IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-6, June 2018 E-ISSN: 2347-2693

An Approach to Regression Testing based on Grounded Theory Specifications

Prabhakar Kandukuri[1*], A. Ananda Rao[2], K. Venugopala Rao[3]

[1] Department of CSE, JNT University, Ananthapur, INDIA
[2] Dept. of CSE, JNT University, Ananthapur, INDIA
[3] Dept. of CSE, GNITS, Hyderabad, INDIA

[*]*Corresponding Author: prabhakarcs@gmail.com, Tel.: +91-97057-95005*

Available online at: www.ijcseonline.org

Accepted: 12/Jun/2018, Published: 30/Jun/2018

***Abstract*—** Regression testing becomes a tedious task while validating the Functional and Non-Functional aspects of a software system at maintenance. It is frequently performed activity on a legacy system for every set of change requests at a moment. Hence, it costs a lot for software maintenance in terms of effort and computing resources. The existing approaches like model driven migration, mutual collaboration, test case-optimization, prioritization, code based test generation, ontological classifications and use case driven approaches are not adequate to handle the above mentioned problem in a cost-effective way. This paper presenting a holistic approach to handle the difficulties during software maintenance and for deriving regression suite from the behavioral models preceded by the Grounded Theory (GT).The grounded theory is used for classifying the change requests to the existing system. And it is used for separating of Functional Requirements (FR's) and Non-Functional Requirements (NFR's) of change request to the existing system. This approach will validate the functional and non-functional aspects of the system, it leads to low maintenance cost, early detection of requirements errors and maximizes the test coverage.

***Keywords*—** Grounded Theory, Regression Testing, Software Maintenance, Test Case, Test Suite.

I. INTRODUCTION

Regression testing is a kind of software testing that confirms that the legacy software preserving its functionalities after its migration or interfaced with other system [1]. It is the critical part of software maintenance, modernization and at software patches. The model based techniques are widely used in generating test cases for integration testing, system testing and regression testing. Among these tests, the regression testing is a tough task at software maintenance. Due to the complexity of the models, it less used in regression testing.

The proposed approach for test suite generation is based on behavioral models of the software system preceded by grounded theory. The grounded theory is used for classifying the changes request/ modifications/functionalities to the existing system. And for separation of Functional Requirements (FR's) and Non-Functional Requirements (NFR's) of new functionalities for the existing system [2, 3]. A grounded theory is the one of the research methodologies for reviewing data (Text) collected, convert into concepts, grouping the concepts, check the reliability of concepts, form the subcategories from concepts, provide the labels and finally add all subcategories to the core categories.

Based on the grounded theory specifications use case diagrams along with annotations are generated. All FR's are represented in the form of use cases, and all NFR's are represented in the form of annotations. From the use case diagram regression test suite is generated to validate the software-intensive system. With this approach, it is easy to generate test suite, estimate the effort and cost of the regression test.

The rest of the paper is organized as follows, the background and related work is discussed in section-2. Regression test suite generation for the given system is discussed in section-3. Regression Test Effort and Cost Estimation are discussed in section-4. Results and Discussions are given in section-5 Concluding remarks and future directions are given in section-6.

II. RELATED WORK

It is mandated to perform regression testing after the maintenance of any software product/ system. According to Fran k Fleurey and H.M. Sneed, more than 70% of software budget is consumed by regression testing [4, 5]. According to L. Erlikh, 85-90% of the projects are under maintenance [6]. These statements shows the importance of regression testing and how much budget it is consuming.

In order to reduce the regression testing cost Dr. Kiran Kumar proposed an approach called cost effective regression testing in black-box testing environment [7], in this method he combined both functional and boundary value analysis

test cases and executed as a combined test suite. Then, he reduced testing effort and cost to certain extent. Mr. Prabhakar, Proposed an approach called cost effective model based regression testing [1]. Here, he generated test cases through use cases and estimated cost and effort. It is giving less time, effort and cost than Dr. Kiran kumar's approach to a great extent. And he generated test cases for reusable components with same approach [9].

Adtha Lawanna, proposed an approach to minimize the test suite which leads to testing time and cost reduction [10] but it is not focusing on NFR's of the system. These methods are efficient and adequate to validate the system in a cost effective way. But, not addressing the NFR's, which are the most important aspects of the software system. The NFR's are the constraints over the FR's [11, 12]. In the proposed approach all FR's and NFR's are represented in a simple behavioral diagram called use case. The use cases diagram is very much close to the behavior of the system and easy way to generate the test cases [1, 14].

The proposed approach a holistic approach to handle the difficulties during software maintenance and generating regression test cases from behavioral diagrams based on grounded theory. This approach is cost effective, efficient and addressing validation all functional and non-functional futures.

III. REGRESSION TEST SUITE GENERATION

The regression test suite is the combination of existing test cases and enhanced test cases of a software system. The regression test suite has to be derived and executed on the system under test, to ensure that the modifications (i.e. the enhanced functionalities) are not affecting the original system [1]. The existing test cases are taken from a system before its modifications and the enhanced test cases are taken from the newly received requirements as per the procedure given in Figure 1. Whenever a customer wants to add new functionalities to the running system, the new requirements are taken in the form of user stories or in form of a text, it depends on the context. If it is an agile development environment user stories are taken into consideration or if any other conventional model it could be a text format. After accepting the change request, it will go through Grounded Theory (GT) process for better specifications.

If the specifications are good, it is easy to generate the test cases. Now the enhanced system will go through the grounded theory specifications. The Grounded theory methodology consists of three categories coding namely open coding, Axial coding, and Selective coding. In this method firstly, Open Coding is applied where user's text is analyzed to form the concepts. These concepts are grouped to form categories, upon these categories Axial Coding is applied to check the reliability. If they found to be reliable, then it can be formed into subcategories. Now as a third step, Selective Coding is applied where a core category is identified and all categories are aligned to it. After these steps, the document is formed where it contains the textual format of these requirements in a categorized format for clear understanding. The result is then undergone into Grounded Requirements Engineering (GRE) process.

The GRE process of three steps namely Structural Transfer, Content Transfer and Detailed Description. Firstly, the Structure Transfer where an initial use case is taken and this step should bring up a rough structure for the requirements and is intended to identify nearly all involved use cases. Secondly, Content Transfer where additional information is added which also includes behavior and conditions of a particular use case. Thirdly, Detailed Description where description of each and every attribute is described.

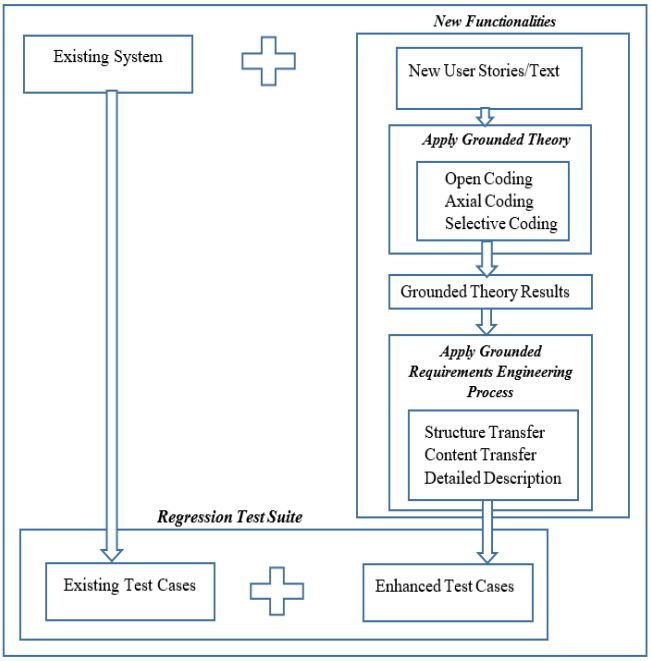

Fig. 1: An Approach to regression test suite generation through Grounded theory Specifications.

After applying the Grounded Requirements Engineering process, the enhanced system gets its behavioral diagrams with all constraints. This system mainly provides Functional and Non-Functional requirements (FR's and NFR's). All FR's are represented in the form of use cases and NFR's are represented in the form of Annotations. Because the NFR's are the constraints over the NFR's and all constraints are represented in the form of annotations.

For the proposed approach, a case study of Library Management system is considered with existing functionalities tabulated in Table 1. The proposed change request *"Login management"* is tabulated in Table 2 in the form of a text. In this approach, first requirements from the

users are elicited and documented using the Grounded Theory. This Grounded Theory results then converted into the Use case template.

Table 1. List of Existing Functionalities in a Library Management System

1. *View particular user details*
2. *View books of user already taken and it's details*
3. *View members who are in the system*
4. *Search for the books in the system*
5. *Reserve searched books*
6. *Issue the books if available*
7. *Renew the books*
8. *Return books and check for due date*
9. *Add/remove books by the Admin in the database*
10. *Update records concurrently*

The grounded theory is applied to the user text of Table 2. Now, the first step in GT is applied and concepts are obtained and Shown in Figure 2. In this process, user stories/ text is broken down into pieces called concepts and these concepts are examined closely, identify the relations among them (like the relation between user and administrator), and finally marks with label/code. Now, these concepts are compared with core existing categories of the system and added to the respective group. Now the related futures are combined. Therefore, the User is combined with first six functionalities of Table 1 and the administrator is combined with all existing functionalities.

Table 2: User Story/ Text for the new functionality

The Library Management System is going to have Login functionality where every user/ member will have their username and password to enter into the system. The administrator will have the same login mechanism as that of a user and highest access permission is given to the administrator to operate all functionalities that exist in the software system. And the lowest access permission is given to the user to search and return books in the online library.

.

Now the Grounded Theory results are given to Grounded Requirements Engineering Process (REP) and it will go through the three step process again which is shown in Table 3. Requirements are taken as use cases and are converted to a use case template which consists of different attributes.

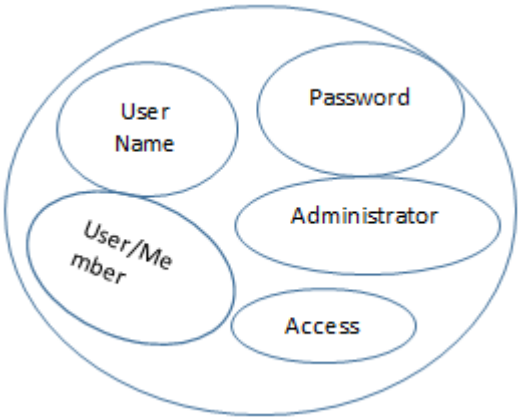

Fig. 2: Building Concepts for the user Stories/ Text.

After the transfer from the GT to the use case template, now take the other approach of extracting the NFR's for the use case of Table 2. Hence, the list of NFR's for their corresponding use cases are obtained. The use cases and their corresponding non-functional requirements are tabulated in table 3.

The extracted list of Use cases and their NFR's are then separated and then written in a detailed manner. By applying the Grounded Theory to Use cases steps we get a template which consists of details about NFR's for a particular Use case. Table 4. Shows how NFR's are associated with a particular use case.

Table 4 shows how the use case needs to have NFR's associated with it. For a particular use case, there can be "multiple NFR's" associated with it or "No NFR's at all". It can be a simple use case having No NFR's associated with it. The NFR's are listed out and the usage of each is written as above for clear understanding. The preconditions and post conditions are useful for extracting exact output for the given input which are composed of NFR's. Also, priority is given for each and every NFR associated for a particular use case. This is useful while developing the system. As care should be taken to the NFR's who have high priority than to those who have low priority.

Now, for the specified Requirements, Use Case Diagram is drawn which consists of Use cases and also the NFR's which are represented in the form of annotations. It is one way of representing FR's and NFR's in the Use Case Diagram. After combing the use case template consisting of FR's and NFR's, Figure 3. Shows the use case diagram for the Library Management System having FR's and NFR's.

Table 3.Template for Use Case and its attributes.

Structure Transfer/ Content Transfer		Detailed Description
Use Case Name: Login		
Description	This is used for a login of Student as well as staff.	The user wants to Login in the system, so details of him are entered for the Login purpose.
Scope	Until the user logins into the website	
Level	User	
Primary Actor	Student/ Staff	Rational Rose tool for displaying the student or staff actors
Stakeholders and interests	Participant wants the Login to be of two kinds, one for Student and one for the staff	Student and staff should have separate logins and these processes should be different
Preconditions	Inputs from the user are to be taken for the Login	
Success guarantees	When on success Login, the User details should be shown	The login of the user is being done successfully
Trigger	Participant wants to Login separately	The details will be taken into consideration for Login.
Priority	Medium	High- While developing the system, it should be done first.

Table 4.Use Cases with its corresponding NFR's

Use cases	Non-functional requirements
1. Login	Integrity/ Security
2. View user details	Correctness, Transparency
3. View books of user	Correctness, Transparency
4. View members	Integrity, Transparency, Accountability
5. Search books	Performance, GUI
6. Reserve books	Verifying user details and book details
7. Issue books	Verifying user details and book details
8. Renew books	Verifying user details and book details
9. Return books	Verifying user details and book details
10. Add/remove books	Maintainability
11. Update records	Maintainability, Backup

Table 5: Login Use Case with NFR's and its Attributes

Use case: Login		
Structure Transfer/ Content Transfer		**Detailed Description**
List Of NFR's		
1. Integrity 2. Security	Used for providing access to authorized user Providing safe login and to avoid misuse of data	The assurance given to the user that it is accessible only by him The assurance given to the user that the details provided are secure from external attack
Preconditions	Details should be entered by the user	In order to provide safe login, details are to be entered
Post conditions	Page should be opened	
Priority for NFR's 1. Integrity 2. Security	 Medium Medium	As integrity and security are important for the Login, they should be implemented soon.

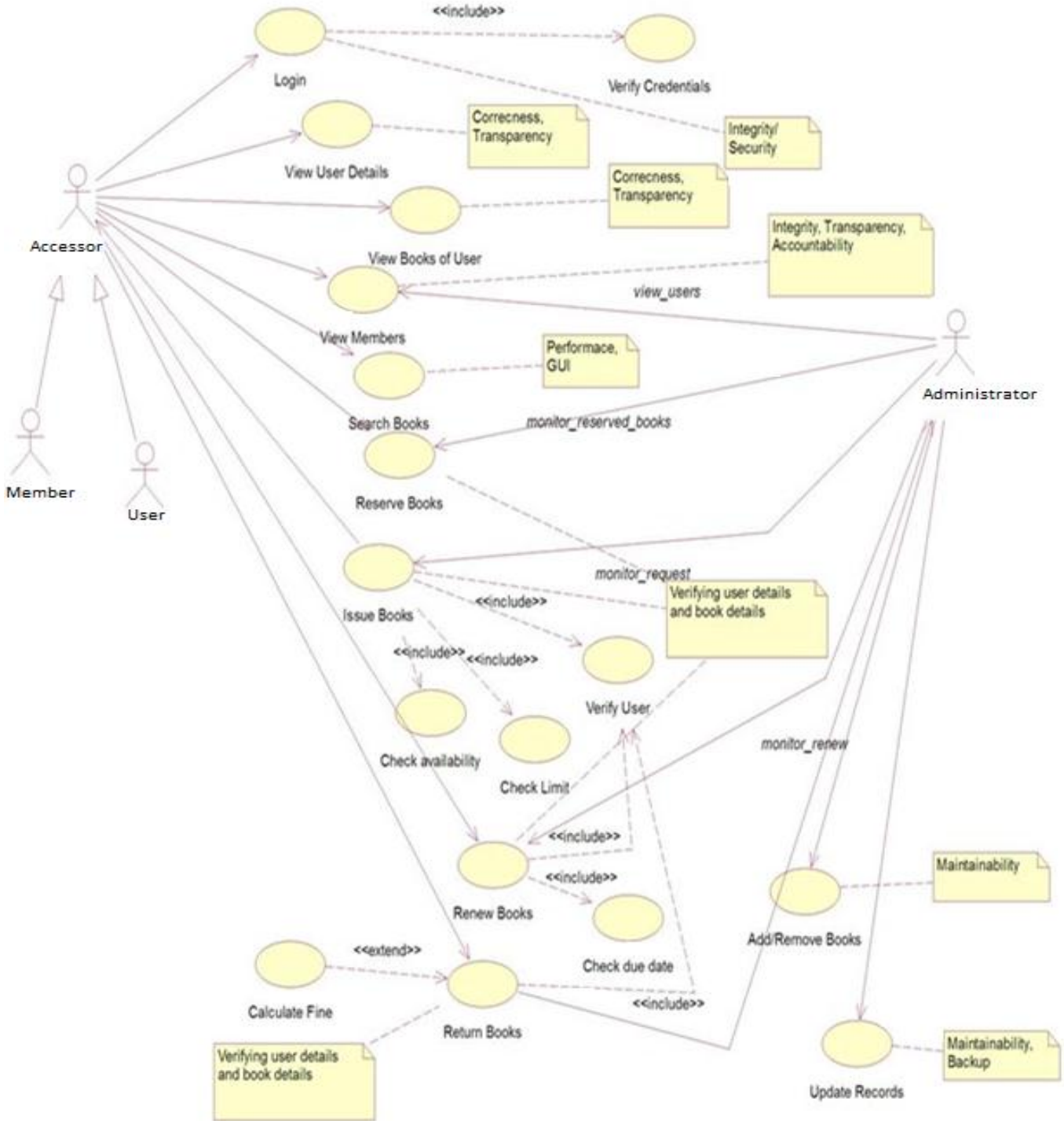

Fig. 3. Use case diagram for the Library Management System shows FR's and NFR's

Table 6: List of Test Cases for Library Management System

TC ID	TC Name	TC Description	Pre -condition	I/P Fields	Expected Results
1	Login	Check the availability of login mechanism	Click on login button	Mouse click on login button	Username and password field gets opened
1.1	Verify Credentials	Verify the use name and password with database	User/administrator Should register with system	Login data and password data	If Inputs match with database allow the member/ administrator to access to system Else reject the access.
1.2	Integrity	Check the integrity of login mechanism	Protocol having strong moral principles	Apply different dishonesty mechanism.	Upholding the integrity of the system being used
1.3	Security	Verify the security mechanism for login	Protocol having strong security principles	Apply different unlock mechanism.	Provide security against threats
2	User Details	Check the availability of view user details mechanism	User must register with the system	Provide id and click on user details	user details window gets opened
2.1	Correctness	Check the correctness of user details	------	Provide correct and incorrect values	System will respond for correct values
2.2	Transparency	Check the Transparency of user details	------	Provide true and false values	System will respond for truth values
3	View Books	Check the availability of view Books mechanism	User must register with system	Click on catalogue	Catalogue will be displayed
3.1	Correctness	Check the correctness of book details	------	Provide correct and incorrect values	System will respond for correct values
3.2	Transparency	Check the Transparency of book details	------	Provide true and false values	System will respond for truth values
4	View Member	Check the availability of view members (user) mechanism	User must register with system	Provide user ID Click on view member	Member window gets opened
4.1	Integrity	Check the integrity of member	Protocol having strong moral principles	Apply different dishonesty mechanism.	Upholding the integrity of the system being used
4.2	Transparency	Check the Transparency of members details	------	Provide true and false values	System will respond for truth values
4.3	Accountability	Check the responsibility of members	------	Provide the responsibilities	Member is responsible for the duties assigned
5	Search Book	Check the availability mechanism for searching the book	User must register with system	Click on catalogue	Catalogue will be displayed
5.1	Performance	Check the performance of a system while	Provide high bandwidth that accommodate	Perform a search or download an article	Provides high performance

		searching for a book	more no. of users		
5.2	Graphical User Interface	checking the screens	System should open different screens	Click/ operate different filed on the screen	Good look and feel mechanisms
6	Reserve Book	Check the availability mechanism for Reserve Book	User must register with system	Click on reserve book option	Book reserved
6.1	Verify User Details	Verify User Details whether right person or not	User must register with system	Provide user ID Click on verify member	User is valid user
6.2	Verify Book Details	Verify Book Details whether it is available or not	Book must register with system	Provide book ID Click on verify	Book details are correct
7	Issue Book	Check the availability mechanism for Issue Book	User must register with system and provide id & Book is available	Click on issue book option	Book issued
7.1	Check Availability	Check Availability of books	Catalogue of book	Provide book ID Click on search	Book available
7.2	Check Limit	Check Limit of books per user	User must register with system and provide id	Enter the limit number	Limit exceeds or not exceeded
7.3	Verify User	Verify User Details whether right person or not	User must register with system	Provide user ID Click on verify member	User is valid user
8	Renew Book	Check the availability mechanism for Renew Book	Book Issued	Provide user ID Click on Renew Book	Book Renewed successfully
8.1	Verify User	Verify User Details whether right person or not	User must register with system	Provide user ID Click on verify member	User is valid user
8.2	Check Due Date	Check the due date of book submission	Fix the due date	Provide book ID Click on verify	Due date exceeded or not exceeded
8.3	Verify Book Details	Verify Book Details whether it is correct one or not while submitting.	Book must register with system	Provide book ID Click on verify	Book details are correct
9	Return Book	Check the availability mechanism for Renew Book	Book Issued	Provide book ID Click on submit	Book submitted successfully
9.1	Verify User Details	Verify User Details whether right person or not	User must register with system	Provide user ID Click on verify member	User is valid user
9.2	Verify Book Details	Verify Book Details whether it is correct one or not while submitting.	Book must register with system	Provide book ID Click on verify	Book details are correct
9.3	Calculate Fine	Calculate Fine if due date exceeds while submitting	Make sure that Due date exceeded	No. of days and fine per day	Payed fine
10	Add/Remove Book	Check the availability mechanism for Add/Remove Book	Check the limit in case add. Make sure that book	Provide book ID Click on Add/ Remove button	Book successfully added/ removed

			available in case delete		
10.1	Maintainability	Check the integrity of login mechanism	catalogue	Maintenance issues	Successfully repaired
11	Update Record	Check the availability mechanism for Update Record	Proper Maintenance issues	Provide accurate details	Record updated successfully
11.1	Maintainability	Check the Maintainability of a system	catalogue	Maintenance issues	Successfully repaired
11.2	Backup	Check the Backup mechanism for Updating the Record	Sufficient storage space	Click on update and update	system updated successfully

IV. REGRESSION TEST EFFORT AND COST ESTIMATION

As testing takes more than a fifty percentage of total budget of a software project [2], it is essential to estimate the regression test cost. The regression testing can be estimated with the help of Equation 4.1 and Equation 4.2 for use case method [1, 12].

Regression test effort = verification of fixed bugs + Bug verification in enhancement. … 4.1

Regression Test effort through use case R_{Uc} *= AUCP x PWE … 4.2*

The enhanced functionalities are added to the use case diagram of the subjective system at design level. Hence, the regression testing effort can be estimated with an existing Equation 4.2 based on use cases [1] and the details of the Equation 4.2 is given in the Table 7 along with effort calculations. Here, PWE is Planning, Writing, and Execution of test cases. For PWE the weighting factor is 2 considered and 0.5 is taken for Total Environment Factor (TEF).

The Table 8 is the detailed description of the Table 7. Table 8 shows the classification of actors and use cases along with their weighting factors. The weighting factor of a use case is given based on the no. of transactions of a particular use case. Here, a transaction is a unit of work seen from the system point of view.

Table 7:Regression Test Effort Estimation

Unadjusted Actor weights(UAW)				Unadjusted Use Case weights(UUCW)			
Actor Name	Actor Type	Factor	Weight	Use Case Name	Use Case Type	Factor	Factor
User	Simple	01	1 x 1=1	View user details, View books, View member, Reserve book, Verify user, Check limit, Check availability, Check due date, Calculate fine, Add/remove books, Update record, Search Books	Simple	12	12 x 1=12
Member	Average	02	1 x 2=2	Verify Credentials, Issue book, Renew book, Return book	Average	04	4 x 2=8
Administrator	Complex	02	1 x 3=3	Login	Complex	03	1 x 3=3
Total UAW			06	*Total UUCW*			23
Unadjusted Use case Point (UUCP) =(UAW+UUCW)							29
Adjusted Use case Point (AUCP) = UUCP x [0.65+(0.01 x TEF)] AUCP=29 x [0.65+(0.01 x 0.50)]							18.99
Total Regression Test Effort =AUCP x 2							37.99

Table 8 : Weighting factors for use case point effort estimation method

Actor weights			Use Case weights		
Actor Type	Description	Factor	Use Case Type	Description	Factor
Simple	Interaction with GUI	1	Simple	Transactions <= 3	1
Average	Interactive or protocol-driver interface		Average	Transactions are between 4-7	2
Complex	Interact with API / low-level interactions	3	Complex	Transactions > 7	3

The total effort estimated through the use case method is 37.99 Man-Hours. The cost of the regression testing can be estimated with existing Equation 4.3.

The total regression testing cost = \$100 x Effort … 4.3

=\$100 x 37.99

=\$3799.

The total estimated cost for performing regression testing on Library Management System is \$3799. The average salary paid to software engineer is \$100 per hour.

V. RESULTS AND DISCUSSIONS

The regression test cases are generated for Library Management System through grounded theory principles. Through the phases of grounded theory all Functional and Non-Functional requirements are categorized. For the given system total 11 functionalities are identified along with their constraints and represented in Use case diagram, it can be observed in Figure 3. From the use case diagram along with annotations, there 37 Test cases are generated and tabulated in the Table 6. The effort and cost of the regression test are estimated with existing formulas [1, 12]. The total effort required to perform regression testing on library management system is estimated as 37.99 Man-Hours. On an average \$100 per hour is the payable salary to the software engineer. Hence the total estimate budget for performing regression testing on a library management system is \$3799.

VI. CONCLUSIONS AND FUTURE DIRECTIONS

This paper introduced a holistic approach to derive regression suite from the behavioural models preceded by the grounded theory. GT is applied for representing functional and non-functional requirements of a subjective system under test. All functional requirements of a change request to the subjective system is represented in the form of use cases and all non-functional requirements are represented in the form of annotations to the use cases. Because all non-functional requirements are the constraints over the functional requirements. And all constraints can be represented in the form of annotations. The regression test cases are generated based on the designed model as per figure 1. This approach has been evaluated with the help of online library management system.

It maximizes the test coverage, early detection of requirements errors with minimum effort and cost. In this experiment, the presented approach generated few repeated test cases, but not extra test cases. Thus, repeated test cases can be eliminated through dependency analysis. This approach leads low maintenance cost of a legacy software. Further research includes usage of state machines to perform dependency analysis for minimizing the test suite. The test suite prioritization techniques can also embed with existing approaches to reduce the test suite execution time. Test suite optimization and prioritization leads to reduction in total budget cost.

REFERENCES

[1] Prabhakar K. et al, "Cost Effective Model Based Regression Testing", "IAENG- World Congress on Engineering", Vol. I-WCE 2017, pp241-246, July 5-7, 2017, London, U.K., 2017.

[2] David Würfel, Rainer Lutz, and Stephan Diehl, Grounded Requirement Engineering: An Approach to Use Case Driven Requirements Engineering, Journals of System and Software Vol. 117, Pp 645-657, Germany, 2016.

[3] Suranjan Chakraborty, Christoph Rosenkranz, Josh Dehlinger, "A Grounded Theoretical and Linguistic Analysis Approach for Non-Functional Requirements Analysis", Thirty Third International Conference on Information Systems, Orlando, 2012.

[4] H.M. Sneed. Risks involved in reengineering pro je ts. In WCRE'99 (Working Conference on Reverse Engineering), pages 204-211, Atlanta, GA, USA, 1999.

[5] Fran k Fleurey, Benoit Baudry, Alain Ni olas, Erwan Breton, and Jean Mar Jézéquel " Model-driven engineering for software migration in a large industrial on text". In Proceedings of MODELS/UML'2007, LNCS, pages 482-497, Nashville, USA, October 2007. Springer.

[6] L. Erlikh, "Leveraging legacy system dollars for e-business", IEEE, IT Professional, Volume: 2, Issue: 3, May/Jun 2000.

[7] Dr. Kiran Kumar J et al "An Approach to Cost Effective Regression Testing in Black-Box Testing Environment", IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 1, ISSN (Online): 1694-0814,May 2011.

[8] Prabhakar K. et al, "A Model Driven Approach to Regression Testing of Reusable Software", International Conference on

Artificial Intelligence & Cognitive Computing, AICC-18., 02-03 Feb 2018, Hyderabad, India, 2018.
[9] Adtha Lawanna, "test case design based technique for the improvement of test case selection in software maintenance", IEEE Xplore: Electronic ISBN: 978-4-907764-50-0, 21 November 2016.
[10] M. Gopi Chand et al, "Four Layered Approach to Non-Functional Requirements Analysis" International Journal of Computer Science and Issues (IJCSI Volume 8, Issue 6, pp 371-379, November 2011.
[11] A.AnandaRao et al, "Layered Approach for performance requirements elicitation" International Journal of Electrical Electronics and Computer Systems(IJEECS),Volume 9,Issue1, pp 568-575, July 2012.
[12] Suresh Nageswaran, "Test Effort Estimation Using Use Case Points", Copyright(c) 2001, Cognizant Technology Solutions, Quality Week 2001, San Francisco, California, USA, June 2001.
[13] Jim Heumann, "Generating Test Cases from Use Cases" Rational edge, Copyright Rational Software 2001 | Privacy/Legal Information, 2001.

Authors Profile

PRABHAKAR KANDUKURI Pursuing Doctor of Philosophy (Ph. D.) in Computer Science & Engineering at JNT University Ananthapur, Anatapuramu, Andhra Pradesh, India. He received his M. Tech. Degree in Computer Science & Engineering from JNTUA College of Engineering, Ananthapuramu, Andhra Pradesh, India, He received his B. Tech. Degree in Computer Science & Engineering from Acharya Nagarjuna University, Guntur, Andhra Pradesh, and He received his Diploma in Computer Engineering (D.CM.E) from the State Board of Technical Education and Training, Hyderabad, erstwhile Andhra Pradesh.

He is an Associate professor of Computer Science & Engineering Department, Vardhaman college of Engineering, Hyderabad, India. His main research interest includes Software Engineering, data Science and Web Services. He published several papers in various international journals/ conferences. Member IAENG.

ANANDA RAO AKEPOGU received the B. Tech. Degree in computer science and engineering from University of Hyderabad, erstwhile Andhra Pradesh, India and the M. Tech. Degree in A.I & robotics from University of Hyderabad, erstwhile Andhra Pradesh, India. He received his Ph. D. Degree from Indian Institute of Technology Madras, Chennai, India.

He is a professor of Computer Science & Engineering Department and currently working as a Director, Research & Development, JNT University Anantapur, Anantapuramu, Andhra Pradesh, India.

Dr. Rao has published more than 120 publications in various national and international journals/conferences. He received the best research paper award for the paper titled: An approach to test case design for cost effective software testing," IAENG-International Conference on Software Engineering, Hong Kong, 2009. He received the best paper award: "Design and analysis of novel similarity measure for clustering and classification of high dimensional text documents," in Proc. 15th ACM-International Conference on Computer Systems and Technologies, pp. 1-8,

Dr. K. Venugopala Rao, received Ph.D. in Computer Science & Engineering from Osmania University, Hyderabad, erstwhile Andhra Pradesh. He received his M. Tech. Degree in Digital Systems & Computer from JNT University, Hyderabad, erstwhile Andhra Pradesh. He received his B. Tech. in Electronics and Communication Engineering from JNTU College of Engineering, Hyderabad, erstwhile Andhra Pradesh.

He is currently working as a Professor, Department of Computer science & Engineering. GNITS, Hyderabad. He received the Best Engineering Teacher award form ISTE. His main research interest includes software engineering and Computer networks, Artificial Intelligence.

