**Background & Overview**

**Software Crowdsourcing** as we know is a novel approach in outsourcing large and complex projects. It is done via cloud to outsource large projects to unknown communities of professional developers[1]. It has been possible because of larger and vibrant online communities[2] having specialized skills in several domains. A software crowdsourcing platform is a fundamental necessity on these grounds as this will lead to progress of a crowdsourcing project. This paper is a measure to study about the crowdsourcing applications, various platforms and specifically focus about the various challenges faced by newcomers into this industry and how they affect a project as a whole. We have shown this by taking a particular platform as a case study in detail and analysed the results in detail in the later sections. As we will see at the various challenges faced by newcomers in this industry it is very important for them to keep on engaging and retaining so as to form a sustainable community altogether[3][2].Throughout our study we have analysed that the main challenges the newcomers face are similar to that of an Open Source Software Development(OSS)[4].Some of the main challenges they face are technical documentation problems and cultural problems because of diversity.This paper is a study conducts a survey for identifying barriers to newcomers in the industry with their contribution to software crowdsourcing projects for paid work[5][2] represented in a competitive or progressive model[6]. The study in this project is basically based on two types of studies :

1. **TopCoder :** It is a well-known crowdsourcing platform. In this paper the authors are using data extracted from this platform for their analysis and results[3].
2. **Feedback :**  They have also studied about the feedbacks from several newcomers(developers) after their contribution to a software crowdsourcing project.[3]

Through this paper the authors, describe all the challenges through various studies and case studies revealing all the characteristics or features of the contributors to these platforms and have also suggested some guidelines or measures to overcome these challenges.

**Crowdsourcing : Overview**

It is nothing but a platform facilitating a participative online activity in which an individual, institution ,a non-profit organization or a company conduct a online survey electing a group individuals with varying knowledge ,heterogeneity and diversity to come together on a common platform to solve a complex project together or individually based on their specialized skillsets, which may be varying[7]. It has several advantages, some of them being reduction of costs, resource accessibility on a wider basis and also facilitates the rapid mobilization of a larger number of people with varied skillsets and abilities to accomplish a mutual task as the number of people are diverse in nature[3]. Simultaneous applications can be performed laterally with different individuals having other skillsets. This is very valuable for a software project because it not only increases the diversity within a project being a multicultural background[1][8] but also adds creativity and problem solving to project development which is actually applicable to progress of a project[6][8]. According to the authors, any crowdsourcing project has been divided into three categories or components :[8][1][3]

* **Requester :** It is the client or the sponsor of the project who sponsors or requests some work to be done by outsourcing it to various people(crowd) via a platform.
* **The Crowd :** It is the multi-diverse and vibrant online community which is being dispersed across the globe consisting of professional developers with different skillsets who work upon the specified project.
* **The Platform :** It is the technological framework or the common access ground where the requester and the Crowd (people) could interact together to accomplish a task. It is nothing but an intermediary between the crowd and the requester.[3]

The authors specifically Mahmood Hosseini mentioned a fourth component[9] in his paper which plays a significant role in a crowd-sourcing platform which is :

* **The Task :** It is nothing but a simple kind of process basically a human intelligence work, based on its complexity they can be subdivided into:
  + 1. **Microtask or Simple Tasks :** It is nothing but a simple task which doesn’t require much cognitive effort altogether. It is repetitive in nature and can be completed in a matter of time[6].Some of the prominent examples

include audio/video transcriptions and image classifications.

* + 1. **Complex Tasks :** Unlike simple tasks , these require specialized skillsets and cognitive efforts. These are usually specialized projects and ask too much of knowledge and expertise from the professionals. Some of the prominent examples of Complex tasks include Big Software development tasks & Innovation projects.

**Crowd-Sourcing : Platforms**

In the industry there are several applications that provide Crowd-Sourcing services. Some of the industry specialists such as Thomas LaToza and André van der Hoek have proposed three efficient software crowd-sourcing scenarios in their research work :[1]

* **Peer-Production model :** This model is based upon accomplishing a task for their own understanding or knowledge basis. It is more of a peer community model where a task is achieved collectively on networks and contacts. The peer group unlike other models do not expect any monetary rewards or gains in this model[1].
* **The Competition model :** This is the standard model of any crowd-sourcing platform. It consists of two components the Requester/Client and the Crowd as we already discussed. Here in this model the Client request the tasks to be done and subsequently pay for their subsequent completion. Monetary benefits are the primary purpose here.
* **The Micro-tasking :** As discussed earlier, this model is used only in simple and repetitive tasks which do not require much cognitive effort. This model is particularly implemented in Testing kind of activities which is repetitive in nature and does not require much cognitive skillsets[8].

Coming to some prominent or widely used crowd-sourcing platforms we have some applications such as Top-Coder, Crowd-Plat, Get-A-Coder. These are some of the examples that support multiple development phases, where developers with all skillsets (not targeting on a single skillset) can come and work on a particular problem. Some of the platforms facilitate target specific phases which focus on delivering or accomplishing tasks based on particular tasks[3]. Some of the examples include:

* **For Requirements analysis :** Crowd-Require
* **For Coding :** Bountify
* **For Design :** Design-Crowd
* **For Testing :** 99Tests
* **Other Supporting applications :** Freelancer websites such as Amazon Mechanical Turk, Stack-Overflow can support as crowd-sourcing platforms.[8][3]

**Top-Coder : Case-Study**

It is a widely used crowd-sourcing platform and the authors in this paper have used this for their analysis. The main goal of this platform is to bring together experts and professionals in design, data-science, open-source development and high-level programming under a common roof[3]. With this platform, they gain the ability to accomplish a mutual task in a more creative and holistic way. The main ideology behind choosing Top-Coder as an analytical study for this paper is its vividness and robust architecture. It a largely used platform and has a very vibrant community as being required by any crowd-sourcing platform[10][11]. It has been used for creating other very famous online platforms and applications. Some of them include AOL, Best-Buy and Facebook[11][10][3].

Top-Coder or any such crowd-sourcing platform is very complex and tedious and involves too many concepts and processes simultaneously. Some of the very important phases include requirements exploration (which is application specific), architecture of the component or modelling of a particular subset of a program , development of a component and finally deployment of the program and testing. The processes are more of similar to that of the Software Development Life Cycle(SDLC)[8][3][5].

*Functioning or Model of Top-Coder*

* In Top-Coder platform different members interact together towards accomplishing a mutual problem. Each problem can be sub-divided into various number of tasks or can be posted as a single problem.
* All of these problems are posted as individual contests. The working members(professionals) can post their respective solutions to any of these contests.
* All of the submissions or the inputs from these experts are analysed thoroughly and based upon the maximum efficiency of the solution the winners are then decided and awarded monetary rewards.
* A co-pilot ensures that the contestants are in well place and the competitions are being undertaken at the right time. Co-pilot takes care of all the expertise requirements and maintains a synchronization between crowd and the client.
* As there are different phases to each problem, the solution to one phase becomes the key solution for unlocking the next problem until the one big problem is being completely analysed and solved.[10][11]

As there are many phases discussed above, the authors have thrown light on some of the most important phases. Some of which include:

* **The Requirements Exploration Phase:** In this phase, all the requirements are gathered, reviewed and specified accordingly based upon their level of priority. This acts as a map to other phases.
* **The Component Architecture Phase :** In this phase each of the component in the platform pass through their development and design considerations. Testing and other related activities are leveraged or performed at this phase so as to identify any final fixes to the submitted solution[3].
* **The Collaboration Phase :** In this final stage, the product is prepared completely by combining all these components together and then is delivered to the customer as a fully functional product[3][8].

Once all these phases are carried out, the final scores are analysed based upon the quality scores they secure at each phases. The first-place participant gets the highest monetary reward so on and so-forth. For each new challenge in the platform their ratings are provisional(based upon the entrance ratings and other phase related scores). The rating is centralized by Top-Coder itself as it provides ratings to each of the member/competitor based on his/her efficiency with respect to the number of competitors[2][3].

**The Quantitative Study**

In the paper, the authors explain about the quantitative study they have conducted to understand the barriers newcomers face in the Software Crowd sourcing project. This was however a preliminary examination of the problem in hand to get a better picture and to help them in the next stage of study. Information was gathered from the platform regarding its members during the period 23 May to 12 June 2016. Then they applied three constraints to select few members among those selected. Firstly, algorithmic competition was chosen as a criterion to make the selection of participants. [1] In Algorithmic competition of Topcoder, members where given the same problem and they were asked to solve them in the same time limit. Algorithmic competitions are the main criteria used to keep members. The participants had to participate in algorithmic competitions recently. The second Criteria was that the authors tried to include Newcomers in the Quantitative study. Members rating was used as a criterion to choose members in this way. The members having rating 1,199 or below were selected for the study. Thirdly, members must have participated in a competition within the past 6 months to be selected for the study.

Filtering out members based on the above-mentioned criteria’s, authors ended up with 3,403 members. The details of these members were also collected like nickname, entry month and year, number of wins and country using a crawler and members who were missing any of them were rejected. In the end the authors ended up with a total of 3100 members and their data. 342 among these were chosen for the study to get a margin of error of ±5 percent and confidence level of 95 percent. Among the information collected in this way, members where mostly from India, China, US, Korea, Japan Bangladesh, Mongolia. 205 among them were from Asian countries, 69 from North America. The Most noticeable thing was that Ninety-Seven percentage of the members selected in this was had never had a success in competitions and the three percent who had chance to win the competitions were having at least nine years of experience. Every member in Topcoder was placed in a tier with color codes showing their success.

Grey 0 to 899

Green - 900 to 1,199

Blue - 1,200 to 1,499

Yellow - 1,500 to 2,199

Red - 2,200 or Higher

So, the members selected in this way was either in First two tiers. A member from Bosnia and Herzegovina had won competitions 134 times and two Indians won 66 times each. Among those members, 113 had never won even once although they were active in Topcoder.

**The Qualitative Study**

For conducting the Qualitative study, the authors invited 20 people half of them university students and others Software Professionals having minimum two years’ experience in the industry. Three Students and five Software professionals were interested. They were asked to understand the basic operating mechanism of Topcoder platform. They were asked to register in the platform and to participate in any development competition which they felt they can complete. Once they completed the task within the allotted time of 12 days, they were asked how they felt during the period. The participants were then asked question ““On the basis of this experience, what main barriers do newcomers face when they want to start contributing to Topcoder? (Please consider technical and nontechnical issues.)” [2]. This was based on research by Igor Steinmacher et al. in the similar field. Open-coding procedures [3] were used to collect and analyze the data.

Based on the response the authors were able to understand the following problems which new comers faced which a listed in descending order of frequency. Firstly, the documentation provider was not good enough for newcomers to comprehend the problem statement. Without proper diagram the requirement appeared ambiguous to them. Even though the Github references were provided, it did not help them much. Secondly, the organization of tasks within Topcoder was not proper so that they were not able to understand which among them they can finish within time and will give them better change of success. Authors inferred that these difficulties occurred because the participants are usually not managing the time properly and usually though only about personal time management.

The third significant problem the newcomers faced was that the code structure was not good so that they can understand it easily. In projects like these, the source code is quite big, so they can only skim through it and cannot spend much time trying to understand it. As the code is lacking good , meaningful names for variable and proper comments, understanding the functionality of different modules was quite challenging. Fourth significant challenge they faced was that they were provided with lot of information which at times they could not comprehend. Authors refer to this as information overload and observes that when a lot of information is provided to Newcomers, rather than being helpful, it will only increase their confusion so that they are not able to make good decisions. As they are relatively new to the platform they may feel stressed out and will gave up in the end.

From the feedback they got the authors observed that the platform was not easy to use. The ISO 9241-11 standard defines usability as “The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use” [4]. Since the completing the competitions are quite demanding, the poor effectiveness of the platform was quite demotivating for the users. Adding to that, the platform was not having a common place where user can discuss their problems. This amplified the problem of usability. The next barrier they had mentioned was the problem of language. As most of the users are from Non-English-speaking countries, using a platform like Topcoder which supports only English is quite difficult for user. Without any option of automatic translation to English, users who could not read English very well faced difficulties.

The authors noted that most of these problems listed are traditional problems which are faced during every software development project. They most severe in this is the problem of documentation as well as understanding the big source code by the newcomers. In Topcoder these problems get increased as the participants won’t get chance to read the whole code and may not understand the real implication and effects of their codes. The participants will not get opportunity to interact with client directly as it is channeled through Topcoder.

**Overcoming Barriers-**

Based on the Quantitative and Qualitative studies, the authors analyzed the challenges newcomers face in crowd sourcing projects and came up with the following measures to overcome them. The first solution the authors propose is to solve the documentation issue. The documentation should be proper, and the platform can use UML diagrams and videos to explain the problem statement. Providing too much information’s are not necessary as it will waste time and effort. When tasks are missing proper documentation, support should be provided so that the participants are able to ask their queries thorough calls, chats etc. There can also be a discussion forum where they can post their queries and get feedback.

Secondly, the platform was lacking a mechanism where newcomers can identify the tasks which are less complex and best suited for them. It should also be easy to estimate the time required to complete a task for the user. This can be done like marking the tasks which are best suited for the newcomers. Thirdly authors mention task information as the important aspect in any crowd sourcing project which enable the newcomers to decide which task will best suit them. LaToza and van der Hoek said that “to learn in today’s crowdsourcing systems, software developers must first join, and acculturate themselves to, software projects” [5]. So, proper information should be provided avoiding unnecessary details which will motivate newcomes. Authors also suggested that providing within platform resources which are specific for newcomers will be good. In this way they will get also the details they need while avoiding information overload.

The next significant problem newcomers faced was understanding code structure. To overcome this, authors suggested using methods so that participants can easily understand the source code. This can be done by including UML diagrams for different parts of the code. The problem here is the size of the program and the copyrights. As mentioned before, the platform is used by people who are from different ethnicity and background and most of them does not have English as their first language. This language barrier is a big demotivator for newcomers and can be overcome by implementing computer assisted translations. So, if automatic translation of information provided in Topcoder to languages of the majority of Topcoders participants are provided they can easily apprehend task at hand.

**Limitations**

The authors studies only about one Topcoder but there are multiple crowdsoucing platforms available in the market with different features. For instance, features of Amazon Mechanical Turk is quite different from that of Topcoder which is even different from CrowdPlat. The barriers we see in one platform can not necessarily appear on another platform. So, a comprehensive study will involve analyzing multiple platform and segregating the barriers newcomers face in all of them. Within the Topcoder platform the authors had analyzed, they studied about only one competition. However, in Topcoder there are multiple competitions and it is good if we study about all them to get a better understanding of problems newcomers face.

Adding to that, the participants who were selected for the qualitative study was chosen by the authors and selected invites. Only those who were willing were then interviewed and involved in the competition. This may lead to a bias where participants selected in the group represents a small group among total participants of Topcoder competitions and so other groups may be underrepresented. So, it will be better if this bias is avoided in the study.

**Future Works**

Authors mentions to use other methodologies like using Topcoders log analysis in future works for covering different areas of the study. They also mention to study other platforms and different competitions to get a holistic picture of the issue at hand. The also plans to make a study with larger number of participants so that data collected is close to real world.

**Conclusion-**

Newcomers are most important part of all crowdsourcing project as they help these platforms to grow and evolve. They can give a new and different approach to competitions. But they should be supported in the beginning so that they are not demotivated in the beginning. The authors observe that even though the solutions suggested in this study are not tried in real platforms, it will help to reduce the problems newcomers face in these projects. Crowdsourcing platforms should try to implement these methods suggested in the paper so that Newcomers can contribute better to software projects.

**References & Bibliography**

[1] T. D. Latoza and A. Van Der Hoek, “Crowdsourcing in Software Engineering : Models , Opportunities , and Challenges,” *IEEE Softw.*, pp. 1–13, 2016.

[2] I. Steinmacher, T. Conte, M. A. Gerosa, and D. Redmiles, “Social Barriers Faced by Newcomers Placing Their First Contribution in Open Source Software Projects,” *Proc. 18th ACM Conf. Comput. Support. Coop. Work Soc. Comput. - CSCW ’15*, pp. 1379–1392, 2015.

[3] A. L. Zanatta, I. Steinmacher, L. S. MacHado, C. R. B. De Souza, and R. Prikladnicki, “Barriers Faced by Newcomers to Software-Crowdsourcing Projects,” *IEEE Softw.*, vol. 34, no. 2, pp. 37–43, 2017.

[4] M. Osterloh and S. Rota, “Open source software development-Just another case of collective invention?,” *Res. Policy*, vol. 36, no. 2, pp. 157–171, 2007.

[5] W. T. Arizona, M. N. Huhns, and S. Carolina, “Cloud-Based Software,” 2014.

[6] K.-J. Stol and B. Fitzgerald, “Two’s company, three’s a crowd: a case study of crowdsourcing software development,” *Proc. 36th Int. Conf. Softw. Eng. - ICSE 2014*, pp. 187–198, 2014.

[7] E. Estellés-Arolas and F. González-Ladrón-De-Guevara, “Towards an integrated crowdsourcing definition,” *J. Inf. Sci.*, vol. 38, no. 2, pp. 189–200, 2012.

[8] A. Kittur *et al.*, “The future of crowd work,” *Proc. 2013 Conf. Comput. Support. Coop. Work - CSCW ’13*, p. 1301, 2013.

[9] M. Hosseini, K. Phalp, J. Taylor, and R. Ali, “Model,” 2014.

[10] K. Li, J. Xiao, Y. Wang, and Q. Wang, “Analysis of the Key Factors for Software Quality in Crowdsourcing Development: An Empirical Study on TopCoder.com,” *2013 IEEE 37th Annu. Comput. Softw. Appl. Conf.*, pp. 812–817, 2013.

[11] J. Hu, “pCoder r ( A ): Develop ping Software through Crowdsourcing,” pp. 1–19, 2010.