Challenges in Automatic Test Case Generation

[1]Dr K P YADAV, [2]Dr SAROJ PATEL,[3]TANNU ARORA
[1]Computer Science, [2]Applied Science, [3]Computer Science
[1]UPTU, Uttar Pardesh, [2,3]JNU,Jodhpur
India
[1]drkpyadav732@gmail.com,[2]drsarojpatel@gmail.com,[3]tannu.arora@gmail.com

Abstract: - Our research approach in this paper will be through survey process using a questionnaire related to automation of test case generation to analyze use of UML and its implication software testing effectiveness. A questionnaire is a means of eliciting the feelings, beliefs, experiences, perceptions, or attitudes of some sample of individuals. As a data collecting instrument, it could be structured or unstructured. A questionnaire is prepared as a data collecting tool from development firms. This paper summarizes a survey in the field of automated software testing. The main aim of this paper is to show different type of challenges faced by testing team during test case generation and also, UML techniques used for automation of software testing process and sources of automatic test case generation. One critical task in software testing is to generate test cases. Since, test case generation has become an optimization problem hence scope remains open to apply some more techniques to achieve better results. The analysis used in this paper shows that automation of test case generation takes place early in the life cycle of software i.e., requirement specification which results in reduced cost of entire development of software.

Key-Words: - Automated software testing; optimization problem; UML Techniques; automatic test case generation; optimization problem; requirement specification

1 Introduction

It is a blunder to assume that test automation is simply imprisoned and play again of a manual test process. In fact, automation is basically different from manual testing, there are entirely different issues and chance. And, even the best automation will never completely replace manual testing, because automation is about inevitability and users are inherently erratic[7]. So, use automation to verify what we expect, and use manual testing for what we don't. A common scenario in software testing is therefore that test data are generated, and a tester manually adds test oracles. As this is a difficult task, it is important to produce small yet representative test sets, and this representativeness is typically measured using code coverage[3].Test process must be well defined. A key thought is that we cannot automate a development that is not already well-defined. A fully manual process may not have the regulation or documentation necessary to hold up well-designed automation documentation[8]. The questionnaire is most frequently a very concise, preplanned set of questions designed to yield specific information to meet a particular need for research information about a pertinent topic. The research information is attained from respondents normally from a related interest area. Requirement Management plays a important and essential role in test case generation. The purpose of Requirements Management is to establish a common understanding between the customer and the software project of the customer's requirements that will be addressed by the software project. Requirements Management involves establishing and maintaining an agreement with the customer on the requirements for the software project. The agreement covers both the technical and nontechnical (e.g., delivery dates) requirements[9]. The agreement forms the basis for estimating, planning, performing, and tracking the software project's activities throughout the software life cycle. Whenever the system requirements allocated to software are changed, the affected software plans, work products, and activities are adjusted to remain consistent with the updated requirements.

2 Problem Statement

Testing is a destructive task in which the goal is to find relevant defects as early as possible. It requires automation to reduce cost and ensure high regression, thus delivering determined quality[4]. In this section, we will state our problems which

include some research questionnaire regarding automation of software testing process. The research questions aims at finding the efficient procedures for automatic test case generation in practice. Moreover, a test case generation process is defined to select more general test cases with variables and parameters that can be instantiated at testing execution time[1]

The questions are:

RQ1: What are the various UML Techniques used for Automated Test Case Generation?

RQ2: Which is the most widely used technique?

RQ3: What are the broad areas covered by these techniques?

RQ4: What type of testing has been done?

RQ5: What are the benefits of using various techniques?

RQ6: From where are these test cases generated?

2.1 Sample Design

Sample Element: - The basis unit, which needs to be examined through the study, are Software development firms.

Population: - The universe of the study will comprise of all the Managers/Executives Software Units located in Delhi & NCR

Method: - The approach of convenience sampling is used for survey.

3 Different Scenario and Their Challenges and Proposed Solution

(i) Product Requirements Specification (PRS) and functional specifications for various product features are the only inputs to the test requirements gathering process

Challenges

• Define clear and complete test requirements

•Manage changes to requirements citing references in the text of the abstract, type the corresponding number in square brackets as shown at the end of this sentence [1].

Proposed solution

• Arrange for a PRS presentation by the product management team

• Arrange for product feature presentations from development team

• Prepare traceability matrices

• Get the traceability matrices reviewed by development team for completeness and clarity

• In case of any requirement changes, make corresponding modifications to traceability matrices at both levels.

• Traceability matrices to serve as a starting point for the new testers deployed for a feature's testing.

In summary, collaborate with development team throughout the testing life cycle starting with test requirements gathering stage till product release.

(ii) Preparation of test plans was being carried out directly on the basis of feature functional specifications.

Challenges

- Functional gaps in the test plans
- Difficulty in review by the development team for large test plans

Proposed Solution

Have the reviewed/approved traceability matrices

(iii)The product to be functionally tested was complex one with a large feature set and a number of supported hardware-software configurations. On the other hand, the time/effort available for testing was limited.

Challenges

- How to utilize the available limited resources/time to provide optimum test

coverage to the product functionality while covering various supported configurations.
- To monitor the exact hardware-software configurations for the test environment.

Proposed Solution

- Prioritization of test cases in test plans.
- Preparation of a consolidated functional test coverage matrix for various test cycles to be executed.
- Preparation of a consolidated configurations coverage matrix for various test cycles.
- Utilize an extensive decision model while deciding on the test matrix for each cycle
- Prepare a detailed test matrix for each test cycle to monitor the details for test environment to a minute level and get the same reviewed by the development team before proceeding with the testing.

3 Results and Analysis

This section reflects the results related to the research question collected through survey.

RQ1- The key techniques found were: UML[5] diagrams, State chart diagrams, sequence diagrams, activity diagrams, class diagrams, collaboration diagrams as shown in Figure 1.

Fig 1 Automated Test Case Generation

RQ2- The most widely used techniques can be best analyzed from the graph given below It can be easily noted that activity diagram and sequence diagram are the majority extensively used approaches so far.

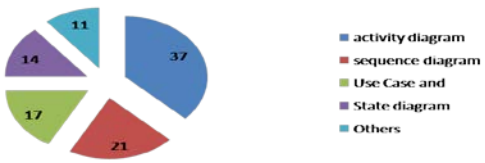

Fig 2 Most widely used Technique

RQ3- The broad areas covered by these techniques consist of Web applications, real time embedded systems, artificial intelligence planning, spreadsheets, and Object oriented systems, SOA interacting services, chip design etc.

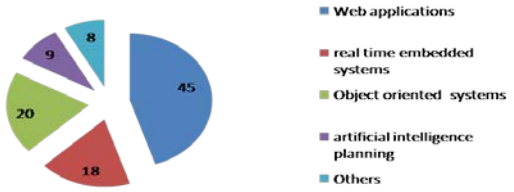

Fig 3 Area covered by UML Techniques

RQ4-Black box testing, white box testing, grey box testing, conformance testing and scenario based testing, system are the major types of testing performed.

Fig 4 Testing Type used

RQ5- The benefits of using various techniques are performing class level testing. Activity Diagrams can represent both conditional and parallel activities.

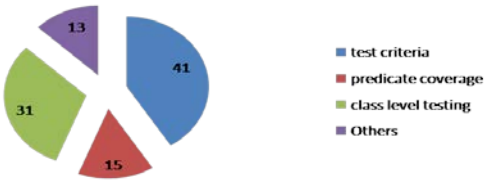

Fig 5 Benefits of Techniques used

RQ6- Test cases are produced either from requirements specification or source code, design specifications. From the chart, we can easily analyze that mostly test cases are generated from the requirement specifications.

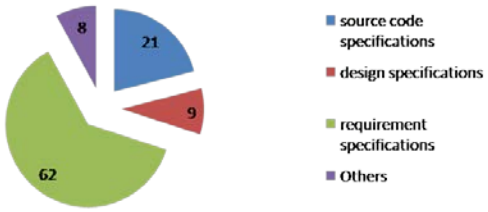

Fig 6 Sources of Test Case Generation

Table 1 summarizes the various results related to automated test case generation

Table 1

S.No	Research Problem	Results of Survey conducted	%age used
1.	UML Techniques used for Automated Test Case Generation	Class Diagrams	36
2.	Most widely used technique	Activity Diagram	37
3.	Broad areas covered	Web Applications	45
4.	Testing Type	White Box Testing	32
5.	Benefits of these techniques	Test Criteria	41
6.	Source of test case generation	Requirement Specifications	62

4 Conclusion

Certainly, many new ideas in software engineering were originally developed by trying them out on real development projects. In action research it is best to be flexible in the choice of methods used for data collection, and questions that demand answers constantly appear when going through the research cycle.AR is a process of social research where both outsiders and problem owners work together to solve problems and reach common goals. As AR promotes and focuses on the interaction between the researcher and the stakeholders. In this way it is necessary to obtain an insight in this project which otherwise would not be possible to conduct such a research when studying from distance. The above results and analysis we have collected shows that for automation of testing, UML diagrams basically activity diagrams and class diagrams are used. These are generally used in web-based applications. And primary source of test case generation is requirement gathering. This means that automation takes place early in the software life cycle.

References:

[1]Wilkerson L. Andrade, Patricia D.L. Machado, "Generating Test Cases for Real-Time Systems Based on Symbolic Models", IEEE Transactions on Software Engineering, vol.39, no. 9, pp. 1216-1229,(2013)

[2]Yann Moffett, Juergen Dingel, Alain Beaulieu, "Verifying Protocol Conformance Using Software Model Checking for the Model-Driven Development of Embedded Systems", IEEE Transactions on Software Engineering, vol.39, no. 9, pp. 1307-13256,(2013)

[3]Gordon Fraser,Andrea Arcuri,"Whole Test Suite Generation",IEEE Transactions on Software Engineering, vol.39, no. 2,pp.276-291, (2013)

[4]Macario Polo, Pedro Reales, Mario Piattini, Christof Ebert, "Test Automation", IEEE Software, vol.30, no. 1, pp. 84-89,(2013)

[5]UML-based Specification Environment (USE), http://sourceforge. net/apps/mediawiki/useocl index.php?title=The_UML-based_ Specification_Environment , (2012)

[6]http://www.edn.com/design/systems-design/4357539/The-top-five-software-testing-problems-and-how-to-avoid-them

[7]http://www.softwaretestpro.com/itemassets/4772/automatedtestinghandbook.pdf

[8]http://www.guru99.com/manual-testing.html

[9]http://www.itib.net/Conform/Models/CMM/tr25_l2a.html