International Journal of u- and e- Service, Science and Technology
Vol.10, No.9 (2017), pp.35-48
http://dx.doi.org/10.14257/ijunesst.2017.10.9.04

Increases the Reliability of Software using Enhanced Non Homogenous Poisson Process (EHPP), Functional Point and Test Point Analysis

Amol K. Kadam[1], S.D. Joshi[1], Debnath Bhattacharyya[2] and Hye-Jin Kim[3]

[1]Bharati Vidyapeeth University College of Engineering, Pune
[2]Department of Computer Science and Engineering,
Vignan's Institute of Information Technology,
Visakhapatnam-530049, India
[3]Sungshin Women's University,
2, Bomun-ro 34da-gil,
Seongbuk-gu, Seoul, Korea
akkadam@bvucoep.edu.in, sdj@live.in, debnathb@gmail.com,
hyejinaa@daum.net
(Corresponding Author)

Abstract

In these proposed Software Reliability Growth Model analyze all codes files of the project. In this model every code file of the project is analyze and provide the suggestions to the user for improving performance of the system. Also this model calculate the cost of the project that cannot be calculate at existing software reliability growth model. This model focused on testing time, testing coverage, functional point analysis and test point analysis to increases the reliability of software, calculate software cost and optimize the software maintenance cost.

Keywords*: Enhanced Non Homogenous Poisson process (EHPP), Function Point Analysis (FPA), Test Point Analysis (TPA), Software Reliability Growth Model*

1. Introduction

Testing is a crucial activity to make sure code quality. Huge organizations will have many development groups with their product being take a look at by full test groups. Take a look at team managers should be able to properly set up their schedules associated resources and estimates for the needed take a look at execution effort will be an extra criterion for take a look at choice, since effort could be restrictive in follow. An honest take a look at execution effort estimation approach will profit each tester managers and code comes [1-2]. There's estimation model associated an expertise primarily based approach for take a look at execution effort.

The probability of failure-less operation in a specified environment in a specific period of time under specific conditions is called as Software Reliability[13]. Software Reliability Growth models (SRGM) is developed for the estimate software reliability measures such as number of remaining faults, software failure rate and software reliability.

Software testing can be defined as a process to detect faults in the entire developed computer software which falls in the category of Software development life cycle (SDLC) phases. Software testing helps us to detect the probable faults and errors in the developed software[9]. Testing of the software for longer time does not ensure bug free software

Received (June 14, 2017), Review Result (August 11, 2017), Accepted (September 1, 2017)

with higher reliability[11]. Optimum amount of code also needs to be covered to make sure that the software is of good quality. It is hard to remove the entire faults in the software due to its complex nature. This is also termed as imperfect debugging[15]. Error generation is defined as phenomena in which remaining faults in the software leads to further generation of faults.

Test estimation consists of the estimation of effort and value for a selected level of testing, exploitation numerous ways, tools, and techniques. The wrong estimation of testing effort typically ends up in associate inadequate quantity of testing, which, in turn, will result in failures of software package systems once they're deployed in organizations. Estimation is that the most crucial activity in software package testing[8], associated an ineluctable one, however it's typically performed hurriedly, with those liable for it simply hoping for the simplest.

Testing is directed toward inputs and program parts wherever errors are a lot of possible. The main target of testing is on finding defects, and defects typically often found abundant quicker by totally different testing attributes. It's vital to balance the relationships between effort, schedule and quality[7]. It's wide accepted that merely estimating one in every of these aspects while not considering the others can lead to phantasmagorical estimations. Classical estimation models are established supported linear or non-linear multivariate analysis, that incorporate mounted input factors and stuck outputs.

2. Objective of the Research Work

- ✓ To improve performance & reduce maintenance cost.
- ✓ Estimate Software Cost.
- ✓ Check the efficiency of development activities
- ✓ Quality and Testability of the test object
- ✓ Interdisciplinary Research Project
- ✓ Industrial Consultancy
- ✓ Academic Research Activities

3. Methodology

In this research work we have developed Software Reliability Growth Model that contain Enhanced Non homogenous poison process[3][5], Function Point Analysis and Test Point Analysis. It was an attempt to overcome difficulties associated with lines of code as a measure of software size, and to assist in developing a mechanism to predict effort associated with software development.

Enhanced Non Homogenous Poisson process (EHPP):

- ⦿ Testing Time: Either the CPU time or calendar time is referred as Testing time[16].
- ⦿ Testing Coverage: The prediction of the software reliability is ensured through testing coverage[10].

A software developer make use of Testing Coverage to evaluate the quality of the tested software and also helps to determine the additional effort required for improving the reliability of the software[6].

Threshold value: The threshold value is interpreted based on previous projects experience and historical information. While considering the threshold value, benchmarks designed by industries also taken into grant. From the team experience and various processes involved the threshold is monitored and updated.

Basic Testing Coverage Measures:

- ⦿ 1. Statement Coverage: Number of lines processed in the program. If number of the lines are exceeded more than threshold range then giving advice to user.

- 2. Path Coverage: It indicates number of viable paths that exist in the code and also find the inheritance tree.
- 3. Decision / Condition Coverage: It tells whether Boolean expressions tested in control structures are true or false. If Boolean expressions are more than threshold range then giving to advice
- 4. Procedure Coverage: It gives number of procedures determined by the testing Software reliability growth models (SRGM). In that also giving advice to user when no of the procedures and functions are more than threshold value[14].

So We have to find out reliability of the software and also giving advice for increases reliability of the software.

Enhanced Non Homogenous Poisson Process on Testing Coverage:

Mathematical model is algebraic expression that shows the quantitative stimulation of software analysis.

Block coverage: overall quantity of blocks that have been executed by test cases

Branch coverage: overall quantity of branches that have been executed by test cases

Test Case 1 :

$$\alpha_1 = \int_{t_0}^{t_1} c_1 \, dt$$

Where

α_1 : No. of faults detected in block 1

t_0 & t_1 : Start time & end time of testing in block 1

C_1 is coverage function over interval time $t_0 \leq t < t_1$

Test Case 2 :

$$\alpha_2 = \int_{t_1}^{t_2} c_2 \, dt$$

Where

α_2 : No. of faults detected in block 2

C_2 is coverage function over interval time $t_1 \leq t < t_2$

Resulting Test Case:

$$\alpha = \int_{t_0}^{t_1} c_1 \, dt + \int_{t_1}^{t_2} c_2 \, dt + \cdots + \int_{t_{n-1}}^{t_n} c_n \, dt$$

Proposed Model:

$$\alpha = \sum_{b=0}^{n} C(t_b)$$

α :Actual faults in software.

Testing Efforts: Test effort refers to the expenses for testing the software[8].

Function Point Analysis:

FPA is the method of calculating the size of the software by using the complexity of software functionalities using the function points. Then function point is used for the estimating the effort to develop it.

Calculate Size of Project: In this phase it calculates the lines of code, blank spaces as well as comments in the project. If the number of lines per class is greater than threshold value then it advice for splitting of the class.

Calculate Number of Object in Class: Total number of object in class are identified For the project given to system then after clicking on calculate number of object of class it calculates number of attributes of the class. if the total number of attributes in class are more than threshold range then advice for splitting the class.

Calculate Number of Methods: In this module it calculates how many methods in each class of the project. If method values don't match between threshold range 3-7 then provide suggestion for splitting the methods.

Test Point Analysis:

Team Structure: Mainly focused on how many employees are required for completing project, suppose employees are project manager, team leader, seiner developer, junior developer and trainee. All these employees perform various task such as how many lines written per day, how many days he worked on that project then based on these parameters it estimates cost of the team structure.

Environmental Factors: Environmental Factors plays keen role in estimating the software cost. Infrastructure, electricity, Maintenance & other miscellaneous are such kind of factors which we have to consider for budgeting the cost of the project[4].

Estimated Cost: By considering the parameters which comes under structure of team as well as environmental factors, which are used in development process, total amount of expenditure taken into consideration it calculates total cost of the software.

4. Proposed System Architecture

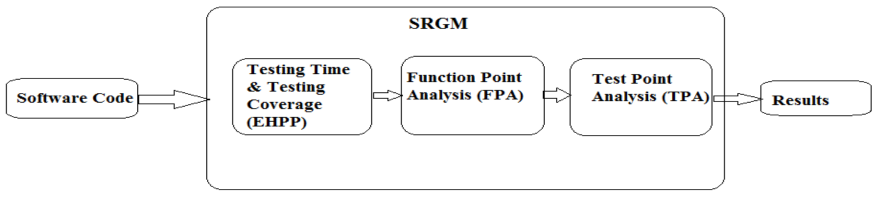

SRGM
Software Code
Testing Time & Testing Coverage (EHPP)
Function Point Analysis (FPA)
Test Point Analysis (TPA)
Results

Figure 1. System Architecture

This system architecture comprises the three modules:

- Input files
- SRGM
  1. Testing Time & Testing Coverage
  2. Function Point Analysis
  3. Test Point Analysis
- Graph Generation

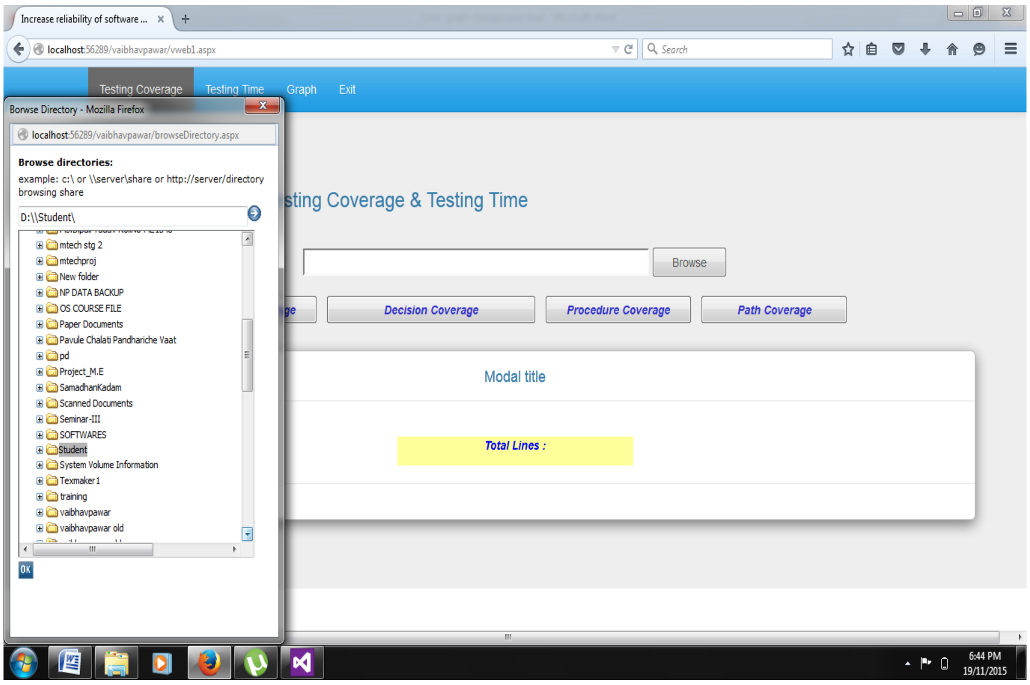
Figure 1 shows the diagram for the architecture of proposed system. Architecture can be depicted as subsequent way:

1. In Put Files is the actual input to the project. It contains the various code modules to be analyzed
2. Develop the EHPP Model using Testing time and testing Coverage.
3. Analysis on input file has performed by means of function point analysis and test point analysis along with SRGM.
4. Then finally result of the processing will shown in the form of graph.

5. Result Analysis

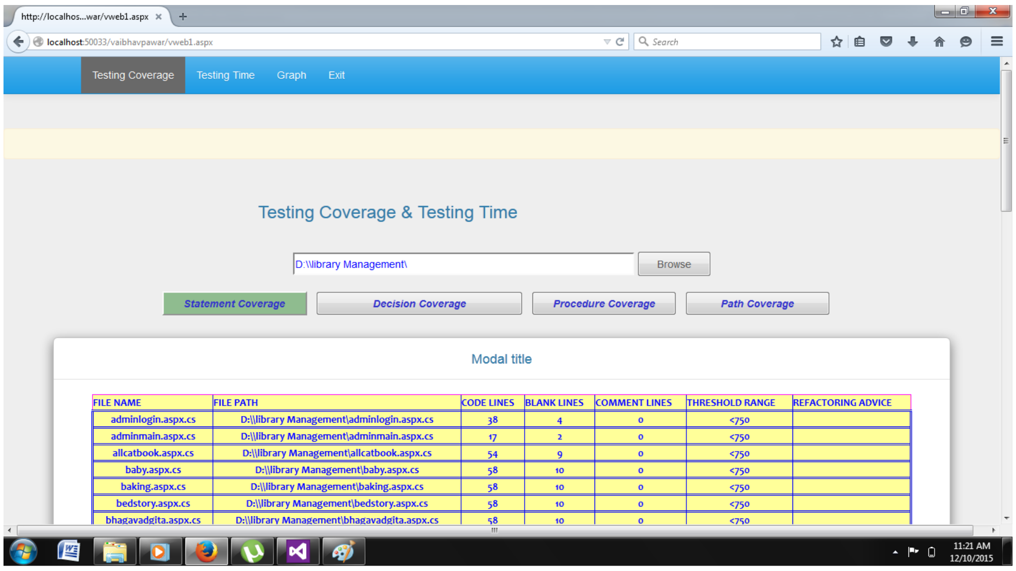
Screen: Testing Time & Testing Coverage

Description: In this there are four buttons for analysis of the project code. First provide input for the system using browse button to select particular project folder.



Screen: Statement Coverage

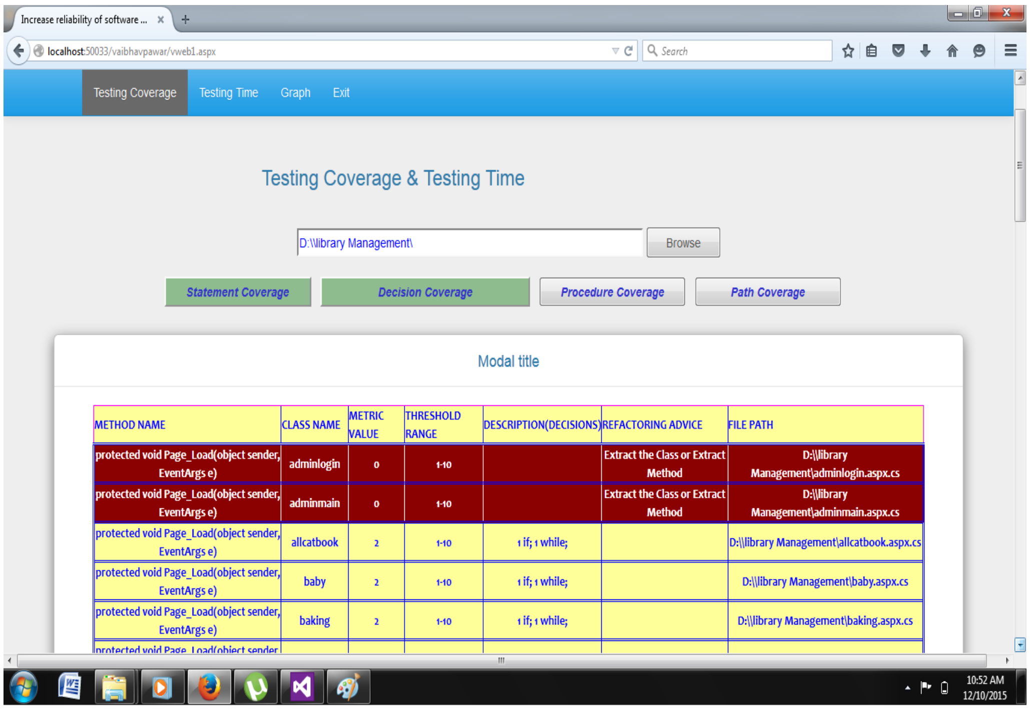
Description: Calculate lines of code executed in project.



FILE NAME	FILE PATH	CODE LINES	BLANK LINES	COMMENT LINES	THRESHOLD RANGE	REFACTORING ADVICE
adminlogin.aspx.cs	D:\\library Management\adminlogin.aspx.cs	38	4	0	<750	
adminmain.aspx.cs	D:\\library Management\adminmain.aspx.cs	17	2	0	<750	
allcatbook.aspx.cs	D:\\library Management\allcatbook.aspx.cs	54	9	0	<750	
baby.aspx.cs	D:\\library Management\baby.aspx.cs	58	10	0	<750	
baking.aspx.cs	D:\\library Management\baking.aspx.cs	58	10	0	<750	
bedstory.aspx.cs	D:\\library Management\bedstory.aspx.cs	58	10	0	<750	
bhagavadgita.aspx.cs	D:\\library Management\bhagavadgita.aspx.cs	58	10	0	<750	

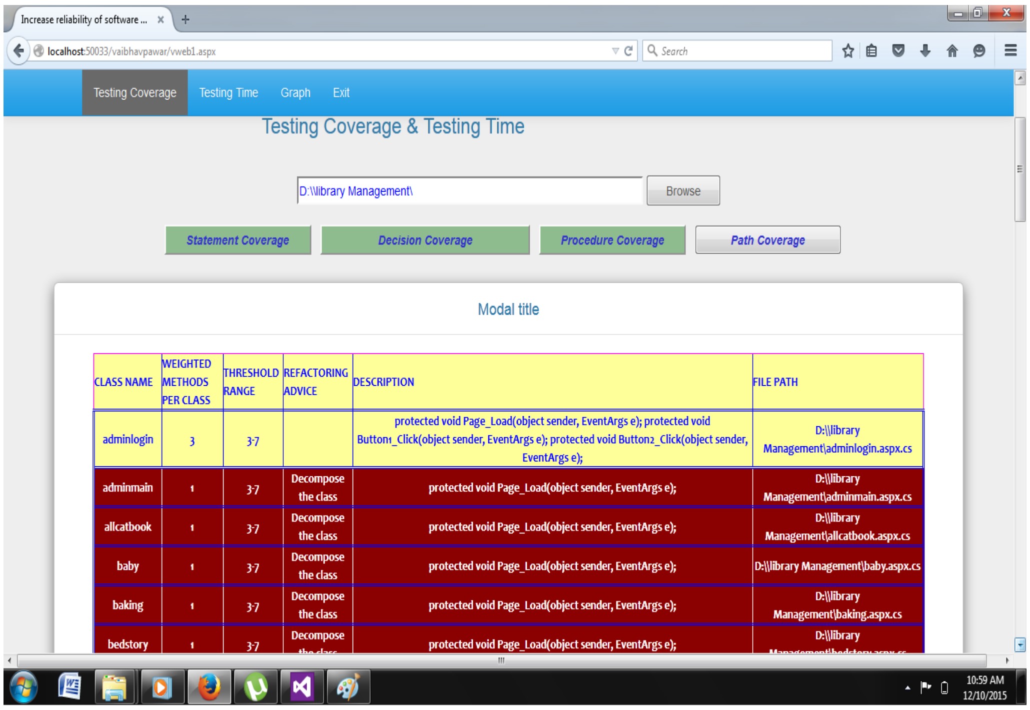
Screen: Decision Coverage

Description: Calculate number of the Boolean expressions executed in project



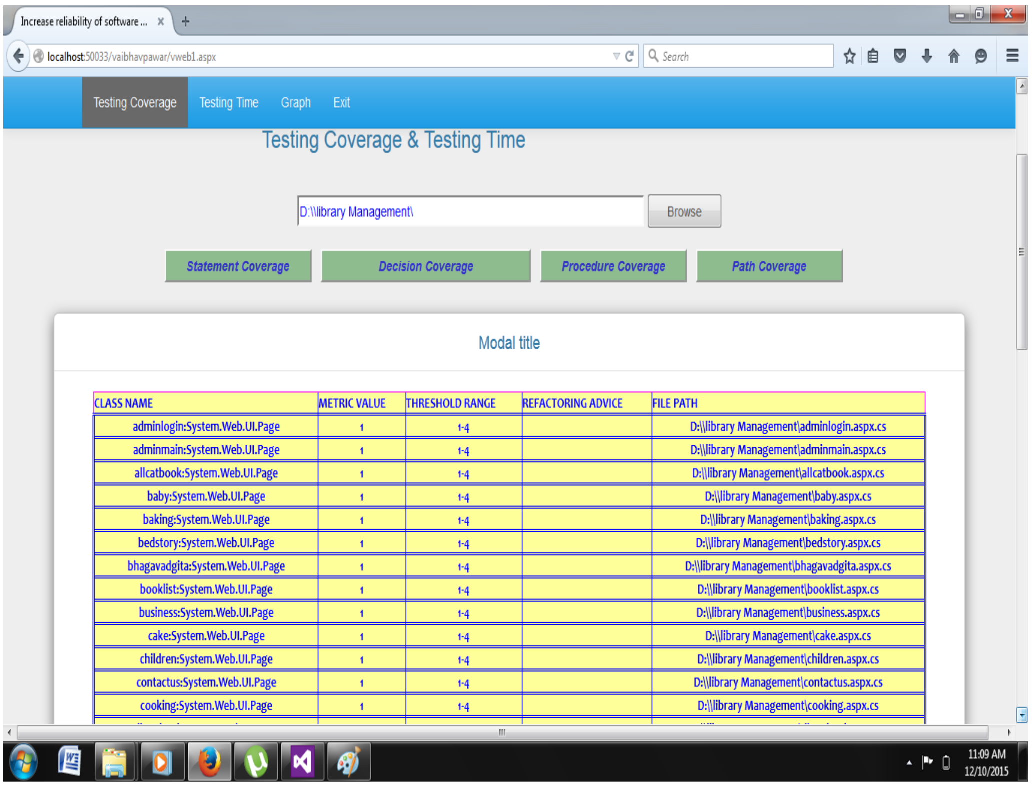
METHOD NAME	CLASS NAME	METRIC VALUE	THRESHOLD RANGE	DESCRIPTION(DECISIONS)	REFACTORING ADVICE	FILE PATH
protected void Page_Load(object sender, EventArgs e)	adminlogin	0	1-10		Extract the Class or Extract Method	D:\\library Management\adminlogin.aspx.cs
protected void Page_Load(object sender, EventArgs e)	adminmain	0	1-10		Extract the Class or Extract Method	D:\\library Management\adminmain.aspx.cs
protected void Page_Load(object sender, EventArgs e)	allcatbook	2	1-10	1 if; 1 while;		D:\\library Management\allcatbook.aspx.cs
protected void Page_Load(object sender, EventArgs e)	baby	2	1-10	1 if; 1 while;		D:\\library Management\baby.aspx.cs
protected void Page_Load(object sender, EventArgs e)	baking	2	1-10	1 if; 1 while;		D:\\library Management\baking.aspx.cs
protected void Page_Load(object sender						

Screen: Procedure Coverage
Description: Calculate number of the methods per class in project



CLASS NAME	WEIGHTED METHODS PER CLASS	THRESHOLD RANGE	REFACTORING ADVICE	DESCRIPTION	FILE PATH
adminlogin	3	3-7		protected void Page_Load(object sender, EventArgs e); protected void Button1_Click(object sender, EventArgs e); protected void Button2_Click(object sender, EventArgs e);	D:\\library Management\adminlogin.aspx.cs
adminmain	1	3-7	Decompose the class	protected void Page_Load(object sender, EventArgs e);	D:\\library Management\adminmain.aspx.cs
allcatbook	1	3-7	Decompose the class	protected void Page_Load(object sender, EventArgs e);	D:\\library Management\allcatbook.aspx.cs
baby	1	3-7	Decompose the class	protected void Page_Load(object sender, EventArgs e);	D:\\library Management\baby.aspx.cs
baking	1	3-7	Decompose the class	protected void Page_Load(object sender, EventArgs e);	D:\\library Management\baking.aspx.cs
bedstory	1	3-7	Decompose	protected void Page_Load(object sender, EventArgs e);	D:\\library

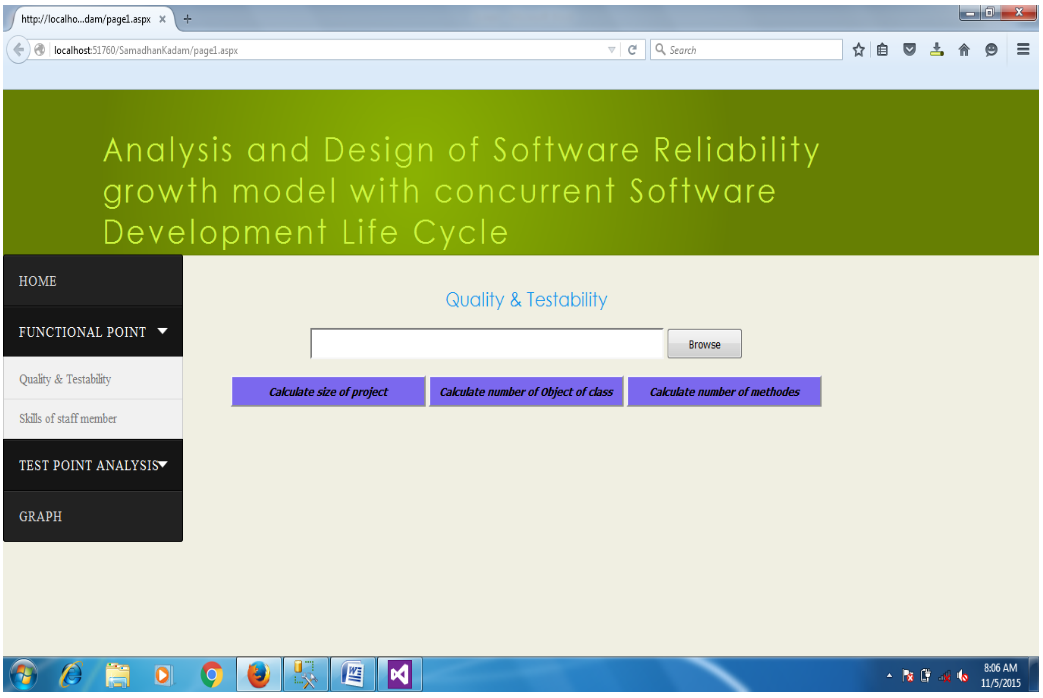
Screen: Path Coverage
Description: Calculate number of possible paths executed in class also finds out inheritance tree



CLASS NAME	METRIC VALUE	THRESHOLD RANGE	REFACTORING ADVICE	FILE PATH
adminlogin:System.Web.UI.Page	1	1-4		D:\\library Management\adminlogin.aspx.cs
adminmain:System.Web.UI.Page	1	1-4		D:\\library Management\adminmain.aspx.cs
allcatbook:System.Web.UI.Page	1	1-4		D:\\library Management\allcatbook.aspx.cs
baby:System.Web.UI.Page	1	1-4		D:\\library Management\baby.aspx.cs
baking:System.Web.UI.Page	1	1-4		D:\\library Management\baking.aspx.cs
bedstory:System.Web.UI.Page	1	1-4		D:\\library Management\bedstory.aspx.cs
bhagavadgita:System.Web.UI.Page	1	1-4		D:\\library Management\bhagavadgita.aspx.cs
booklist:System.Web.UI.Page	1	1-4		D:\\library Management\booklist.aspx.cs
business:System.Web.UI.Page	1	1-4		D:\\library Management\business.aspx.cs
cake:System.Web.UI.Page	1	1-4		D:\\library Management\cake.aspx.cs
children:System.Web.UI.Page	1	1-4		D:\\library Management\children.aspx.cs
contactus:System.Web.UI.Page	1	1-4		D:\\library Management\contactus.aspx.cs
cooking:System.Web.UI.Page	1	1-4		D:\\library Management\cooking.aspx.cs

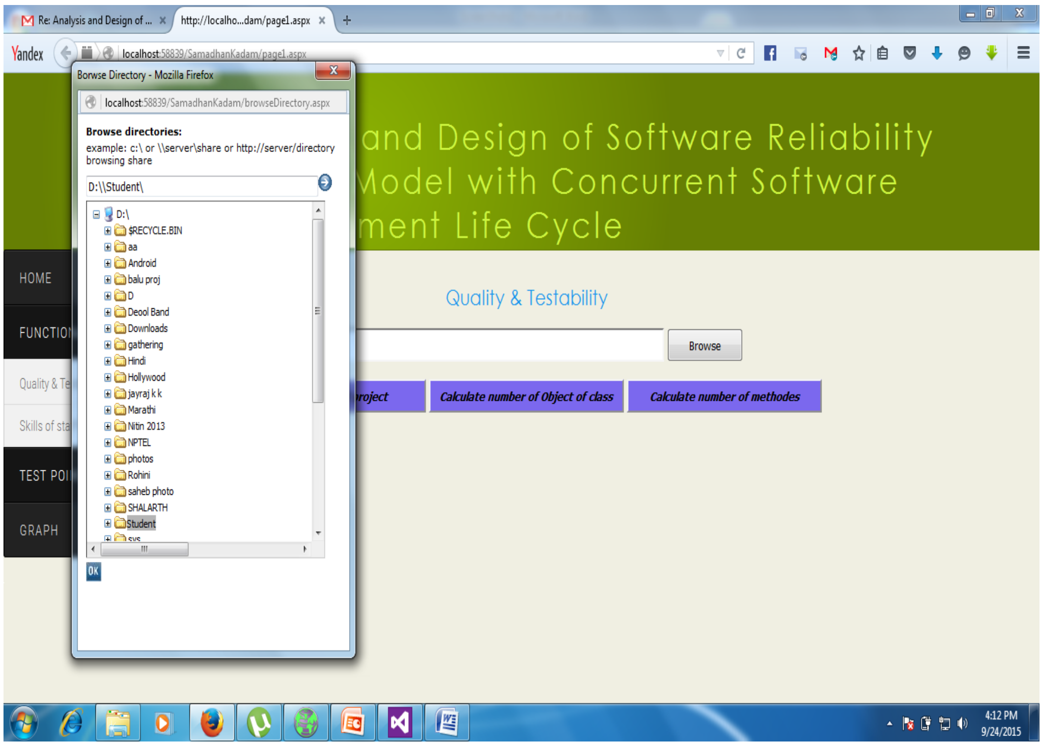
Screen: Home Screen for module II

Description: This is home screen of the research work. In that screen we can see the different tabs like Functional Point, Test Point & Graph.


Analysis and Design of Software Reliability growth model with concurrent Software Development Life Cycle
HOME
FUNCTIONAL POINT
Quality & Testability
Skills of staff member
TEST POINT ANALYSIS
GRAPH
Quality & Testability
Browse
Calculate size of project
Calculate number of Object of class
Calculate number of methodes

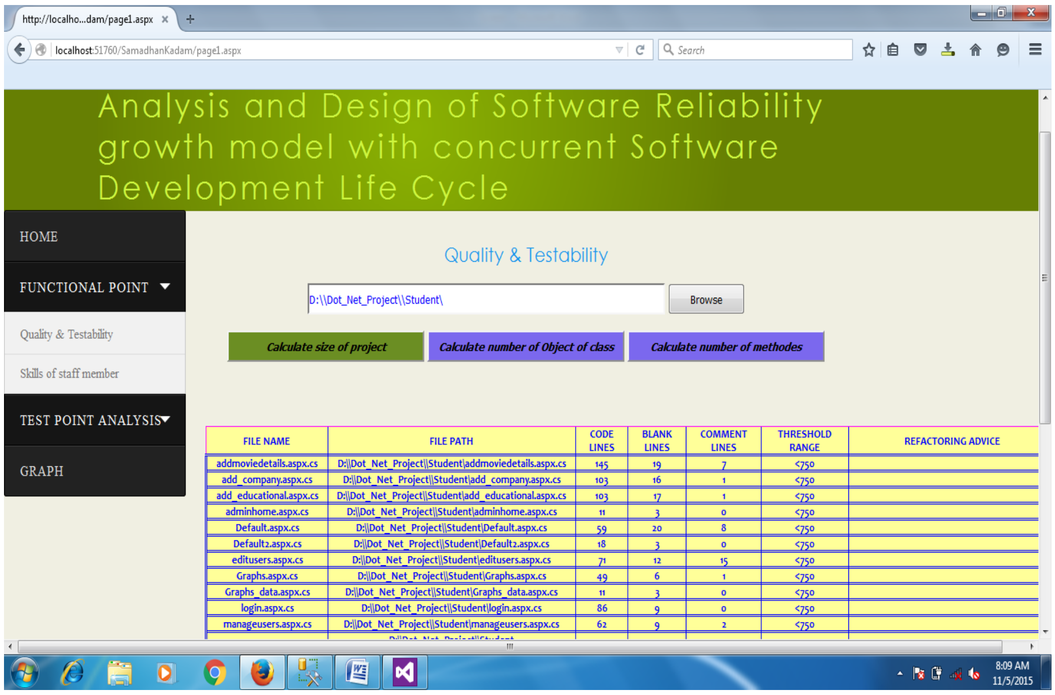
Screen: Quality & Testability

Description: In this there are three buttons for analysis of the project code. First provide input for the system using browse button to select particular project folder.


Borwse Directory - Mozilla Firefox
localhost:58839/SamadhanKadam/browseDirectory.aspx
Browse directories:
example: c:\ or \\server\share or http://server/directory browsing share
D:\\Student\
D:\
$RECYCLE.BIN
aa
Android
balu proj
D
Deool Band
Downloads
gathering
Hindi
Hollywood
jayraj k k
Marathi
Nitin 2013
NPTEL
photos
Rohini
saheb photo
SHALARTH
Student
OK
and Design of Software Reliability
Model with Concurrent Software
ment Life Cycle
Quality & Testability
Browse
Calculate number of Object of class
Calculate number of methodes
HOME
GRAPH

Screen: Calculate Size of Project

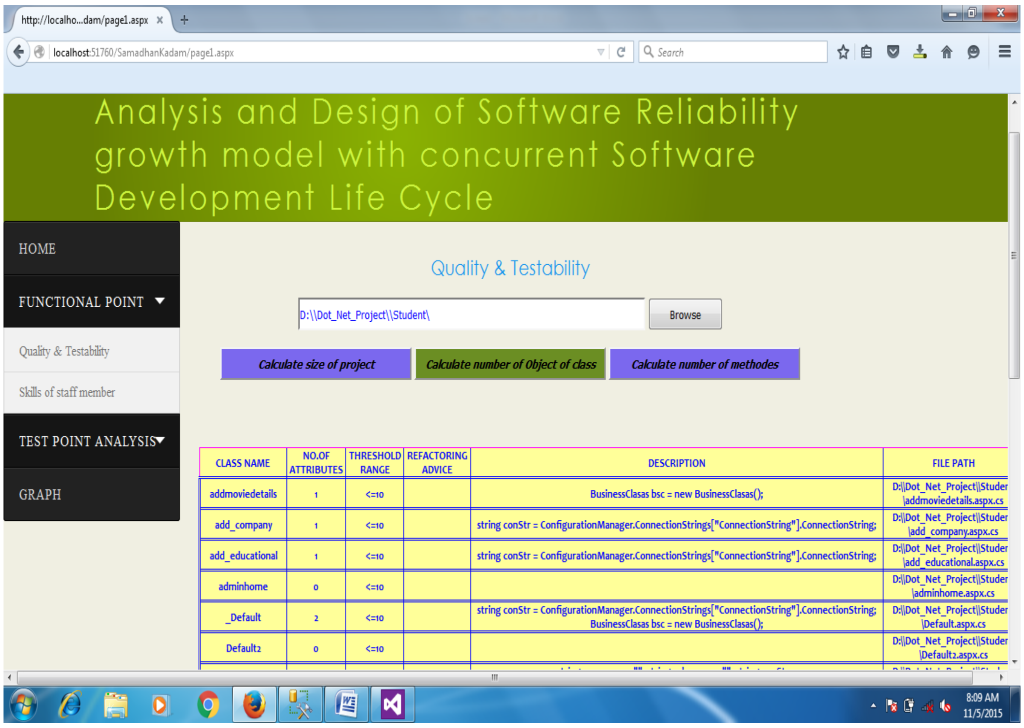
Description: In this screen analysis all coding files of the project and calculate size of the project. If line of code increases more than 750 then suggest the split the class.


localhost:51760/SamadhanKadam/page1.aspx
Analysis and Design of Software Reliability growth model with concurrent Software Development Life Cycle
HOME
FUNCTIONAL POINT
Quality & Testability
Skills of staff member
TEST POINT ANALYSIS
GRAPH
Quality & Testability
D:\\Dot_Net_Project\\Student\
Browse
Calculate size of project
Calculate number of Object of class
Calculate number of methodes

FILE NAME	FILE PATH	CODE LINES	BLANK LINES	COMMENT LINES	THRESHOLD RANGE	REFACTORING ADVICE
addmoviedetails.aspx.cs	D:||Dot_Net_Project||Student|addmoviedetails.aspx.cs	145	19	7	<750	
add_company.aspx.cs	D:||Dot_Net_Project||Student|add_company.aspx.cs	103	16	1	<750	
add_educational.aspx.cs	D:||Dot_Net_Project||Student|add_educational.aspx.cs	103	17	1	<750	
adminhome.aspx.cs	D:||Dot_Net_Project||Student|adminhome.aspx.cs	11	3	0	<750	
Default.aspx.cs	D:||Dot_Net_Project||Student|Default.aspx.cs	59	20	8	<750	
Default2.aspx.cs	D:||Dot_Net_Project||Student|Default2.aspx.cs	18	3	0	<750	
editusers.aspx.cs	D:||Dot_Net_Project||Student|editusers.aspx.cs	71	12	15	<750	
Graphs.aspx.cs	D:||Dot_Net_Project||Student|Graphs.aspx.cs	49	6	1	<750	
Graphs_data.aspx.cs	D:||Dot_Net_Project||Student|Graphs_data.aspx.cs	11	3	0	<750	
login.aspx.cs	D:||Dot_Net_Project||Student|login.aspx.cs	86	9	0	<750	
manageusers.aspx.cs	D:||Dot_Net_Project||Student|manageusers.aspx.cs	62	9	2	<750	

Screen: Calculate Number of Object Class.

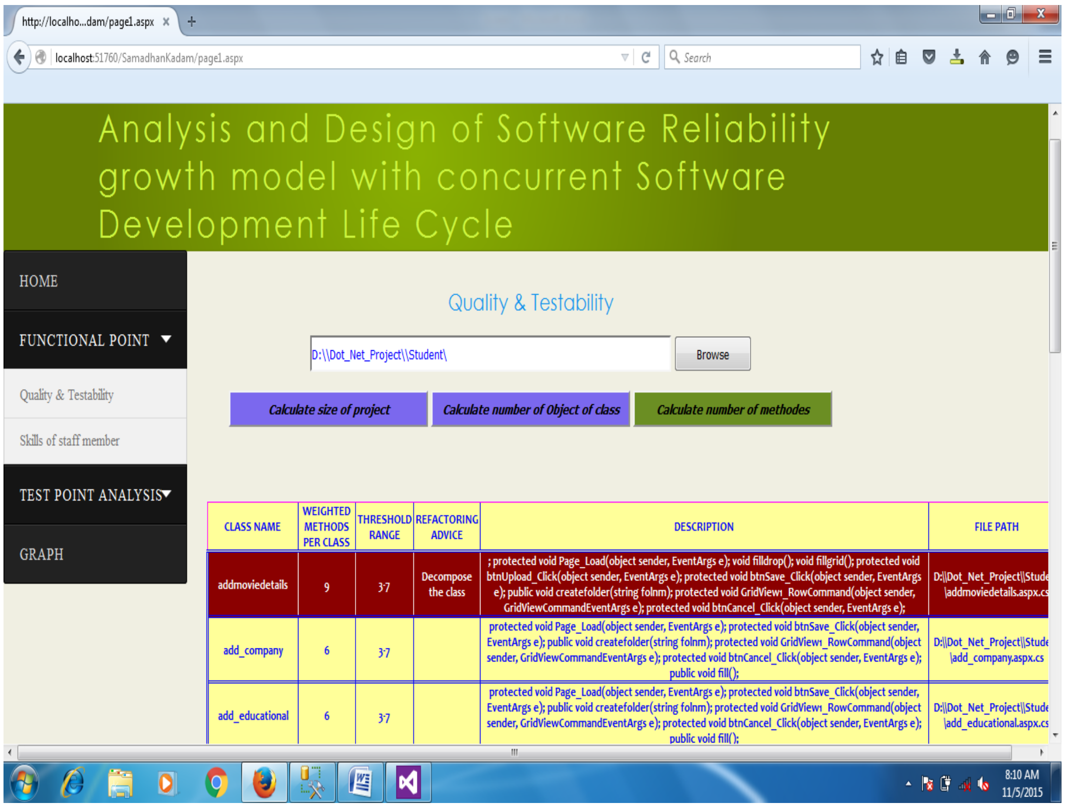
Description: In this screen calculate how many objects of each class in the project. If Objects are more than 10 then provide suggestion to the user.


http://localho...dam/page1.aspx
localhost:51760/SamadhanKadam/page1.aspx
Search
Analysis and Design of Software Reliability growth model with concurrent Software Development Life Cycle
HOME
FUNCTIONAL POINT
Quality & Testability
Skills of staff member
TEST POINT ANALYSIS
GRAPH
Quality & Testability
D:\\Dot_Net_Project\\Student\
Browse
Calculate size of project
Calculate number of Object of class
Calculate number of methodes
8:09 AM
11/5/2015

CLASS NAME	NO.OF ATTRIBUTES	THRESHOLD RANGE	REFACTORING ADVICE	DESCRIPTION	FILE PATH
addmoviedetails	1	<=10		BusinessClasas bsc = new BusinessClasas();	D:\\Dot_Net_Project\\Studer \addmoviedetails.aspx.cs
add_company	1	<=10		string conStr = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;	D:\\Dot_Net_Project\\Studer \add_company.aspx.cs
add_educational	1	<=10		string conStr = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;	D:\\Dot_Net_Project\\Studer \add_educational.aspx.cs
adminhome	0	<=10			D:\\Dot_Net_Project\\Studer \adminhome.aspx.cs
_Default	2	<=10		string conStr = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString; BusinessClasas bsc = new BusinessClasas();	D:\\Dot_Net_Project\\Studer \Default.aspx.cs
Default2	0	<=10			D:\\Dot_Net_Project\\Studer \Default2.aspx.cs

Screen: Calculate Number of Methods.

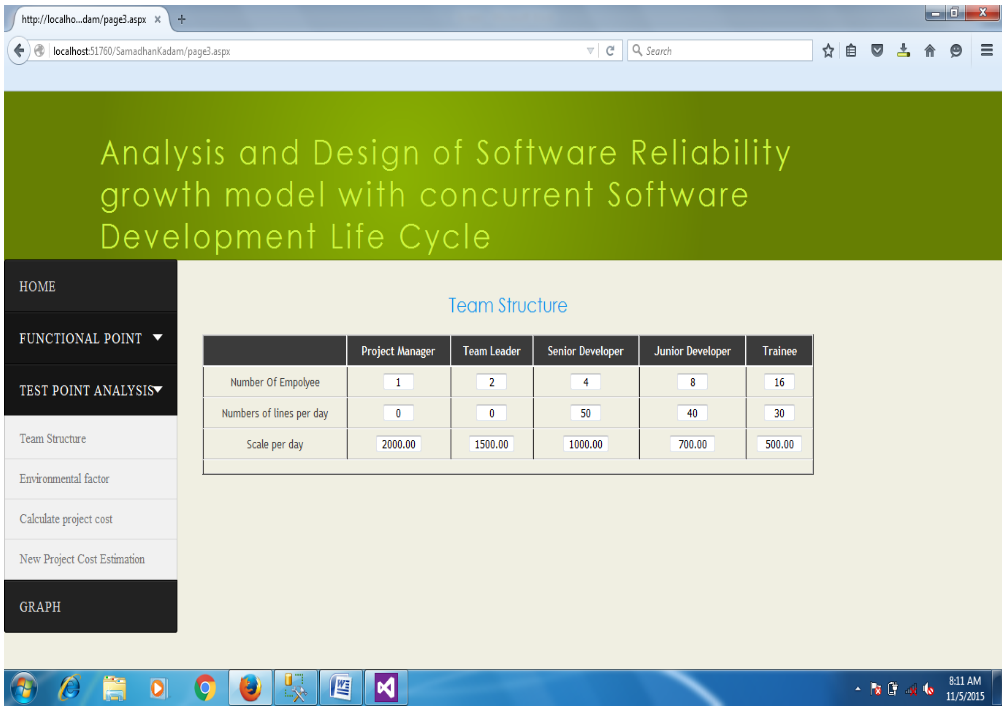
Description: In this screen calculate how many methods in each class of the project. If method values don’t match between 3-7 then provide suggestion for splitting the methods.


http://localho...dam/page1.aspx
localhost:51760/SamadhanKadam/page1.aspx
Search
Analysis and Design of Software Reliability growth model with concurrent Software Development Life Cycle
HOME
FUNCTIONAL POINT
Quality & Testability
Skills of staff member
TEST POINT ANALYSIS
GRAPH
Quality & Testability
D:\\Dot_Net_Project\\Student\
Browse
Calculate size of project
Calculate number of Object of class
Calculate number of methodes
8:10 AM
11/5/2015

CLASS NAME	WEIGHTED METHODS PER CLASS	THRESHOLD RANGE	REFACTORING ADVICE	DESCRIPTION	FILE PATH
addmoviedetails	9	3-7	Decompose the class	; protected void Page_Load(object sender, EventArgs e); void filldrop(); void fillgrid(); protected void btnUpload_Click(object sender, EventArgs e); protected void btnSave_Click(object sender, EventArgs e); public void createfolder(string folnm); protected void GridView1_RowCommand(object sender, GridViewCommandEventArgs e); protected void btnCancel_Click(object sender, EventArgs e);	D:\\Dot_Net_Project\\Stude \addmoviedetails.aspx.cs
add_company	6	3-7		protected void Page_Load(object sender, EventArgs e); protected void btnSave_Click(object sender, EventArgs e); public void createfolder(string folnm); protected void GridView1_RowCommand(object sender, GridViewCommandEventArgs e); protected void btnCancel_Click(object sender, EventArgs e); public void fill();	D:\\Dot_Net_Project\\Stude \add_company.aspx.cs
add_educational	6	3-7		protected void Page_Load(object sender, EventArgs e); protected void btnSave_Click(object sender, EventArgs e); public void createfolder(string folnm); protected void GridView1_RowCommand(object sender, GridViewCommandEventArgs e); protected void btnCancel_Click(object sender, EventArgs e); public void fill();	D:\\Dot_Net_Project\\Stude \add_educational.aspx.cs

Screen: Team Structure.

Description: In test point analysis team structure, environmental factor & processing these sections are provides. In team structure provide the how many employees are required for completing project.

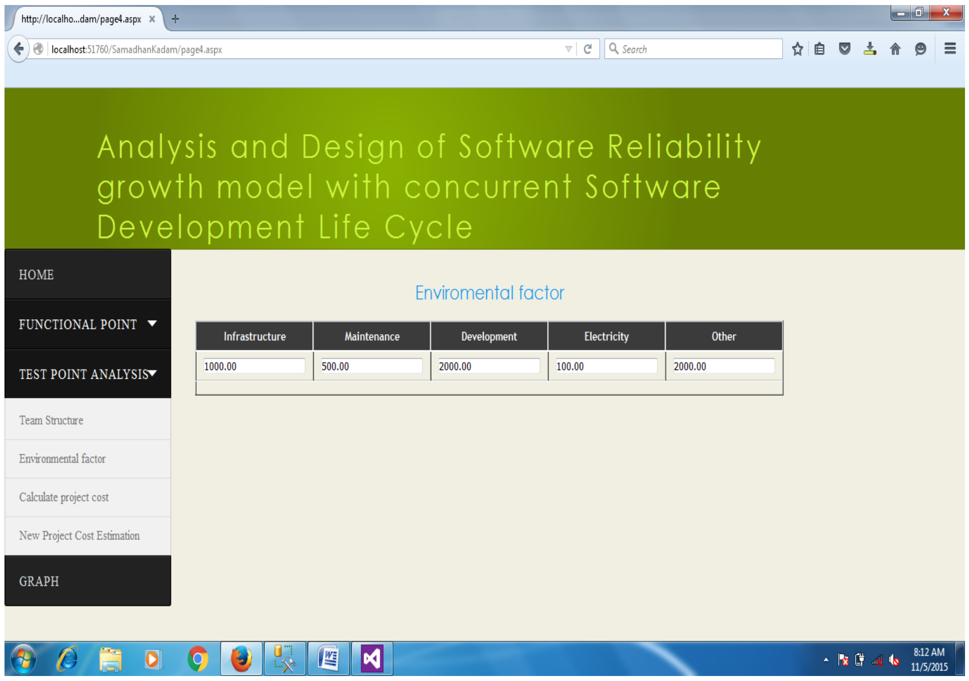


Analysis and Design of Software Reliability growth model with concurrent Software Development Life Cycle

Team Structure

	Project Manager	Team Leader	Senior Developer	Junior Developer	Trainee
Number Of Empolyee	1	2	4	8	16
Numbers of lines per day	0	0	50	40	30
Scale per day	2000.00	1500.00	1000.00	700.00	500.00

Screen: Environmental factor.
Description: In environmental factor provide values for infrastructure, electricity, development and other cost.

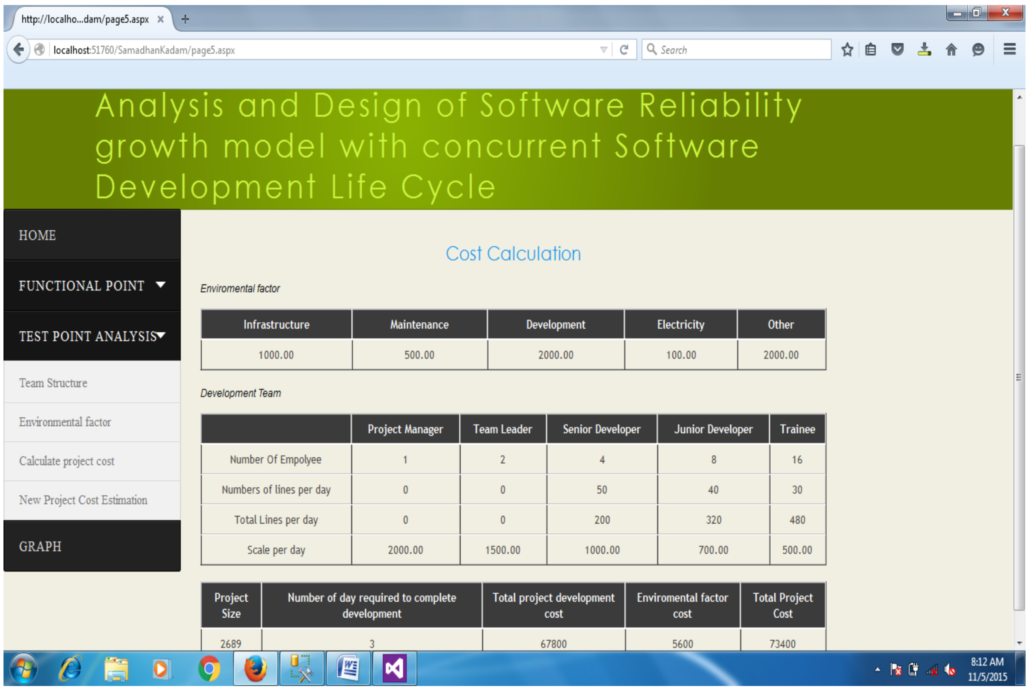


Analysis and Design of Software Reliability growth model with concurrent Software Development Life Cycle

Enviromental factor

Infrastructure	Maintenance	Development	Electricity	Other
1000.00	500.00	2000.00	100.00	2000.00

Screen: Cost Calculation.
Description: This screen is the final screen of the research work. In that calculate the final cost of the project using team structure & environmental factor.



Analysis and Design of Software Reliability growth model with concurrent Software Development Life Cycle

Cost Calculation

Enviromental factor

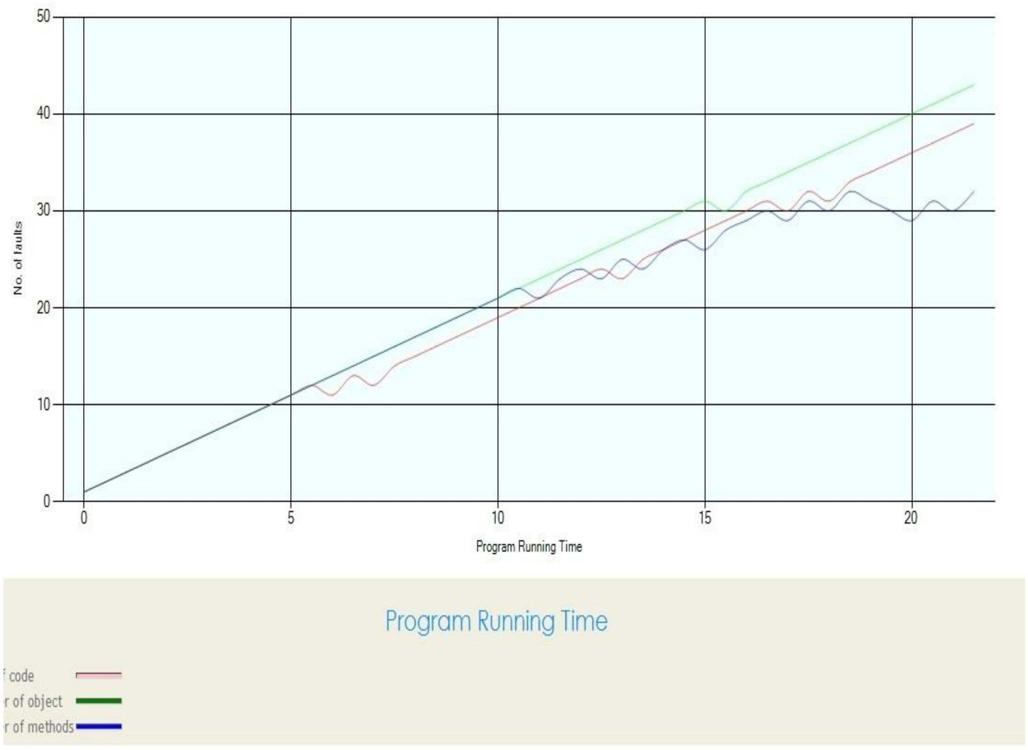
Infrastructure	Maintenance	Development	Electricity	Other
1000.00	500.00	2000.00	100.00	2000.00

Development Team

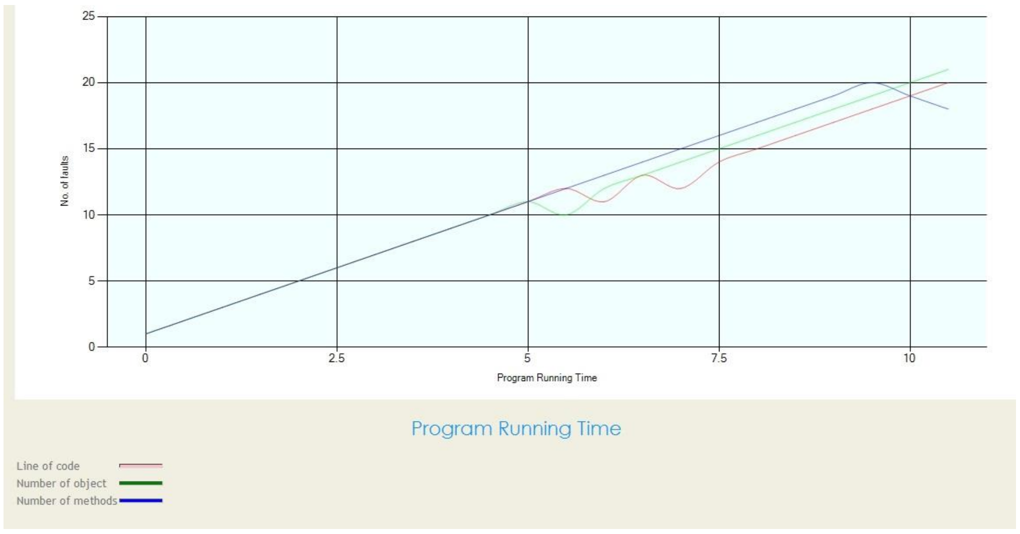
	Project Manager	Team Leader	Senior Developer	Junior Developer	Trainee
Number Of Empolyee	1	2	4	8	16
Numbers of lines per day	0	0	50	40	30
Total Lines per day	0	0	200	320	480
Scale per day	2000.00	1500.00	1000.00	700.00	500.00

Project Size	Number of day required to complete development	Total project development cost	Enviromental factor cost	Total Project Cost
2689	3	67800	5600	73400

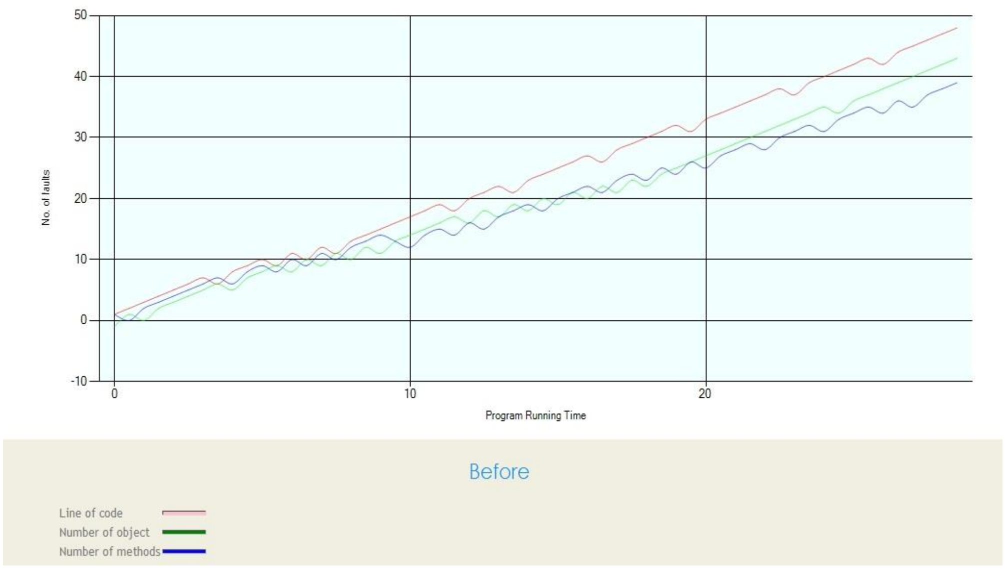
Resultant Graph:
Before Threshold value: Complexity Increase



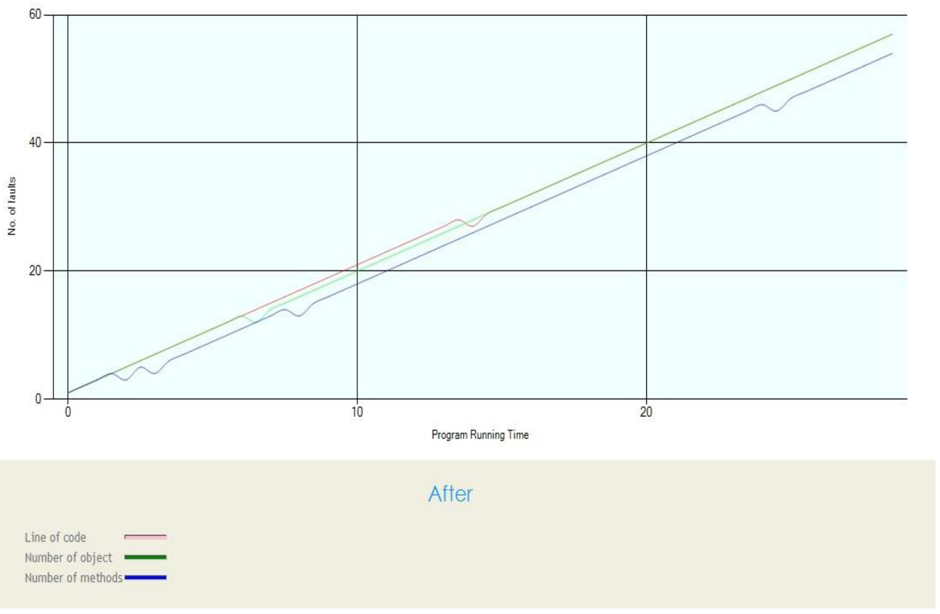
After Threshold value: Complexity Decrease



Before Threshold value: Complexity Increase



After Threshold value: Complexity Decrease



On the above survey complexity of the software is reduces and Increases the reliability of the Software.

8. Conclusion

This research presented a modified software reliability growth model that is based on debugging software and detecting faults that might be removed from the software. I implement this reliability growth model along with quality and testability analyzer. Quality and testability analyze the coadaded file from input project and give the suggession to the users for improving performance. This Reliability Growth Model also calculate the cost of the project that cannot calculated existing model. This model check the developers development skills according to written code files of developer. Also checks efficiency of the developments activities.

References

[1] W. Afzal, S. Alone, K. Glocksien and R. Torkar, "Software test process improvement approaches: A systematic literature review and an industrial case study", Journal of Systems and Software, vol. 111, **(2016)**, pp. 1–33.

[2] J. Yang, Y. Liu, M. Xie and M. Zhao, "Modeling and analysis of reliability of multi-release open source software incorporating both fault detection and correction processes", Journal of Systems and Software, vol. 115, **(2016)**, pp. 102-110.

[3] A. H. S. Garmabaki, A. Barabadi, F. Yuan and J. Lu, "Reliability modeling of successive release of software using NHPP", IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), **(2015)**, pp. 761-766.

[4] M. Zhu, X. Zhang and H. Pham, "A comparison analysis of environmental factors affecting software reliability", Journal of Systems and Software, vol. 109, **(2015)**, pp. 150-160.

[5] B. Roy, S. Kr. Misra, A. Basak, A. Roy and D. Hazra, "A Quantitative Analysis of NHPP Based Software Reliability Growth Models", International Journal of Innovative Research in Computer and Communication Engineering, ISSN (Print): 2320-9798, vol. 2, no. 1, **(2014)**, pp. 2432-2438.

[6] D. Alrmuny, "A Comparative Study of Test Coverage-Based Software Reliability Growth Models ", 11th International Conference on Information Technology: New Generations (ITNG), ISBN:978-1-4799-3187-3, **(2014)**, pp. 255-259.

[7] S. Ikemoto, T. Dohi and H. Okamura, "Quantifying software test process and product reliability simultaneously", IEEE 24th International Symposium on Software Reliability Engineering (ISSRE) , ISSN :1071-9458, **(2013)**, pp. 108-117.

[8] B. Purnaiah, V. R. Krishna and B. Venkata, "Fault Removal Efficiency in Software Reliability Growth Model", Advance in computational research, ISSN 0975-3273, vol. 4, no. 1, **(2012)**.

[9] P.K. Kapur1, Anu G. Aggarwal2 and Abhishek Tandon3 'Two Dimensional Flexible Software Reliability Growth Model with Two Types of Imperfect Debugging' 5th national conference INDIACOM 2011

[10] Shuanqi Wang, Yumei Wu, Minyan Lu, Haifeng Li, "Software reliability modeling based on test coverage ", 9th International Conference on Reliability, Maintainability and Safety (ICRMS),pp. 665-671, ISBN: 978-1-61284-667-5 2011.

[11] Hoang Pham, Editor "Handbook of Reliability Engineering", Springer **(2003)**.

[12] C. D. Scott and R. E. Smalley, "Diagnostic Ultrasound: Principles and Instruments", Journal of Nanosci. Nanotechnology., vol. 3, no. 2, **(2003)**, pp. 75-80.

[13] Ganesh Pai, "A Survey of Software Reliability Models", Dec. 6, 2002

[14] M. H. Chen, Mi. R. Lyu and W. E. Wong, "Effect of Code Coverage on Software Reliability Measurement", IEEE Transactions on Reliability, vol. 50, no. 2, **(2001)**.

[15] P. K. Kapur and R. B. Garg, "Optimal software release policies for software reliability growth model under imperfect debugging," RAIRO, vol. 24, **(1990)**, pp. 295–305.

[16] A. L. Goel and K. Okumoto, "Time dependent error detection rate model for software reliability and other performance measures," IEEE Trans. on Reliability, vol. R-28, no. 3, pp. 206–211, 1979.