A Coordination Approach to Support Crowdsourced Software Design Process

Ohoud Alhagbani and Sultan Alyahya *

Posted Date: 24 October 2024

doi: 10.20944/preprints202410.1694.v1

Keywords: Crowdsourcing; Software design; Crowdsourcing software design; Coordination; Process model; Platform

Article

A Coordination Approach to Support Crowdsourced Software Design Process

Ohoud Alhagbani and Sultan Alyahya *

Department of Information Systems, College of Computer and Information Science, King Saud University
* Correspondence: sualyahya@ksu.edu.sa

Abstract: Crowdsourcing software design (CSD) is the completion of specific software design tasks on behalf of a client by a large, unspecified group of external individuals who have the specialized knowledge required by an open call. Although current CSD platforms have provided features to improve coordination in the CSD process (such as email notifications, chat, and announcements), these features are insufficient to solve the coordination limitations. A lack of appropriate coordination support in CSD activities may cause delays and missed opportunities for participants, and thus the best quality of design contest results may not be guaranteed. This research aims to support the effective management of the CSD process through identifying the key activity dependencies among participants in CSD platforms and designing a set of process models to provide coordination support through managing this activity. In order to do this, a five-stage approach is used: First, the current CSD process is investigated by reviewing 14 CSD platforms. Second, the review resulted in the identification of 17 possible suggestions to improve CSD. These suggestions were evaluated in stage 3 through distributing a survey to 41 participants who have experience of using platforms in the field of CSD. In stage 4, we designed ten process models that can meet the requirements of suggestions, while in stage 5, we evaluated these process models through interviews with domain experts. The result shows that coordination support in the activities of the CSD can add valuable contribution to the development of CSD platforms.

Keywords: Crowdsourcing; Software design; Crowdsourcing software design; Coordination; Process model; Platform

1. Introduction

Crowdsourcing software design (CSD) is usually organized around platforms. The platforms allow clients to request to perform design tasks and allow designers in finding tasks that are required from clients. Crowdsourcing platforms rely on contests which represents all the CSD activities that occur between independent designers that create solutions towards winning a reward for the completion of tasks. CSD is used in many design areas, such as web, mobile, embedded systems, cloud computing, etc.

When performing tasks in CSD, some dependencies occur between activities as different participants collaborate and it has many forms, the most common one occurs if two activities are related to each other, and a second activity cannot start before the first activity ends. When activities are related, if there is a problem in one of the activities, it must share this information with the others [1]. Managing these dependencies among activities requires coordination to implement the activities on time [2].

Regardless of the wide use of CSD, the lack of coordination is one of the major issues in CSD, as the lack of good coordination can lead to project failure and long delivery delays [3]. The literature identified multiple limitations associated with coordinating its process. For example, many designs will likely repeat due to a lack of coordination among participants, and potential waste of time in selecting best designs from a larger pool of designs [4–6]. Additionally, many designers can create similar designs for the same task without considering each other [7]. A lack of coordination support in CSD activities results also in delays and missed opportunities for designers [8]. The absence of coordination among these activities may lead to disturbances in performance.

 doi:10.20944/preprints202410.1694.v1

Therefore, many researchers have emphasized the need to study how to coordinate activities among a group of participants in CSD and what mechanisms need to be adopted to ensure process applied most effectively [1,6,9,10]. Additionally, CSD should support more complex and valuable work by defining tasks, interpreting the dependencies that occur when tasks are completed, finding and allocating appropriate designers to perform those tasks, monitoring and supervising their performance, and evaluating that their performance is of a high quality [7].

Coordination theory provides a lens through which to analyze tasks, participants, dependencies, and coordination mechanisms. This theory assumes that the compatibility or incompatibility of the dependencies and available coordination mechanisms may explain the issues participants face in achieving their goals [11–13].

The aim of this research is to support the effective management of CSD process through identifying the key activity dependencies among participants in CSD platforms and designing a set of process models to provide coordination support through managing this activity.

This paper is structured as the following: Section 2 presents a background of the software design, and the crowdsourcing concept and characteristics, and then it discusses the use of CSD. In Section 3, the approach is described in detail. Section 4 the current CSD process and activities are discussed. Section 5 identifies the activities that require coordination support. Section 6 discusses the results of the evaluation of the importance of supporting coordination in CSD activities, which have been done by conducting online surveys. Section 7 presents the design of the proposed models of CSD activities. Section 8 evaluates the proposed approach to manage the CSD process, while Section 9 concludes the paper.

2. Related Work

2.1. Crowdsourcing Software Design (CSD)

CSD is defined as the accomplishment of specified software design tasks on behalf of a client (typically, an organization) by a large and typically undefined group of external individuals with the requisite specialist knowledge through an open call [14].

In addition to the basic web and mobile design, Mao et al. [15] identified some possible design areas that can benefit from CSD. In Architectural Design, the requester asks the crowd of designers to create an architectural design for his program based on the specifications he chooses. In software design review, crowdsourcing designers are used to provide feedback on designs, enabling designers to improve their work [16].

Many research studies [2,14,17–20] defined three four main components to any crowdsourced software development system:

- Requesters: Companies or individuals, who post the issues they want to solve as a freelance job or contest.
- Crowd: The community of online workers who have signed up for the platforms and then decided as individuals or small teams whether to take on an open call challenge.
- Platforms: Computer systems that provide an online marketplace for workers and clients to meet. This marketplace mediates and connects requesters with online workers to solve tasks.

2.2. Literature Review

Many research papers discuss and contribute to the CSD process. Latoza et al. [21] proposed a limitation of the award-winner selection process, where the role is conducting two contests on user interface design and architectural design. Designers in both were able to review and rate their peers' designs. The researchers found that the quality of the designs was higher as all the designers benefited from each other's feedback on the initial designs and improvements implemented into the revised designs. Some designers can produce the best designs in much less time than others have produced low-rated designs. While recombination enables most designers to improve, it does not enable weak designers to produce strong designs even in adapting stronger designers' ideas. Murray-Rust et al. [22] stated that crowdsourcing software development handles tasks in a simplified format.

 doi:10.20944/preprints202410.1694.v1

To support the crowdsourcing of software development processes, the system needs to activate coordination mechanisms. It is often necessary to impose certain restrictions on the coordination and quality of these collaborations in order to manage them. Coordination protocols provide a high-level organization of activities (including planning and evaluation/feedback), while a set of coordination and quality constraints direct the assignment of workers to tasks. This allows the system as a whole to strike a balance between imposed restrictions and creative freedom in software development. The approach was evaluated by implementing an initial version of the coordination model for a case study and simulating its behavior on a heterogeneous group of workers.

Latoza and Hoek [21] discusses methods of exploring software development that enable multitudes of developers to contribute effectively through creating, distributing, and coordinating tasks. Underlying these approaches are three essential considerations: analysis, coordination, and quality. All of them influence how tools enable software to be built with a crowd. In this research, each aspect was studied, including analyzing a wide range of software development work into nuanced tasks, coordinating contributions across the crowd, and ensuring the quality of software produced by the crowd. Some challenges in coordination include how workers choose tasks, allocate the most efficient workers for the task that they have to do, how the system can track the tasks to be done, and how the dependencies between tasks are revealed and managed.

Alyahya and Alamer [23] investigated the different types of dependencies between the activities of the crowdsourcing platform. Through the search, five types of dependencies occurred between activities, among them, they dealt with two types. The first type was a hard prerequisite dependency, which is if there are two activities in a task and Activity 1 depends on Activity 2. If the developer of Activity 1 tries to finish it, the system must close Activity 2 because of its association with Activity 1. This type of dependency is very complex because parameters must be set that coordinate this sequence for all the related activities. The second type was a soft prerequisite dependency. This dependency also occurs between two activities. When a developer begins work on Activity 1, the system will send a message to the developer of Activity 2 that to this effect. This dependency is simpler than the previous one.

Xiao and Paik [7] mentioned that crowdsourcing software platforms allow clients to advertise their tasks and help them find workers to do the tasks, but the problem is the coordination among workers to complete the tasks. Through this research, a conceptual framework was formed to bring coordination support to crowdsourcing software platforms. It is not enough for these platforms to support crowdsourcing for creative, complex, and organized work that needs multiple workers with varied expertise. Coordination among these highly skilled workers is needed so that each worker, as a service that can be self-described and explored, can be dynamically assembled into complex teamwork. A workflow-based scheme was defined to structure and represent the work of the crowd. Crowdsourcing was then performed as the coordinate protocol was administered. The coordination protocol was designed to initiate and manage the crowdsourcing process by finding the worker, obligating him/her to carry out the tasks, and coordinating him/her in the protocol, where work was identified that consisted of a set of interrelated activities.

Machado [24] mentioned that Crowdsourcing software development platforms depend on a competitive approach in which workers independently create solutions while they compete against each other via financial rewards for completing tasks. Contest usually reduces cooperation. This research focused on studies related to cooperation between the primary software and the client in the assigning of crowd workers, presenting tasks, and the effect of cooperation among workers of the crowd on the quality of the task solutions. The research aimed to identify the characteristics of collaboration and the barriers that crowd members face in competitive crowdsourcing software.

Amrit [25] addressed an important part of coordination in software development, which is the problem of assigning tasks among members of the software development team. This research provided an analytical framework for research on teamwork. A vision for a potential future for the crowdsourcing business is laid out and entails worker considerations, such as motivation, feedback, and pay. This can be addressed through mechanisms to preserve reputation, provide better customer interaction, and increase pay. Also, customer considerations, such as coordination, the dismantling

 doi:10.20944/preprints202410.1694.v1

of tasks, and quality control, are considered. These can be addressed through workflow mechanisms such as collaboration.

Although there is research supporting coordination among participants in CSD platforms [26], it has not covered all aspects. To develop and contribute to the development of CSD, the importance of this research comes in bridging the gap related to the study of coordination among participants in CSD.

3. Approach

The aim of our research is to investigate the current process of CSD practice and to search for possible advancements to the process. To achieve this aim, we use a five-stage approach as follows:

Stage 1: Investigating current CSD process: This stage investigates 14 CSD platforms and identifies the key activities involved in the CSD process.

Stage 2: Identifying coordination needs: This stage is concerned with identifying the main dependencies involved in the activities extracted from the previous stage. The potential limitations of managing these dependencies along with suggestions of coordination mechanisms that may improve dependency management are identified. We used the approach of studying coordination dependencies in order to provide process improvement. This approach has been intensively used in the literature (e.g., [11,27]).

Stage 3: Evaluation of existing and proposed coordination mechanisms: This stage is the evaluation and comparison of the existing coordination mechanisms with the proposed ones. Data will be collected from questionnaires to assess existing coordination mechanisms and then participants will be interviewed to evaluate proposed coordination mechanisms based on an understanding of the way tasks are carried out and the dependencies that occur between activities.

Stage 4: Design process models: This stage presents the design of the proposed model to manage CSD activities. A set of process models has been created for the proposed activities agreed on in the previous evaluation. These models provide a visual representation of how the activities can be developed.

Stage 5: Evaluation of the process models: This stage evaluates the proposed model and discusses the outcome through interviews with experts in CSD.

4. Stage 1: Investigating Current CSD Process

CSD platforms allow clients to create a contest to perform specific design tasks for example web design sites, (HTML), (JS) services, etc. The CSD platforms covering various regions around the world were selected based on the number of users, number of clients, and appearance in internet search results and blogs, and the platforms most used as a case study in scientific research.

In addition to the above, the selected platforms provide all software design services such as (Architectural design, Component design, Algorithm design, Database design, System Interface design, User Interface design, and Logical design), the investigation included general crowdsourcing software platforms that contain software design services. These platforms are as presented in Table 1.

Table 1. Summary of investigated platforms.

Platforms	Platform type	Year of Foundation	Head Quarter	Community Size
99design [28]	Design	2008	Australia	4 million
DesignCrowd [29]	Design	2008	Australia	616,000
DesignHill [30]	Design	2014	India	125,000
CrowdSpring [31]	Generic	2008	USA	210,000
CrowdSite [32]	Generic	2009	Netherland	532,000
Freelancer [33]	Generic	2009	Australia	34 million
Upwork [34]	Generic	2000	Athens	5 million

 doi:10.20944/preprints202410.1694.v1

Designfier [35]	Design	2017	USA	Unknown
DesignContest [36]	Design	2003	USA	160,000
Guerra Creativa [37]	Design	2011	Argentina	14,000
Hatchwise [38]	Design	2008	USA	8 million
110designs [39]	Design	2011	USA	22,000
MOJO Marketplace [40]	Generic	2009	USA	5.8 million
Fiverr [41]	Generic	2010	USA	Unknown

The review of fourteen CSD platforms was performed to identify current CSD operations and their activities. An investigation of the activities in CSD platforms was conducted using the following methods: reading the official description of the nature of the work of the platform, which is available on the website of the CSD platform, watching the videos on the platform that explain how it works, reading the comments of the participants on the platform., contacting platform support and ask about the features offered to participants, communicating with some clients and designers on the platform and inquiring about how to work on contests, and visiting the forums of the platform that bring together the participants in the platform.

To understand the workflow of the current design process in CSD, the platforms are analyzed and accordingly the crowdsourcing workflow for software design is drawn. Figure 1 illustrates the current CSD workflow. Most CSD platforms follow a general process, which can be seen and understood from the flow of activities across the platforms. As mentioned in the Approach Section, the result of the review is presented in [99].

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 24 October 2024 doi:10.20944/preprints202410.1694.v1

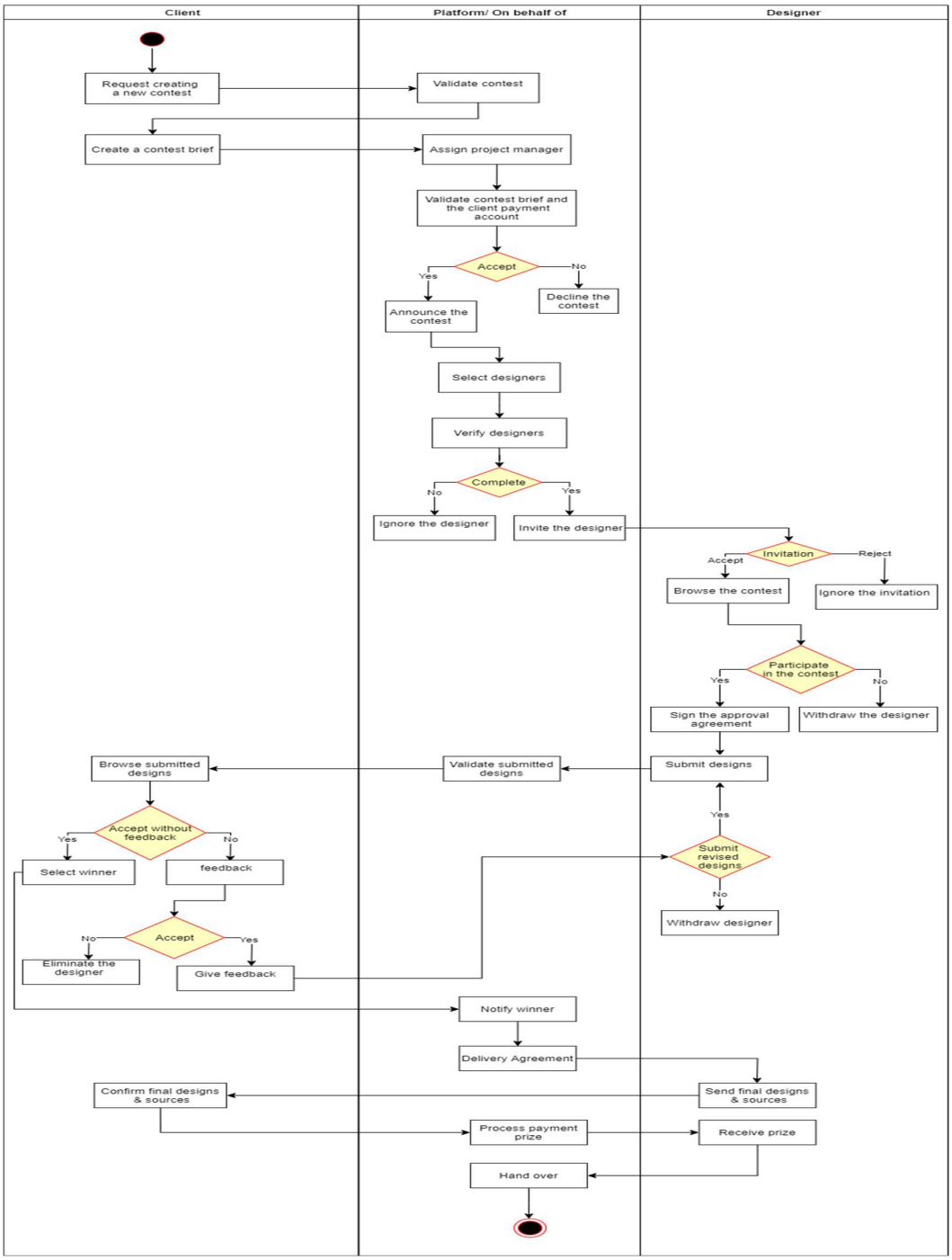

Figure 1. The workflow of the design process in crowdsourcing software design platforms.

5. Stage 2: Identifying Coordination Needs

Coordination is important to ensure that activities are implemented in the CSD on time, so it was necessary to communicate between clients, designers and platform representatives (i.e., project managers) to build a contest. Based on coordination theory, coordination is defined as the process of managing the dependencies between activities. Thus, studying coordination means analyzing the dependencies that appear between tasks in the system and determining how to manage those dependencies [89].

 doi:10.20944/preprints202410.1694.v1

An analysis of CSD activities on available CSD platforms and the results of current research in this field show that the coordination support for CSD activities needs to be improved for a better crowdsourcing contest. This section analyzes the limitations of coordination support provided by the current CSD process and suggests a set of coordination mechanisms that can help overcome these limitations.

We followed heuristics provided by Crowston and Osborn [12] for analyzing dependencies and coordination mechanisms in a situation to make dependency focused analysis. They proposed this procedure: identify dependencies, and then search for coordination mechanisms. In other words, look for dependencies, then ask which coordination support is used to manage those dependencies. Crowston and Osborn [12] suggested asking questions such as the following to discover dependencies and coordination mechanisms:

- What are the inputs/outputs to each activity (e.g., informational and other necessary pre-conditions/post-conditions)?
- What potential performance problems associated with this process? Do these problems reflect unmanaged [or poorly managed] dependencies?

Table 2 emphasizes to what extent the current CSD platforms support these activities, identifies the dependencies in each activity, discusses the potential limitations, and discusses how current CSD platforms can be improved to better coordinate and support CSD activities.

Table 2. Coordination support needed for CSD activities.

Activity	Dependency	Current Platform Support	Potential Limitations	Proposed Coordination Mechanisms
A1: Create contest brief	The project manager is assigned to the contest after the client creates the contest brief.	The project manager will only be assigned by filling out the contest form, as the platform provides a form for the client to manually write the contest brief and attach all files and contest details, and this form helps reduce time and effort in selecting the project manager.	L1. The unavailability of a form including all the information related to the contest that will be conducted may lead to a misunderstanding of the requirements of the contest.	S1. The platform provides a detailed form that includes all the requirements to ensure the proper functioning of the contest (i.e., client expectations, design preferred colors, design plans).
			L2. The client may ignore the need to attach important files to the contest, which may lead to design errors.	S2. Set constraints on the client to ensure they add all the information and details related to the contest in terms of the objectives and the expected final result from the contest and that they attach the files related to the design.
A2: Assign a project manager	Platforms assign a project manager for each contest and then start an announcement for the contest.	The platform assigns the project manager manually based on the contest brief and client requirements, and the project manager must validate the contest, then publish the contest announcement on the platform page.	L3. Manually selecting a project manager may take a significant amount of time away from the contest.	S3. Automatically select the project manager based on the type of contest.
			L4. A project manager who is busy with too many contests may be selected, and therefore, the project manager's schedule would not fit the new contest and thus may affect the effectiveness of the contest.	S4. Set constraints on capacity related to the association of contests with project manager.

 doi:10.20944/preprints202410.1694.v1

Activity	Dependency	Current Platform Support	Potential Limitations	Proposed Coordination Mechanisms
A3: Validate contest brief	Before the contest can be announced, the project manager must validate the contest and payment.	The project manager manually validates the contest, reviews the contest feed, verifies all client accounts and data, and then publishes the contest announcement.	L5. A project manager may publish unqualified contests because he may not be in his fields of interest or competence.	S5. It is good to categorize the contest by making a list of contest types by field and competency and to ensure that the project manager has a background related to the type of contest to help validate the contest efficiently.
A4: Announce the contest	The project manager chooses the designers after the contest is announced.	The project manager announces the contest by announcing it on the platform page, and thus the project manager can select the designers following the requirements attached to the contest brief.	L6. There is a lack of multiple communication channels available to send contest announcement messages to designers, thus resulting in a potential missed opportunity for different designers to participate in the contest.	S6. Send messages or notifications on social media accounts or email regarding a new contest.
A5: Select designers	The project manager selects suitable designers to invite to participate in the contest.	The project manager selects the designers based on the client's requirements, the selection is based on the designer's rank, grade, and the number of times to participate in contests and similar designs, then the project manager sends invitations to the designers.	L7. The availability of designers on the platform is not considered so there is no distinction made between active and inactive designers when they are invited to participate in the contest. This may lead to a lack of diversity in designs because not all different designers who have been invited participate, and this may lead to the loss of the opportunity to participate for new active designers.	S7. Set a constraint to select active designers only.
			L8. New designers may not be selected because they did not participate in any contests.	S8. Reserve some invitations for new designers to participate in the contest.
A6: Verify designers	The project manager verifies the designers, to invite them to participate in the contest.	The project manager can manually determine the suitability of designers who have a design history registered in the platform.	L9. It may be difficult for the project manager to verify some designers because new designers do not have a design history or because there is no history of the designers in the contest field.	S9. The suitability of designers without history can still be evaluated by asking them to submit a sample of similar designs so that the project manager can check their skill level.
A7: Invite designers	The project manager sends invitations to designers to participate in the	The project manager automatically invites the appropriately selected designers via email or through notifications on	L10. Designers may not check their email for contest notifications.	S10. To communicate with designers through various communication channels, such as WhatsApp or text messages.

 doi:10.20944/preprints202410.1694.v1

Activity	Dependency	Current Platform Support	Potential Limitations	Proposed Coordination Mechanisms
	contest, so the designers can browse the contest and accept the invitation or not participate.	the designer page, thus through the invitation designers can browse the contest and see the contest brief, which helps them decide whether to accept the invitation and participate in the contest or not to accept the invitation to participate.	L11. The time difference between countries is not considered when sending invitations to designers.	S11. Take into account the time difference when sending invitations to designers; The project manager must check the time before sending out invitations to the designers, and an additional day can be added to the invitation if there is a significant difference in time zones before the start of the contest.
A8: Approval Agreement	Before designers can browse the contest, they must accept or reject the approval agreement on the invitation.	The project manager sends the invitation to the designers by email. If the designers decide to participate in the contest, they move to the contest page and then move to the approval agreement. The designer can, then agrees and browse the contest brief.	L12. The platform does not remind designers about the need to agree to the Approval Agreement if they do not respond.	S12. Remind the designers that the approval agreement has not been completed and that he is not currently considered a participant in the contest. It is also a good idea to provide automated support to respond to the approval agreement through notifications or multiple communication channels, such as WhatsApp or text messages.
A9: Participate in the contest	Designers work on the contest and submit their designs for validation, based on the time specified in the contest brief.	Platform can only prevent designers from submitting their works if deadline is reached.	L13. There is a lack of mechanisms to monitor contest progress and designers who do not submit designs to the contest.	S13. Send notifications to designers about the progress of the contest or notify inactive designers to fulfill their obligation to submit designs; otherwise, they must be withdrawn to allow new designers to join the contest.
A10: Validate submitted designs	The project manager waits until a sufficient number of designs have been sent by the designers, and the project manager validates the designs before submitting them to the client.	The Project Manager validates designs for compliance with the contest brief, browses, reviews and ratings, and presents fair designs in proportion to the contest brief to the client to save his time and effort in searching for designs.	L14. The validation process takes place in the final stages of the contest. Thus, the validation process may take long time. This may waste the designers' time and effort.	S14. Provide support by notifying the client to validate the designs as soon as the project manager receives the designs from designers.

 doi:10.20944/preprints202410.1694.v1

Activity	Dependency	Current Platform Support	Potential Limitations	Proposed Coordination Mechanisms
A11: Browse submitted designs	The client browses and reviews submitted designs that meet the requirements of the contest brief	The client browses the designs submitted by the designers, and reviews them in accordance with the requirements of the contest brief. The client either selects the designs that are compatible with feedback, or cancels the designs that are not appropriate.	L15. The client eliminates some designs during the evaluation phase when they are evaluating the designs, which in turn may result in wasted designer effort.	S15. Setting constraints and obligations on the client to oblige him to browse all designers' designs that are in the meet with the contest brief that was sent by the project manager in the early stages of the contest.
A12: Give Feedback	The client has to browse the designs in the first round and give feedback on the designs, until they move to the final stage of the contest and the designers submit the revised designs.	The client browses the designs submitted in the first stage and chooses, while browsing, the designs that suit the requirements of the contest, while giving manual feedback to make the required modifications so that they can move to the final stage.	L16. Clients do not provide designers with direct feedback nor communicate with designers during the contest period. This may result in poor designs.	S16. Provide a mechanism for communication through the platform that enables clients to provide feedback to the designers during the contest.
			L17. It is possible that two designs are submitted, but neither can be accepted in their current states. Currently, there is no mechanism to enable clients to request to combine both designs to produce a better design.	S17. Provide a mechanism during the feedback phase that enables the client to combine two designs that complement each other.

6. Stage 3: Evaluation of Existing and Proposed Coordination Mechanisms

In order to evaluate the need for the proposed coordination support, an online survey was conducted. The survey is distributed through email and social media and received 41 responses. It is taken into account that the selected people must have a strong background in the domain of CSD.

The respondents were asked to rank the limitations and suggestions affecting contest coordination in CSD platforms on a Likert scale of 1-5 where 1 means strongly disagree and 5 means strongly agree. Seventeen limitations and seventeen suggestions affecting contest coordination in CSD platforms were evaluated. The descriptive statistical analysis [94] was used to find out the respondents' opinions regarding the limitations and suggestions in the stages of the design contest (Table 3).

Table 3. Statistical results for the limitation and suggestions of existing and proposed coordination mechanisms.

Limitations / Suggestions	Average	Interpretation	Limitations / Suggestions	Average	Interpretation
L1	4.09	Positive	S9	3.41	Positive
S1	3.88	Positive	L10	3.53	Positive
L2	3.76	Positive	S10	3.32	Moderate
S2	3.68	Positive	L11	3.37	Moderate
L3	3.71	Positive	S11	3.26	Moderate
S3	3.62	Positive	L12	3.52	Positive
L4	3.82	Positive	S12	3.35	Moderate
S4	3.74	Positive	L13	3.77	Positive
L5	3.91	Positive	S13	3.65	Positive

 doi:10.20944/preprints202410.1694.v1

S5	3.65	Positive	L14	3.47	Positive
L6	3.88	Positive	S14	3.56	Positive
S6	3.79	Positive	L15	3.32	Moderate
L7	3.52	Positive	S15	3.35	Moderate
S7	3.27	Moderate	L16	3.68	Positive
L8	3.38	Positive	S16	3.71	Positive
S8	3.56	Positive	L17	3.53	Positive
L9	3.65	Positive	S17	3.47	Positive

The survey findings show that it is clear that a large number of participants agreed on that these CSD activities suffer from the identified coordination limitations and that the proposed coordination mechanisms are valid, although not all of them did. This is normal due to their different roles, responsibilities, and needs. However, the goal of this evaluation is to make sure that these CSD activities would provide more coordination support for CSD process. The full results of this evaluation can be found in this external webpage[1].

7. Stage 4: Design Process Models

This section presents the design of new process models incorporating the proposed coordination mechanisms identified earlier in this research. The list of these models and how they cover the proposed mechanisms are shown in Table 4.

Table 4. Proposed process models and corresponding coordination mechanisms.

The design contest part	The corresponding process model	Activity covered	The Proposed Coordination mechanisms
Creating design contest	P1: Create contest brief	A1	S1. The platform provides a detailed form that includes all the requirements to ensure the proper functioning of the contest (i.e., client expectations, design preferred colors, design plans).
			S2. Set constraints on the client to ensure they add all the information and details related to the contest in terms of the objectives and the expected final result from the contest and that they attach the files related to the design.
	P2: Assign a project manager and Validate contest brief	A2	S3. Automatically select the project manager based on the type of contest.
			S4. Set constraints on capacity related to the association of contests with project manager.
		A3	S5. It is good to categorize the contest by making a list of contest types by field and competency and to ensure that the project manager has a background related to the type of contest to help validate the contest efficiently.
	P3: Announce the contest	A4	S6. Send messages or notifications on social media accounts or email regarding a new contest.
Performing designs and following up on contest design activities	P4: Select designers	A5	S7. Set a constraint to select active designers only.
			S8. Reserve some invitations for new designers to participate in the contest.
	P5: Verify designers	A6	S9. The designer who wants to participate in a new type of contest or for new designers who do not have a history to submit a sample of similar designs so that the project manager can check their skill level.

[1] https://github.com/salyahya99/CSD-Process

 doi:10.20944/preprints202410.1694.v1

The design contest part	The corresponding process model	Activity covered	The Proposed Coordination mechanisms
	P6: Invite designers	A7	S10. To communicate with designers through various communication channels, such as WhatsApp or text messages.
			S11. Take into account the time difference when sending invitations to designers; The project manager must check the time before sending out invitations to the designers, and an additional day can be added to the invitation if there is a significant difference in time zones before the start of the contest.
	P7: Approval agreement	A8	S12. Remind the designers that the approval agreement has not been completed and that he is not currently considered a participant in the contest. It is also a good idea to provide automated support to respond to the approval agreement through notifications or multiple communication channels, such as WhatsApp or text messages.
	P8: Participate in the contest	A9	S13. Send notifications to designers about the progress of the contest or notify inactive designers to fulfill their obligation to submit designs; otherwise, they must be withdrawn to allow new designers to join the contest.
	P9: Validate designs and provide feedback by client.	A10	S14. Provide support by notifying the client to validate the designs as soon as the project manager receives the designs from designers.
		A11	S15. Set constraints and obligations on the client to oblige him or her to provide feedback to the designers in the first stages of the contest in addition to browsing all the designs submitted in the first stage.
	P10: Combine two designs that complement each other through feedback	A12	S16. Provide a mechanism for communication through the platform that enables clients to provide feedback to the designers during the contest.
			S17. Provide a mechanism during the feedback phase that enables the client to combine two designs that complement each other.

Ten process models are suggested. Due to the page limit of this article, we discuss in details the first three ones below, while the rest of models can be found in the Appendix.

7.1. P1: Create Contest Brief

This process model covers the proposed activity A1: Create contest brief. The process model in Figure 2 demonstrates that the platform shall provide a detailed form that includes all the requirements to ensure the proper functioning of the contest.

 doi:10.20944/preprints202410.1694.v1

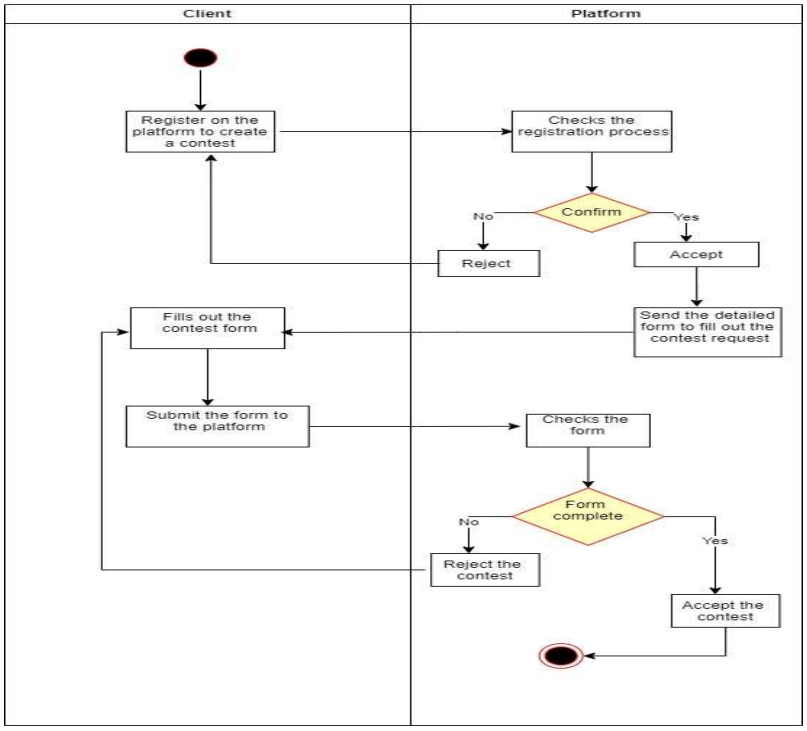

Figure 2. Process model 1 'Create contest brief'.

When a client registers on the platform to request a design contest, then the platform determines automatically the type of contest and sends customized form that matches that type. The platform sends a detailed form for the contest that includes the following information: the name of the client, type of design, design language category, the main purpose and objectives of the contest, the potential impact of the contest, what are the main elements of the design contest, and the actions you want the designers to take in the contest. The client fills out the contest form and optionally attaches the files he wishes, then sends them to the platform, and the platform verifies the detailed form of the contest.

7.2. P2: Assign a Project Manager and Validate Contest Brief

The process model in Figure 3 demonstrates how to assign a project manager and validate contest brief. The platform automatically verifies the project managers present on the platform in accordance with the type of design contest. After that, the platform checks the mechanism of participation of the project manager in the contests, whether full-time or part-time and the current number of contests, as the platform appoints the project manager who has the least number associated with the contests. The platform then will send notification to the project manager that he has been selected to be the project manager for the new contest.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 24 October 2024 doi:10.20944/preprints202410.1694.v1

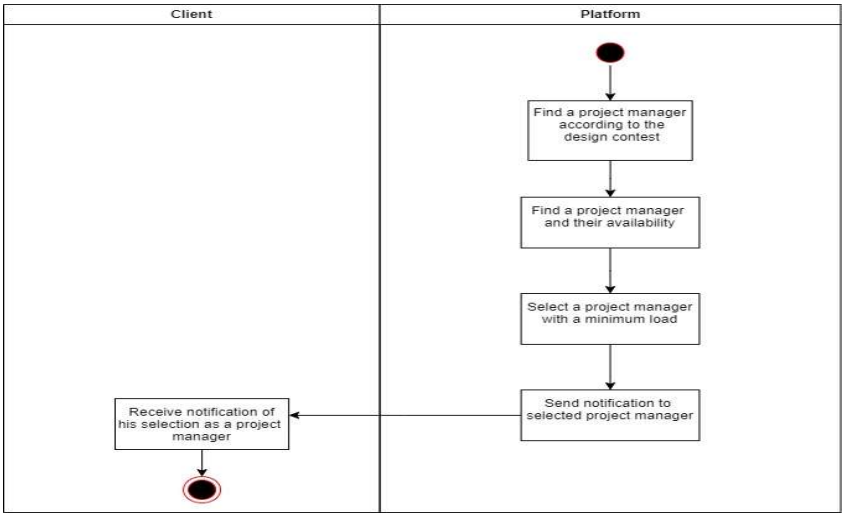

Figure 3. Process model 2 'Assign a project manager and Validate contest brief'.

7.3. P3: Announce the Contest

The process model in Figure 4 illustrates the importance of having multiple communication channels to send contest announcement messages to designers.

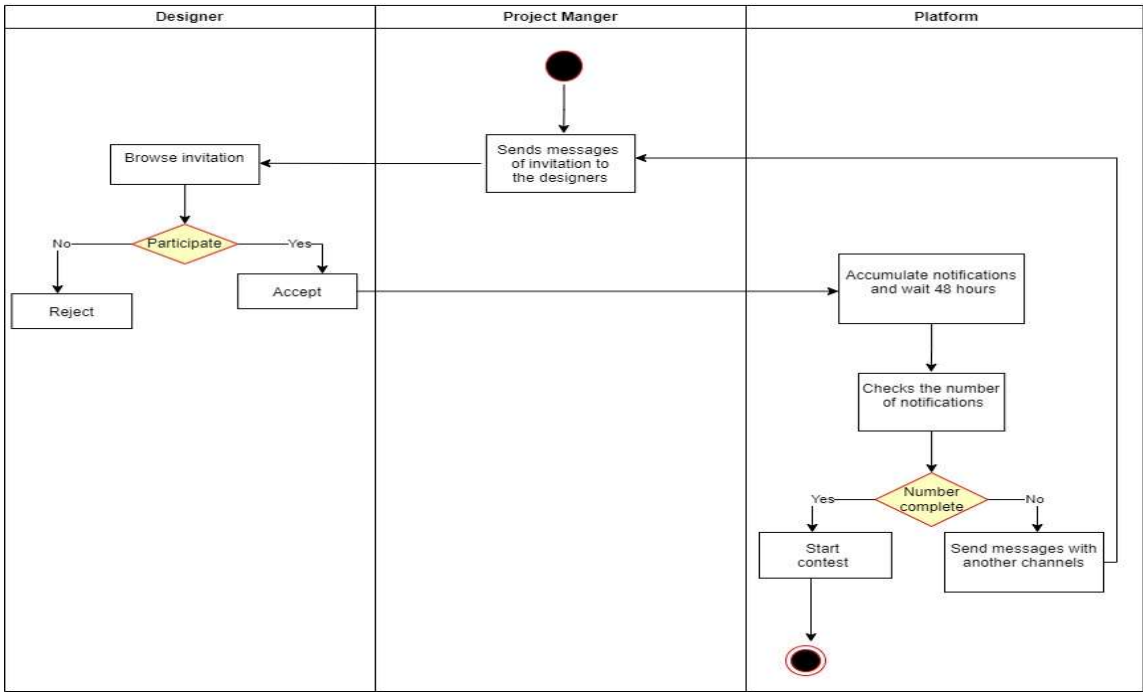

Figure 4. Process model 3 'Announce the contest'.

The project manager sends invitation messages to participate in the contest to designers who have been selected in accordance with the requirements and needs of the contest. The designers browse the invitation, then two decisions are made either to not accept the invitation to participate in the design contest and or accept it. The platform accumulates the acceptance responses and waits for a particular period of time (i.e., 48 hours), then the platform checks the number of acceptance responses again.

 doi:10.20944/preprints202410.1694.v1

if the number of designers is incomplete to participate in the contest, invitations are sent to the designers with another communication channels, either through social media or WhatsApp messages.

8. Stage 5: Evaluation of the Process Models

In this research, semi-structured one-to-one interviews with domain experts were conducted for the evaluation of the proposed process models to manage CSD activities. The evaluation using process models, provide more thorough discussion since these models visualize the CSD process, giving a better understanding about the activities and the coordination required.

In semi-structured interviews, the interviewer prepares a list of predetermined questions and participants have the opportunity to explore issues in much depth and from as many angles as they please, during answering the open-ended questions. In addition, the interviewer freely explores different fields and asks specific questions during the interview [96].

The sample of interviewees should share critical similarities related to the research question. In order to maximize the depth and richness of collected data, the selected interview participants are practitioners from the crowdsourcing design domain. Interviewing experts on the domain made the conversations more valuable as they shared their experiences and common challenges. The interviews had an average time of 55 minutes. Their views are reported here to evaluate the proposed CSD activities.

8.1. Interviewees Background

The interviews involve 7 people who are four designers (D1, D2, D3, and D4), two clients (C1, C2) and one project manager (PM1). Table 5 shows the characteristics and experience of the participants in the domain of CSD.

Table 5. Interviewee's characteristics and experiences.

	D1	D 2	D 3	D4	C1	C 2	PM1
Years of Experience	3 Years	2 Years	1 Year	3 Years	2 Years	1 Year	2 Years
Job Title	Web designer and developer	Web developer	Web designer	Web designer and developer	Software company manager	Application team leader	Project Manager
Use of Platforms	99designs, Upwork, Crowdsit, Freelance, Fiverr.	99designs, Freelancer, Upwork.	99designs, Freelance, Upwork.	Upwork, Freelancer, Fiverr.	99designs, Upwork, Freelancer.	99designs, Upwork, Freelancer, Fiverr.	99designs, Upwork, Freelancer.

8.2. Interview Findings

The participants discussed the difficulties they face with CSD projects and how they could be overcome through the proposed models. Most suggestions are agreed on by all participants. However, we have received some comments and these are detailed in Table 6. The process models not mentioned in the tables mean did not receive comments and they are agreed on as designed.

Table 6. Comments of the interviewees.

Process model	Proposed Suggestion	Comments of the interviewees	Remarks of authors
	S1: It is good that a platform provides a detailed form that includes all the requirements to ensure the proper	D1: Agreed with S1, but he gave recommendations for the use of artificial intelligence technology to initially classify	Using artificial intelligence technology for review is a good idea, but it could be expensive

 doi:10.20944/preprints202410.1694.v1

Process model	Proposed Suggestion	Comments of the interviewees	Remarks of authors
	functioning of the contest (e.g., client expectations, design preferred colors, design plans).	the contest according to specialization.	for platforms. However, it can be considered in future studies.
		D4: Agreed with S1, but he recommended that a blank box is placed in the form for any additions by the client.	The recommendation can be reflected in the process model
P1: Create contest brief		C2: Not agreed with S1, because he believed that providing a specific form would restrict the client from adding contest requirements.	Placing a blank box in the form for any additions by the client helps to add more details and information.
	S2: It is good to set constraints on the client when writing the contest brief to ensure they add all the information and details related to the contest in terms of the objectives and the expected final result from the contest and that they attach the files.	D1/C2: Not agreed with S2, because they believe that constraints placed on the client do not allow him to work freely in attaching files, and do not give complete freedom in detailing the requirements of the contest.	The recommendation can be reflected in the process model so that the constraints are not mandatory but optional.
	S3: It is good to automatically select the project manager based on the type of contest.	D1: Agreed with S3, but he recommended that the client could determine the need for having a project manager depending on the complexity and rating of the contest.	Having project manager is essential. The absence of a project manager for each contest leads to not obtaining the final results of the required quality.
		D1: Agreed with S3, but made recommendations that the client can accept or reject the project manager chosen by the platform, and the client decides to appoint the project manager himself.	The recommendation can be reflected in the process model so that the client can suggest a particular project manager, but his suggestion can be considered only if the chosen project manager has capacity.
	S4: It is good to set constraints on capacity related to the association with contests on the project manager.	C1/C2: Agreed with S4, but suggested that a method is put in place by which the project manager selected by the platform could be approved or rejected by the client.	The suggestion was outside the scope of the limitation definition but can be considered in future studies.
P2: Assign a project manager and Validate contest brief		PM1: Not agreed with S4, he state that there is no specific mechanism to distinguish between a full-time project manager and a part-time project manager, which in turn leads to a loss of opportunity for the full-time project manager to participate in the contest.	The recommendation can be reflected in the process model so that the capacity of the project managers with the contest is determined based on the mechanism of the project managers association with the contest, either full-time or part-time.
	S10: It is good to message designers through different communication channels, such as WhatsApp or text message.	D4: Agreed with S10, but recommended that the communication mechanism is specified in the contest page.	The recommendation can be reflected in the process model
		PM1: Not agreed with S10, because he believes using	The opinion was not agreed on, because the process of sending

 doi:10.20944/preprints202410.1694.v1

Process model	Proposed Suggestion	Comments of the interviewees	Remarks of authors
P6: Invite designers		multiple channels to communicate with designers increases the cost, in addition to more time and effort.	notifications via WhatsApp does not require effort, and does not require an increase in cost because the cost is marginal compared to the benefit.
	S11: It is good to consider the time difference when sending invitations to designers; the project manager should check the time before sending invitations to designers, and an extra day may be added to the invitation if there is a significant difference in time zones before the start of a contest.	D2: Agreed with S11, but recommended to send the invitation from 7 am to 7 pm, in order to consider the time zone differences.	In fact, adding extra day can solve the time zone differences.
P9: Validate submit designs, Browse submitted designs and Providing feedback	S15: It is good to set constraints and obligations on the client to oblige him to provide feedback to the designers in the first stages of the contest in addition to browsing all the designs submitted in the first stage.	C1/C2: Not agreed with S15, stating that there should be no obligations on the client to give feedback on all designs.	The recommendation can be reflected in the process model so that written feedback on all designs are not mandatory, but optional. However, the client at least gives a rating to the designers, while for the designers closest to winning.
P10: Providing feedback	S17: It is good to provide a mechanism during the feedback phase that enables the client to combine two designs that complement each other.	D2/D4: Not agreed with S17, Because of designer property rights and the possibility of misunderstanding between designers.	The idea of combining two designs serves the interest of the designers and the interest of the client so that instead of choosing one designer whose design could be incomplete, two designs are chosen that complement each other. This still preserves the property rights of each designer regarding his own design.

9. Conclusions

This research provides several mechanisms to support the the CSD process. Seventeen CSD improvements, related to twelve CSD activities, were identified.This complies with the emerging research in the CSD literature for more efficient solutions to overcome current limitations (e.g., [8,11,12,15]).

Although the CSD platforms have provided features to improve the CSD process (e.g., email notifications, chat, announcements), these features are insufficient to solve the coordination limitation. The proposed approach extends the scope of the existing coordination mechanisms by structuring the contest form, selecting project managers, selecting designers, avoiding inactive designers, improving notifications and reminders, , in addition to tracking the progress of the contest and giving feedback.

The authors believe that the results and suggestions that have been reached will add valuable information that contributes to the development of the field of design contests in CSD

Seventeen CSD improvements, related to twelve CSD activities, were identified along with the identification of the coordination support required for each CSD activity.

Appendix: Process Models

 doi:10.20944/preprints202410.1694.v1

P4: Select Designers

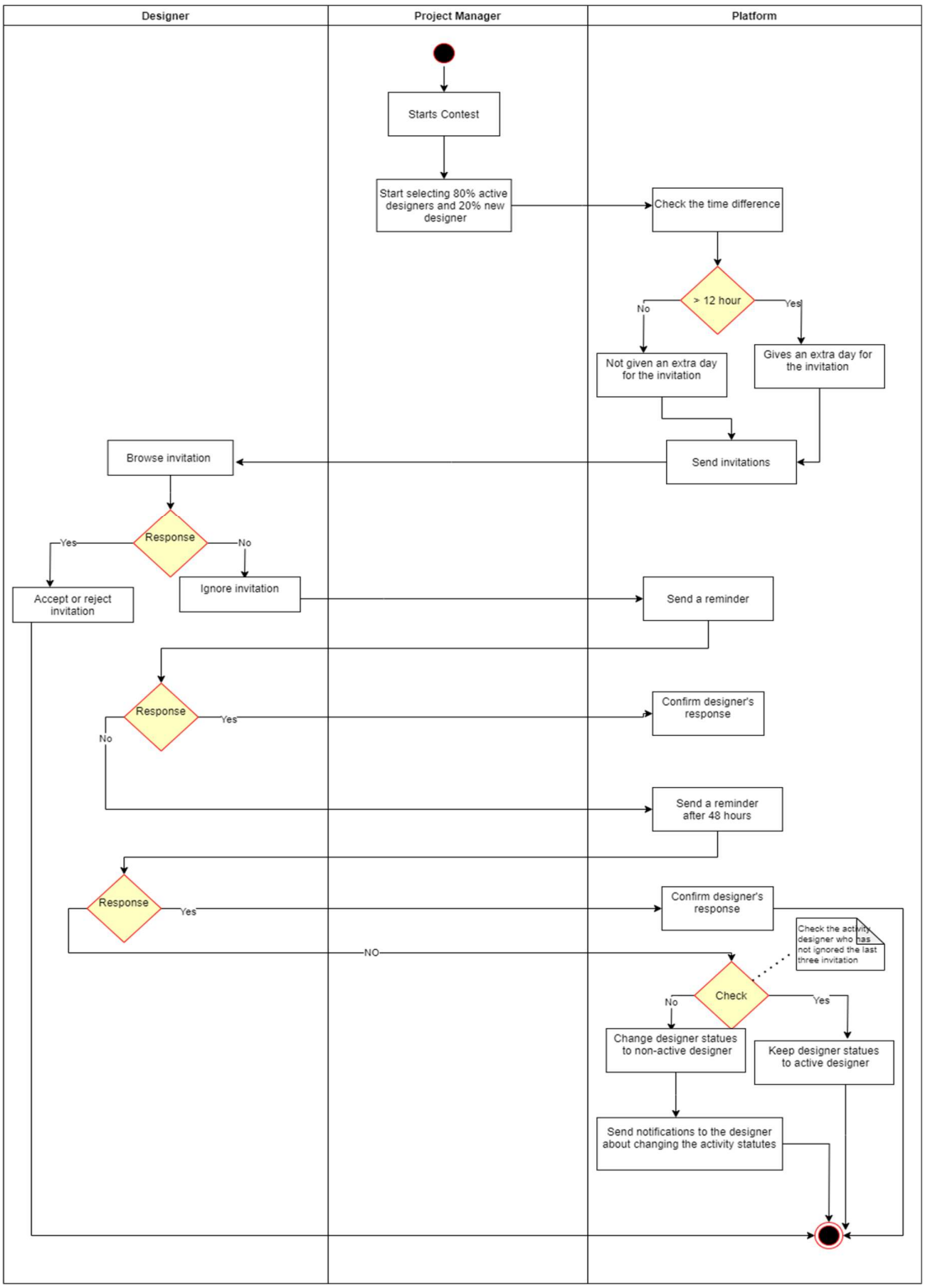

Figure A1. Process model 4 'Select designers'.

P5: Verify Designers

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 24 October 2024 doi:10.20944/preprints202410.1694.v1

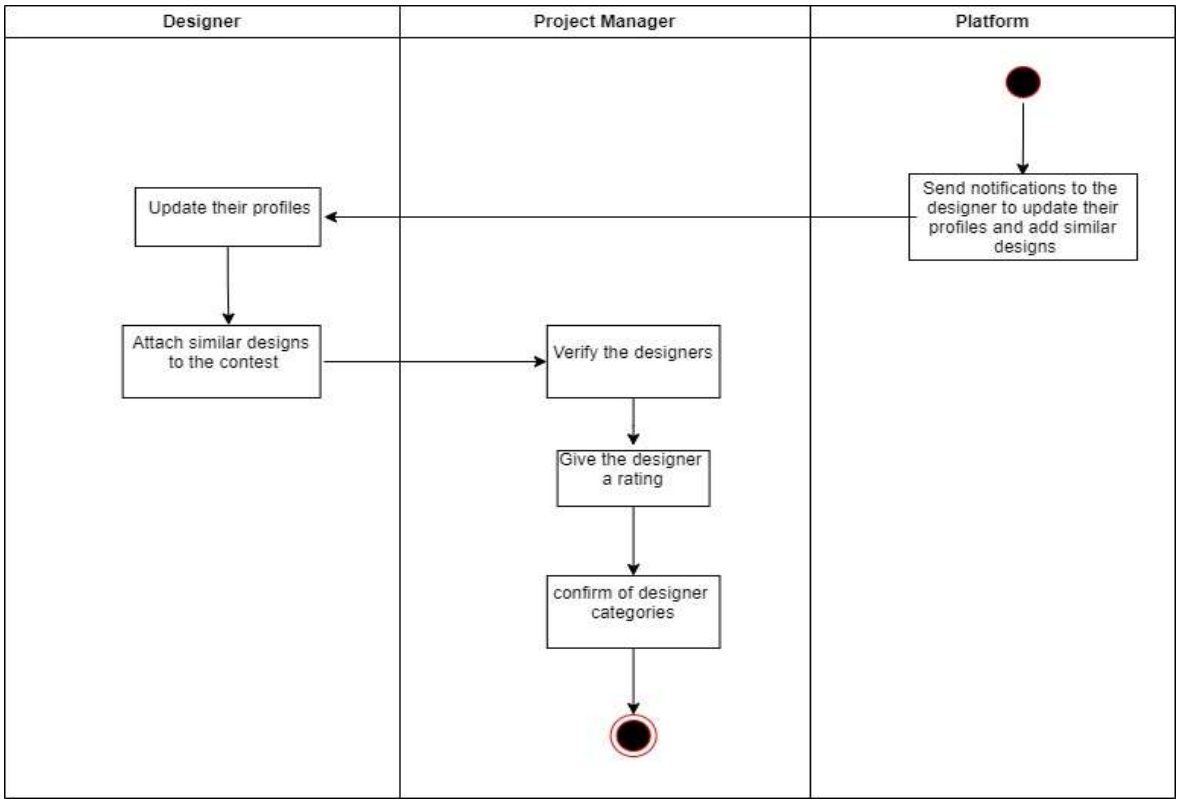

Figure A2. Process model 5 'Verify designers'.

P6: Invite Designers

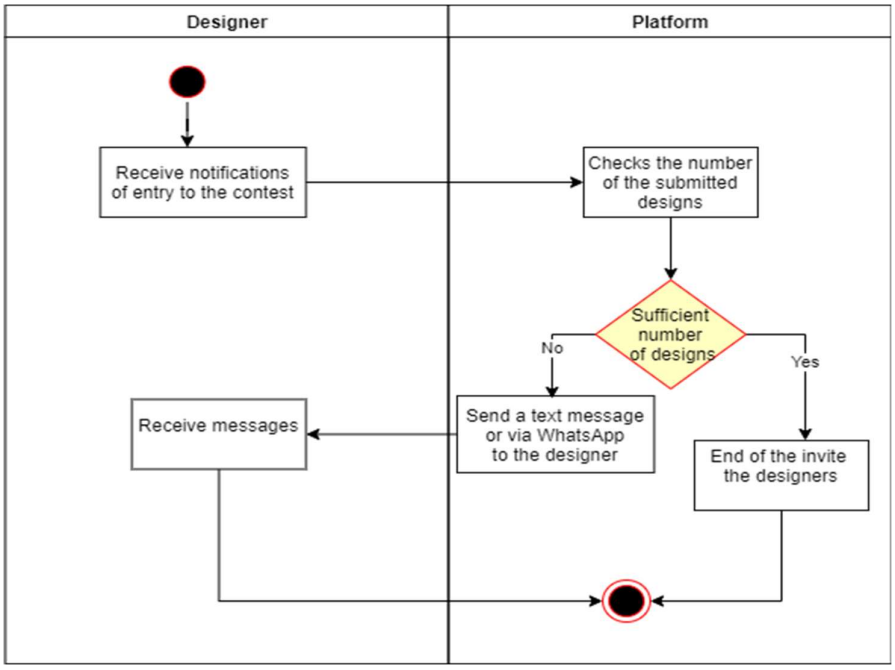

Figure A3. Process model 6 'Invite designers'.

P7: Approval Agreement

 doi:10.20944/preprints202410.1694.v1

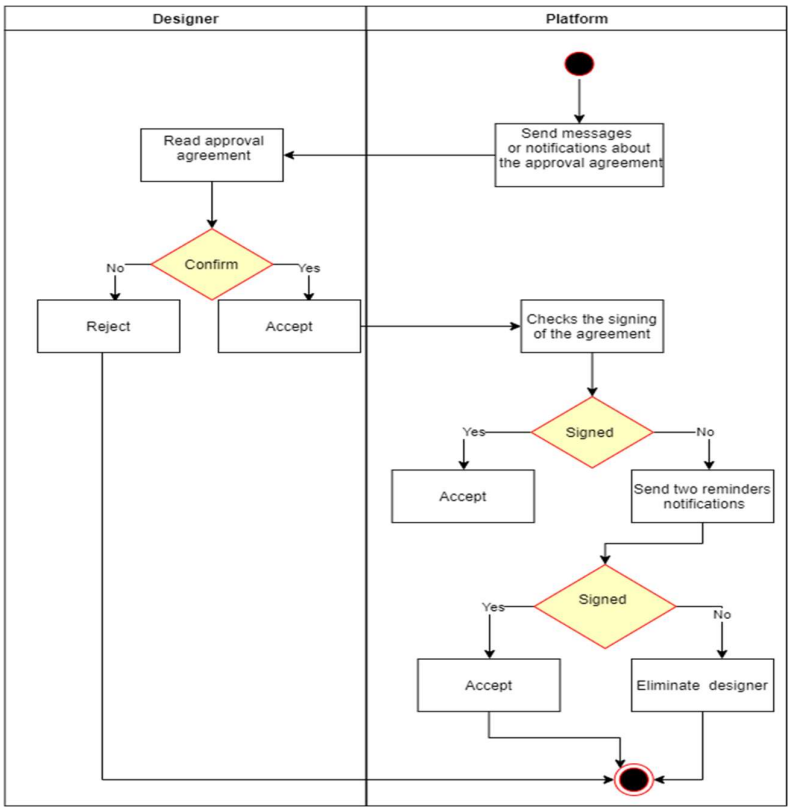

Figure A4. Process model 7 'Approval agreement'.

P8: Participate in the Contest

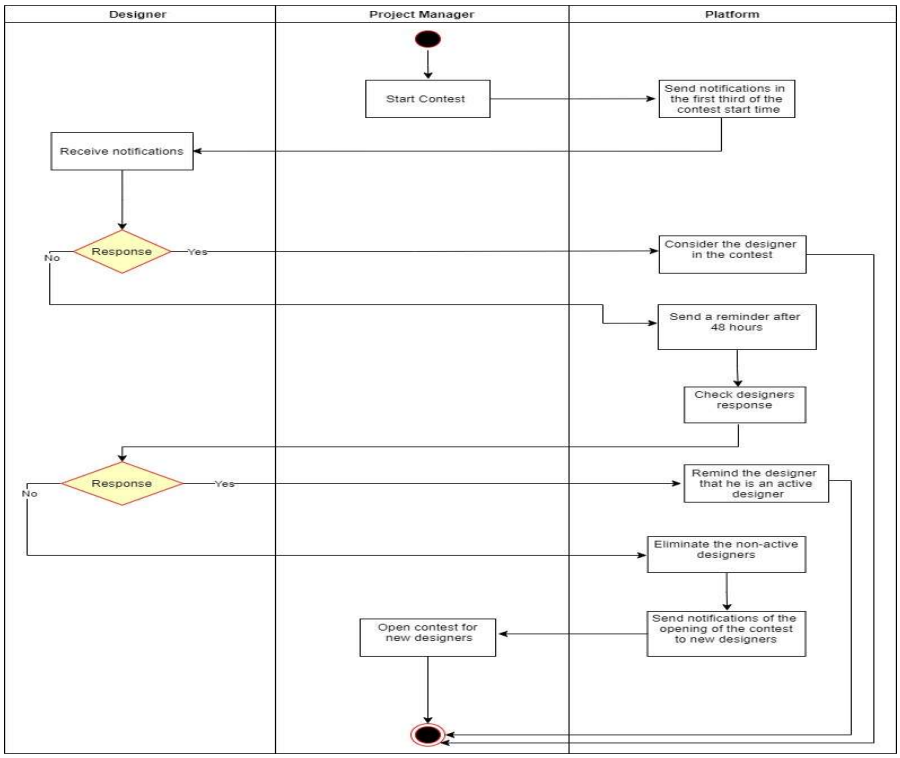

Figure A5. Process model 8 'Submit designs'.

 doi:10.20944/preprints202410.1694.v1

P9: Validate Designs and Provide Feedback by Client

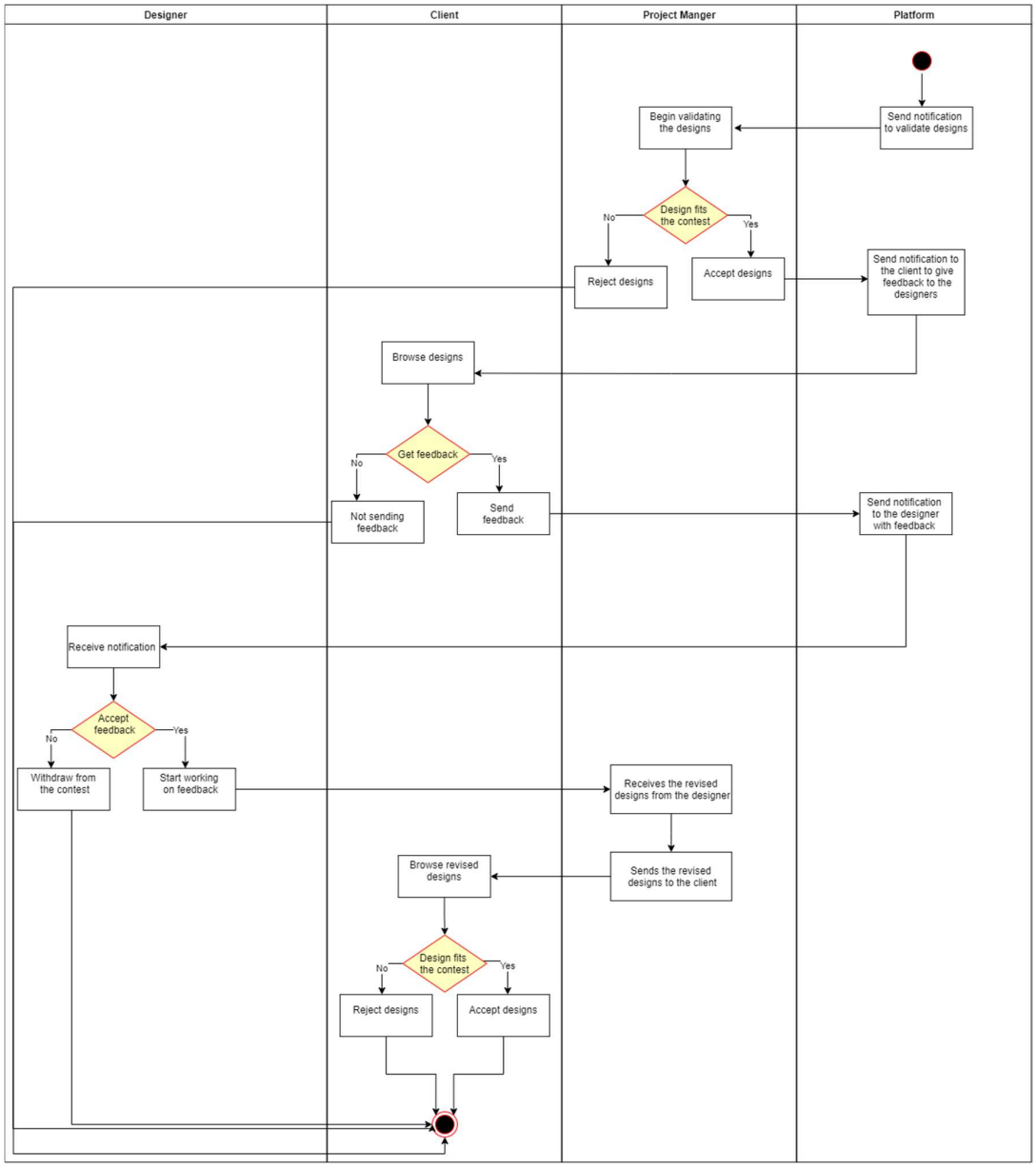

Figure A6. Process model 9 'Validate designs and provide feedback by client'.

P10: Combine Two Designs That Complement Each Other Through Feedback

 doi:10.20944/preprints202410.1694.v1

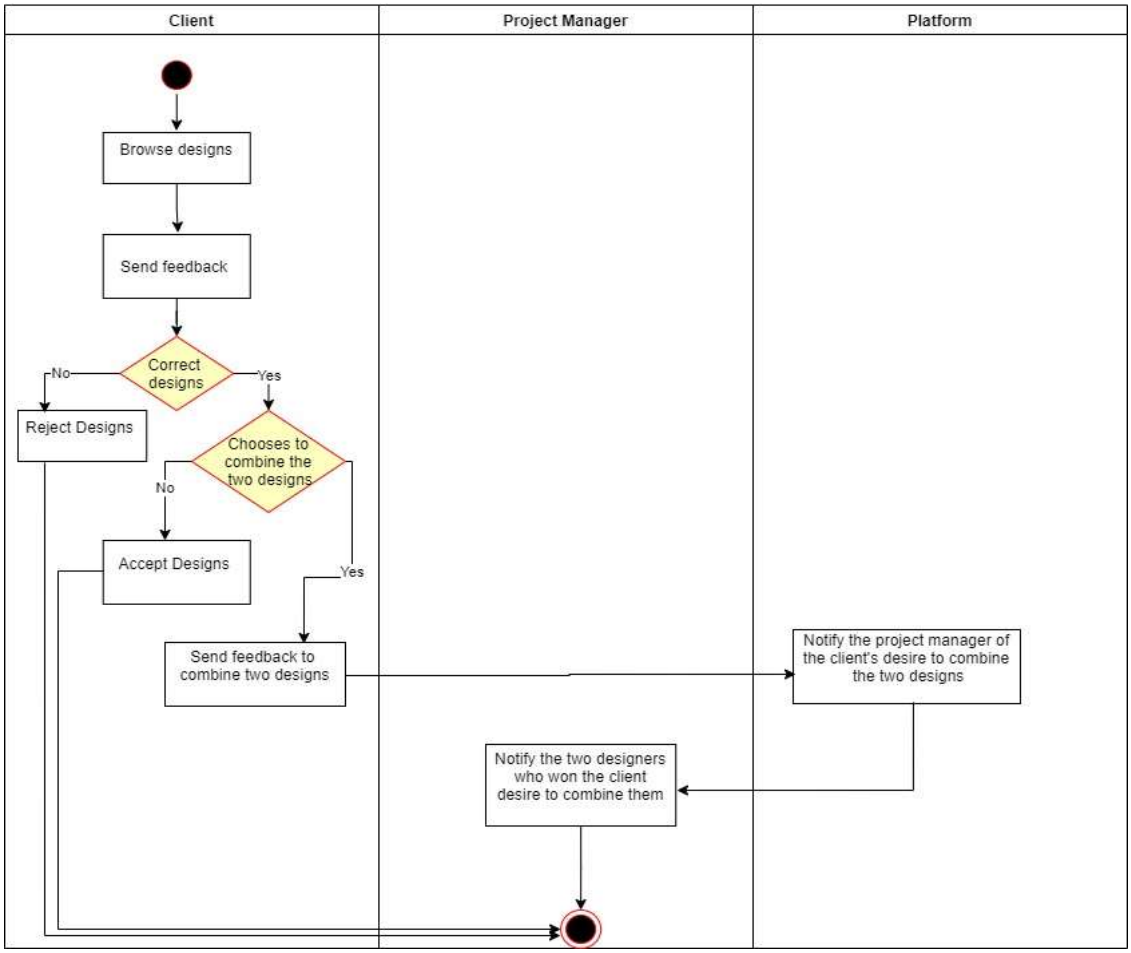

Figure A7. Process model 10 'Combine two designs that complement each other through feedback'.

References

[1] X. Peng, M. Ali Babar, and C. Ebert, "Collaborative software development platforms for crowdsourcing," *IEEE Softw.*, 2014, doi: 10.1109/MS.2014.31.

[2] K. Stol and B. Fitzgerald, "Research Protocol for a Case Study of Crowdsourcing Software Development," *Lero Tech. Rep. Lero*, no. 10, 2014.

[3] C. Amrit, J. van Hillegersberg, and K. Kumar, "Identifying coordination problems in software development: finding mismatches between software and project team structures," *arXiv Prepr. arXiv1201.4142*, 2012.

[4] I. Blohm, S. Zogaj, U. Bretschneider, and J. M. Leimeister, "How to manage crowdsourcing platforms effectively?," *Calif. Manage. Rev.*, vol. 60, no. 2, pp. 122–149, 2018, doi: 10.1177/0008125617738255.

[5] S. Coleman, S. Hurley, C. Koliba, and A. Zia, "Crowdsourced Delphis: Designing solutions to complex environmental problems with broad stakeholder participation," *Glob. Environ. Chang.*, vol. 45, pp. 111–123, 2017, doi: 10.1016/j.gloenvcha.2017.05.005.

[6] H. Wu, J. Corney, and M. Grant, "An evaluation methodology for crowdsourced design," *Adv. Eng. Informatics*, vol. 29, no. 4, pp. 775–786, 2015, doi: 10.1016/j.aei.2015.09.005.

[7] L. Xiao and H.-Y. Paik, "Supporting Complex Work in Crowdsourcing Platforms: A View from Service-Oriented Computing," *2014 23rd Australian Software Engineering Conference*. IEEE, pp. 11–14, 2014, doi: 10.1109/aswec.2014.11.

[8] A. SARI and G. I. ALPTEK\.IN, "An Overview of Crowdsourcing Concepts in Software Engineering," *Int. J. Comput.*, vol. 2, 2017.

[9] X. Niu, S. Qin, H. Zhang, M. Wang, and R. Wong, "Exploring product design quality control and assurance under both traditional and crowdsourcing-based design environments," *Adv. Mech. Eng.*, vol. 10, no. 12, 2018, doi: 10.1177/1687814018814395.

[10] E. Simperl, "How to use crowdsourcing effectively: Guidelines and examples," *Lib. Q.*, vol. 25, no. 1, 2015, doi: 10.18352/lq.9948.
[11] K. Crowston, E. Mitchell, and C. Østerlund, "Coordinating Advanced Crowd Work: Extending Citizen Science," *Citiz. Sci. Theory Pract.*, 2019, doi: 10.5334/cstp.166.
[12] K. Crowston and C. S. Osborn, "A coordination theory approach to process description and redesign," *Organ. Bus. Knowl. MIT Process Handb.*, 2003.
[13] O. Almughram and S. Alyahya, "Coordination support for integrating user centered design in distributed agile projects," 2017, doi: 10.1109/SERA.2017.7965732.
[14] K. J. Stol and B. Fitzgerald, "Researching crowdsourcing software development: Perspectives and concerns," 2014, doi: 10.1145/2593728.2593731.
[15] K. Mao, L. Capra, M. Harman, and Y. Jia, "A survey of the use of crowdsourcing in software engineering," *J. Syst. Softw.*, 2017, doi: 10.1016/j.jss.2016.09.015.
[16] S. T. D. Yuan and C. F. Hsieh, "An impactful crowdsourcing intermediary design - a case of a service imagery crowdsourcing system," *Inf. Syst. Front.*, 2018, doi: 10.1007/s10796-016-9700-8.
[17] S. Alyahya, "Collaborative Crowdsourced Software Testing," *Electron.*, vol. 11, no. 20, 2022, doi: 10.3390/electronics11203340.
[18] S. Alyahya, "Crowdsourced software testing: A systematic literature review," *Information and Software Technology*, vol. 127. 2020, doi: 10.1016/j.infsof.2020.106363.
[19] Y. Zhao and Q. Zhu, "Evaluation on crowdsourcing research: Current status and future direction," *Inf. Syst. Front.*, 2014, doi: 10.1007/s10796-012-9350-4.
[20] M. Hosseini, K. Phalp, J. Taylor, and R. Ali, "The four pillars of crowdsourcing: A reference model," 2014, doi: 10.1109/RCIS.2014.6861072.
[21] T. D. LaToza and A. Van Der Hoek, "Crowdsourcing in software engineering: Models, motivations, and challenges," *IEEE Softw.*, 2016, doi: 10.1109/MS.2016.12.
[22] D. Murray-Rust, O. Scekic, and D. Lin, "Worker-Centric Design for Software Crowdsourcing: Towards Cloud Careers," *Progress in IS*. Springer Berlin Heidelberg, pp. 39–50, 2015, doi: 10.1007/978-3-662-47011-4_3.
[23] S. Alyahya and G. Alamer, "Managing work dependencies in open source software platforms," 2019, doi: 10.23919/ELINFOCOM.2019.8706495.
[24] L. dos S. Machado, "Empirical studies about collaboration in competitive software crowdsourcing," The Pontifical Catholic University of Rio Grande, Brazil, 2018.
[25] C. Amrit, "Coordination in software development: The problem of task allocation," 2005, doi: 10.1145/1083106.1083107.
[26] R. Aliady and S. Alyahya, "Crowdsourced software design platforms: Critical assessment," *J. Comput. Sci.*, vol. 14, no. 4, 2018, doi: 10.3844/jcssp.2018.546.561.
[27] A. Sultan, W. K. Ivins, and W. A. Gray, "A holistic approach to developing a progress tracking system for distributed agile teams," 2012, doi: 10.1109/ICIS.2012.7.
[28] "99designs." https://99designs.com.
[29] "DesignCrowd." https://www.designcrowd.com/.
[30] "DesignHill." DesignHill.com.
[31] "CrowdSpring." CrowdSpring.com.
[32] "CrowdSite." CrowdSite.com.
[33] "Freelancer." .
[34] "Upwork." Upwork.com.
[35] "Designfier." .
[36] "DesignContest." DesignContest.com.
[37] "Guerra Creativa." https://www.guerra-creativa.com.
[38] "hatchwise." .
[39] "110designs." .
[40] "MOJO Marketplace." .
[41] "Fiverr." .