A Goal Model for Crowdsourced Software Engineering

Fiza Siyal[1] and Xavier Franch[2]

[1]Universitat Politècnica de Catalunya - BarcelonTech, Barcelona, Spain
[2]Universitat Politècnica de Catalunya - ESSI, Campus Nord, Barcelona, Spain

engr_fizza2001@quest.edu.pk
franch@essi.upc.edu

Abstract. Crowdsourced Software Engineering (CSE) is the act of undertaking any external software engineering tasks by an undefined, potentially large group of online workers in an open call format. Using an open call, CSE recruits global online labor to work on various types of software engineering tasks, such as requirements extraction, design, coding and testing. The field is rising rapidly and touches various aspects of software engineering. CSE has grown significance in both academy and industry. Despite of the enormous usage and significance of CSE, there are many open challenges reported by various researchers. In order to overcome the challenges and realizing the full potential of CSE, it is highly important to understand the concrete advantages and goals of CSE. In this paper, we present a goal model for CSE, to understand the real environment of CSE, and to explore the aspects that can somehow overcome the aforementioned challenges. The model is designed using RiSD, a method for building Strategic Dependency (SD) models in the *i** notation, applied in this work using iStar2.0. This work can be considered useful for CSE stakeholders (Requesters, Workers, Platform owners and CSE organizations).

Keywords: Crowdsourced software engineering, Goal modeling, i*, iStar, SD model, Crowdsourced software development.

1 Introduction

The term 'Crowdsourcing' was first widely accepted in 2006 [1]. Howe defined as the act of a company or institution taking a function once performed by employees and outsourcing it to an undefined (and generally large) network of people in the form of an open call [2]. In 2008, Brabham [3] discussed that crowdsourcing has become popular in all type of companies, e.g. crowdsourced t-shirt design, photography and other media design, etc., and remarkably, to solve scientific problems.

More than ten years after, Mao et al. published a comprehensive literature review on crowdsourcing [4]. Since Howe's definition was the most widely accepted crowdsourcing definition in the papers surveyed in this review, Mao et al. chose to define

Crowdsourced Software Engineering (CSE) simply as an instantiation of Howe's definition: "CSE is the act of undertaking any external software engineering tasks by an undefined, potentially large group of online workers in an open call format" [4]. Using an open call, CSE recruits global online labor to work on various types of software engineering tasks, such as requirements extraction, design, coding and testing [4]. CSE tasks demand reliable developers and enough participation to guarantee a qualified asset that can be delivered to the client.

CSE is implemented by many successful crowdsourcing platforms, such as TopCoder (www.topcoder.com), uTest (www.utest.com), Mob4Hire (www.mob4hire.com) and Bugcrowd (www.bugcrowd.com). Therefore, developers may have difficulties in selecting which one is the most suitable for their needs. We suggest in this paper that CSE stakeholders could use a model-based approach to select the platform with all mentioned features that best fits to their needs. We propose the use of goal models expressed in iStar2.0 in order to understand the most important actors, goals, dependencies and resources, among others, involved in CSE. The purposes of goal models in this context are: 1) to visualize and understand the real environment of CSE using iStar2.0 constructs (mainly actors and dependencies); 2) to explore (based on challenges and proposed solutions from literature) how to realize the full potential of CSE in order to make it an ideal paradigm for software engineering projects: 3) knowing the requirements of the best CSE platforms; 4) improving the work processes of CSE environment.

The rest of the paper is organized as follows: Section 2 describes the background of CSE, Section 3 presents the proposed goal model of CSE, and finally Section 4 concludes the work.

2 Background

In 2017, Mao et al. [4] presented a comprehensive survey of the use of CSE. They reviewed the definitions of crowdsourcing and derived their own definition (already quoted in the previous section) together with a taxonomy. Mao et al. further analyzed the software engineering domains, tasks and applications for crowdsourcing and the platforms and stakeholders involved in realizing CSE solutions. As they report, CSE generally involves three types of actors: Employers (aka Requesters), who have software development work that needs to be done; Workers, who participate in developing software; Platforms, which provide an online marketplace within which requesters and workers can meet. Fig. 1 shows these actors and their relationships as depicted in [4].

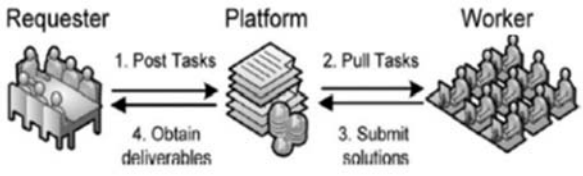

Fig. 1. General process & actors in CSE [4]

Mao et al. [5] stated that crowdsourcing utilizes an open call format to attract online developers to accomplish various types of software development tasks such as architecture, component design, component development, testing and bug fixing. They stated the problems from two different perspectives: developer's perspective, and platform's perspective. To tackle the reported problems, the authors proposed a developer recommendation framework. They performed an empirical study for the evaluation of the proposed framework. The experimental results show that their recommender system outperforms the baseline method and can achieve promising accuracy and diversity.

Other papers, e.g. Latoza and van der Hoek [6], Peng et al. [7] and Stol et al. [8] among others, discussed on challenges, motivations, forces, limitations and current practices. Due to space reasons we cannot survey them here, but they have influenced our proposal.

3 Goal Model for CSE

This section presents a goal model for CSE. The model is designed by following the R*i*SD methodology proposed by Franch et al. [9]. We use the iStar 2.0 notation [10] and for modeling, the pIstar tool [11]. We will transform the goals into tasks, resources or quality at different steps during the phase, which is actually what the R*i*SD methodology is.

The methodology consists of three phases: domain analysis, social system construction, and socio-technical system construction. Due to space reasons, we focus on the second one (considering that the background section above is a kind of substitute of the domain analysis).

This second phase builds the social system model in the form (SD) model. The model is constituted iteratively. This model does not include the software system and therefore it focuses on the stakeholder needs [9]. For this reason, and taking the statement from [7], we identify a platform vendor as a CSE organization, an actor who owns a platform, and is a manager/admin/vendor of the platform. We apply next the activities defined in [9] for Phase II.

Activity II.1: Identify departing actors.

Fig. 2 shows the actors involved in CSE are: Requester, Worker, and CSE organization. Requester and Worker are the entities who achieve their goals with the contribution of the CSE organization, whereas the CSE organization is responsible for providing online marketplace to Requester for receiving the outsourced requests of software engineering (SE) tasks and Workers to develop the solutions of outsourced tasks on payment.

Stol and Fitzgerald [12] described that an important consideration for a crowdsourcing customer (Requester) is to decide on an appropriate remuneration that will attract sufficient participants (Worker) to a crowdsourcing contest. Based on this we come to know that appropriate remuneration is the major concern of a worker, so we identified the main goal of worker as "Money earned".

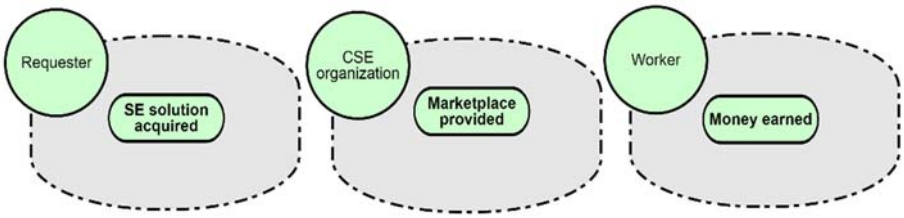

Fig. 2. Departing actors in CSE

Activity II.2: Establish goal dependencies among actors

Fig. 3 illustrates the goal dependencies among the actors. Goals of the actors depend on other actors. Latoza and van der Hoek [6] mentioned that the competition model used by the topcoder crowdsourcing platform is similar to traditional outsourcing, in which a client requests work and pays for its completion, so one more dependency added "Payment received" from CSE organization to Requester.

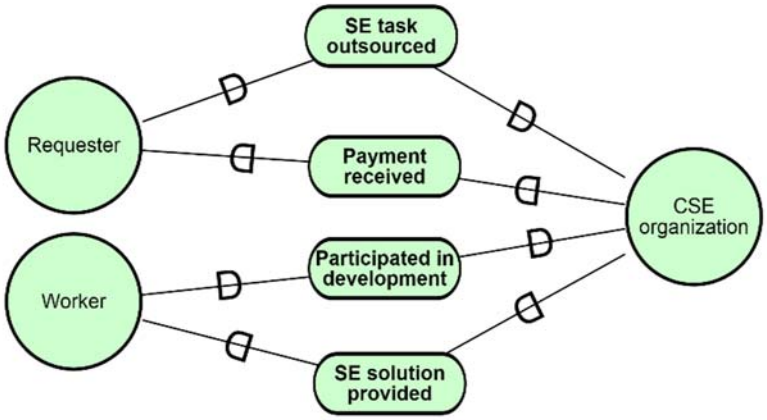

Fig. 3. Goal dependencies among actors

Activity II.3: Classify and rename the dependum of the added dependencies

Fig. 4 shows the transformations of dependencies. Transformations are made as: the goal "SE task outsourced" can be achieved by executing a task "Outsource SE task"; the goal "payment received" transformed to a task "Receive payment"; the goal "Participated in development" classified and renamed to a task " Develop SE task"; and the goal "SE solution provided" changed to a resource "SE task solution".

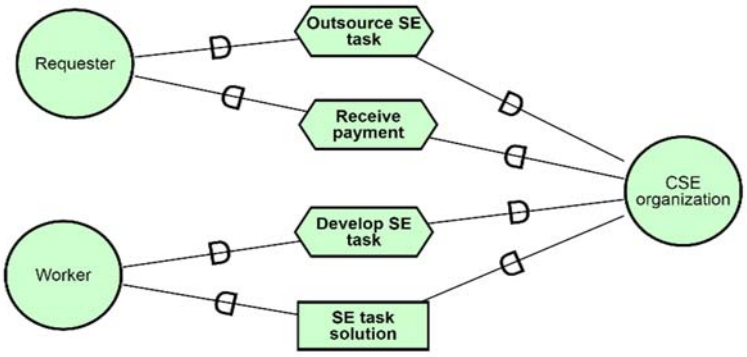

Fig. 4. Classified and renamed added dependencies

Activity II.4: Check for new actors and/or dependencies

To elaborate more the CSE social scenario, new dependencies have been added. Fig. 5 shows the new dependencies named: “Task reqs. Identified”, “Solution accepted”, and “Appreciation received”. These dependencies are identified in connection with previous dependencies. Starting from Requester, after submitting the outsource request to CSE organization, the next step would be to identify the task requirements by CSE organization. Similarly CSE organization may accept a solution provided by a Worker as a result of developing a task. Further a Worker depends on a Requester for acceptance of his provided solution, and it is shown by introducing dependency "Appreciation received". Also a task "Receive payment", identified in Fig. 4 has been transformed into a resource "Payment of solution".

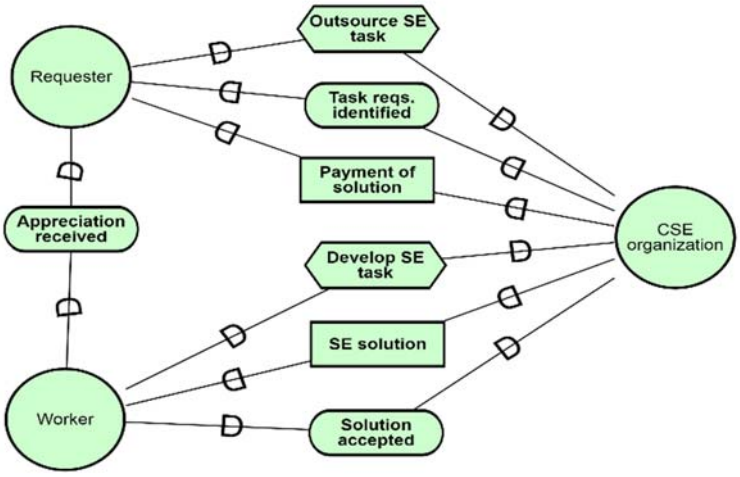

Fig. 5. Strategic Dependency Model for CSE (First refinement)

Fig. 5 has been further refined into Fig. 6 as: the goal "Task reqs. identified" is refined into a source "Task requirements"; the goal "Solution accepted" refined into a quality "High quality solution", and the goal "Appreciation received" transformed into a quality "Requester's satisfaction". Finally, a qualification link is established from quality “high quality solution” to resource “SE solution” to show that high quality refers to a solution provided by a Worker.

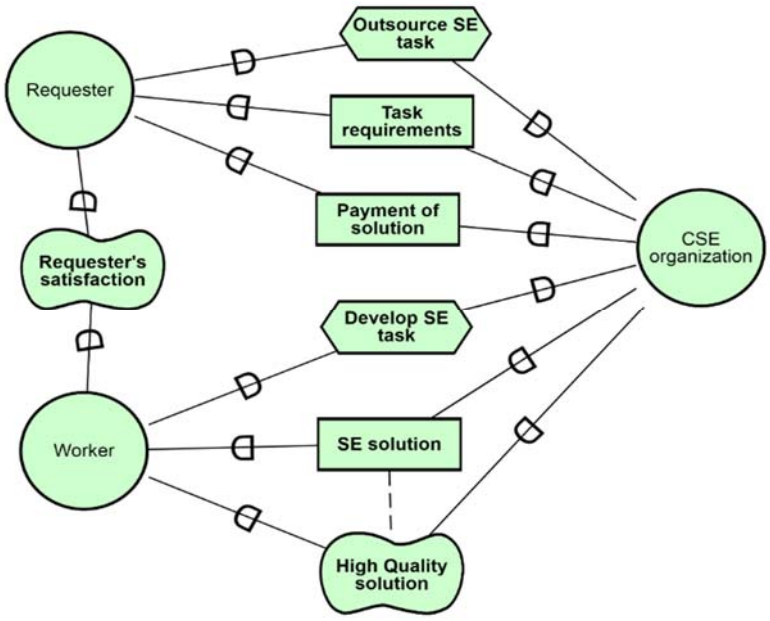

Fig. 6. Final Strategic Dependency Model for CSE (second refinement)

4 Conclusion

We have proposed a goal model for CSE in order to explore its real environment. Since past few years CSE has been the point of attraction for the researchers in the field of software engineering. From literature it has been noted that crowdsourcing is a concept been used as a problem solving approach in variety of domains. We selected the CSE domain to explore its environment by considering CSE process, actors, goals, and dependencies among actors involved. The model has been designed by following the R*i*SD methodology. We used the iStar 2.0 notations and for modeling, the pIstar tool. In this work, we focused on constructing the social system (known as strategic dependency model) for CSE. At present, the models show the work processes and associations among stakeholders of CSE based on goal dependencies. The final model resulting from the full application of the R*i*SD methodology will be based on proposed solutions of challenges reported in literature. A simple questionnaire-based survey could be used to evaluate all the models that part of this paper and also built as future work. Target participants would be all CSE stakeholders, including CSE organizations, platform

vendors, and workers/developers. Once the final models are available, usual techniques can be applied to analyse the results, for instance definition and evaluation of metrics [13], satisfaction analysis [14] and risk analysis [15]. Also, using them for reengineering current software development processes [16] would be a possible line of action.

References

1. The Rise of Crowdsourcing, Wired, https://www.wired.com/2006/06/crowds/, 1 June 2006.
2. Crowdsourcing: A Definition, http://crowdsourcing.typepad.com/cs/2006/06/crowdsourcing_a.html, 2 June 2006.
3. Brabham, D. C.: Crowdsourcing as a Model for Problem Solving. Convergence: The International Journal of Research into New Media Technologies. Sage Publications London, Los Angeles, New Delhi and Singapore. 14, 1, 75–90 (2008)
4. Mao, K., Capra, L., Harman, M., Jia, Y.: A survey of the use of crowdsourcing in software engineering. The Journal of Systems and Software. 126, 57–84, (2017)
5. Mao, K., Yang, Y., Wang Q., Jia, Y., Harman, M.: Developer Recommendation for Crowdsourced Software Development Tasks. In: IEEE Symposium on Service-Oriented System Engineering, (2015)
6. Latoza, T. D., Hoek, A.: Crowdsourcing in Software Engineering: Models, Motivations, and Challenges. In: IEEE Software , vol. 33, no. 1, pp. 74-80, (2016)
7. Peng, X., Babar, M. A., Ebert, C.: Collaborative Software Development Platforms for Crowdsourcing. In: IEEE Software , vol. 31 , no. 2, pp. 30-36, (2014)
8. Stol, K.-J., LaToza, T. D., Bird, C.: Crowdsourcing for Software Engineering. In: IEEE software, vol. 34, no. 2, pp. 30-36. (2017)
9. Franch, X., Grau, G., Mayol, E., Quer, C., Ayala, C., Cares, C., Navarrete, F., Haya, M., Botella, P.: Systematic Construction of i* Strategic Dependency Models for Sociotechnical Systems. International Journal of Software Engineering and Knowledge Engineering. 17, 1, 79-106, (2007)
10. Fabiano Dalpiaz, Xavier Franch, Jennifer Horkoff. iStar 2.0 Language Guide. arXiv:1605.07767, (2016)
11. piStar tool, http://www.cin.ufpe.br/~jhcp/pistar/
12. Stol, K.-J., Fitzgerald, B.: Researching Crowdsourcing Software Development: Perspectives and Concerns. In: Proceedings of the 1st International Workshop on CrowdSourcing in Software Engineering CSI-SE. 7-10. ACM, New York, (2014)
13. Franch, X.: A Method for the Definition of Metrics over i* Models. In: 21st International Conference Advanced Information Systems Engineering (CAiSE), pp. 201-215 (2009)
14. Horkoff, J., Yu, E.: Comparison and Evaluation of Goal-Oriented Satisfaction Analysis Techniques. Requirements Engineering Journal (REJ), Springer, pp. 1-24, (2012).
15. Franch, X. et al.: Managing Risk in Open Source Software Adoption. In: Proceedings of the 8th International Joint Conference on Software Technologies (ICSOFT), pp. 258-264 (2013).
16. Grau, G., Franch, X., Maiden, N.: A goal based round-trip method for system development. In: Proceedings of 11th Int. Working Conference on Requirements Engineering: Foundations for Software Quality (REFSQ), pp. 71-86 (2005).