerenciamento que foi validado em um quase-experimento crowdsourcing. A aplicação do modelo demonstrou capacidade e eficiência para realizar o gerenciamento entre os participantes dispersos e as atividades.*

## 1. Introdução

O termo *crowdsourcing* (CS) foi originalmente cunhado em 2006 por [Howe 2006] que definiu o conceito no "ato de desenvolver uma atividade por um agente designado ou terceirizar através de uma chamada para um grande grupo de pessoas". O CS tornou-se um modelo de negócios utilizando a inteligência coletiva para solucionar problemas ou completar atividades [Pan and Blevis 2011]. Segundo [Schwartz et al. 2015] a aplicação do CS providenciou de forma simples o acesso, o fornecimento e a geração de conhecimento, oferecendo benefícios para realização de serviços, produção de ideias ou conteúdo a partir da solicitação da contribuição de um grupo [Benedek et al. 2015].

Segundo [LaToza and van der Hoek 2016] fatores como redução de custo, paralelização de atividades e a colaboração de especialistas impulsionaram sua prática na Engenharia de Software, tornando-se um tópico popular e emergente no desenvolvimento de projetos de software [Dwarakanath et al. 2016, Tajedin and Nevo 2013].

Entretanto, os desafios causados pela dispersão geográfica, realização de atividades e descentralização de grupos de trabalho denotam a complexidade de gerenciar o CS [Satzger et al. 2014]. A necessidade de gerenciar o CS é paralelo a realidade das novas plataformas que focam sua gestão em fatores humanos (*feedbacks* e comunicação) e tecnológicos (*microtasks*) [Kucherbaev et al. 2016]. Porém, apesar do crescimento de utilização do CS em projetos de software, existe uma escassez de pesquisas concentradas no desenvolvimento e gerenciamento do CS nestes projetos [LaToza and van der Hoek 2016, Mattauch 2013].

Neste estudo, nós apresentamos um modelo de gerenciamento do CS em projetos de software a partir da coordenação dos participantes e atividades, contribuindo para os aspectos sociais e humanos do desenvolvimento de software em ambientes CS. Nós conduzimos uma análise da literatura e identificamos cinco práticas caracterizadas pela atomização de atividades, documentação do projeto, formação dos grupos, comunicação entre participantes e realização de *feedbacks*. Estas práticas foram compiladas em um modelo de gerenciamento CS validado em um quase-experimento. Os resultados demonstraram que o modelo auxiliou a execução do projeto, facilitando o gerenciamento dos participantes e das atividades desenvolvidas.

Este trabalho foi estruturado em 7 seções. Na primeira seção foi apresentada a contextualização, as lacunas da área e o objetivo deste estudo. Na segunda seção abordamos tópicos referentes à caracterização do CS em projetos de software. Um levantamento de publicações com respostas parciais ao deste estudo é encontrado na terceira seção. Na quarta seção apresentamos a construção do modelo de gerenciamento. Na quinta seção apresentamos a execução do quase-experimento e na sexta seção as análises e discussões dos resultados. Por fim são demonstrados os pontos alcançados com a condução deste estudo, seus fatores de limitação e os trabalhos para pesquisas futuras.

## 2. Crowdsourcing

O CS é formado pelos participantes do projeto, a organização responsável, a plataforma e as atividades que serão executadas [Hosseini et al. 2014]. Segundo [Zakariah et al. 2015] a utilização do CS como um modelo de negócios oferece benefícios para colaboradores e organizações como a flexibilidade de tempo e localização, aquisição de habilidades, diminuição de gastos, atração de talentos, aquisição de expertise externa, entre outros [Hossain 2012, Pedersen et al. 2013].

Apesar de similaridades com o *open source*, o CS geralmente agrega participantes motivados por fatores financeiros. Outra característica percebida no CS se refere ao desenvolvimento concorrente, diferente do desenvolvimento distribuído. Desta maneira, são amplificados os desafios de gerenciamento do CS devido ao nível de conhecimento entre os participantes, recrutamento de colaboradores, verificação do desenvolvimento de múltiplas atividades, coordenação das contribuições e compartilhamento de informações para membros do projeto [Hosseini et al. 2015, LaToza and van der Hoek 2016].

## 3. Trabalhos Relacionados

No estudo conduzido pelos autores [Kucherbaev et al. 2016] é apresentada uma abordagem para possibilitar o gerenciamento por meio de um processo CS. Sua análise produziu um conjunto de considerações com objetivos de direcionar novas pesquisas e entusiastas

da área. Entretanto, a pesquisa possui um caráter explanatório e suas conjecturas ainda não foram avaliadas a fim de comprovar, potencializar ou confrontar resultados.

Uma pesquisa conduzida pelos autores [Cullina et al. 2016] direcionou a necessidade de identificar, regular e refinar sub processos CS. Uma contribuição inicial refere-se a definição entre os conceitos chaves identificados na literatura e a geração de um modelo conceitual para facilitar sua aplicação. Porém, os autores declararam que o modelo está em um estágio de maturidade inicial e pode receber modificações para a sua implementação. A Tabela 1 apresenta a comparação entre os estudos similares e o modelo de gerenciamento CS desenvolvido neste artigo:

**Tabela 1. Comparação Entre os Estudos**

| Estudo | Métodos | Validação | Contribuição |
|---|---|---|---|
| [Kucherbaev et al. 2016] | Análise da Literatura | - | Teórica |
| [Cullina et al. 2016] | Análise da Literatura | - | Teórica |
| Modelo desenvolvido | Análise da Literatura | Quase-experimento | Prática |

Os dois estudos relacionados utilizaram métodos de pesquisa baseado na análise da literatura e não apresentaram validação em ambientes CS, assim a contribuição deu-se apenas de forma teórica. O modelo desenvolvido neste estudo também foi gerado com base na análise da literatura, porém a validação ocorreu através de um quase-experimento simulando a aplicação do CS e demonstrando os resultados de forma prática.

## 4. Modelo de Gerenciamento Crowdsourcing

Com o objetivo de compreender o estado da arte sobre o gerenciamento do CS nós conduzimos uma análise da literatura, selecionando estudos sob quatro critérios, apresentados a seguir:

- C1: O estudo corresponde ao âmbito do *crowdsourcing*;
- C2: O estudo corresponde ao âmbito de projeto de software;
- C3: O estudo apresenta práticas sobre como realizar o gerenciamento;
- C4: O estudo está completo e foi publicado em uma conferência, revista, jornal ou *workshop*.

Nós realizamos buscas combinando os termos *"management", "crowdsourcing", "crowd sourcing"e "software"* em bibliotecas digitais que possuem amparo de pesquisa com a Universidade Tecnologica Federal do Paraná - Campus Cornélio Procópio. As bibliotecas possuem um acervo de estudos na área computacional, apresentando publicações voltadas ao CS e projetos de software. As bibliotecas utilizadas foram: *IEEE Xplore (http://ieeexplore.ieee.org)*, *ACM Digital Library (http://dl.acm.org)* e *ScienceDirect (http://www.sciencedirect.com/)*.

Após a realização das buscas nas bases especificadas, nós selecionamos publicações que atendiam os quatro critérios (C1, C2, C3 e C4). Os estudos selecionados foram analisados para extração de informações sobre as fases do CS, definições sobre o gerenciamento do CS e se é uma característica referente ao fator humano (H) e/ou tecnológico (T), a relação de estudos é apresentado na Tabela 2.

A partir da análise dos estudos foram identificadas cinco práticas para a criação do modelo de gerenciamento do CS, a atomização das atividades, a documentação do projeto, a formação do grupo, a realização de comunicação e feedbacks.

**Tabela 2. Estudos Analisados**

| Referência | Fases do CS | Definição | Fator |
|---|---|---|---|
| [Dwarakanath et al. 2015] | Design | Elicitação e análise de um projeto | H |
| | Documentação | Descrever os componentes do projeto | H/T |
| | Implementação | Criação e postagem das tarefas em uma plataforma | H |
| | Verificação | Execução de testes unitários | H |
| [Tsai et al. 2014] | Apenas definições/ atividades | (a) requisitos; (b) design; (c) codificação (d) teste; (e) comunicação e (f) documentação. | H/T |
| [Saremi and Yang 2015] | Atomização | Distribuição com o máximo possível de tarefas | T |
| | Plataforma | Enviar para uma plataforma *crowdsourcing* as tarefas | T |
| | Registro | Mecanismo para o registro de tempo utilizado | T |
| | Comunicação | Comunicação e feedbacks entre colaboradores e companhia | H |
| [Murray-Rust et al. 2015] | Tarefas Independentes | Coordenação de colaboradores em ambiente de desenvolvimento usando tarefas atômicas e paralelas, canais de comunicação e feedbacks | H/T |
| [Stol and Fitzgerald 2014] | Apenas definições/ atividades | Utilizar a atomização de tarefas, coordenação, feedbacks, comunicação, planejamento e conhecimento. Assegurar a qualidade, motivação e remuneração. | H/T |

**Atomização:** o CS é executado a partir do desenvolvimento de atividades por um grupo de participantes. Desta forma, os autores [Saremi and Yang 2015, Murray-Rust et al. 2015, Stol and Fitzgerald 2014] sugerem que para ocorrer o gerenciamento do CS é necessário minimizar ou atomizar as atividades. O processo de atomização cria o conceito de *microtasks* definidas como tarefas que podem ser executadas de forma independente e em curto prazo de tempo. Porém, a atomização de atividades pode gerar um conjunto grande de *microtasks*, dificultando o gerenciamento. Com isto, cada projeto deve ser analisado para estabelecer um critério de atomização. O fator de atomização foi considerado tecnológico, pois impactará no tamanho do projeto.

**Documentação:** os estudos de [Dwarakanath et al. 2015, Tsai et al. 2014] apresentaram que conceitos de documentação são necessários para gerenciamento das atividades. O projeto deve possuir um conjunto de manuais, diagramas ou materiais de apoio para a realização das *microtasks*. A documentação foi considerada como fator tecnológico, pois deve estar presente nas atividades do projeto.

**Formação de grupos:** o CS é formado por um grupo de pessoas com diferentes níveis de conhecimentos e habilidades. Projetos com diversidade de participantes podem oferecer uma gama maior de soluções, porém também maximizam os desafios de gerenciamento devido aos fatores culturais, fronteiras geográficas e o tamanho do grupo. Com isto, os estudos de [Murray-Rust et al. 2015, Stol and Fitzgerald 2014] sugerem que a formação do grupo possua uma coordenação para selecionar participantes que possam auxiliar, estejam engajados e familiarizados com o projeto. A formação de grupos refere-se a um fator humano de gerenciamento.

**Comunicação:** A comunicação é um fator humano de gerenciamento citada nos estudos de [Tsai et al. 2014, Saremi and Yang 2015, Murray-Rust et al. 2015]. Todos os participantes devem possuir acesso a um canal de comunicação para realizar interação entre o grupo. A prática da comunicação durante o CS é um processo para auxiliar o encontro de informações, tomada de decisões e verificar dúvidas ou soluções encontradas.

**Feedbacks:** O *feedback* é um processo circular de interação entre participantes e o(s) responsável(is) pelo desenvolvimento do projeto e foi citado nos estudos de [Saremi and Yang 2015, Murray-Rust et al. 2015, Stol and Fitzgerald 2014]. O *feedback*

deve ser utilizado como uma prática para verificar o andamento das *microtasks* e para os participantes apresentarem eventuais dúvidas ou sugestões do projeto. Do outro lado, ao(s) responsável(is) do projeto, cabe salientar pontos como correção de atividades e a visão geral do projeto. O *feedback* é um fator de gerenciamento humano.

A partir da compilação deste conjunto de práticas formado por fatores humanos e tecnológicos, foi gerado um modelo de gerenciamento do CS para projetos de software, apresentado na Figura 1.

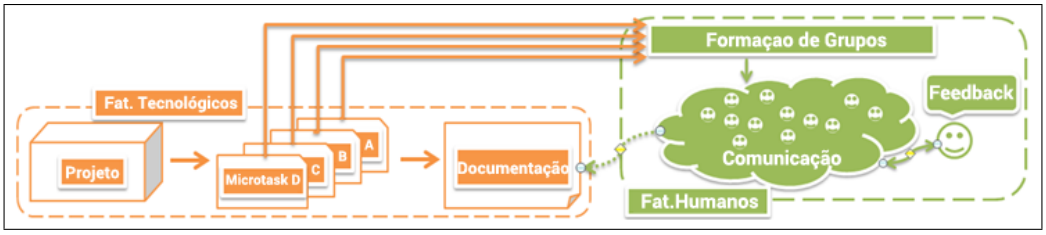


**Figura 1. Modelo de Gerenciamento CS**

O modelo foi organizado pela execução das cinco práticas anteriormente descritas. Após o projeto ser instanciado todas as atividades devem ser atomizadas. Para auxiliar o gerenciamento, as atividades atomizadas devem possuir documentação (diagramas, planilhas, casos de testes, etc...) que facilitem sua compreensão e execução. Após estas duas etapas é iniciado a formação de grupos, utilizando as *microtasks* e documentação como apoio para seleção de participantes. O projeto deve possuir um mecanismo que providencie a comunicação, como um chat ou wiki. Por fim, são performados *feedbacks* durante a execução das atividades entre o responsável do projeto e os participantes.

## 5. Validação do Modelo de Gerenciamento

Para efetuar o procedimento de validação e verificar o gerenciamento de CS foi utilizado a execução de um quase-experimento em uma iniciativa CS a fim de verificar o comportamento do modelo de gerenciamento previamente apresentado.

### 5.1. Iniciativa Crowdsourcing

A iniciativa pertence à utilização de robôs Legos Mindstorms para demonstração dos conceitos de gerenciamento de projetos e programação para os discentes dos cursos de graduação em Análise e Desenvolvimento de Sistemas e Engenharia da Computação da Universidade Tecnológica Federal do Paraná - Campus Cornélio Procópio.

Os robôs Lego Mindstorms possuem um conjunto de peças[1] que podem ser combinadas para a montagem de quase 25 modelos a partir da documentação oficial[2]. O processador permite o recurso de programação utilizando a linguagem Java e C por meio de um ambiente para aprendizado e ensino de programação. Os robôs Mindstorms apresentam possibilidades de apresentar noções de robótica, programação e trabalho em equipe.

Selecionamos uma turma de graduação que possuía alunos de Análise e Desenvolvimento de Sistemas e de Engenharia da Computação para verificar a execução do modelo fornecendo gerenciamento com a perspectiva dos fatores humanos e tecnológicos.

[1]Disponível em: http://arstechnica.com/gadgets/2013/08/review-lego-mindstorms-ev3-means-giant-robots-powerful-computers/

[2]Disponível em: http://www.lego.com/en-us/mindstorms/build-a-robot/

### 5.2. Execução do Quase-Experimento

A turma recebeu um treinamento de montagem e de programação do kit Lego Mindstorms. No início do semestre letivo foi concordado que uma das notas que integravam a média final dos alunos era composta pela participação em experimentos e quase-experimentos da disciplina, desta forma não foi necessário a assinatura de um termo de consentimento.

O quase-experimento foi executado em uma turma com os 27 discentes organizados em dois grupos. A composição dos grupos misturou os cursos, porém devido ao número ímpar uma equipe reteve mais participantes. Este desbalanceamento não foi percebido como um motivo de falha ou desafio na condução, visto que ambos os grupos concluíram o projeto de forma gerenciada e em tempos similares.

Para a configuração do ambiente CS estabeleceu-se as seguintes características: **Canal de comunicação:** Cada grupo deveria, obrigatoriamente, utilizar o chat eletrônico da plataforma acadêmica moodle[3]. **Dispersão:** Cada grupo era formado por dois papéis, estruturado como *Crowd* (equipe responsável por planejar estratégias) e *Crowdsourcer* (participante responsável por executar a estratégia). **Base de informação:** *Crowd* e *Crowdsourcer* possuíam acesso à documentação, porém apenas o *Crowdsourcer* possuía acesso às peças do robô. Com esta configuração enquanto o *Crowd* deveria adotar estratégias de montagem e enviá-las por chat ao *Crowdsourcer*, este deveria receber, executar e reportar o andamento do projeto para o *Crowd*. Com esta configuração foi verificado se o modelo poderia gerenciar o CS e realizar a construção de um robô, obedecendo as seguintes regras:

- O *Crowdsourcer* ficou afastado de seu respectivo *Crowd*.
- Todas as ordens deveriam acontecer pela troca de mensagens eletrônicas entre *Crowd* e *Crowdsourcer*.
- Todos os discentes tiveram acesso à documentação com as seguintes informações: código, imagem, quantidade, cor e categoria de todos os componentes do kit Lego.
- O robô deveria possuir o processador central e ao menos um motor.
- O *Crowd* deveria definir as estratégias de montagens e enviar os códigos de peças a serem encaixados para o *Crowdsourcer*.
- O *Crowdsourcer* deveria consultar o código na documentação, identificar a peça, realizar a montagem e enviar o *feedback* do encaixe e o andamento do projeto.

Não foi definido um limite mínimo de peças ou tempo para realizar a montagem do robô. A motivação foi trabalhada apenas na competição de qual grupo concluiria primeiro a montagem. O protocolo do quase-experimento é exposto na Tabela 3.

Para otimizar a aderência de *feedbacks*, a cada 30 minutos de duração do quase-experimento um membro do *Crowd* poderia deslocar-se até ao *Crowdsourcer* para verificar o andamento do projeto. Esta regra foi aplicada para evidenciar dois fatores: comunicação e *feedback*. O tempo máximo utilizado para a realização da montagem do robô foi de 44 minutos e 25 segundos. E somente um *feedback* deste estilo foi necessário para a conclusão do projeto, todos os outros *feedbacks* ocorreram via chat eletrônico.

[3]Disponível em: https://moodle.org

**Tabela 3. Modelo de gerenciamento aplicado ao quase-experimento**

| Prática | Aplicação |
|---|---|
| Atomização | Para garantir a atomicidade, selecionamos o seguinte conjunto de *microtasks*: 1) *Crowd* deve identificar o componente; 2) *Crowd* deve enviar o código para *Crowdsourcer*; 3) *Crowdsourcer* deve identificar o componente; 4) *Crowdsourcer* deve realizar a montagem e 5) *Crowdsourcer* deve enviar feedback. |
| Documentação | A documentação foi aplicada a partir de uma planilha descritiva dos componentes do kit Lego Mindstorms, descrevendo o código, imagem, quantia, cor, categoria, nome e encaixes possíveis de cada peça |
| Formação de Grupos | Para garantir que os grupos fossem formados com certo engajamento, foram realizadas apresentações sobre a iniciativa Lego Mindstorms aos participantes, também ocorreu a apresentação da documentação descrita anteriormente. A formação de dois grupos ocorreu aleatoriamente, apenas o processo de seleção dos papéis a serem executados foram discutidos entre cada grupo. |
| Comunicação | O *Crowd* e seu respectivo *Crowdsourcer* deveriam comunicar-se através do chat eletrônico. |
| Feedbacks | Os feedbacks foram reportados pelos *Crowdsourcer* a cada execução de uma ordem. |

## 6. Discussão e Análise

Os dois grupos apresentaram êxito para a conclusão do projeto, ambos concluíram o desafio utilizando a capacidade de inteligência coletiva do *Crowd* e do *Crowdsourcer*. Destacando ainda aderência ao conjunto de regras apresentadas.

Para a nomenclatura dos grupos optou-se por identificá-los como Grupo A e Grupo B. A primeira equipe que concluiu a montagem do robô foi o Grupo A utilizando um total de 39m e 24s. O Grupo A utilizou o total de 167 interações de mensagens eletrônicas durante todo o desenvolvimento do projeto. A segunda equipe, Grupo B, empregou 44m e 25s para o desenvolvimento do projeto, e utilizou 103 interações de mensagens eletrônicas. O produto final de ambas as equipes está apresentado na Figura 2.

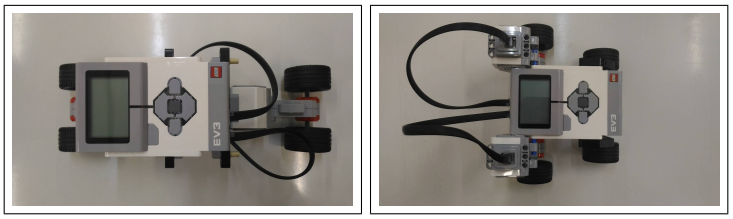

**Figura 2. Estrutura final do robô do Grupo A (esquerda) x Grupo B (direita)**

A análise individual dos log's do chat de cada grupo identificou que todas as ordens enviadas do *Crowd* para o *Crowdsourcer* foram executadas. Cada *Crowdsourcer* enviou o status do *feedback* sobre a montagem. Uma característica percebida se refere ao fato dos *Crowdsourcers* enviarem para o *Crowd* não somente o *feedback*, mas também sugestões sobre componentes para otimizar a montagem do robô. Fortalecendo e maximizando o ciclo presente nas práticas de Feedbacks e Comunicação apresentadas anteriormente.

Apesar de todas as ordens terem sido executadas, alguns componentes desenvolvidos não foram utilizados na montagem final do robô, devido ao fato da *Crowd* optar por outras táticas durante o desenvolvimento do projeto.

### 6.1. Atomização

A atomização foi executada a partir da definição de cinco *microtasks* (ver Atomização na Tabela 3) que deveriam ser executadas sequencialmente. Sua correta utilização forneceu mecanismos para o gerenciamento do CS, oferecendo balanceamento de *microtasks* entre o *crowd* e *crowdsourcer*.

### 6.2. Documentação

Ficou evidenciado que a documentação, neste caso a planilha descritiva dos componentes, foi uma prática importante para o gerenciamento do CS. Oferecendo mecanismos de autogestão para o *crowd* com a descrição de cada componente e facilitando a concepção de estratégias para o procedimento de montagem.

### 6.3. Formação de Grupos

A formação de grupos (ver Tabela 3) foi realizada para proporcionar aos participantes do quase-experimento um engajamento de conhecimento sobre os robôs Legos Mindstorms. A documentação contribuiu para a formação do grupo, visto que os participantes possuíam a mesma base conceitual para o desenvolvimento facilitando o gerenciamento.

### 6.4. Comunicação

Para garantir que houvesse constante comunicação durante o quase-experimento, os participantes utilizaram um chat eletrônico para a troca de mensagens no seu grupo. A comunicação foi explorada como forma de dispersar o andamento do projeto para todos os participantes e oferecer autogestão do *Crowd*. Oferecendo a possibilidade de visão geral do projeto entre todo o grupo, facilitando o gerenciamento do CS apesar da dispersão. Nós encontramos exemplos em que o *Crowd* perguntou se o *Crowdsourcer "compreendeu a estratégia de montagem"* e se poderia *"retornar o andamento da montagem"*.

### 6.5. Feedback

Apesar do quase-experimento possuir um mecanismo formal para a ocorrência de *feedbacks*, com o aumento da maturidade do projeto os *feedbacks* foram mais proveitosos. Cada *Crowdsourcer* percebeu que a sua função poderia auxiliar o desenvolvimento do projeto não somente executando ordens, mas apresentando soluções ou otimizações para as estratégias adotadas pelo *Crowd*. Um exemplo ocorreu quando o *Crowdsourcer* enviou ao *Crowd* se *"apenas faltava a base do robô"* e verificou *"se poderia utilizar uma determinada peça para concluir a montagem"*.

## 7. Conclusão

O CS é um emergente modelo de negócios para o desenvolvimento de projetos de software. Analisando a literatura sobre o gerenciamento do CS foi identificado um conjunto de práticas que foram compiladas em um modelo de gerenciamento do CS. O modelo pode ser aplicado como um guia prático para maximizar as chances de êxito em projetos de software.

Conduzimos um quase-experimento verificando o gerenciamento a partir dos aspectos humanos e tecnológicos em uma iniciativa CS que evidenciou o modelo do CS de competição e recrutamento. O projeto final apresentou resultados satisfatórios, gerenciando as atividades desenvolvidas, a colaboração dos participantes e a compreensão do projeto no ambiente disperso. Todos os grupos alcançaram o objetivo final dentro do projeto e apresentaram aderência as ordens preestabelecidas.

Nós acreditamos que este estudo contribuiu para a compreensão e tratamento de aspectos sociais e tecnológicos de gerenciamento em projetos de software aplicados em

ao novo modelo de desenvolvimento CS. O objetivo deste estudo em analisar o gerenciamento do CS verificou que a partir da utilização de um conjunto bem especificado de etapas e atividades, os fatores humanos e tecnológicos do CS podem ser executados com a atomização de atividades, documentação, formação de grupos, realização de *feedbacks* e comunicação entre o grupo.

### 7.1. Limitações e Trabalhos Futuros

Os projetos de software podem oferecer desafios maiores aos simulados no quase-experimento, com isto os fatores de atomização, documentação, formação de grupos, feedbacks e comunicação podem ter sido limitadas. Com isto, o modelo pode oferecer insuficiências quando aplicado em determinados cenários. Um risco percebido refere-se ao *Crowdsourcer* executar ordens sem avisar o *Crowd*, constatado somente com a análise dos log's do chat. Outro adendo é que não foi aplicado conceitos de programação.

Uma contribuição futura refere-se ao amadurecimento do conceito de gerenciamento de CS em projetos de software, através de experimentações com o modelo apresentado neste estudo.

## Referências

Benedek, A., Molnar, G., and Szuts, Z. (2015). Practices of crowdsourcing in relation to big data analysis and education methods. In *Intelligent Systems and Informatics (SISY), 2015 IEEE 13th International Symposium on*, pages 167–172.

Cullina, E., Conboy, K., and Morgan, L. (2016). Choosing the right crowd: An iterative process for crowd specification in crowdsourcing initiatives. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pages 4355–4364.

Dwarakanath, A., Chintala, U., Shrikanth, N. C., Virdi, G., Kass, A., Chandran, A., Sengupta, S., and Paul, S. (2015). Crowd build: A methodology for enterprise software development using crowdsourcing. In *CrowdSourcing in Software Engineering (CSI-SE), 2015 IEEE/ACM 2nd International Workshop on*, pages 8–14.

Dwarakanath, A., Shrikanth, N. C., Abhinav, K., and Kass, A. (2016). Trustworthiness in enterprise crowdsourcing: A taxonomy & evidence from data. In *Proceedings of the 38th International Conference on Software Engineering Companion*, ICSE '16, pages 41–50, New York, NY, USA. ACM.

Hossain, M. (2012). Crowdsourcing: Activities, incentives and users' motivations to participate. In *Innovation Management and Technology Research (ICIMTR), 2012 International Conference on*, pages 501–506.

Hosseini, M., Phalp, K., Taylor, J., and Ali, R. (2014). The four pillars of crowdsourcing: A reference model. In *2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS)*, pages 1–12.

Hosseini, M., Shahri, A., Phalp, K., and Ali, R. (2015). Recommendations on adapting crowdsourcing to problem types. In *2015 IEEE 9th International Conference on Research Challenges in Information Science (RCIS)*, pages 423–433.

Howe, J. (2006). The rise of crowdsourcing. *Wired magazine*, 14(6):1–4.

Kucherbaev, P., Daniel, F., Tranquillini, S., and Marchese, M. (2016). Crowdsourcing processes: A survey of approaches and opportunities. *IEEE Internet Computing*, 20(2):50–56.

LaToza, T. D. and van der Hoek, A. (2016). Crowdsourcing in software engineering: Models, motivations, and challenges. *IEEE Software*, 33(1):74–80.

Mattauch, T. (2013). Innovate through crowd sourcing. In *Proceedings of the 41st Annual ACM SIGUCCS Conference on User Services*, SIGUCCS '13, pages 39–42, New York, NY, USA. ACM.

Murray-Rust, D., Scekic, O., Papapanagiotou, P., linh Truong, H., Robertson, D., and Dustdar, S. (2015). A collaboration model for community-based software development with social machines. *EAI Endorsed Transactions on Collaborative Computing*, 1(5).

Pan, Y. and Blevis, E. (2011). A survey of crowdsourcing as a means of collaboration and the implications of crowdsourcing for interaction design. In *Collaboration Technologies and Systems (CTS), 2011 International Conference on*, pages 397–403.

Pedersen, J., Kocsis, D., Tripathi, A., Tarrell, A., Weerakoon, A., Tahmasbi, N., Xiong, J., Deng, W., Oh, O., and de Vreede, G. J. (2013). Conceptual foundations of crowdsourcing: A review of is research. In *System Sciences (HICSS), 2013 46th Hawaii International Conference on*, pages 579–588.

Saremi, R. L. and Yang, Y. (2015). Dynamic simulation of software workers and task completion. In *CrowdSourcing in Software Engineering (CSI-SE), 2015 IEEE/ACM 2nd International Workshop on*, pages 17–23.

Satzger, B., Zabolotnyi, R., Dustdar, S., Wild, S., Gaedke, M., Gobel, S., and Nestler, T. (2014). Chapter 8 - toward collaborative software engineering leveraging the crowd. In Mistrik, I., , Bahsoon, R., , Kazman, R., , and Zhang, Y., editors, *Economics-Driven Software Architecture*, pages 159 – 182. Morgan Kaufmann, Boston.

Schwartz, C., Borchert, K., Hirth, M., and Tran-Gia, P. (2015). Modeling crowdsourcing platforms to enable workforce dimensioning. In *Telecommunication Networks and Applications Conference (ITNAC), 2015 International*, pages 30–37.

Stol, K.-J. and Fitzgerald, B. (2014). Researching crowdsourcing software development: Perspectives and concerns. In *Proceedings of the 1st International Workshop on CrowdSourcing in Software Engineering*, CSI-SE 2014, pages 7–10, New York, NY, USA. ACM.

Tajedin, H. and Nevo, D. (2013). Determinants of success in crowdsourcing software development. In *Proceedings of the 2013 Annual Conference on Computers and People Research*, SIGMIS-CPR '13, pages 173–178, New York, NY, USA. ACM.

Tsai, W. T., Wu, W., and Huhns, M. N. (2014). Cloud-based software crowdsourcing. *IEEE Internet Computing*, 18(3):78–83.

Zakariah, Z., Janom, N., and Arshad, N. H. (2015). Business model of crowdsourcing: Review paper. In *2015 IEEE 6th Control and System Graduate Research Colloquium (ICSGRC)*, pages 66–69.