Crowd-Centric Requirements Engineering

Remco Snijders, Fabiano Dalpiaz
Department of Information and Computing Sciences
Utrecht University, the Netherlands

Mahmood Hosseini, Alimohammad Shahri, Raian Ali
Faculty of Science and Technology
Bournemouth University, United Kingdom

***Abstract*—Requirements engineering is a preliminary and crucial phase for the correctness and quality of software systems. Despite the agreement on the positive correlation between user involvement in requirements engineering and software success, current development methods employ a too narrow concept of that "user" and rely on a recruited set of users considered to be representative. Such approaches might not cater for the diversity and dynamism of the actual users and the context of software usage. This is especially true in new paradigms such as cloud and mobile computing. To overcome these limitations, we propose crowd-centric requirements engineering (CCRE) as a revised method for requirements engineering where users become primary contributors, resulting in higher-quality requirements and increased user satisfaction. CCRE relies on *crowdsourcing* to support a broader user involvement, and on *gamification* to motivate that voluntary involvement.**

I. Introduction

The primary measure for the success of a software system is the degree to which it meets the purpose for which it was intended [1]. Requirements Engineering (RE) is the branch of software engineering that treats the early and preliminary stages for achieving this objective. RE is concerned with identifying the real-world goals that justify a software system, the functions that such a system should deliver, and the existing constraints [2, p. 315]. Poor RE is widely acknowledged as one of the main factors of software project failure [1], [3].

User involvement has been recently studied in RE. It has been shown that users are seldom involved, despite the common agreement that doing it would result in better RE and higher chances for project success [4]. Generally, according to the Standish CHAOS Report, user involvement seems the most important success factor for IT projects [5].

Existing methods to enhance user involvement exist [6], [7], but they represent only partial solutions:

- The concept of user is narrow; priority is given to customers (as they request and pay for the system), while end users have a marginal role, despite the fact that they will ultimately experience and benefit from the system.
- User involvement has been explored in the early phases of RE, where selected samples of users have a say in the requirements for the system (e.g., via focus groups). However, such early input can later be discarded by developers and customers, which conduct requirements negotiation, prioritisation, and specification in private sessions. Thus, user needs may be finally missed.
- No incentives are provided to foster user involvement which is often voluntary.

II. Harnessing the Power of Crowdsourcing and Gamification

In order to address the limitations that we described in Section I, we suggest to harness the power of the two recent mechansims of crowdsourcing and gamification.

Crowdsourcing has been studied as a scalable and inexpensive way to involve users in RE. Lim and Finkelstein [6] have described the StakeRare method that supports requirements elicitation via "a crowdsourcing approach to identify and prioritise stakeholders and their requirements". The CrowdREquire [7] platform supports requirements specification, which comes along with a use case model, business model and market strategy. Hosseini et al. [8] also make the case for exploiting crowdsourcing in requirements elicitation.

We argue that crowdsourcing has a high potential in most of the activities that constitute RE. A large network of both internal and external stakeholders (e.g., users, customer representatives, developers) can perform requirements elicitation, negotiation and prioritisation, potentially leading to higher user satisfaction and innovation in requirements.

In order to increase the motivation for participation, we see gamification as a natural complementary mechanism to crowdsourcing. We aim to turn the RE process into a participatory and enjoyable design activity where users are rewarded by (i) *social recognition*, as user participation makes them more visible to the community through game elements such as leaderboards and badges; and (ii) *satisfaction*, as the actual decision making is driven by the crowd, rather than developers.

III. The Crowd-Centric RE (CCRE) method

We advocate Crowd-Centric Requirements Engineering (CCRE) as a method that guides software product companies in effectively applying crowdsourcing throughout RE. This is a challenging task and we expect that each activity in RE requires different approaches on how to set up the crowdsourcing, e.g., recruiting the crowd and interaction platforms.

Figure 1 compares traditional RE methods (left), RE with crowdsourced requirements elicitation including stakeholders identification [6], [7], [8] (middle), and CCRE (right). In traditional RE methods, customers are asked to propose their requirements to the developer company. Such company analyses these requirements, makes a prioritisation based on their assumptions and preferences, and ends with the specification activity. In RE with crowdsourced elicitation, crowdsourcing is proposed within the scope of the initial activity of RE.

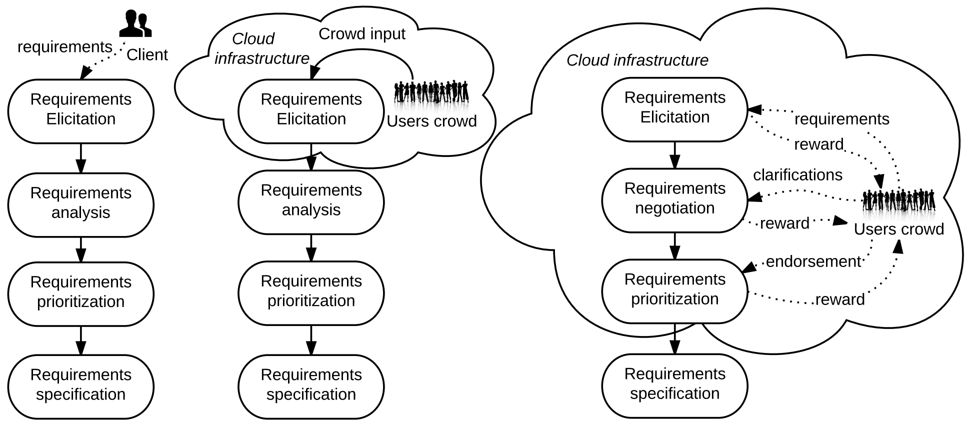

Fig. 1. Comparing traditional RE, RE with crowdsourced elicitation, and CCRE

The software company will take on from the crowd input and specify requirements in a relatively independent manner.

In the CCRE method, we propose to analyse each of the RE activities to determine where crowdsourcing and gamification could add value. The crowd is not only involved in the initial elicitation activity, but can also be in other activities such requirements negotiation and prioritisation. Since requirements specification needs a certain expertise, we propose that this should still be done by the software company's requirements analysts. Specification is done for requirements prioritised by the crowd, as suggested by the agile requirements refinery [9].

In CCRE, the users crowd (and not their elite representatives) is continuously involved in developing requirements and is, thereby, enabled to verify and evolve these requirements. The relative priorities of users are clear, as the crowd reviews the whole set of requirements iteratively. CCRE goes beyond elicitation and enables co-creation and co-design. However, crowd involvement is not guaranteed and requires motivation.

Gamification is a key component of the CCRE method, as demonstrated by the "reward" arrows in the figure. Members of the users crowd receive points for proposing, commenting on and voting for requirements. These points can be used by a member to promote their own proposed requirements or to vote on requirements that were proposed by other members. Badges can be earned by providing constructive feedback and thereby helping in the process of requirements analysis.

The **Cloud** is the infrastructure that supports CCRE, enabling the interaction between the users crowd and the software company throughout the RE activities (proposing requirements, clarifying them, etc.). CCRE is meant for software with users using the software in a dynamic context (e.g., cloud services), hence the value of our participatory approach.

IV. Challenges and Directions

Software companies need a large degree of openness to employ crowdsourcing massively. Indeed, users can see the ideas of the company and other users. User-defined priorities or feature discussions could raise expectations that the company might not want, or is unable, to meet. This may create complex politics and give the company a feeling of lost control.

Assuring the quality of the requirements obtained from the crowd is challenging. Users might not know the standards and guidelines of requirements documents of a company and, thus, provide ambiguous requirements. In such case, The moderation of requirements analysts would still be necessary.

Generic challenges of crowdsourcing and gamification continue in CCRE. Overlooking minorities, malicious participants, and the emergence of dominance and unhealthy dependencies in the crowd are examples of the CCRE challenges from the crowdsourcing angle. The impression of trivialising the task, clustering, opting-out, and introducing dishonesty to win the reward are typical challenges when applying gamification.

V. Acknowledgements

The research was partially supported by the Fusion Investment Fund of Bournemouth University—BUUU project.

References

[1] B. Nuseibeh and S. Easterbrook, "Requirements Engineering: a Roadmap," in *Proc. of FOSE'00*, 2000, pp. 35–46.

[2] P. Zave, "Classification of Research Efforts in Requirements Engineering," *ACM Computing Surveys*, vol. 29, no. 4, pp. 315–321, Dec. 1997.

[3] M. Chemuturi, *Requirements Engineering and Management for Software Development Projects*. Springer, 2013.

[4] S. Kujala, M. Kauppinen, L. Lehtola, and T. Kojo, "The Role of User Involvement in Requirements Quality and Project Success," in *Proc. of RE'05*. IEEE, 2005, pp. 75–84.

[5] The Standish Group, "CHAOS Summary 2009," Tech. Rep., 2009. [Online]. Available: http://emphasysbrokeroffice.com/files/2013/04/Standish-Group-CHAOS-Summary-2009.pdf

[6] S. Lim and A. Finkelstein, "StakeRare: Using Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation," *IEEE Transactions on Software Engineering*, vol. 38, no. 3, pp. 707–735, 2012.

[7] A. Adepetu, K. A. Ahmed, Y. A. Abd, A. A. Zaabi, and D. Svetinovic, "CrowdREquire: A Requirements Engineering Crowdsourcing Platform," in *Proc. of the AAAI Spring Symposium: Wisdom of the Crowd*, 2012.

[8] M. Hosseini, K. Phalp, J. Taylor, and R. Ali, "Towards Crowdsourcing for Requirements Engineering," in *Proc. of the Empirical Track of REFSQ 2014*, 2014, pp. 82–87.

[9] K. Vlaanderen, S. Jansen, S. Brinkkemper, and E. Jaspers, "The Agile Requirements Refinery: Applying SCRUM Principles to Software Product Management," *Information & Software Technology*, vol. 53, no. 1, pp. 58–70, 2011.