International Journal of Enhanced Research in Management & Computer Applications
ISSN: 2319-7471, Vol. 10 Issue 6, June, 2021, Impact Factor: 7.751

Simulation Result: Engage Software Tester Early In Software Development Life Cycle a Duration Based Models

Dr. Maneesh V. Deshpande

Asst. Professor, Computer Science Department, S.S.Maniar College, Nagpur, Maharashtra, India

ABSTRACT

Many Research have been Proposed on Software Engineering and Software testing but a lesser amount of research on engaging software tester early in software development life cycle. Also very few provide the guideline to actually implement engaging software tester early. This paper provides the information and stated newly created models based on the durationregarding software development life cycle for the software companies. Duration is important and this paper includes two durations mainly short and long, comparison based on traditional SDLC. Result of the simulation depends on traditional SDLC and newly created models. Major consideration of comparison (Result based on)

- **Time required completing all the phases of software development.**
- **Number of bug count**
- **Average bug finding ratio.**

Key terms: Software Development Life Cycle (SDLC), Software Testing (ST), Orange Human Resource management (HRM) Software, My Info module, Comma separated value files (CVS), bug, bug counting ratio.

1. INTRODUCTION

Development Process is essential in every change related process. Due to the same human being can able to portray the final product. If the development process is proper and implement in correct form then quickly can reach up to final stage without much more difficulty, but if not implemented properly or ignore some steps then can't fulfill the users requirements. The same case happened with software development and if software product should be proper incase of error free and user friendliness then software tester must act like an actor in film. In SDLC software Developer and software Tester are the basic building blocks. But in most companies these two are work independently. Because of their independent work many problems arises. But if they work in joint venture then this software development is called as Joint Application.[1]. A software development process, also known as a software development life cycle (SDLC), is a structure imposed on the development of a software product. It is often considered as a subset of system development life cycle. Software Testing is essential and very much crucial, not only for customers but for software companies also. Because of the awareness most software companies are using software testing. But some companies are still not implementing testing in proper stage. This paper tried to prove the same thing and mentioning some suggestions for engage software testers early in software development life cycle. Paper organized as follows our work in the context of prior work (Literature review), finding based on simulation result considering newly created models, and finally involves conclusion acknowledgement and references.

2. LITERATURE REVIEW

Mark L. Gillenson, PhD, CCP, October 16, 2006 [1]. The University of Memphis Research Proposal."Engaging Testers Earlier in the SDLC", have the following views: We believe that software development is fertile ground for the use of cross-functional teams, with testers being integral members of those teams at all stages of development. An additional contribution will be testers seeing themselves as stakeholders in the quality of the finished applications by virtue of their work throughout the SDLC. This will lead to the further development of systems testing as a recognized and respected specialty within information systems organizations. Finding and fixing these problems early (i.e. at the requirements or

design phase) will reduce the overall risk and cost of the product [2], this paper focuses on the software inspection approach. Next Paper based on the measuring the software quality during life cycle software development, with the help of improving the ISO 9126 [3]. Author describes the importance of testing throughout software development [4] and further believes that software testability analysis can play a crucial role in quantifying the likelihood that faults are not hiding after testing does not result in any failures for the current version. How software testing changes its nature in terms of working from organization to organization is stated in this paper. It is advisable to carry out the testing process from the initial stages, with regard to the Software Development Life Cycle or SDLC to avoid any complications [5]. Testing continues to represent the single largest cost associated with the development of sophisticated, software intensive, military systems. If the concept of testing begins very early in the development process significant savings can be achieved [6]. The focus of this paper is to present the first phase of development of decision models that could be used to determine the best uses of software testing resources. The ultimate model could be used to reduce overall costs while applying resources where they provide the greatest value throughout the systems development process [7].

3. FINDINGS

Following are newly created models, for "Engage Software Tester Early in Software development Life Cycle." Considering Mark Gillenson (2007) models, newly created models are conceptually same but the difference is that to go more in depth of the concept. In terms of mentioning Tester A and Tester B [1], categorized the testers on the basis of the experience i.e. senior tester and junior tester. All these models are implemented on each phases of traditional software development life cycle. Based on the work assign to individuals the concern models took birth as described below.

Model 1: Tester Centric Software Development Model for Short Duration
Following figure shows software development model for short duration considering tester first. This is a small (Duration) model. There are only three terms mentioned, so quickly we can reach to final product. So many projects which have to be completed in few months, for those this model might be suitable.

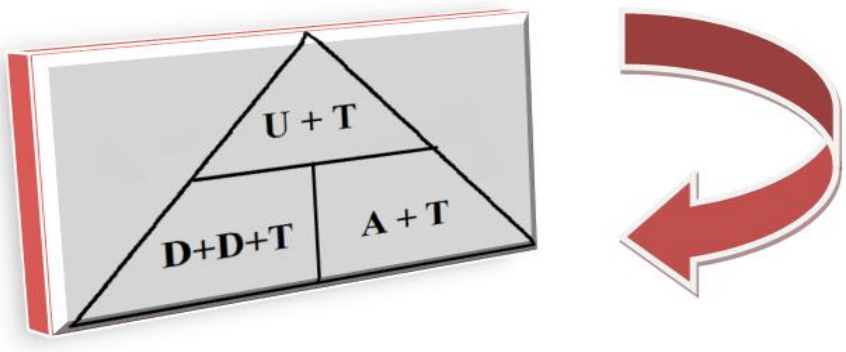


Figure 1- Tester Centric Software Development Model for Short Duration

The abbreviation of above mentioned characters is mentioned below
Where U+T= Consider User Requirements First and related Testing done…
A+T= Analysis and Testing done on Analysis Phase.
D+D+T= Design, Development and then Testing.

Testers are working in all possible combination of development. One can say that it is also a filtering process since the defects are corrected at each individual phase and the output of one phase will be directed as an input to another. The flow mention clock wise direction since after requirements and proper testing comes to next phase i.e. analysis phase where analyst does all important findings and concerning to this phase testing is done. Finally came to the main design, development phase where all documents are converted to identical computerized format and lastly again testing is done to find out errors. Major problems are likely to seen in this phase corresponding serious testing takes place at this stage.

Model 2: Tester Centric Software Development Model for Long Duration:
Following figure indicates the same phases but having changes according to improve the quality of final intended product. Considering the mentality of users, they onetime compromised with the time or duration but never compromised with the product having bugs. Considering the same we tried to introduce the model which actually took time for reaching the goal but the quality now is much more improved after implemented the following changes in SDLC.

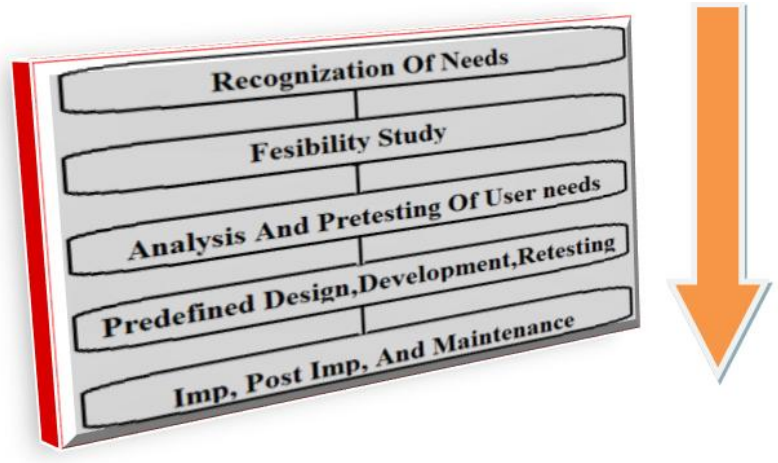


Figure 2- Tester Centric Software Development Model for Long Duration

The above figure is a Tester Centric Software Development Model for Long Duration. Where the first two phases are same as of SDLC. From next phase we tried to introduce tester to test the requirements and correct errors if it observed in the feasibility as well as analysis phase. After this phase usual terms consider i.e. predefined design, development and on the basis of that once again testing is done called as "Retesting". Finally with the implementation, post implementation and maintenance phase mentioned.

4. SIMULATION RESULTS

For successful creation of simulator Orange HRM Open Source HR Management software used. Our research has focused on Orange HRM – **My Info Module**. The reason of choosing the above software is because it's easily available on internet (Open Source), most of IT companies used the same software. The technical details are as follows.

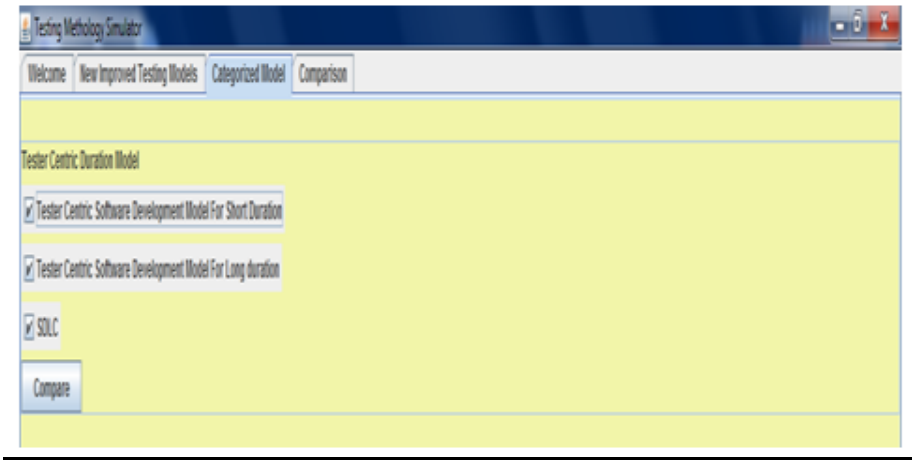
- ✓ Orange HRM version 2.7
- ✓ Frontend- Java version 1.7
- ✓ Backend- excel , Comma Separated Values file (CVS)
- ✓ Ubantu- Operating System
- ✓ Bugzilla- version 4.4.9.

Simulation software based on the new created models, in which Software Testing field gives the prime importance. All the comparison based on

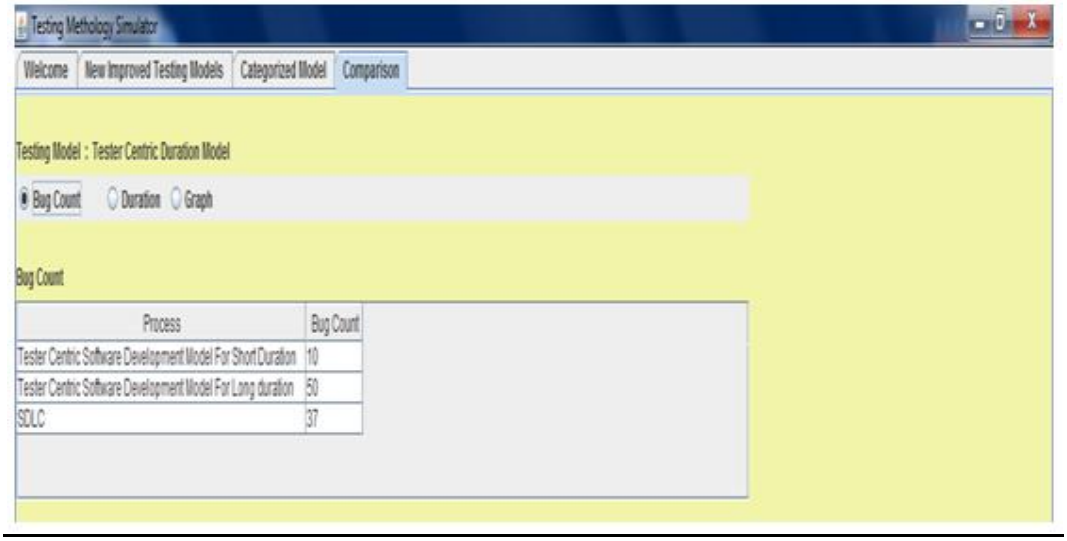
A) **Duration** : (Time required for completing all the phases of software development in hours)
B) **Bug Count** : (Total Quantity of bug found during software development)
C) **Graph** : (Including Design Efforts, Execution Efforts and Total Efforts)

First "Welcome Page" Snapshot Related To Simulator, "For Engage Software Tester Early In Software Development Life Cycle".

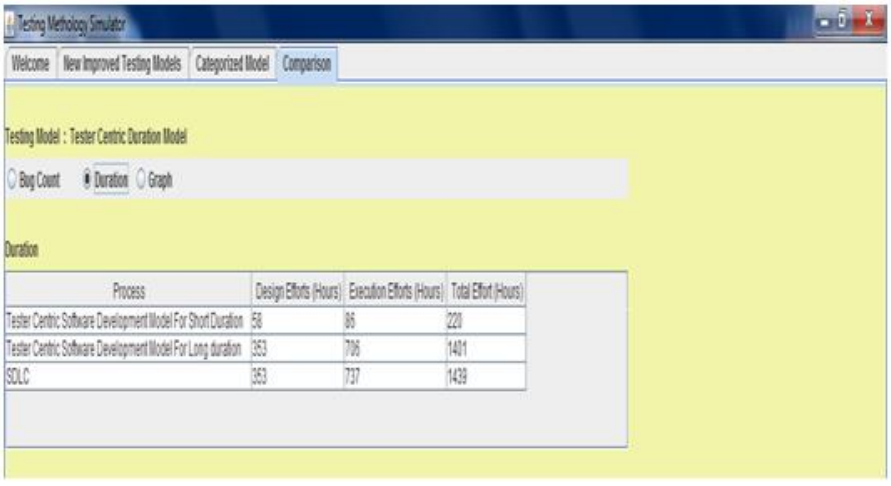
Comparison On the basis Of Duration(Graphical Representation)between Testers Centric Short Duration, Long Duration, and SDLC Model: Now here comparison took place between all the above two newly created tester centric models with normal SDLC. Following snapshot shows the same below.



Bug Count: Following bug count graphical representation appears with the three concern models.



Duration: Following duration graphical representation appears with the three concern models.



Graph: Finally the graph which shows Design efforts, Execution efforts and Total efforts. A following graph consists of testing method on X-axis and Total hours required on Y-axis with the comparison of three concern models.

A) **Comparison based on Total time required to complete software development:** Following table shows all important findings based on the simulation software result. Calculated the bug percentage ratio is based on following formula.
Bug percentage ratio= bug Count *100/ Required time in Hrs.

Table 1: Result of simulation based on time required.

S. No.	Models Type	Required Time In Hrs.	Bug Count	Distribution Time in Hrs.(Design+Execution+Other)	Bug Percentage ratio
1.	Normal Software Development Life Cycle	**1439**	37	353+737+349	*2.57%*
2.	Tester Centric Software Development Model For Short Duration	**220**	10	58+86+75	*4.54%*
3.	Tester Centric Software Development Model For Long Duration	**1401**	50	353+706+342	3.56%

B)Comparison based on Bug Count considering Stated Models: Following table shows total bug count during all the phases of software development. Starting with Normal Software development and comparison with all newly created models. Newly created models numbers have their usual meaning mentioned earlier.

Table 2: Result of simulation based on bug count.

Normal SDLC Bug Count	New Stated Models Bug Count
37 (SDLC)	**1) 10 (Short Duration)** 2) **50 (Long Duration)**

C) Comparison Based On percentage of Bug Count Considering Stated Models:
Next consideration is based on the bug finding ratio which is also shown in following table.

Table 3: Result of simulation based on bug finding Ratio

Normal SDLC Bug Finding Ratio	New Models Bug Finding Ratio	Average Bug Findings Ratio
2.57%	1) 4.54% 2) 3.56%	**4.05**%

CONCLUSION

This paper suggests how to engage software tester early in software development life cycle. Software Product is big issue for customer as well as Software Company. Because of the same software product should be delivered on time with budget and without much more errors. While implementing the above stated models certainly software companies required a less time as compared to normal software development as mentioned in following table.

Sr. N	Models Type	Required Time In Hrs.
1.	Normal **Software Development** Life Cycle	1439
2.	Tester Centric Software Development Model **For Short Duration**	220
3.	Tester Centric Software Development Model **For Long Duration**	1401

While considering the above table normal SDLC requires time 1439 (in hours) to complete the process, but if stated models implemented then it requires much less time in the form of 220 and 1401. So time management is reduced in this case. Secondly bug counting is also essential, and tried to improved the same as normal SDLC as shown in following table.

S. No.	Models Type	Required Time In Hrs.	Bug Count	Findings
1.	Normal Software Development Life Cycle	1439	37	0.025
2.	Tester Centric Software Development Model For Short Duration	220	10	0.045
3.	Tester Centric Software Development Model For Long Duration	1401	50	0.035

So normal SDLC bug counting is 0.025, and all stated models much more improved as well in bug counting also as 0.045 and 0.035. Above table findings generated by simply dividing bug count with required time in hours. Lastly consider bug counting ratio in terms of percentage mentioned in following table as well.

Normal SDLC Bug Finding Ratio	New Models Bug Finding Ratio
2.57%	1) 4.54% 2) 3.56%

In Normal SDLC the same above ratio is 2.57, but stated models ratio is much superior. So finally we conclude that if above stated models used then following main three terms observed.

- ✓ Time requires is less (mentioned in Hrs)to complete the software development life cycle.
- ✓ More Bugs found because of the engagement of software tester early.
- ✓ Improved bug finding ratio.

ACKNOWLEDGEMENT

I would like to thank all anonymous reviewers for their valuable comments that were used to improve this paper. I am grateful to my Respected guides Dr. Suryakant B. Thorat , Dr. UjjwalLanjewar, Dr. Pradeep K. Butey for giving proper guideline and provide necessary help to me. Finally last but not the least I am very thankful to my family, my all mentors starting from my childhood and my friends for their kind support.

REFERENCES

[1]. "Engaging Testers Early and Throughout the SDLC includes seven model" By Mark L. Gillenson, Xihui Zhang, Sandra Richardson.
[2]. Finding and Fixing Problems Early:A Perspective-Based Approach to Requirements and Design Inspections by Dr. Forrest Shull and Dr. IoanaRus , Dr. Jeffrey C. Carver
[3]. Measuring the Software Product Quality during the Software Development Life-Cycle: An International Organization for Standardization Standards Perspective ByRafa E. Al-Qutais (Journal of Computer Science 5 (5): 392-397, 2009 ISSN 1549-3636 © 2009 Science Publications)
[4]. Improving the Software Development Process Using Testability Research Author: Jeffrey M. Voas Keith W. Miller
[5]. Importance of Testing in Software Development Life Cycle Author**:** T.Rajani Devi (International Journal of Scientific & Engineering Research Volume 3, Issue 5, May-2012 1 ISSN 2229-5518)
[6]. Testing and Software Technical Risk Assessments Author: Brad Neal, SimVentions
[7]. Development of Decision Models for Best Use of Software Testing ResourcesAuthor: Charles J. Campbell, Judith C. Simon, Ronald B. Wilkes