Journal of Computer Science

Original Research Paper

Crowdsourced Software Design Platforms: Critical Assessment

Reem Aliady and Sultan Alyahya

Department of Information Systems, King Saud University, Riyadh, Saudi Arabia

Article history
Received: 03-02-2018
Revised: 05-03-2018
Accepted: 17-04-2018

Corresponding Auhor:
Sultan Alyahya
Department of Information Systems, King Saud University, Riyadh, Saudi Arabia
Email: sualyahya@ksu.edu.sa

Abstract: There is a great interest towards the use of crowdsourcing into software development activities including: Requirements engineering, implementation, testing, evaluation and the focus of this research, design. Software design is one of the least explored activities within the concept of crowdsourcing. Therefore, this research provides a comprehensive coverage on the current state-of-the-art in the use of crowdsourcing in software design. It analyzes the current existing major crowdsourced software design platforms and discusses their workflows. The analysis results in identifying a set of limitations in the current platforms and improvements are proposed to overcome those limitations. Both findings of this research including (limitations and improvements) are validated by creating and distributing a questionnaire to software practitioners to justify the need to overcome those limitations through the proposed improvements.

Keywords: Crowdsourcing, Software Design, Crowdsourced Software Design, Software Development

Introduction

Software development using crowdsourcing has been gaining an increasing attention in the recent years. Crowdsourced software development has different characteristics than the traditional software development done through in-house sourcing. It is performed through an open call where anyone can participate rather than a job assigned to a known team member.

Moreover, software development using crowdsourcing has been used in many software development activities such as requirements engineering, implementation, testing, verification and design.

In this research, crowdsourced software design is the main focus. A comprehensive investigation has been made on its major activities which include architecture design, user interface design and design revision. The majority of crowdsourced software design platforms are presented with an elaboration on their workflows.

The motivation behind this research is the fact that there have not been many studies on crowdsourced software design. In the course of this research, it has been found that there are few papers discussing crowdsourced software design individually (Bernstein, 2010; Latoza *et al*., 2015; Nebeling *et al*., 2012; Bao *et al*., 2011; Nebeling and Norrie, 2011a; Akiki *et al*., 2013; Lasecki *et al*., 2015). To the researchers' knowledge, there has not been any research that provided a complete comparison and workflow evaluation on crowdsourced design platforms as it will be discussed in this study. Thus, this work aims to fill the need for such research and contribute with a comprehensive analysis and assessment of the current major platforms along with proposing improvements in the crowdsourced software design domain. This shall contribute to improving the currently used practices in the domain of crowdsourcing for software design.

This paper is divided as follows: The next section provides some background information about the main concepts in this research. Then, the literature review is provided. The activities offered in the current crowdsourced software design platforms are then described. Finally, the limitations and potential improvements of the platforms are discussed and evaluated before concluding the paper.

Background

This section will provide a background on software development, software design, crowdsourcing crowdsourced software development and crowdsourced software design.

Software Design

The major phases in software development include requirements analysis, design, implementation, testing and evaluation (Bassil, 2012).

Software design is the phase after requirements analysis and before the implementation. It is the process of creating specifications of the software product based on given constraints and goals (Sommerville, 2011).

Based on Ralph (2010), there are two views on how design process should be conducted. The first approach is called reason-centric. It views the design process as a rational decision-making process that is plan-driven. A designer takes a problem-solving approach and follows a plan and a set of constraints to achieve a goal state. The other approach is called action centric and views the design process as creative-driven improvised task. This approach uses reflection on action where a designer designs the solution as a reflection of conversation between him and the situation in which he alternates between conceptualizing the problem and making a move then evaluating that move. This move is an action dedicated for situation improvement.

Moreover, the design process activities include:

- *architectural design*: Where relationships among software components are highlighted.
- *Interface design:* It is the process of considering how each functionality is presented.
- *Component design*: Where services are assigned to components.
- *Database design*: It is specifications of data structures in a database.

Those activities lead to four outputs which are system architecture, user interface specifications, database specification and component specification (Sommerville, 2011).

Crowdsourcing

Crowdsourcing is a distributed problem-solving model that uses the computer and human computation combination. The term itself has been created by Howe (2006). Crowdsourcing is constructed of two words, 'crowd' which means the people contributing to and 'sourcing' which means the processes used to find, assess and engage suppliers of services.

Based on (Estellés-Arolas and González-Ladrón-de-Guevara, 2012), crowdsourcing can be defined as "a business practice that means literally to outsource an activity to the crowd". Similarly, a definition was made by Alonso and Lease as "the outsourcing of tasks to a large group of people instead of assigning such tasks to an in-house employee or contractor" (Alonso and Lease, 2011).

There are many characteristics make crowdsourcing different from outsourcing and traditional jobs. crowdsourcing is based on open calls, as anonymous individuals, groups and organizations are able to work simultaneously on a given project (Alonso and Lease, 2011). In addition, openness is a distinguishing characteristic. That is, individuals within the crowd provide inputs to the client on a voluntary basis or against payment not according to a contract.

Generally, three actors are included in crowdsourcing: Clients or Requesters, Platform and Workers. Clients are the ones in need of work to be done, Platform is where clients and workers can meet and Workers are the ones responsible for performing the work.

Crowdsourced Software Development

Crowdsourced Software Development is the application of crowdsourcing in all software development activities whether the activity itself yields a software or not such as, test case refinement and requirement elicitation (Ke Mao *et al*., 2015). It eliminates the difference between developers and end-users by enabling a co-creation practice, for instance, an end-user becomes a co-designer or co-tester.

Software development activities that have crowdsourcing applied to them include:

- Requirements, there have been several applications of crowdsourcing on requirement engineering such as acquisition, categorization and documentation. In addition, there are many commercial platforms that support crowdsourced requirements such as StakeSource (Lim *et al*., 2010) and CrowdREquire (Adepetu *et al*., 2012). Moreover, using crowdsourcing with requirements has opened other participation roles for stakeholders to be involved in such as joining in release planning and prioritizing and not only be a source of requirements
- Coding, crowdsourcing is used in three tasks; IDE enhancement, program optimization and programming support. It can be considered to be thoroughly explored in research and have many commercial platforms. Platforms include TopCoder (Lakhani *et al*., 2010), Crowdboost (Cochran *et al*., 2015) and Collabode (Goldman, 2012)
- Testing, it is the most explored activity within crowdsourcing. Crowdsourced testing offers testing capabilities to regular users and not only experts as in regular testing. Quadrant of euphoria (Musson *et al*., 2013) and iTest (Yan *et al*., 2014) are some of the examples on crowdsourced testing tools
- Verification, it is used with crowdsourcing mainly to minimize costs and skill needs due to the high cost of skilled experts used in regular verification (Schiller and Ernst, 2012). An example of a verification tool is CrowdMine (Li *et al*., 2012)

- Evolution and Maintenance, it has been one of the earliest benefactors from the application of crowdsourcing. Evolution has many research works published including runtime adaptation systems, scalability mechanisms and others (Ali *et al*., 2011; Bacon *et al*., 2009)
- Design, crowdsourcing is used for three tasks; user interfaces (GUI), architecture and design revision (will be discussed in more details in the next section)

Crowdsourced Software Design

Utilizing crowd participations in software design solves the problems in traditional designing approaches that face all stakeholders involved. Those problems include high cost, slowness and high risk because the resulted design might be of no good for customer. On the other hand, designer's problems include shortage in designing jobs no matter what qualification they have (Sommerville, 2011).

Involving the crowds in software design is a new trend that is gaining popularity among individual users and companies due to its many advantages; yet no many research works have been made on crowdsourced software designs specifically. Its main advantages include reducing overhead for companies as they do not have to hire many experts and instead they can benefit from the crowds in the software design process. In addition, it cuts expenses for the requester and allow the access to many design options (Wu *et al*., 2013).

Crowdsourced software design is used in three design areas (Ke Mao *et al*., 2015): First, it can be used in generating software user interfaces. Crowds are asked by a requester to generate interfaces based on a description and a possible set of constraints. User interface is a crucial part of the software and their correctness is important for the overall correctness of the software (Memon *et al*., 2003). This is more beneficial for non-expert users either because they are unable to design such interfaces or perhaps cannot design at that level of proficiency.

Second, crowdsourced software design can be used in architecture design. A requester asks the crowd of workers to create an architectural design for his software based on specifications that he chooses.

Last is the use of crowdsourcing in software design revision. It uses the crowds to give feedback on designs. It enables designers to enhance their work and get a feeling of their design's impact on the crowds (Bao *et al*., 2011).

Related Work

There are several research papers discussing and contributing to crowdsourced software design. The crowd powered interfaces design space is explored in Bernstein (2010), the exploration is conducted through using three prototype systems and different data collection themes such as friend sourcing, data mining and outsourcing. It provides supporting evidence that crowd sourced data whether from a crowd, labor market or social network site can be used to contribute in interface enhancements that allow high level tasks such as personalization.

Moreover Latoza *et al*. (2015) examined a limitation of the crowdsourced software design process and introduced a method to overcome it. The limitation is that the process is linear in that a winner is chosen and the potential of the rest of the crowd is thrown away. The experiment they conducted included two competitions where they requested two types of designs an initial then a revised design. The revised design was based on lessons learned from the crowd. The two competitions were on user interface design and architecture design. Designers in both competitions were able to revise and evaluate their peer's designs. By doing so the researchers found that the quality of the designs was higher as all designers benefited from each other's feedback on their initial designs and implemented enhancements in their revised designs.

Another paper by Nebeling *et al*. (2012) introduced an approach to involve crowds in the design process of web interfaces and created a platform: CrowdDesign which supports two models. First model is sharing and reuse which is based on constructing a common library of components that enables sharing either an interface or information component within the community. The second model is active crowdsourcing that enables an active request-response cycle. This enables the requester to directly request from the crowd of designers. The second model gives control over the time and cost.

In addition, Nebeling and Norrie (2011a), provide a model for using crowdsourcing in adapting user interfaces for websites. It presents the fact that most websites use a fixed interface layout and they are moving towards having several versions for their interface. This paper's method is to use a crowdsourcing model where developers create the initial version with adaptive features in which, at runtime, can evolve by assistance from users.

Nebeling and Norrie (2011b) contribute by creating a new crowdsourcing approach to web site user interface adaptation. It introduces a model for implementing interface adaptations within web sites to allow users to contribute and deploy their interface adaptations into the web site's interface. This is done through a double role for users, the first allows them to be seen as regular users who take advantage from the used user interface adaptations and the other allows them to be active contributors and deploy their own personal interface adaptations. Those contributions or changes are deployed into the CSS as modifications that can be

downloaded into the original website. This allows for a light weighted adaptation because no extra versions of the HTML document need to be managed. Also, this adaptation model allows computation to happen on the client side by linking the CSS to the web document which reduces costly server side computations.

Furthermore, Akiki *et al*. (2013) develop a crowdsourcing user interface adaptation approach to decrease visual complexity through decreasing bloat in interfaces. This paper explains that bloat in interfaces occur when many features are offered in an interface that are of no interest to the user and only distinct amount of features are of interest. This is handled through user interface (UI) feature-set reduction. It uses the same role-based user interface adaptation mechanism.

Wu *et al*. (2015) introduced an approach of the designing method of crowdsourced design when major variations in key parameters are present, such as variations in payment or assessment. The task submission quantity of different payments was easily measured but the quality was more challenging

Weidema *et al*. (2016) explored crowdsourced software design through micro-tasking and its ability to generate designs. It experimented using Amazon Mechanical Turk (AMT) and results showed three findings. First that it is possible for a crowd to produce a wide range of designs. Second is that there are major quality variations among those designs. Last is that although tasks were small, they are seen to be big by some workers.

Stol and Fitzgerald (2014) state that competition's prominent popularity in software activities is due to its variety of inputs. For instance, 99designs enables customers to crowdsource visual design tasks and web designs to choose the best choice. Bug bounties, as well, fall in this classification; diverse laborers may distinguish distinctive bugs, improving the probability a bug is found.

Li *et al*. (2016) introduced an agent-based crowdsourcing design in which it describes crowdsourcing as a distributed problem-solving approach and workers on the same software project are viewed as agents. It presents the framework as five tuples and six-step process. Tuples include tasks, collaborations, roles, agents, constraints. The suggested process in this study is:

- Step one: Crowdsourced design task (task1) is released in the platform by the requester agent.
- Step two: A number of worker agents submit their design proposals before the due date of the task.
- Step three: The platform performs under the following condition: If the proposal number is not equal to zero, then all proposals are released as a new task (task2) to be evaluated by the crowds and voted for the best. Else if the proposals equal zero then return to step one.
- Step four: A number of new worker agents will participate in task 2 and choose the best design proposal and submit their votes before a deadline. The design with most votes will be sent back to the requester in (task1). All workers participating in voting will get rewarded.
- Step five: If the sent design proposal is accepted by the requester then head to step six. Else the requester will release a new task asking for
- Step six: The winning design proposal worker will get a reward. End of process.

Research on design revision is done in Bao *et al*. (2011) where it explores design evaluation mechanisms and performs an evaluation on them to get better understanding on how they function. Two evaluation mechanisms are evaluated: Prediction voting and Likert scale rating. Prediction voting requests that the crowd predicts if a design solution is the winner of a contest and records the votes. On the other hand, Likert scale rating provides a five-point Likert scale to crowds to evaluate design solutions and then it uses the average score as the fitness value for that solution. Finally, Apparition system is proposed by Lasecki *et al*. (2015) which enables users to sketch their interfaces with a natural language description and the crowd with recognition algorithms will transform them into actual user interfaces.

From the coverage of the literature above, it is clear that there is no research providing comprehensive study of the current crowdsourced software design platforms including analyzing and comparing several platform workflows as what this research aims to do.

Crowdsourced Software Design Activities

An investigation has been made on crowdsourced software design platforms. Six platforms were investigated and are presented with their design activities (a summary is shown in Table 1). The investigation was done through using the platforms, watching tutorials and reading their help center answers. The following platforms were chosen because of their high numbers of active users. Therefore, making them most likely the preferred choice of platform to work on.

Covered Platforms

99Designs

It is a web-based platform for crowdsourcing user interfaces founded by Mark Harbottle and Matt Mickiewicz in 2008. It has over a million designers. It is considered to be a pioneer in crowdsourcing designs.

DesignCrowd

It is a crowdsourcing web-based platform that enables crowdsourcing user interfaces. It was established in 2008 by Alec Lynch and Adam Arbolino.

Table 1: Summary of platforms based on supported/unsupported current design activities

	Platforms					
Design activity	99Designs	DesignCrowd	DesignHill	CrowdSpring	CrowdSite	Freelancer
Verify designers				•		
Create design project	•	•	•	•	•	•
Comment on project briefs				•	•	
Notify designers		•				
Invite designers	•		•		•	
Invite clients	•					
Pre-selection of crowd designers	•					
Browse contests	•	•	•	•	•	•
Participation of crowd of designers	•	•	•	•	•	•
Eliminate designers	•	•	•	•	•	•
Withdraw design	•	•	•	•	•	•
Give feedback	•	•	•	•	•	•
Crowd submits revised designs	•	•	•	•	•	•
Link design		•				
Hold design		•				
Browse submitted designs	•	•	•	•	•	•
Select 1st round candidates	•					
Submit 2nd round designs	•					
Create voting poll/ Focus group			•	•	•	•
Select winner	•	•	•	•	•	•
Handover/ Wrap up design	•	•	•	•	•	•
Send Message	•	•	•	•	•	•

Key: • Demonstrates that the platform has the activity; Demonstrates that the platform does not have the activity

DesignHill

It provides web user interfaces through crowdsourcing and was founded by Rahul Aggarwal and Varun Aggarwal in 2014.

CrowdSpring

It was founded in 2008 by Ross Kimbarovsky and Michael Samson and it enables customers to crowdsource their user interface designs.

CrowdSite

It is a crowdsourced software design platform that enables user interface design crowdsourcing and was founded in 2009 by Roel Masselink.

Freelancer

It is a web-based platform for crowdsourcing software architecture and UML designs. It was found in 2009 by Matt Barrie.

Design Activities

The platforms include the design activities:

Verify Designers

This activity is available in CrowdSpring platform. Before allowing any designer to participate in any contest, the system asks him to submit three designs to each design contest category that he wants to participate in. Then the platform panel evaluates the designs. Based on the quality of those designs, the designer gets verified or not. If he is verified, he will be allowed to work on projects.

Create Design Project

This activity exists in all platforms. The requester creates a design contest and fills a form along with a features selection to complete the brief. The information in the form that he should fill includes background information on the organization he wants a design for, his industry, comments on the visual style whether colors or themes, number of pages needed to be designed if it was for an interface design, upload any images or files that are useful, contest title, contest duration, promotion options to have his contest promoted on twitter or other platforms, selects a design package in which it specifies the number of designers that will be participating in his contest and the billing information.

Notify Designers

This activity exists in DesignCrowd platform. The system will notify the **designers** of available contests. A notification will be sent to his email to notify him of any new contest posted.

Invite Designers

This activity is in 99Designs, DesignHill and CrowdSite platforms. The requester can invite designers to participate in his design contest project through clicking a button on the designer's profile in the platform.

Invite Clients

This activity exists in 99Designs platform. The crowd of designers can invite clients by email to perform design work for them through the platform. This is done when the designer clicks on the invite clients button in his clients list tab. Then the designer fills in the client's email and a description of the project that he wants to do for the client.

Pre-Selection of Crowd Designers

This activity exists in 99Designs platform. The system chooses which designers are allowed to participate. This activity is available only in highly paid contests to ensure high quality designs. This is done when a requester chooses a premium design package when creating his design contest brief. Only premium designers will participate.

Browse Contests

This activity exists in all platforms. The crowd of designers can view available contests by pressing on the contest tab in their account and filtering contest categories.

Participation of Crowd of Designers

This activity exists in all platforms. The designers choose to participate and submit their designs to the contest.

Hold Design

This activity exists in DesignCrowd. The system holds the submitted design for 24 h if the designer has a performance score rating, based on accumulation of previous performance scores, below than 2.5/5. This ensures that first designs a requester receives are of highest quality. This is done when a designer submits a design, it will have an 'under review' design status in the submissions list in the designer's account.

Browse Submitted Designs

This activity exists in all platforms. The requester can view the designs submitted to his project through the platform. He selects the contest he wishes to view and all the submitted designs to it will be shown.

Eliminate Designers

This activity exists in all platforms. The requester can remove designers from participating in his project.

Give Feedback

This activity exists in all platforms. The requester gives feedback as rating and comments to the submitted designs.

Comment on Project Briefs

This activity exists in CrowdSpring and CrowdSite. The requester and crowd of designers can communicate through the comment section in the design project brief.

Withdraw Design

This activity exists in all platforms. The designer can withdraw and remove his design submissions before a contest closes.

Crowd Submits Revised Designs

This activity exists in all platforms. The crowd of designers submit their designs after changing them based on the requester feedback.

Link Designs

This activity exists in DesignCrowd. The crowd of designers can link their final designs with the previous versions submitted.

Select First Round Candidates

This activity is available in 99Designs platform. The requester chooses 6 designers based on their submitted designs to move into the second round.

Submit Second Round Designs

The crowd of designers submit their final designs.

Create Voting Poll/Create Focus Group

This design activity is offered in DesignHill, CrowdSite and Freelancer as a voting poll and in CrowdSpring as a focus group. The requester, using the platform, creates a voting poll or a focus group to vote on 10 designs of his choice. Then the voting link is created and the requester can send it to the audience of his choice. The responses are then sent to the requester's email. This makes selecting a winning design easier.

Select Winner

This activity exists in all platforms. The requester chooses a winner from the participants.

Hand Over Design

This activity exists in all platforms. The requester has the right to request additional design changes after he chooses a winner within a period of time then pays the reward.

Send Message

This activity exists in all platforms. The requester can send private message designers through the message service on the platform.

Crowdsourced Software Design Workflow

A comprehensive software design crowdsourcing workflow is presented here based on analysis of the previously mentioned platforms. This helps better understand the current best practice process in crowdsourcing software designs. Figure 1 illustrates the workflow.

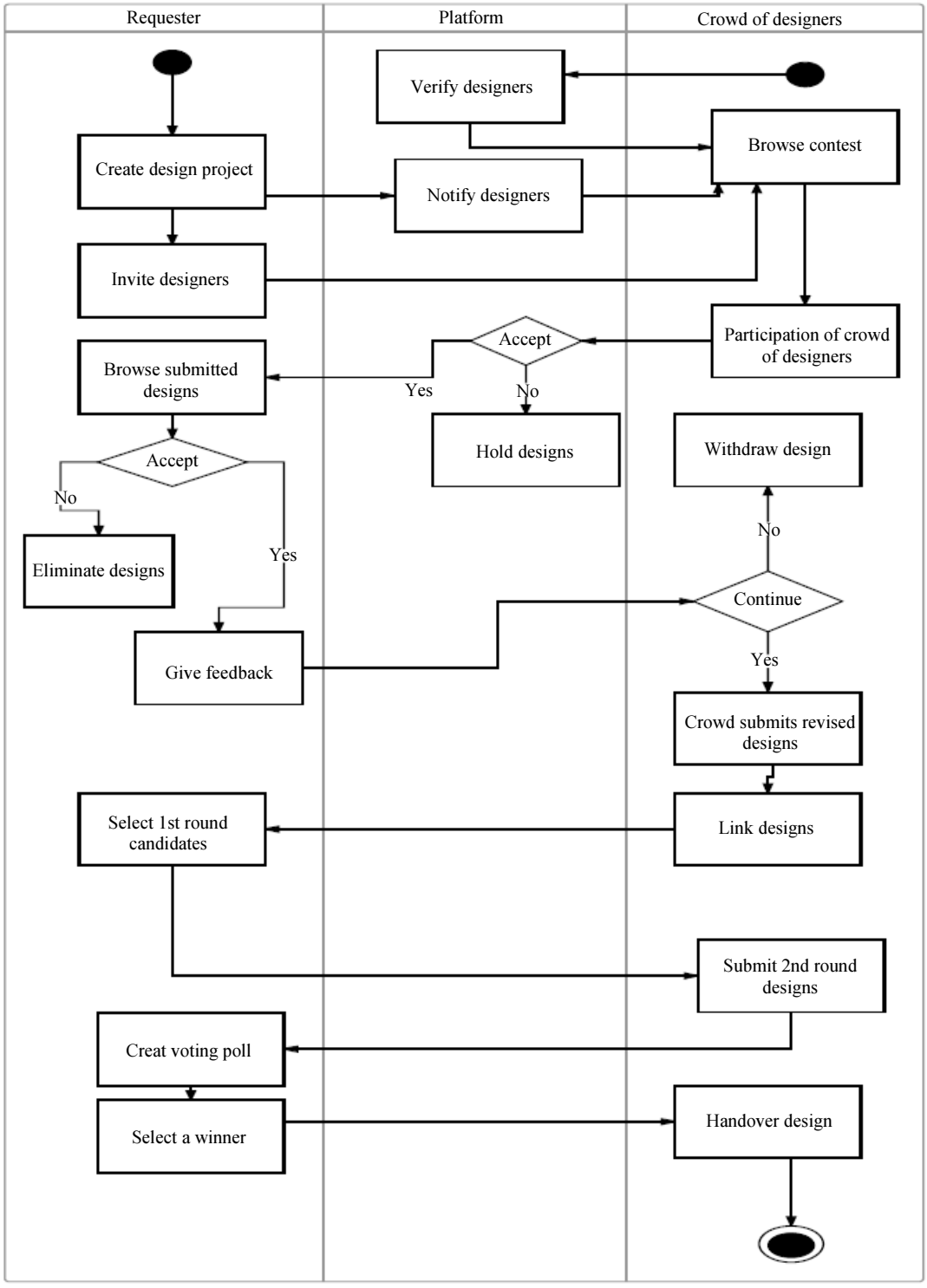

Fig. 1: Crowdsourced software design workflow

When a designer creates an account, he will need to be **verified**. This is done by choosing the design categories he wishes to work on then uploading three original designs to the platform. Those designs are sent to the verification panel. If they are approved, he will get verified and can participate in any contest under that category in the future but if they are not approved he will not get verified and will not be able to participate on the platform.

A requester after signing in the platform **creates a project** that performs as a contest. He specifies his design needs, provides complete information on the design requirements, the price and duration of contest. Then he either **invites designers** of his choice by clicking on the invite button on their profiles or the platform **sends a notification to the designers** who choose to be informed whenever a new contest is created. The contest will be visible to everyone but designers who choose to be notified are keeping themselves updated with new contests as they occur.

Designers then can **browse contests** either through the notifications of new contests, if they are subscribed to notifications, or through browsing the platform directly. It should be noted that designers can browse a contest even if they were not invited and even if they were not notified. They can do so by simply browsing the search engine within the platform.

Designers can **participate in contests**. Their designs go through another panel where their design is checked if it is of high quality, it will be sent to the requester but if it is of low or medium quality, it will be **hold** and released after 24 h to the requester. This is done to ensure that only highest quality designs are sent to the requester first.

The requester then can **browse the submitted designs**. He can either accept them and provide **feedback** to the designer to improve or if he does not accept them or does not want to work with a particular designer, he can **eliminate the designs** from his contest.

Once the designer gets the feedback from the requester he can work on improving his designs then **submits his revised design** or **withdraw his design**. He can withdraw his design any time before the contest ends. The designers also may **link** their first designs together with their final design so that the requester knows the improvement in their work based on his feedback.

After all the designers submit their revised designs, the requester chooses a number of designers to be **1st round candidates** that can go to the 2nd round.

Next, the candidates **submit their 2nd round designs.** Through the first and second round the requester can **gives feedback** to designers.

The requester may **create a voting poll** so that he can get assistance from audience of his choice to narrow down the best preferable designs. Then he can **select a winner**.

After the requester selects a winner he and the designer go through the **hand over** process where the requester can request additional minor modifications to the designs and the designer hands in the intellectual rights for his designs to the requester and gets paid.

Limitations/Improvements Analysis

This section mentions the limitations that were observed in the current crowdsourced software design platforms. In addition, a set of improvements are suggested for each limitation.

Verify Designers

Limitation

When a designer chooses to participate for the first time in a contest category, he will have to verify the quality of his designs in that category. The verification activity is done manually by an evaluation panel. This might take up to a week to be processed thus delaying the designer from contributing. This is not time consuming especially for designers in need to start working immediately.

Improvement

Instead of the current manual method of assuring quality of designs, an enhancement is proposed. The enhancement is to make the process semi-manual and reduce the load on the evaluation panel. This is done by making each designer who needs to be verified after submitting his 3 designs to also evaluate three designs for a random designer by selecting either ‘verify’ or ‘don not verify’ and add his justification. Then once a designer gets three verifications from three different peer designers, he will get verified automatically.

This way the evaluation panel will keep track on all verification requests with two lists: Designers waiting to be verified and designers verified by peers along with the peer justification. Designers who are not verified will still be in the waiting list for the evaluation panel to take a decision on whether to verify them or not.

Although, to the researchers’ knowledge, there has not been any mentions of an experiment to test or validate this improvement the following questionnaire, that was distributed to practitioners of the field, results illustrated that this improvement is justified.

Participation of Crowd of Designers

Limitation

There might be less number of crowds that are interested in participating because the possibility of a win is small as the highest experts usually win most contests (Yuen *et al*., 2011; Zheng *et al*., 2011). More frequently a crowd worker stops contributing after few none wins and this makes a gap between the number of crowds originally signed in and the active ones. This affects the requester’s contest as less designs are submitted to him and he gains less from the diversity of crowds.

Improvement

As Fried (2010) explained that usually motives for the crowd’s participation is social rather than financial. The amount of reward is important and affects the

number of participants but the gain in reputation and social status have more effect on crowd's participation.

Therefore, providing a feature that calculates points for reputation in two methods is suggested. The first is willingness, for example: Once a designer submits a design to a contest he gets a point. The second is quality, for example: When the requester chooses a winner, he has to choose 2nd and 3rd runner ups and the designer gets points if he is in 2nd place or 3rd place. This way the designers will gain even if they do not win a monetary prize.

Crowd Submits Revised Designs

Limitation

As shown in Latoza *et al*. (2015) experiment results, after a requester provides feedback to a designer, the changes might be major or even a complete re-work for the designers. This is considered to be a crucial set back to the designer as it might take him a very considerable amount of time to implement all the changes the requester needs. This is time consuming and takes a lot of effort from the designer's side.

Improvement

In order to improve this process and justified based on results of Latoza *et al*. (2015) experiment that one designer suggested to add a pre-design process, in which a requester initially receives wireframes instead of complete designs and gives his feedback. By doing so designers will make changes to wireframes. This indicates that this improvement is supported by designers and will save time and effort.

Therefore, when they submit their final wireframes and the requester chooses the ones to go forward into the contest, any further feedback he will give is going to be minor. This will likely reduce the amount of changes that the designers will do.

Select 1st Round Candidates

Limitation

When the requester receives the revised designs from all the designers, he chooses a number of them to enter the 2nd round of the competition. The limitation in this is that the requester needs to choose a certain number of designers and there might be two designers who have very similar designs that complement each other and work better as one design but are not sufficient individually. This can cause a loss of talent and it is considered to be a poor utilization of human capabilities.

Improvement

Allowing the requester to initiate a combine request to combine efforts of two designers and work as one can increase collaboration and quality. As one designer might capture certain elements that a requester needs in a design, another designer might have strength in using certain techniques and combining their efforts may result in a higher quality design that meets the requester's needs.

The combine feature would allow two designers that are not the best among their peers but might have potential if combined together.

Latoza *et al*. (2015) experiment has shown that there was a recommendation for a combine feature that allows collaboration. Although in that study the recommendation was to combine the two best designers to provide even better design nonetheless it still indicates that collaboration in any form increases quality and enhances outcomes.

Submit 2nd Round Designs

Limitation

Designers can submit their final designs in the 2nd round based on feedback from the requester only. This can be problematic as there is only the requester's perspective and the designers cannot learn from each other's work and cannot view the design from different angles.

Improvement

To improve this process, a peer evaluation feature can be added to all candidates participating in the last round. The designers will be able to view each other's work and borrow ideas. In addition, they will be able to evaluate each other's designs and provide ratings and comments within that specific contest. Based on Latoza *et al*. (2015) experiment results showed that exposing designers to each other's work will remarkably enhance the quality of the designs.

Limitations/Improvements Evaluation

An initial questionnaire was distributed to evaluate both the limitations and the proposed improvements. The questionnaire had 14 questions. Three questions were on the respondent's background. The rest were to validate the limitations and their improvements using a rating scale. Likert scale was used, it is a scale of five choices: (Strongly Disagree, Disagree, Neutral, Agree, Strongly Agree) that evaluates the respondent's level of agreement (Sullivan and Artino, 2013). The targeted respondents were with software development background from various regions across the globe. The questionnaire was distributed to 100 people and received 15 responses.

The following initial results were obtained from the responses.

The three questions on the background of the respondents had those results:

The first question concerning whether they had experience with using crowdsourcing in software engineering has received majority of Yes answers (Fig. 2).

The second question was about the respondent's field of expertise in software development. Responses varied with highest in design. One respondent added another choice, which was requirements engineering (Fig. 3).

The third question requested the number of experience years they had in software development. Most answers were around the three first options but none chose the plus 10 years of experience choice (Fig. 4).

For the rest of the questionnaire, the questions were about how much the respondents agreed with the limitations and their improvements. The results were as following:

The question concerning limitation1 has received responses around neutral, agree and strongly agree. The highest percentage was in agree (Fig. 5).

The first improvement question has received only 13.33% disagree (Fig. 6).

The question regarding limitation 2 has received responses in all choices except for strongly disagree and responses percentages were close. Overall, 60% either agree or strongly agree (Fig. 7).

The question concerning the second improvement has received majority of responses in agree then strongly agree (Fig. 8).

The question regarding limitation 3 has received majority of responses in agree (Fig. 9).

The question about the third improvement has received responses centered around agree and strongly agree with few responses in neutral (Fig. 10).

The question concerning limitation 4 had diverse responses. Nonetheless, majority of responses were in agree (Fig. 11).

The question about the fourth improvement had diverse responses while it had equal responses in strongly disagree, disagree and neutral, the highest response percentage was in agree then strongly agree (Fig. 12).

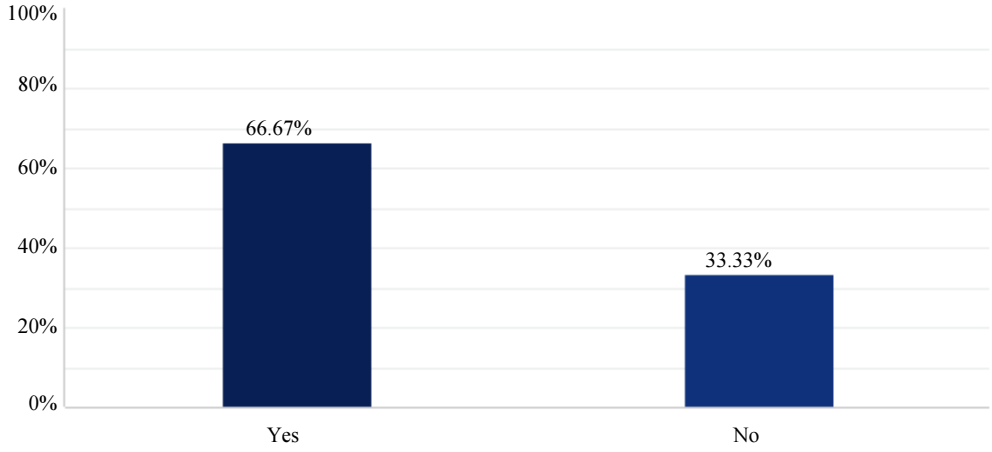

Fig. 2: First background question responses

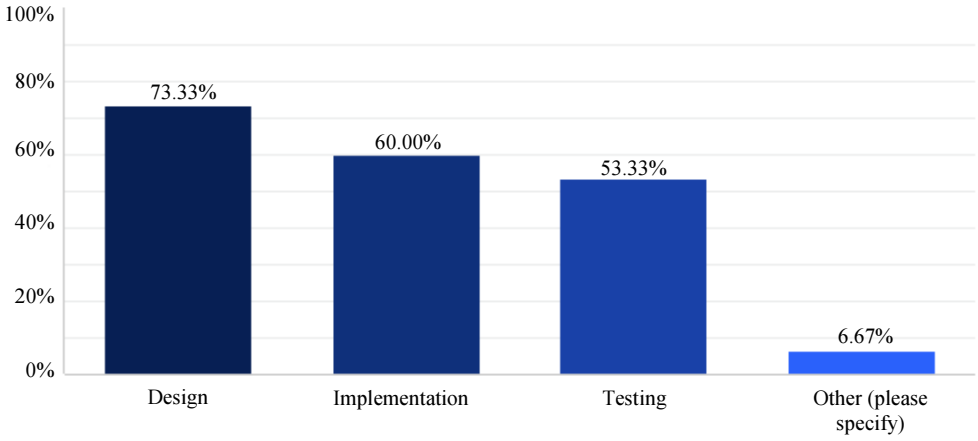

Fig. 3: Second background question responses

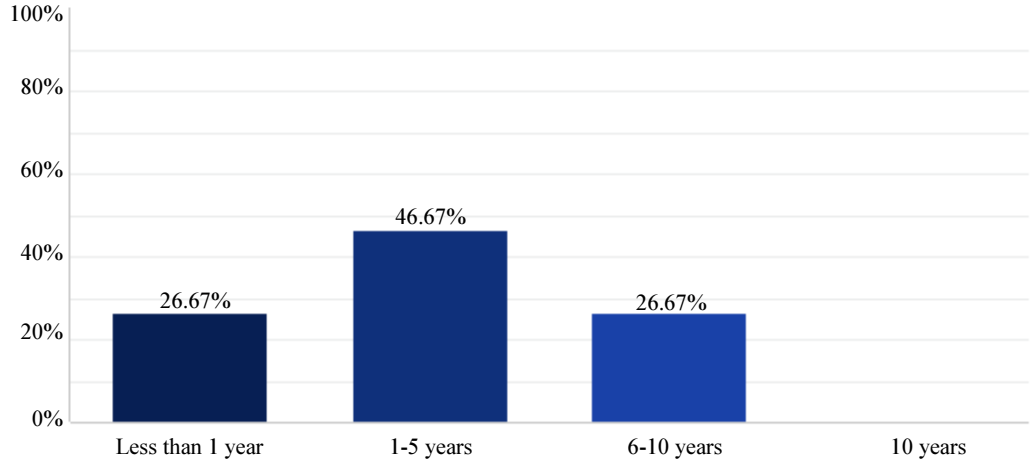

Fig. 4: Third background question responses

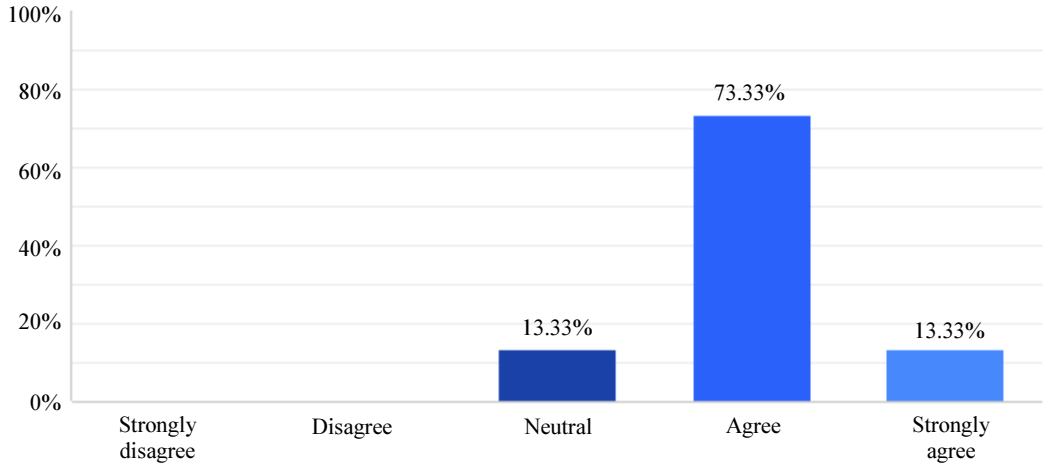

Fig. 5: First limitation responses

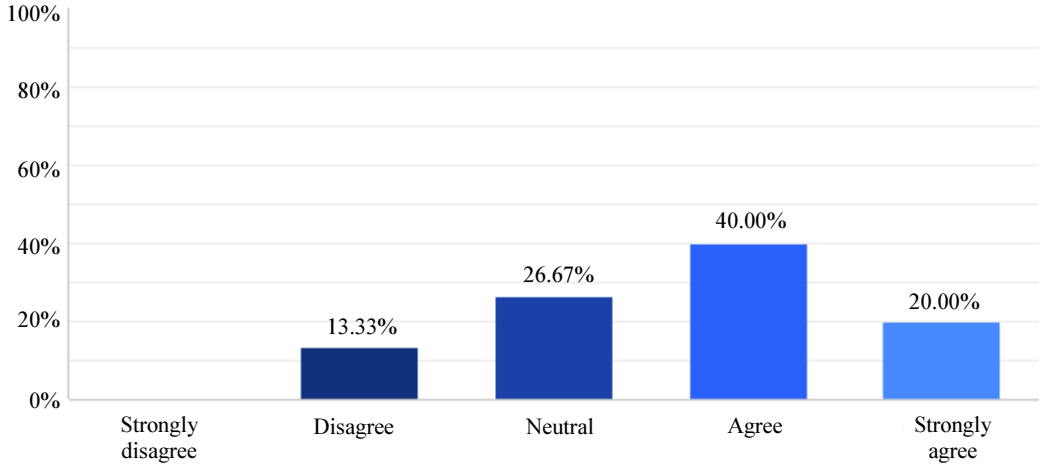

Fig. 6: First improvement responses

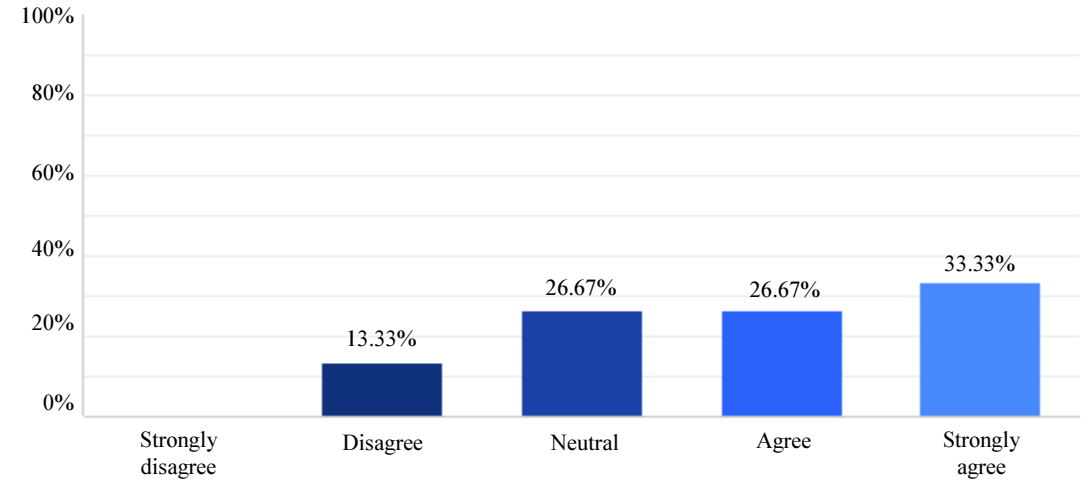

Fig. 7: Second limitation responses

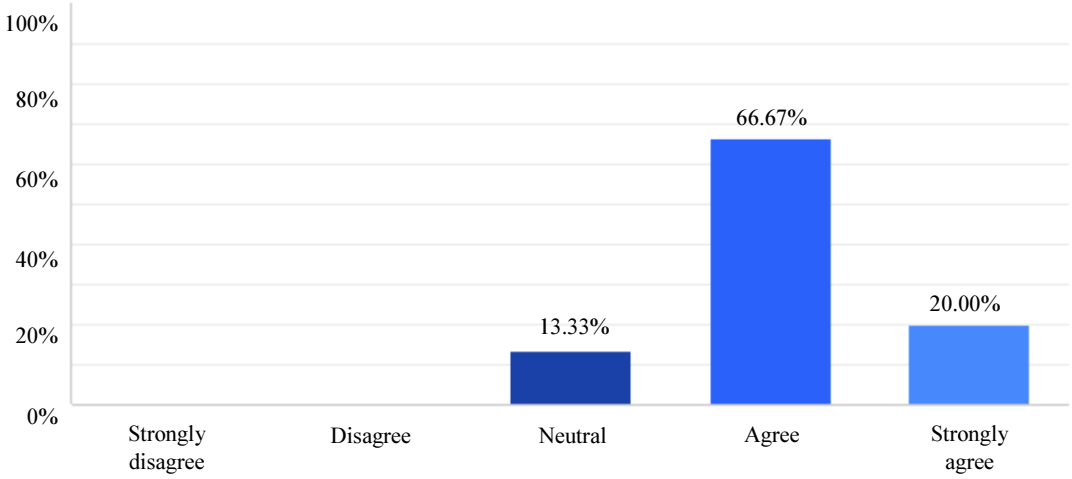

Fig. 8: Second improvement responses

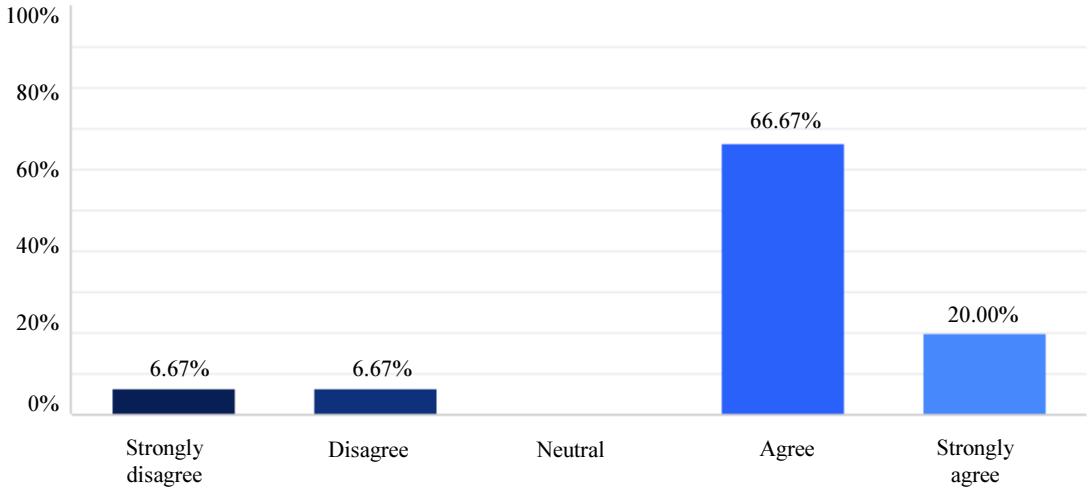

Fig. 9: Third limitation responses

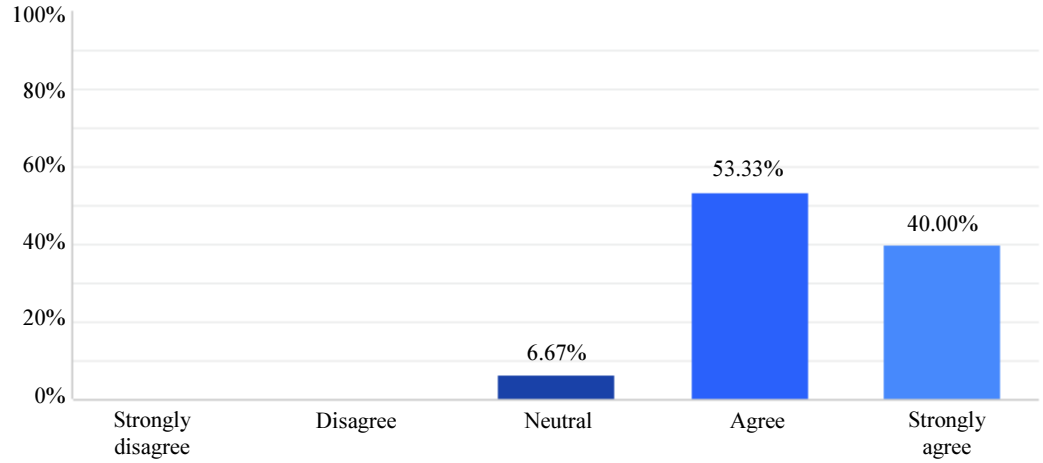

Fig. 10: Third improvement responses

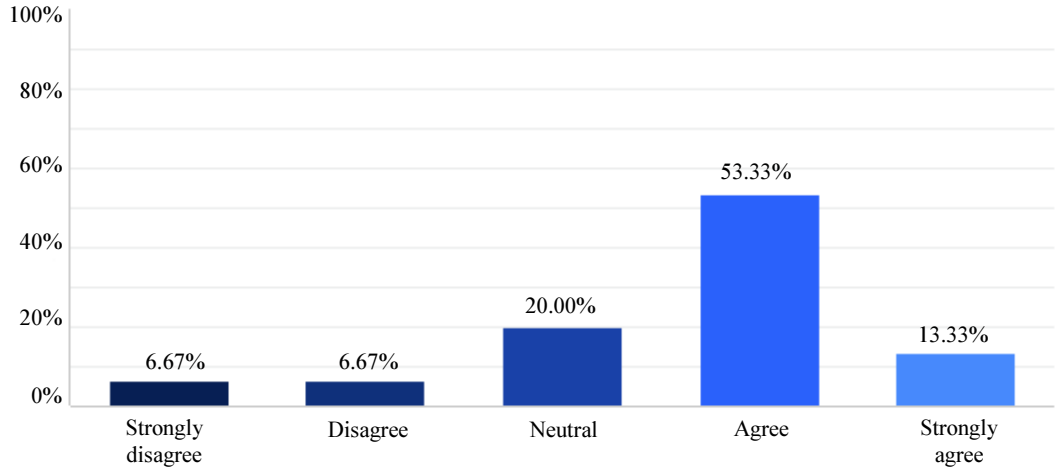

Fig. 11: Fourth limitation responses

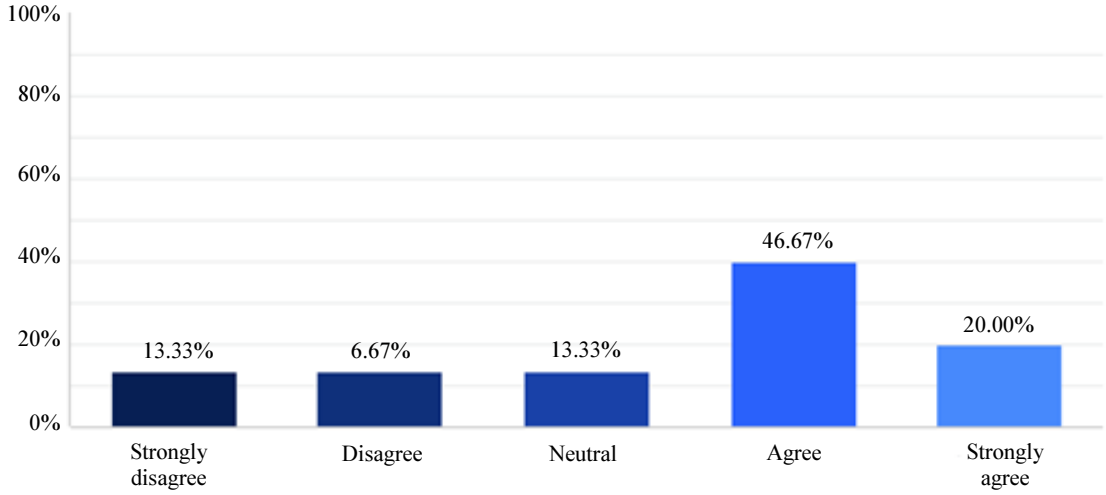

Fig. 12: Fourth improvement responses

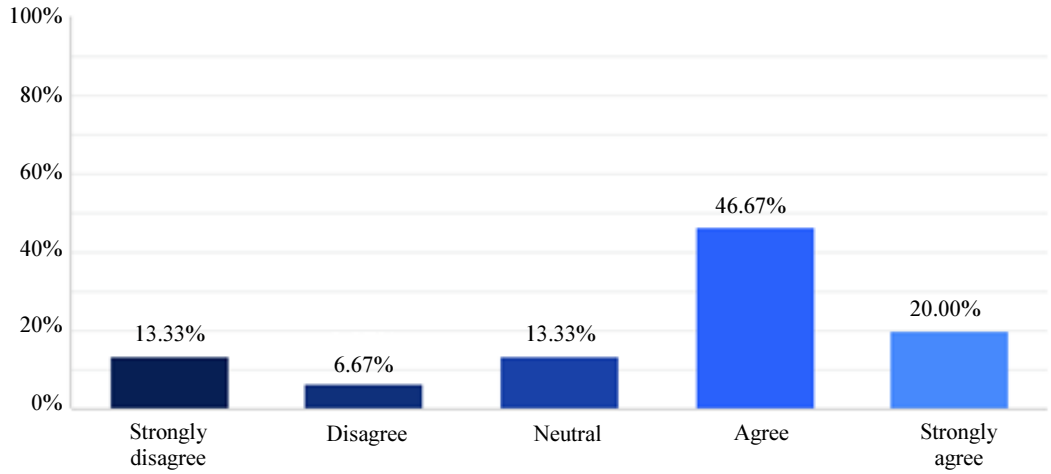

Fig. 13: Fifth limitation responses

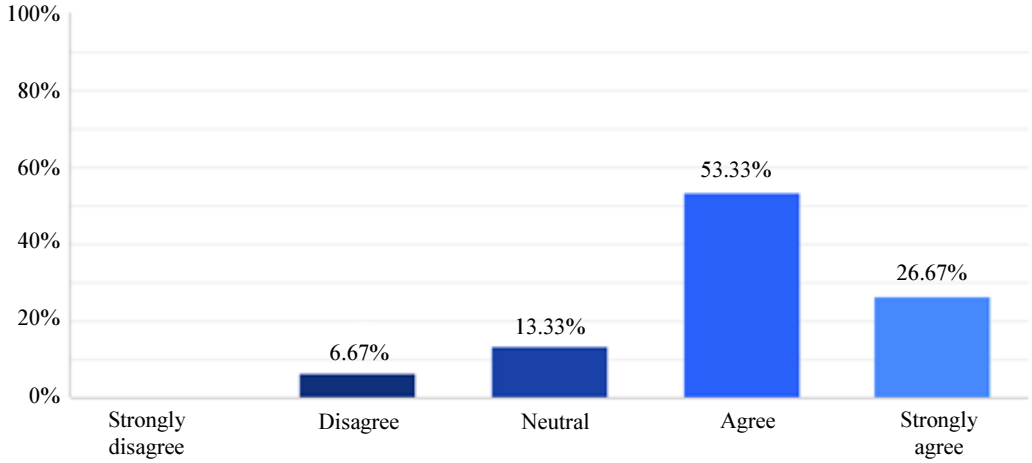

Fig. 14: Fifth improvement responses

The question considering limitation 5 has received different responses with highest in agree then strongly agree (Fig. 13).

The question about the fifth improvement had responses in every choice except for strongly disagree. More than half of responses were in agree (Fig. 14).

From the responses collected in this questionnaire it is evident that the limitations and their proposed improvements are valid and there is an actual need for them. It is true that there were some negative opinions on those results but the overall votes were in favor of this research's findings.

Conclusion

Crowd sourced software development has been used to support various aspects of the software development lifecycle activities including design.

In this research, the six major platforms that are used for crowdsourcing software design were analyzed and the basic design activities were listed and defined in detail. Afterwards, their corresponding workflow is illustrated. Subsequently, a number of five limitations were deduced from the platforms' workflow and five improvements to overcome those limitations were proposed.

Finally, those limitations and proposed improvements were evaluated through the use of a questionnaire that gathered the responses of software developers who are for the most part familiar with the use of crowdsourcing in software design. The questionnaire's results justified and validated the existence of such limitations and the need to overcome them as well as the effectiveness of the proposed improvements.

The intended extension for this research is to design an independent solution that solves the limitations through applying the proposed improvements. This

platform would be as a proof of concept and is not intended to be used by one specific platform.

Author's Contributions

Reem Aliady and Dr. Sultan Alyahya: Worked in all the research elements described in this work.

Ethics

This article is original contribution of the authors and is not published elsewhere. There is no ethical issue involved in this article.

References

Ali, R., C. Solis, M. Salehie, I. Omoronyia and B. Nuseibeh *et al*., 2011. Social sensing: When users become monitors. Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering, Sept. 05-09, ACM, Szeged, Hungary, pp: 476-479. DOI: 10.1145/2025113.2025196

Adepetu, A., K.A. Ahmed, Y. Al Abd, A. Al Zaabi and D. Svetinovic, 2012. CrowdREquire: A requirements engineering crowdsourcing platform. AAAI Technical Report SS-12-06, Wisdom of the Crowd.

Alonso, O. and M. Lease, 2011. Crowdsourcing 101: Putting the WSDM of crowds to work for you. WSDM, Hong Kong,

Akiki, P., A. Bandara and Y. Yu, 2013. Crowdsourcing user interface adaptations for minimizing the bloat in enterprise applications. Proceedings of the 5th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, Jun. 24-27, ACM, London, United Kingdom, pp: 121-126. DOI: 10.1145/2494603.2480319

Bernstein, M.S., 2010. Crowd-powered interfaces. Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology, Oct. 03-06, ACM, New York, pp: 347-350. DOI: 10.1145/1866218.1866220

Bao, J., Y. Sakamoto and J.V. Nickerson, 2011. Evaluating design solutions using crowds. Proceedings of the 17th Americas Conference on Information Systems, Aug. 4-7, Detroit, Michigan. DOI: 10.2139/ssrn.2201651

Bassil, Y., 2012. A simulation model for the waterfall software development life cycle. Int. J. Eng. Technol., 742-749: 2049-3444.

Bacon, D.F., Y. Chen, D. Parkes and M. Rao, 2009. A market-based approach to software evolution. Proceedings of the 24th ACM SIGPLAN Conference Companion on Object Oriented Programming Systems Languages and Applications, Oct. 25-29, ACM, Orlando, Florida, pp: 973-980. DOI: 10.1145/1639950.1640066

Cochran, R.A., L. D'Antoni, B. Livshits, D. Molnar and M. Veanes, 2015. Program boosting: Program synthesis via crowd-sourcing. ACM SIGPLAN Notices, 50: 677-688. DOI: 10.1145/2775051.2676973

Estellés-Arolas, E. and F. González-Ladrón-de-Guevara, 2012. Towards an integrated crowdsourcing definition. J. Informat. Sci., 38: 189-200. DOI: 10.1177/0165551512437638

Fried, D., 2010. Crowdsourcing in the software development industry. Nexus of Entrepreneurship and Technology Initiative Fall.

Goldman, M., 2012. Software development with real-time collaborative editing. PhD Thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, United States.

Howe, J., 2006. The rise of crowdsourcing. Wired Magazine, 14: 1-4.

Ke Mao, L.C., M. Harman and Y. Jia, 2015. A survey of the use of crowdsourcing in software engineering. University College London, United Kingdom.

Lasecki, W.S., J. Kim, N. Rafter, O. Sen and J.P. Bigham *et al*., 2015. Apparition: Crowdsourced user interfaces that come to life as you sketch them. Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, Apr. 18-23, ACM, Seoul, Republic of Korea, pp: 1925-1934. DOI: 10.1145/2702123.2702565

Latoza, T.D., M. Chen, L. Jiang, M. Zhao and A. Van Der Hoek, 2015. Borrowing from the crowd: A study of recombination in software design competitions. Proceedings of the 37th International Conference on Software Engineering, May 16-24, IEEE Xplore Press, Florence, Italy, pp: 551-562. DOI: 10.1109/ICSE.2015.72

Li, W., S.A. Seshia and S. Jha, 2012. CrowdMine: Towards crowdsourced human-assisted verification. Proceedings of the 49th Annual Design Automation Conference, Jun. 3-7, ACM, San Francisco, CA, USA, pp: 1254-1255. DOI: 10.1145/2228360.2228590

Lim, S.L., D. Quercia and A. Finkelstein, 2010. StakeSource: Harnessing the power of crowdsourcing and social networks in stakeholder analysis. Proceedings of the 32th ACM/IEEE International Conference on Software Engineering, May 2-8, ACM, Cape Town, South Africa, pp: 239-242. DOI: 10.1145/1810295.1810340

Li, H., L.Y. Hao, X. Ge, J. Gao and S. Guo, 2016. An agent-based approach for crowdsourcing software design. Proceedings of the Chinese Control and Decision Conference, May 28-30, IEEE Xplore Press, Yinchuan, China, pp: 4497-4501. DOI: 10.1109/CCDC.2016.7531795

Lakhani, K.R., D.A. Garvin and E. Lonstein, 2010. Topcoder (a): Developing software through crowdsourcing. Harvard Business School, Boston, USA.

Musson, R., J. Richards, D. Fisher, C. Bird and B. Bussone *et al*., 2013. Leveraging the crowd: How 48,000 users helped improve Lync performance. IEEE Software, 30: 0740-7459. DOI: 10.1109/MS.2013.67

Memon, A.M., I. Banerjee and A. Nagarajan, 2003. GUI Ripping: Reverse Engineering of Graphical User Interfaces for Testing. Proceedings of the 10th Working Conference on Reverse Engineering, Nov. 13-17, IEEE Xplore Press, Victoria, B.C., Canada, pp: 260-269. DOI: 10.1109/WCRE.2003.1287256

Nebeling, M., S. Leone and M. Norrie, 2012. Crowdsourced web engineering and design. Proceedings of the 12th International Conference on Web Engineering, Jul. 23-27, Springer-Verlag, Berlin, Germany, pp: 31-45. DOI: 10.1007/978-3-642-31753-8_3

Nebeling, M. and M.C. Norrie, 2011a. Context-aware and adaptive web interfaces: A crowdsourcing approach. Proceedings of the International Conference on Web Engineering, (CWE' 11), Springer, Berlin, pp: 167-170. DOI: 10.1007/978-3-642-27997-3_17

Nebeling, M. and M. Norrie, 2011b. Tools and architectural support for crowdsourced adaptation of web interfaces. Proceedings of the 11th International Conference Web Engineering, Jun. 20-24, Paphos, Cyprus, pp: 243-257. DOI: 10.1007/978-3-642-22233-7_17

Ralph, P., 2010. Comparing two software design process theories. Proceedings of the 5th international conference on Global Perspectives on Design Science Research, Jun. 04-05, Springer, Gallen, Switzerland, pp: 139-153. DOI: 10.1007/978-3-642-13335-0_10

Sommerville, I., 2011. Software Design Process. In: Software Engineering, Hirsch, M. (Ed.), Pearson Education Inc., United States, ISBN-10: 0-13-703515-2, pp: 37-40.

Schiller, T.W. and M.D. Ernst, 2012. Reducing the barriers to writing verified specifications. ACM SIGPLAN Not., 47: 95-112. DOI: 10.1145/2398857.2384624

Stol, K.J. and B. Fitzgerald, 2014. Two's company, three's a crowd: A case study of crowdsourcing software development. Proceedings of the 36th International Conference on Software Engineering, May 31-Jun. 07, ACM, Hyderabad, India, pp: 187-198. DOI: 10.1145/2568225.2568249

Schmidt, F.A., 2015. The design of creative crowdwork–from tools for empowerment to platform capitalism. Royal College of Art.

Sullivan, G.M. and A.R. Artino Jr., 2013. Analyzing and interpreting data from Likert-type scales. J. Graduate Med. Educ., 5: 541-542. DOI: 10.4300/JGME-5-4-18

Wu, W., W.T. Tsai and W. Li, 2013. Creative software crowdsourcing: From components and algorithm development to project concept formations. Int. J. Creative Comput., 1: 57-91. DOI: 10.1504/IJCRC.2013.056925

Wu, H., J. Corney and M. Grant, 2015. An evaluation methodology for crowdsourced design. Adv. Eng. Inform., 29: 1474-0346. DOI: 10.1016/j.aei.2015.09.005

Weidema, E.R., C. López, S. Nayebaziz, F. Spanghero and A. van der Hoek, 2016. Toward microtask crowdsourcing software design work. Proceedings of the IEEE/ACM 3rd International Workshop on CrowdSourcing in Software Engineering, May16-16, IEEE Xplore Press, Austin, TX, USA, pp: 41-44. DOI: 10.1109/CSI-SE.2016.015

Yan, M., H. Sun and X. Liu, 2014. iTest: Testing software with mobile crowdsourcing. Proceedings of the 1st International Workshop on Crowd-Based Software Development Methods and Technologies, Nov. 17-17, ACM, Hong Kong, China, pp: 19-24. DOI: 10.1145/2666539.2666569

Yuen, M.C., I. King and K.S. Leung, 2011. A survey of crowdsourcing systems. IEEE 3rd International Conference on Privacy, Security, Risk and Trust, Oct. 9-11, IEEE Xplore Press, Boston, MA, USA, pp: 766-773 DOI: 10.1109/PASSAT/SocialCom.2011.203

Zheng, H., D. Li and W. Hou, 2011. Task design, motivation and participation in crowdsourcing contests. Int. J. Electronic Commerce, 15: 57-88. DOI: 10.2753/JEC1086-4415150402