

Requirement and Risk Based Testing

A new approach to ensure user's satisfaction

Luis Cavalheiro

Senior Consultant - AtYourSide Consulting
luis.cavalheiro@atyoursideconsulting.com

Abstract - *This document provides an overview of the Requirement's Risk Based Testing process and an overview of Automated Test Designer (ATD), the tool that supports this process. The intended audience for this paper is project managers, development managers, developers, test managers and test practitioners who are interested in understanding requirements-based testing and how it can be applied to their organization.*

Keywords – Model Based Testing, MBT, Requirements Based Testing, Risk Based Testing, RBT, Automated Test Designer, ATD, Test Case Generation, Test Data Generation, Test Script Generation.

1. INTRODUCTION

Model-based testing is a new and evolving technique for generating a suite of test cases from requirements. Models are used to understand, specify and develop systems in many disciplines. From DNA and gene research to the development of the latest fighter aircraft, models are used to promote understanding and provide a reusable framework for product development. In the software engineering process, models are now accepted as part of a modern object oriented analysis and design approach by all of the major OO methodologies.

The objective of this paper is to present an approach for test case generation based on graphical model (Cause/Effect definitions tree) that describes the behaviour of the application under test (AUT). This approach is based on Automated Test Designer (ATD) tool, a unique tool for test case, data and script automation generation.

2. REQUIREMENT BASED TESTING

The requirements-based testing process is comprised of two phases: ambiguity reviews and cause-effect graphing. An ambiguity review is a technique for identifying ambiguities in requirements specification to improve the quality of those requirements. Cause-effect graphing is a test-case design modelling technique that derives the minimum number of test cases to cover 100 percent of the functional requirements.

This approach stabilizes the application interface definition early because the requirements for the user interface become well defined and are written in an unambiguous and testable manner. This allows the use of capture/playback tools sooner in the software development process.

3. SOFTWARE DEVELOPMENT PROCESSES

There are many software development methodologies. Each has its own characteristics and approaches, but most software development methodologies share the following four main activities:

- **Requirements engineering:** There is a description of what has to be delivered.
- **Design:** There is a description of how the requirements will be implemented.
- **Code:** The application is implemented from the requirements and the design.
- **Test:** The behaviour of the code is compared to the expected behaviour described by the requirements.

All these activities are part of the waterfall model and are performed sequentially by the presented order. The waterfall model does not work very well in practice for various reasons. If a defect is found after coding, there is a good deal of rework to correct the code, and possibly the design, test cases, and requirements as well. Testing often is a bottleneck activity and most of the times not even executed. This often raises several difficulties that lead to low software quality, over budget activities and dramatic delays.

3.1 MAIN REASONS WHY SOFTWARE FAIL

The Standish Group and other studies show the main top reasons why software projects fail are all related to requirements and their specifications. Inadequate requirements often are the reason for failing projects.

The Requirement Based Testing process addresses all these issues:

- It begins at the first phase of software development process (Requirements Engineering) where the correction of errors is the least costly.

- Is in this first phase also where the largest portion of bugs has their root cause.
- It addresses improving the quality of requirements.

One improvement of the waterfall model is to begin incorporating verification activities such as testing much earlier in the development process rather than waiting until coding is done to begin testing. Having the test team working alongside the development team leads to better requirements, a clear vision on resources and more realistic plans. But this is only possible with suitable tools.

4. AUTOMATED TEST DESIGNER

Automated Test Designer (ATD) is a unique tool for generating test cases based on requirements. ATD is an easy to use windows client / server tool designed to define Test Cases, Test Data and Automated Test Scripts based on functional requirements. Since these Test Cases are based on actual, documented requirements, the test team can be sure that they are testing 100% of the application's functionality.

ATD is composed by three main modules:

- **Test Case Generator (TCG):** advanced and rigorous Neural Network Optimization algorithm capable of generate a complete set of test cases to certify 100% of your requirement rules.
- **Test Data Generator (TDG):** after identifying your test data based on your requirement specification ATD generated all the test data needed to execute the generated test cases.
- **Test Script Generator (TSG):** automated test scripts improve the speed and the reliability of test cases execution.

4.1 REQUIREMENTS SPECIFICATION

This module provides the means for requirements rules to be specified. Requirements specifications are translated from natural language to formal language using Cause / Effect Definitions tree. Using the Cause / Effect Definitions tree, the project team can analyze every aspect of the requirements in ATD. ATD can identify several characteristics that testable requirements must comply with, such as unambiguous and deterministic. These situations must be corrected before proceeding.

At the same time Cause/Effect definitions tree is being created, the requirements specification is verified for accuracy. At that point, we are considering the requirements specification and not the requirement itself. The requirement is a need of the user. The requirement specification is our imperfect attempt at

documenting this need. Using a formal language like the one provided by Cause/Effect definition tree, we are contributing for an excellent requirements specification.

The Cause/Effect definition tree serves as input for ATD. The output will then be the generated test cases.

4.2 TEST CASE GENERATOR (TCG) MODULE

ATD uses an advanced and rigorous Neural Network Optimization algorithm in order to generate a complete set of test cases to certify 100% of your requirement rules.

ATD supports proven methods for reducing and prioritizing the number of test cases while also achieving high requirements coverage. ATD implements a proprietary N-Way Testing method based on Risk factor which extends the concept of *PairWise* Testing, so Quality Assurance Team can reduce the number of test cases accordingly to their objective.

Despite ATD generates the minimum test cases, this number can be greater than hundreds or even thousands test cases depending on the requirement being specified. Each Cause has a Risk attribute that allows ATD to generate the minimum number of test cases in order to cover the most important functional requirements of the AUT based on each case risk factor. With TCG module the minimum number of test cases is created in order to discover the maximum number of defects excelling this way the testing process.

These generation methods taking advantage of multi-server processing to dispatch the compilation of Test Case Generation Agents on other computers reducing dramatically the time required to process larger amounts of requirements.

4.3 TEST DATA IDENTIFICATION

After specified the requirements and generated all associated test cases you still need to execute one of the hardest tasks before start running the generated test suite. You need to identify and create / extract all the data needed to run the different test cases. This is a very complex and very time consuming task. ATD can help you doing this. The new ATD Test Data Generator (TDG) module allows you to identify your functional test data.

Using ATD you can easily identify data to use as valid or invalid input for causes as well data to use as output for effects in order to validate your expected results. ATD is a requirement based testing tool. That means

you identify the data for the generated test cases based on your requirement needs. This test data identification serves as input for ATD TDG module.

4.4 TEST DATA GENERATOR (TDG) MODULE

After identifying your test data based on your requirement specification you can start generating the test data needed to execute your test cases. The produced output will enable your testers to start executing test cases. All relevant information such as Actions, Expected Results and respective data to use as input and output are available in an easy to read tabular format.

4.5 TEST AUTOMATION

At this time test plan is complete with all relevant test cases and test data generated. Now it's time to start executing the test. Testing activities must be heavily automated to allow them to be executed quickly. The test case design approach (ATD TCG method) should produce the minimum number of test cases to reduce the amount of time needed to execute tests. In order to be efficient and not waste time in creating test scripts.

ATD provides the TSG module that allows automating the generated test cases. You start by identifying all user interface objects (GUI or not) that are part of the AUT. This work can be done even if the AUT it's not developed yet. This information will then be used as input for the TSG module.

4.6 TEST SCRIPT GENERATOR (TSG) MODULE

The biggest benefit of automated test tools is that they (if implemented properly) allow you to be more productive, testing your systems quicker without losing quality. With automated test scripts you will improve the speed and the reliability of your test cases execution. The ATD TSG module allows you to automatically generate test scripts for the main automation tools in the market, such as Mercury's WinRunner, Mercury's QuickTest, Compuware's TestPartner or IBM Rational Robot.

With this module your testers can focus on requirements specification rather than on test scripts creation and maintenance.

5. CONCLUSION

ATD's approach significantly decreases the amount of time required to design and build tests, reducing the overall testing effort.

ATD is designed to help project managers, QA and testing teams:

- Reducing time needed to produce Test Cases.
- Achieve 100% functional coverage with Test Cases.
- Reduce time and effort maintaining Test Cases.
- Manage project risk based on test strategy.
- Optimize Quality / Cost ratio.

Every time the requirements change, the tester just has to focus his work on updating the requirements specification, test data, and GUI objects the AUT is composed by. After provided this input ATD generates and automates all test cases.

ATD can provide project management, test and QA teams the right tool to achieve better quality, faster validation and fewer costs.

At the same time you can give effectiveness to your testing process by using a RBT methodology approach, you can give efficiency by automating all generated test cases. ATD approach using a requirement and risk based testing ensure user's satisfaction.