A Proposed Methodologies on Multidimensional Software Testing Approach

[1]Manish Kumar, [2]Santosh Kumar Singh, [3]Dr. R.K. Dwivedi
[1] Assistant Professor, [2] Assistant Professor, [3] Associate Professor
[1] University Department of Computer Applications,
[1] Vinoba Bhave University, Hazaribag, Jharkhand

***Abstract* - This paper majorly focuses on the concept of multidimensional testing techniques for time related scenarios. Here we are proposing a multi dimensional technique where the combination of time and component for a particular risk category is considered for test execution. Here we are trying to combine the factors or components into two dimension pairs and each pair is analyzed against their risk based on their likelihood and impact of the time factors. In this paper, we are proposing a new test technique where a dimensional analysis is done on the test set. The proposed testing methodology will improve the effectiveness and efficiency for testing the time related scenarios.**

***Keywords* - time related testing, multidimensional testing, risk based classification.**

I. Introduction

Multidimensional Testing (MDT) provides an efficient, systematic approach for detecting a product's hardware and software incompatibilities. Multi-dimensional Testing can reduce both product development times and test escapes. Potential software-application customers can apply the pass-fail testing concepts of MDT to determine if an application software package that they are considering purchasing will later have compatibility issues in their work environment.

Software testing is the process of exercising or evaluating a system or system components to determine whether it satisfies the specification and executes in its intended world [1]. The time and effort required to have an acceptable coverage grows when complexity and size of the software grows. Testing time related scenarios are always challenging from various aspects like components, interactions, browsers, time zones and many more. Code based testing is quite difficult for testing the different time zones or any time related scenarios as it does not guarantee the acceptable satisfactory results. A tester cannot completely depend on the source code for selecting the test cases [2][3].

The classification tree method [4][5][6] by Groclitman and Grime is used for identifying the classifications and their associated classes from a specification. But here they assumed testers can construct the classification tree from their experience and expertise. There was no recommendation available for the actual process.

Pair-wise testing technique discussed in [7] cannot be used in a time related testing situation. It requires all the parameters to be repeated for the same number of times when a new factor is added to the list [8]. This can be ignored by adding 'don't care' values for the missing ones [9], but this unnecessarily creates extra tests for pair-wise testing.

II. Purpose of Multidimensional Testing? [10]

Organizations often discover incompatibilities and failures in self-developed software and new hardware introductions. During development and pilot testing, a product might perform satisfactory; however, design problems might initially not be detected until the device experiences a variety of customer-usage situations. These problems can occur even though the development test organization meticulously followed their design test plan.

Businesses often recognize that there is a need to improve the efficiency and effectiveness of their software and hardware development process so that they can do more in less time and have fewer of the above described development-test-problem escapes. To address this need, development test teams would like to be able to identify all system incompatibilities so that any identified issues can be timely resolved before product-customer distribution. Product test organizations would need to assess the acceptability of this product-design-logic pass/fail response in addition to other product requirements; e.g., speed-of-performance measurement.

When following standard compatibility testing, organizations might list all the potential incompatibilities and then test each one against the new software or hardware. This is called one-at-a-time testing. This testing approach might look good in theory, but can miss the evaluation of compatibility situations that later cause significant customer issues.

With computer software and hardware testing, often combinational issues of factor levels cause logic pass/fail failures. For example, one state of a factor may not work when exercised in unison with particular state conditions of two other factors; e.g., a certain state level of three factors causes a software failure.

Example of how Multidimensional Testing is used [9]

Smarter Solutions' Multidimensional Testing (MDT) provides a methodology that can significantly reduce or even eliminate compatibility escapes from a product-development process. This tool blends traditional statistical design of experiments

techniques with unique aspects of compatibility testing. The compatibility testing results are called "logic pass-fail" data. For this form of testing, hardware and software setting combinations are evaluated to determine if their function is satisfactory.
Example 1: Evaluating a new mobile phone handset
Example 2: Evaluating a new pc for compatibility with current hardware and networks

Features of Multidimensional Testing [9]
Multidimensional testing can be used to create test designs for any number of factors and levels considerations or assesses the test-coverage effectiveness of current test-case plans. The generated test cases can have a scripting so that those familiar with the product can readily understand or execute each test case scenario.
Multi-dimensional testing provides a platform for not only building upon team knowledge for test design creation but also offers a hierarchical testing methodology so that software/hardware problems can be detected and resolved early in the overall development cycle, as opposed to early-development-cycle-time issues being first detected in the product's final verification test. Inputs and outputs are in a form that is easy for developers to understand.

III. Current Scenario
Testing time-related scenarios have a lot of challenges involved based on the number of components, interactions, geographical time zones and other factors. There will always be situations where many factors might be ignored due to time constraints. In client server architecture, it is possible that the clients behaving differently for various time zones and the server might be tagged to a particular time zone.
Given below are some time-related scenarios:

- Log-in based triggers according to time zones. In such a situation, it might be necessary to restart the system to mark up to the new time zone by changing the start and end time of the transactions.
- Triggers based on the various date formats if the system is working on different date formats.
- Will the system be able re-initiate a task or save the current piece of task when the server or client goes done due to power shortage or any other calamities?
- Is the system intelligent enough to handle the daylight savings?
- Will the system respond to an impossible timing instance like a one second gap between start and stop time for transaction life cycle?

Importance of Risk in the approach:
Risk determines the priority of tests required to complete the time related scenario. Time is an important factor for testing time related scenarios. So, risk also should be considered as an important factor to consider which scenario (factors) require more priority based on its impact and likelihood.

Testing and verification in various time zones, fixed time based testing, component aging based testing are some of the situations where the proposed method can help.

IV. Algorithm
Here, we are proposing a new test technique where a dimensional analysis is done on the test set. Preconditions
1. The requirement documents and use cases should be analyzed and studied.
2. The requirements should be categorized to identify the time related scenarios..
3. The components and their time dependency can be found from the available system requirement documents (eg: BRD, SRS etc).
4. Verify whether any client-server related timing scenarios from the use cases has to be tested in various environments.

Multidimensional Time Related Testing Technique.
5. Map the system components to one dimension based on their time related behavior.

5.1 To illustrate on how to map the components, let's consider a simple software application containing two components A and B (Table 1).

Component A	Component B

Table 1: Components.

5.2 Not every component or combination of components has the same time zone or time dependency in fulfilling its end result (actual results). Let's take the time parameter (multiple time zones) and its values.

- Parameter T_A has values T_1 and T_2.
- Parameter T_B has values T_3 and T_4.

5.3 For each component or combination of components (interactions) there might be one or more time related test to be performed. Now let's map a component to the corresponding time parameter to be tested. The time is mapped to a different dimension as it involves a number of cases and time zones.

In this example the system time dependency can be on both components A and B or either of these components (Table 2)

Component A	Component B
Component A	
	Component B

Table 2: Component distribution

Let's map the components to their corresponding time test (data Table 3).

Component A	Component B	T_1	T_2
Component A		T_1	T_2
	Component B	T_1	T_2
Component A	Component B	T_3	T_4
Component A		T_3	T_4
	Component B	T_3	T_4

Table 3: Component with the time related dependency.

5.4 The risk associated for each time related component test case row has to be calculated for finding the priority. Risk is assessed from the likelihood and impact of each test roe in the business. The likelihood is the probability of occurrence of the test scenario in the entire business and impact determines the effect or impression of the corresponding the test scenario for the business.

5.5 For each row based on the business value of the time component (high and low relevance) and risk associated for the test row, a priority reference number will be assigned to the each row. Here the

5.6 priority reference strategy is illustrated as follows (Table 4,5):

Time Zone or Time Component Value	High relevance	5
	Low Relevance	0

Table 4: Time Relevance mapping

Priority	High	10
	Medium	5
	Low	0

Table 5: Risk related mappiing

6. The test rows can fall under any of the above three priority numbers (0, 5, 10). The ones with the highest number are to be considered ba sed on the test plan and strategy. The priority reference strategy helps to categorize the test execution.

V. Actual Results

We have created a three dimensional time related testing technique for one of the projects. The testing required an in-depth time related sc enario testing based on risk. The project had a lot of time zones and time related updates to be done real time. We have used an in house tool for generating the dimensions for generating the test rows. Multiple updates by the application which is time driven was tested in multiple systems using a perfor mance driven approach using the proposed approach.

Fig 1: Snapshot of the multidimensional time related testing scenario.

In the above figure (Fig1) the risk factor for each time factor mapped to the component is given. The testing technique compared to the existing techniques was more efficient and gave us better understanding of the right test conditions which cou ld be mapped directly to the test rows. The testing was more easier as the tester had more understanding what was to be tested and when the testing should be taken forward.

VI. Conclusion

The proposed testing technique can be applied to any mission critical scenario testing as it gives a complete idea to the tester regarding the testing. This technique provides an in-depth test scenario analysis and test execution which in turn provides an effective and efficient testing. Depending on the safety integrity levels, the system might require multiple levels of testing like component, integration, system and security requirement testing. Ti me related scenarios in various domains usually share a higher risk. Our empirical result shows that multidimensional testing strategy for time related scen arios gives a more understanding and clear idea to the tester about the factors involved and the tester can accommodate any dynamic changes based on the scenarios. In addition the tester has the flexibility to select the test case rows based on their priority ref rence number. Also the test execution can be directly aligned to the test plan and strategy. The testing technique can be easily adapted to any feature related scenario and to reuse existing test sets when there is any modification to the current system due to change in input parameters. Thus this approach provides a broader view and in-depth test coverage for time related scenario. In this approach the tester and the testing stakeholder are aware on the exact combination o f test cases (with all their interactions and factors), c ycle in which each test row should be executed and prioritizing the testing efforts.

References

[1] Naresh Chauhan, “Software Testing: Principles and Practices”, Oxford Universit y Press, 2010
[2] A.Pretschner, J.Philipps, “Methodological Issues in Model-Based Testing”, [29], 2005, pp. 281-291.
[3] J.Philipps, A.Pretschner, O.Slotosch, E.Aiglstorfer, S.Kriebel, K.Scholl, “Model Based test generation for smart cards”, Proc 8th Intl. Workshop on Formal Method for Industrial Critical System Transfer 5 (2-3), 2004, pp – 1095-1106.
[4] M.Grotchtmann, K.Grimm, “ Classification Trees for partition testing”, Software testing, verification and Reliability, 3(2), 1993, pp. 63 -82.
[5] M.Grotchtmann, K.Grimm, J.Wegener, “Test Case Design using Classification Tree and Classification Tree Designer CTE”, Proc. Of the 8th International quality week (QW’ 95).
[6] H.Singh, M.Conrad, S.Sadi ghipour, “Test Case Design based on Z and Classification Tree”, Proc. On first Int’l IEEE Conference on Formal Engineering Methods, pp. 81-90.
[7] D.M Cohen, S.R. Dalal, M.L Fredman and GC Patton, “The Combinatorial Design Approach to Automate Test Generation” IEEE Software, vol. 13, no. 5, pp 83-89, Sept. 1996.
[8] R. Mandl, “Orthogonal Latin Squares: An application of Experimental Design to Compiler Testing”, Comm, ACM, vol. 28, no. 10, pp. 1054-1058, Oct. 1985.
[9] A.W.Williams and R.L Probert, “A Practical Strategy for testing Par-wise Coverage for Network interfaces”, Proc. IEEE Int’l Symp, Software Reliability Eng. Pp 246-254, 1996.