Improving Software Reliability Using Crowd Sourced Testing

[1]Y. Vamsidhar, [2]Dr. Y.Srinivas, [3]S. N. V. Satyanarayana P.
[1,3]Dept. of CSE, Swarnandhra College of Engg.& Tech, Narsapur, AP, India
[2]Dept. of IT, GIT, GITAM University, Visakhapatnam, AP, India

Abstract
In the present world, applications such as mobiles, gaming consoles and social networking apps are becoming more popular and those products must be delivered with promising quality. Improving Reliability yields quality software out. In this paper, we focus on an emerging software testing strategy called "Crowd Sourced testing" and its contribution to enhance the software reliability of the products that have global user base. We also present the results of our study.

Keywords
Software Quality, Software Reliability, Crowd Sourced Testing.

I. Introduction

The quality of a software product is the most important aspect that the customer will expect implicitly. The quality of the software products can be profoundly affected by various factors like correctness, efficiency, Usability, integrity, reusability, flexibility, interoperability, and reliability [1].
In this paper, we first presented introductory lines about Software Reliability, Coutinho Reliability growth model and Crowd sourced testing in section I. The section II presents some preliminaries related to the reasons of our work. In section III, we furnished our related work and results. Finally we present our conclusions in section IV and references in section V.

A. Software Reliability

Since reliability is the most significant software quality attribute, software reliability depends highly on software quality. According to IEEE, the Software Reliability is defined as "the ability of a system or component to perform its required functions under stated conditions for a specific period of time" [2].
High reliability ensures that the software product is failure free for a specified period of time.

B. Coutinho Reliability Growth Model

Coutinho adapted the Duane growth model to represent the software testing process [4]. Coutinho plotted the cumulative number of deficiencies discovered and the number of correction actions made vs the cumulative testing weeks on log-log paper. Let N(t) denote the cumulative number of failures and let t be the total testing time. The failure rate, $\lambda(t)$, model can be expressed as
$\lambda(t)=N(t)/t$

C. Crowd Sourced Testing

Crowd Sourced Testing is a promising trend in software engineering industry for software quality assurance. This newer testing method examines a product or application under assorted platforms for improved reliability and enhanced end-user representation. The crowd sourced testing team consists of a diverse community of testers for observing the broad set of usage patterns in order to perform precise scrutiny or inspection on the software. The more diverse the crowd sourced testing team; the feedback that comes in is invaluable.
Mostly the products or applications that are to be widely used in diverse user communities are likely to undergo crowd sourced testing. Such applications include mobile applications, social networking applications, gaming applications etc., But crowd sourced testing is not a hassle free solution. It has its own challenges. Also there are some occasions when it is not suitable to adopt crowd sourced testing.

II. Preliminaries

A. Reasons for Using Crowd Sourced Testing

Testing inside the lab is valuable but no longer sufficient. Modern apps require more test coverage and new mindset. As we discussed earlier, crowd sourced testing is useful for the software products that have global user base or that can be used across different environments. According to our experiences/literature survey, crowd sourced testing is most valuable in the following situations [3]:

- When the product has a global user base.
- When the product needs specific domain knowledge that is difficult to get internally from our test team.
- When the product requires user environments that are difficult and complex to set up in the internal test lab.

B. What to Test Using Crowd Sourced Testing

There are some specialized areas which are suitable to have the crowds to test and provide feedback on[3]:

1. Compatibility where the product will be used in a dissimilar set of platforms, environments and configurations.
2. Localization, where we need to accommodate context and culture based verifications.
3. Performance, where for instance we simulate realistic end-user loads using diverse connectivity band width.

Similarly, there are some typical areas that do not make sense to use crowd sourced testing:

- When the products have more environment dependencies, they cannot be simulated outside.
- If the products are very IP sensitive then they cannot be tested on different systems.
- If the product needs to be safe guarded or protected against any leaks then it is not suitable for crowd sourced testing.
- Also when the product has a limited user base, there is no need of performing crowd sourced testing.

It is very important to note that crowd sourced testing works great as a supplemental test technique rather than as a stand-alone test technique.

III. Related Work

We consider the failure dataset of Online IBM data entry software package [4]:

ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)

Table 1:

Testing Time (day)	Failures	Cumulative failures
1	2	2
2	1	3
3	1	4
4	1	5
5	2	7
6	2	9
7	2	11
8	1	12
9	7	19
10	3	21
11	1	22
12	2	24
13	2	26
14	4	30
15	1	31
16	6	37
17	1	38
18	3	41
19	1	42
20	3	45
21	1	46

The following is the graph showing cumulative failures of the above failure dataset:

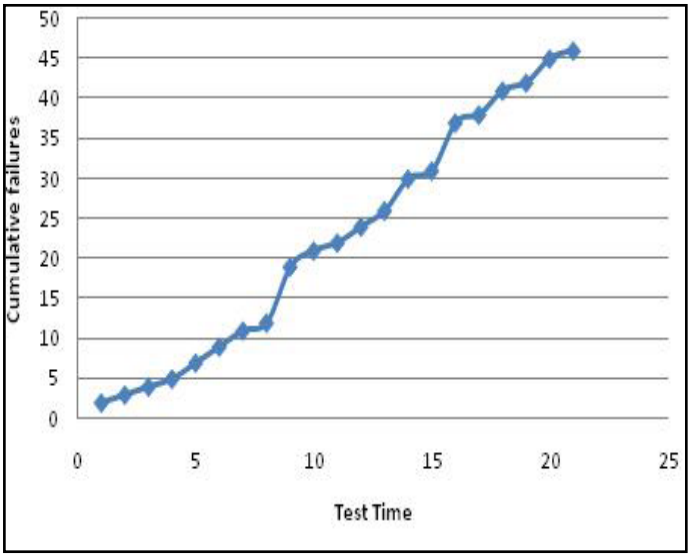

Fig. 1:

The graph shown above indicates that the cumulative failures are increasing exponentially along with the testing time.
By using the Coutinho Reliability Growth model, failure rate of the above dataset is:
$\lambda(t)=N(t)/t$
$= 46/21$
$= 2.190476$
After the product has under gone the crowd sourced testing, the failure dataset and its graph are shown below:

Table 1:

Testing Time (day)	Failures	Cumulative failures
1	2	2
2	3	5
3	1	6
4	2	8
5	2	10
6	3	13
7	4	17
8	1	18
9	7	25
10	3	28
11	0	28
12	0	28
13	0	28
14	1	29
15	1	30
16	1	31
17	2	33
18	0	33
19	0	33
20	0	33
21	0	33

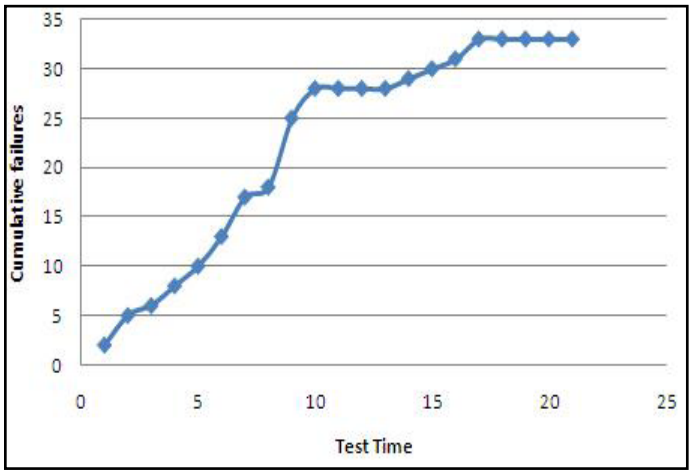

Fig. 2:

From the graph, it is clear that the failures have been reduced and at after some point (day-16) it gets stabilized.
The comparison graph for the two datasets is shown below:

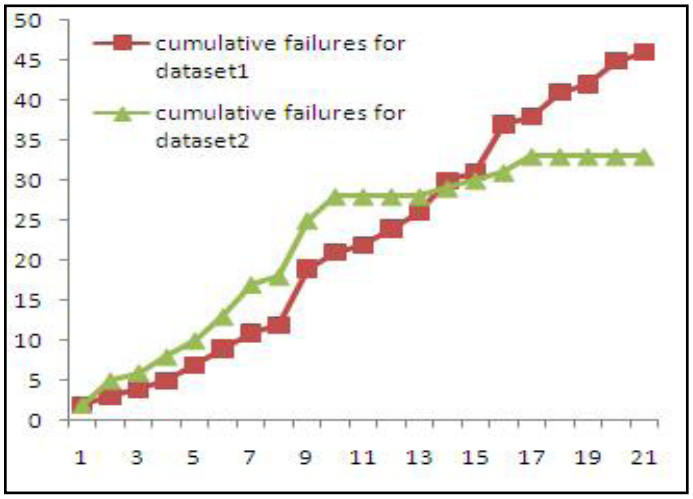

Fig. 3:

The failure rate of the system after crowd sourced testing is:

$\lambda(t)=N(t)/t$

$= 33/21$

$= 1.5714$

From the above value, we got significant reduction on failure rate after crowd sourced testing. Hence this reduction in the failure rate clearly indicates the improvement in software reliability.

IV. Conclusion

From the above results, the failure rate after crowd sourced testing of the product has been substantially reduced. Thereby we can know that reliability increased significantly. We are also concluding that our test labs are not sufficient to have more test coverage for the products that have diverse user community. So we adopted the crowd sourced testing technique and proved that crowd sourced testing will substantially improve the reliability of the product. We also presented the scenarios where the crowd sourced testing may not be used.

References

[1] Dr. Linda Rosenberg, Ted Hammer and Jack Shaw," Software Metrics and Reliability".

[2] "Glossary of Software Engineering Terminology". IEEE Standard 610.12-1990.

[3] "A Technical paper on Best Practices in Crowd Sourced Testing", Rajini P, Star East 2012.

[4] System Software Reliability, Hoang Pham, Springer Series in Reliability Engineering, pp. 137 & 171.