IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 3, Ver. III (May-Jun. 2014), PP 121-128
www.iosrjournals.org

Integrated Test Frame Work for defect free Embedded Product delivery

A.chandra suresh[1],Dr. Kvnm.Prasad[2]
[1](Department of ECE sviet, Engineering college,India)
[2](Jenco Department, Vidhyuth southa, India)

Abstract:*Now a days, the industries are looking for zero defect embedded products.Of late the companies are recalling the products due to the non functioning and performance of the products up to their expectations as per the design conditions and estimations. So many uncovered defects are highlighted even their operation which eventually triggers the products recall later times. Recently M/s Toyota recalled 60 lacks vehicles globally for non functioning wheel cable.With this we can get a question that "Are all delivered products are defect free?" Probably, answer would be Not. Even Though organizations spending lot off investment during product life cycle .*

Due to this embedded product margin being minimized to the manufacturing Industries and is going to be a non profit product.Hence there is immense need to deliver defect free product by the manufacturers .There is a need to maintain or adopt sophisticated/test Proven test process, test coverage techniques.This paper proposes an integrated test frame work which fills the gap among Test levels,process,Testcoverage,Prototype testing, Virtual testing model based testing etc,… to deliver defect free embedded product

Keywords: *Test process,proto type testing, test iterations, test coverage, test levels, virtual testing.*

I. Introduction

Defect free product shall be delivered in minimal time means productivity shall be better with better quality. Focus on productivity is highly important for organizations to minimize product cost. To improve productivity. Software testing provides a means to reduce errors, cut maintain and overall software costs. Numerous software development and testing methodologies, tools, process and techniques have emerged over the last few decades promising to give Quality product without defect. But defect free software is not possible without using standard test process .Now the organizations are using some test process ,that may or may not produce defect free product. In this paper we only concentrated on testing methods and challenges faced during testing and how can we design a standard test process.

Testing and test process is an important activity which normally accounts for more than thirty percent of the project life cycle. Many organizations do not give sufficient focus on test process. In this paper we discuss about challenges in testing and propose some research areas for testing. This paper tells how to design a sophisticated test process that will suitable for all embedded software testing..

II. TEST PROCESS

Testing is a process rather than a single activity. This process starts from test planning then designing test cases, preparing for execution and evaluating status till the test closure. So, we can divide the activities within the fundamental test process into the following basic steps.

- Test Planning and control
- Analysis and Design
- Levels of Testing
- Implementation
- Test Execution
- Evaluating exit criteria and Reporting
- Test Closure activities
- Test Metrics
- Test Measurements
- Test reporting

2.1 TEST PLANNING:Testing like any project should be driven by a plan ..One single test plan can be prepared to cover all phases and all teams or there can be separate plans for each phase or for each type of testing . For ex. there needs to be plans for unit testing ,integration testing, performance testing, and so on. They can all be part of a single plan or could be covered by multiple plans. In situations where there are multiple test plan, there

should be one test plan ,which covers the activities common for all plans. This is called Master test plan.The test plan acts as the anchor for the execution, tracking, and reporting of the entire testing projects covers

- What needs to be tested the scope of testing includes clear identification of what will be tested what will not be tested. Test plan determines the scope and risk and identify the objective.
- How the testing is going to be performed-breaking down the testing in to small and manageable tasks and identifying the strategies to be used for carrying out the tasks.
- What resources are needed for testing – computer as well as human resources. And implement the test strategy.
- To determine the required test resources like people, test environments an pcs.
- The time lines by which the testing activities will be performed.
- To schedule test analysis and design tasks, test implementation, execution and evaluation
- To determine the Exit criteriawe need to set criteria such as Coverage criteria

Coverage criteria are the percentage of statements in the software that must be executed during testing. This will help us track whether we are completing test activities correctly. They will show us which tasks and checks we must complete for a particular level of testing before we can say that testing is finished.

2..2 Test control: Test control**hasthefollowing major tasks**

- To measure and analyze the results of reviews and testing
- To monitor and document progress, test coverage and exit criteria
- To provide information on testing
- To initiate corrective actions
- To make decisions

2.3 **Test analysis And Test Design::**Test analysisandTest Designhas the following major tasks: To review the test basis**.** The test basis is the information we need in order to start the test analysis and create our own test cases. Basically it's a documentation on which test cases are based,such as requirements, design specifications, product risk analysis, architecture and interfaces. We can use the test basis documents to understand what the system should do once built.

- To identify test conditions.
- To design the tests.
- To evaluate testability of the requirements and system.
- To design the test environment set-up and identify and required infrastructure and tools.

2.4**Test implementation**: Test implementation has the following tasks,

- To develop and prioritize our test cases by using techniques and create **test data** for those tests. In order to test a software application you need to enter some data for testing most of the features. Any such specifically identified data which is used in tests is known as test data
- We also write some instructions for carrying out the tests which is known as **test procedures**
- We may also need to automate some tests using **test harness** and automated tests scripts
- A test harness is a collection of software and test data for testing a program unit by running it under different conditions and monitoring its behavior and outputs.
- To create test suites from the test cases for efficient test execution.Test suite is a collection of test cases that are used to test a software program to show that it has some specified set of behaviors'.A test suite often contains detailed instructions and information for each collection of test cases on the system configuration to be used during testing. Test suites are used to group similar test cases together
- To implement and verify the environment.

2.5 Test Execution:following major task

- To execute test suites and individual test cases following the test procedures
- To re-execute the tests that previously failed in order to confirm a fix. This is known as confirmation testing or re-testing**.**
- To log the outcome of the test execution and record the identities and versions of the software under tests. The **test log** is used for the audit trial.A test log is nothing but, what are the test cases that we executed, in what order we executed, who executed that test cases and what is the status of the test case (pass/fail).These descriptions are documented and called as test log.
- To Compare actual results with expected results
- Where there are differences between actual and expected results, it report discrepancies as Incidents

2.6**Evaluating Exit criteria and Reporting:** Based on the risk assessment of the project we will set the criteria for each test level against which we will measure the "enough testing". These criteria vary from project to project and are known as exit criteria. Exit criteria come into picture, when Maximum test cases are executed with certain pass percentage. Bug rate falls below certain level. When achieved the deadlines.

- To check the test logs against the exit criteria specified in test planning
- To assess if more test are needed or if the exit criteria specified should be changed
- To write a test summary report for stakeholders

2.7**Test Closure activities** :Test closure activities are done when software is delivered.

- When all the information has been gathered which are needed for the testing
- When a project is cancelled
- When some target is achieved
- When a maintenance release or update is done

2..8 Test Reporting: Testing requires constant communication between the test team and other teams(like Development teams).Test reporting is a means of achieving this communication. There are three types of test reports are used.

2.8.1 Test Incident Report: A test incidence report is a communication that happens through the testing cycle as and when defects are encountered. A test incident Report is nothing but an entry made in the defect repository. Each defect has a unique ID and this is used to identify the incident. The high impact test incidences are highlighted in the test summary.

2.8.2 Test cycle report: Test projects take places in units of test cycles .A test entails planning and running certain test in cycles, each cycle using a different build of the product. A test cycle rport at end of each cycle gives

- A summary of the activities carried out during that cycle.
- Defects that were uncovered during that cycle, based on their severity and impact;
- Progress from the previous cycle to the current cycle in terms of defects fixed.
- Outstanding defects that are yet to be fixed in this cycle.

2.8.3 Test Summary Report: The final step in a test cycle is to recommend the suitable of product for release. A report that summarizes the results of a test cycle is the test summary report. The summary reports should presents

- A summary of the activities carried out during the test cycle or phase.
- Variance of the activities carried out from the activities planned this includes
  - The tests that were planned to be run but could not be run(with reason)
  - Modifications to tests from what was in the original test specifications.
  - Additional tests that were not in plan(that were not in test original test plan)
  - Any other deviation from test plan.
- Summary of test results should include
  - Test that failed, with any root cause description
  - Severity of impact of the defects uncovered by the tests.
- Comprehensive assessment and recommendation for the release should include
  - Fit for Release assessment
  - Recommendation of release.

2.9Test Metrics: Testing is the penultimate phase before product release ,it is essential to measure the progress of testing and product quality .Tracking test progress and product quality can give a good idea about the release –whether it will be met on time with known quality .Measuring and producing metrics to determine the progress of testing in thus very important. Thus ,metrics are needed to know test case productivity and to estimate test completion. It is not testing alone that determines the date at which the product can be released. The number of days needed to fix all outstanding defects is another crucial data point.

Hence, metrics helps in estimating the total days needed for fixing defects. Once the time nedded for testing and time for defects fixing are known, the release date can be estimated. Testing and defect fixing are activities can be executed simultaneously.

Metrics are classified in to different types based on what they measure and what area they focus on. At very level metrics are classified into three categories.

2.9.1 Project metrics: A set of metrics that indicates how the project is planned and executed.

2.9.2 Progress metrics: A set of metrics that tracks how the different activities of the project are progressing. The activities include both development activities and testing activities. Progress metrics helps in finding put the status of test and are also good indicator of product quality.

2.9.3 Productivity metrics: A set of metrics that takes into account various productivity numbers that can be collected and used for planning and tracking testing activities. These metrics help in planning and estimating of testing activities.

III. Levels of Testing

There are different levels during the process of Testing. In this a brief description is provided about these levels. Levels of testing include the different methodologies that can be used while conducting Software Testing. Following are the main levels of Software Testing

- Functional Testing
- Non Functional Testing

3.1 Functional Testing**:**This is a type of black box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to conform to the functionality it was intended for. Functional Testing of the software is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. There are five steps that are involved when testing an application for functionality.

Step 1:	The determination of the functionality that the intended application is meant to perform.
Step 2:	The output based on the test data and the specifications of the application.
Step 3:	The output based on the test data and the specifications of the application
Step 4:	The writing of Test Scenarios and the execution of test cases.
Step 5:	The comparison of actual and expected results based on the executed test cases.

3.1.1 Unit Testing: This type of testing is performed by the developers before the setup is handed over to the testing team to formally execute the test cases. Unit testing is performed by the respective developers on the individual units of source code assigned areas. The developers use test data that is separate from the test data of the quality assurance team. The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality.

Limitations of Unit Testing*:* Testing cannot catch each and every bug in an application. It is impossible to evaluate every execution path in every software application. The same is the case with unit testing.There is a limit to the number of scenarios and test data that the developer can use to verify the source code. So after he has exhausted all options there is no choice but to stop unit testing and merge the code segment with other units

3.1.2 Integration Testing**:** The testing of combined parts of an application to determine if they function correctly together is Integration testing. There are two methods of doing Integration Testing Bottom-up Integration testing and Top Down Integration testing.

Bottom-up integration: This testing begins with unit testing, followed by tests of progressively higher-level combinations of units called modules or builds .

Top-Down integration**:** This testing, the highest-level modules are tested first and progressively lower-level modules are tested after that.

In a comprehensive software development environment, bottom-up testing is usually done first, followed by top-down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic those it will encounter in customers' computers, systems and network.

3.1.3System Testing: This is the next level in the testing and tests the system as a whole. Once all the components are integrated, the application as a whole is tested rigorously to see that it meets Quality Standards. This type of testing is performed by a specialized testing team. System testing is so important because of the following reasons

- System Testing is the first step in the Software Development Life Cycle, where the application is tested as a whole
- The application is tested thoroughly to verify that it meets the functional and technical specifications .
- The application is tested in an environment which is very close to production environment where the application is deployed.
- System testing unable us to test ,verify and validate both the business requirements as well as the application Architecture.

3.1.4 Regression Testing: Whenever a change in a software application is made it is quite possible that other areas within the application have been affected by this change. To verify that a fixed bug hasn't resulted in another functionality or business rule violation is Regression testing. The intent of Regression testing is to

ensure that a change, such as a bug fix did not result in another fault being uncovered in the application. Regression testing is so important because of the following reasons:

- Minimize the gaps in testing when an application with changes made has to be tested**.** Testing the new changes to verify that the change made did not affect any other area of the application .
- Mitigates Riskswhen regression testing is performed on the application.
- Test coverage is increased without compromising timelines.
- Increase speed to market the product

3.1.5 Acceptance Testing **:** This is arguably the most importance type of testing as it is conducted by the Quality Assurance Team who will gauge whether the application meets the intended specifications and satisfies the client.s requirements. The QA team will have a set of pre written scenarios and Test Cases that will be used to test the application. More ideas will be shared about the application and more tests can be performed on it to gauge its accuracy and the reasons why the project was initiated. Acceptance tests are not only intended to point out simple spelling mistakes, cosmetic errors or Interface gaps, but also to point out any bugs in the application that will result in system crashers or major errors in the application. By performing acceptance tests on an application the testing team will deduce how the application will perform in production. There are also legal and contractual requirements for acceptance of the system.

3.1.6 Alpha Testing**:** This test is the first stage of testing and will be performed amongst the teams (developer and QA teams). Unit testing, integration testing and system testing when combined are known as alpha testing. During this phase, the following will be tested in the application.

- Spelling Mistakes
- Broken Links
- Cloudy Directions
- The Application will be tested on machines with the lowest specification to test loading times and any latency problems

3.1.7 Beta Testing: This test is performed after Alpha testing has been successfully performed. In beta testing a sample of the intended audience tests the application. Beta testing is also known as pre-release testing. Beta test versions of software are ideally distributed to a wide audience on the Web, partly to give the program a "real-world" test and partly to provide a preview of the next release. In this phase the audience will be testing the following.

- User will install, run the application and send their feedback.
- Typographical errors, confusing application flow, and even crashes.
- Getting the feedback, the project team can fix the problems before releasing the software to the actual users.
- The more issues you fix that solve real user problems, the higher the quality of your application will be.
- Having a higher-quality application when you release to the general public will increase customer satisfaction

3.2 Non –Functional Testing:This section is based upon the testing of the application from its non-functional attributes. Non-functional testing of Software involves testing the Software from the requirements which are non functional in nature related but important a well such as performance, security, user interface etc. Some of the important and commonly used non-functional testing types are mentioned as follows.

Performance Testing*:* It is mostly used to identify any bottlenecks or performance issues rather than finding the bugs in software. There are different causes which contribute in lowering the performance of software:

- Network delay.
- Client side processing.
- Database transaction processing.
- Load balancing between servers.
- Data rendering.

Performance testing is considered as one of the important and mandatory testing type in terms of following aspects:

- Speed (i.e. Response Time, data rendering and accessing)
- Capacity
- Stability
- Scalability

3.2.1 Load Testing **:**A process of testing the behavior of the Software by applying maximum load in terms of Software accessing and manipulating large input data. It can be done at both normal and peak load conditions. This type of testing identifies the maximum capacity of Software and its behavior at peak time**.**Most of the time, Load testing is performed with the help of automated tools such as Load Runner, AppLoader, IBM Rational Performance Tester, Apache JMeter, Silk Performer, Visual Studio Load Test etc. Virtual users (VUsers) are defined in the automated testing tool and the script is executed to verify the Load testing for the Software. The quantity of users can be increased or decreased concurrently or incrementally based upon the requirements.

3.2.2 Stress Testing **:**This testing type includes the testing of Software behavior under abnormal conditions. Taking away the resources, applying load beyond the actual load limit is Stress testing**.** The main intent is to test the Software by applying the load to the system and taking over the resources used by the Software to identify the breaking point. This testing can be performed by testing different scenarios such as:

- Shutdown or restart of Network ports randomly.
- Turning the database on or off.
- Running different processes that consume resources such as CPU, Memory, server etc

3.2.4 Security Testing**:**Security testing involves the testing of Software in order to identify any flaws ad gaps from security and vulnerability point of view. Following are the main aspects which Security testing should ensure. Confidentiality.

- Integrity.
- Authentication.
- Availability.
- Authorization.
- Non-repudiation.
- Software is secure against known and unknown vulnerabilities.
- Software data is secure.
- Software is according to all security regulations.
- Input checking and validation.
- SQL insertion attacks.
- Injection flaws.
- Session management issues.
- Cross-site scripting attacks.
- Buffer overflows vulnerabilities.
- Directory traversal attacks.
-

IV. Test Methodologies

Here , I have considered two testing methodologies and types :

1. White box testing
2. Black box testing

4.1 White box testing : White box testing takes into account the program code ,code structure, and internal design flow.In this testing the number defects come about because of incorrect transalation of requirements and design into program code..White box testing is classified into two kinds

1. Static testing
2. Structural Testing

4.1.1 Static Testing: Static Testing is a type of testing which requires only the source code of the product,not the binaries or executables. Static testing testing does not involve executing th programs on the computers but involves select people going through the code to find whether

i. The code works according to the functional requirement.

ii. The code has been written in accordance with the design developed earlier in the project life cycle.

iii. The code for any functionality has been missed out.

iv The cede handles errors properly.

4.2.2 Structural Testing: Structural testing takes into account the code,code structure ,internal design, and how they are coded.The Structural testing tests are actually run by computer on the built product. Structural testing covers following steps.

1. Statement coverage
2. Path coverage

3.Condition Coverage
4.Function Coverage

4.2Black box testing**:**No knowledge of internal design or code required. Tests are based on requirements and functionality

4.2.1 Black Box - Testing Technique
- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access
- Performance errors
- Initialization and termination errors

1.2. Black box
- Based on requirements and functionality
- Not based on any knowledge of internal design or code
- Covers all combined parts of a system
- Tests are data driven

V. Challenges in testing

The most important mechanism for defect free embedded products is selecting or adopting a standard test process. During the development of Defect free embedded products test process face a lot of challenges at different levels of testing. That are

- Lack of Qualitative test engineer
- Lack of Standard Test process
- Problems in Automation Testing
- Test Environment

5.1 Lack of Qualitative test Engineers: At the initial stage of project life cycle there is no separate person dedicated as test engineer. And there is no specific focus on Testing process of the respective product. In such case Quality of the Embedded product requires adopted testing method by test engineer. The test engineer should upgrade their knowledge on testing .And organizations provide such kind of environment. The test engineer never restricted to a particular testing .He should informative about the total functionality of the Embedded product.

5.2 Challenges in Automation testing**:**Automation Testing should not be viewed as a panacea for all problems nor should it be perceived as a quick-fix solution for for all the quality problem in a product. Automation tekes time and effort and pays off in the long run .However automation requires significant intial outplay of money as well as a steep learning curve for the test engineers before it can start paying off. Management should have patience and persist with automation. The main challenge here is because of the heavy front –loading of cost of test automation, management start to look for any early payback.

Even big organizations not aware of when to automate and how to automate the product testing..And more over the Automate testing is not suitable for medium and small level products. Writing the script for the Automation takes more time and ambiguous. And no efficient and effective test professionals for automation .Automation testing is not suitable for embedded products which change their requirements continuously. Lack of Guidelines for automated test frame work .Focus of the testing is only concentrates on test scipt execution rather than functionality of the product. Review mechanism is not effective in case of script review.

5.3 Lack of standard Test Process: Many of the organizations are not follows the sophisticated and standard test process and they are releasing the products with nominal testing. Hence they are failed to identify several defects in the product. This causes major recalling of the product .One organization develop one product and execute all test cases nominally and released in to market ,some times it may not identify uncovered defects and organizations felt that they test process is standard and defect free. But The test process is not standard in real. The test process framed or developed by the organization own, may not be suitable for different applications of that organization. Organizations are not willing to change their own test process because test engineers and test managers are very familiar with their own test process.

5.4 Test Environment: According to software development the test environment should grow equally. Other wise there may be huge gap among testing life cycle and software life cycle. If the test environment is not ready

at time of test execution the test engineers execute test cases on virtual machine .The execution of test cases is better on target machine compared to virtual machine.

VI. Conclusions

Conclusions and Work directions:

6.1 Servey on recalled products: Approach the organizations which have recalled products and collecting the product specifications, and test procedure after that doing the analysis from the collected data to identify origin or root cause of the defect which was unidentified by the organization test process. By doing repeatedly above procedure among various recalled products. We will get one systematic approach towards standard test process which gives zero defect embedded product.

6.2 Survey on Existing test Maturity models :

At this stage we simultaneously did an active effort to improve the way of testing gathering the information about existed test models doing the study on gathered data, while studying we came across a number of models for testing which aided in accessing as well improving the test process. Some of the available test models

Software testing Maturity model
Test process improvement
Test organization Maturity model
Testing assessment program

6.3 Mapping the Models and collected data:

By doing a comparative study among existed test models and collected data we can develop an integrated test frame model that is suitable for all embedded products.which gives zero defect products. By mapping the maturity models with collected data we can frame a integrated test frame process that gives zero defect embedded product.

Acknowledgements

Summarizing the quite broad and active field of software Testing research has been a tough challenge. While I remain the only responsible of imprecision's or omissions, there are lot of people whom I am indebted. First, with the aim of being as comprehensive and unbiased as possible, I asked several colleagues to send me both a statement of what they considered the topmost outstanding challenge faced by software testing research, and a reference to relevant work (indifferently other authors' or their own paper)that this paper could not miss to cite. Out of the many I invited, I warmly thank for contributing: Adi Narayana mallea **sr**. Project manager in Whirpool india. their contributions have been edited incorporated in the paper. And he gave valuable suggestions

REFERENCES

[1] L. Baresi and M. Young. Test oracles. Technical report,Dept. of Comp. and Information Science, Univ. of Oregon,2001. http://www.cs.uoregon.edu/m˜ ichal/pubs/oracles.html.
[2] E. Bayse, A. R. Cavalli, M. N´u˜nez, and F. Za¨ıdi. A passivetesting approach based on invariants: application to the wap.Computer Networks, 48(2):235–245, 2005.
[3] B. Beizer. Software Testing Techniques (2nd ed.). Van Nostrand Reinhold Co., New York, NY, USA, 1990.
[4] A. Belinfante, L. Frantzen, and C. Schallhart. Tools for test case generation. In [21].
[5] S. Berner, R.Weber, and R. Keller. Observations and lessons learned from automated testing. In Proc. 27th Int. Conf. onSw. Eng., pages 571–579. ACM, 2005.
[6] G. Bernot, M. C. Gaudel, and B. Marre. Software testing based on formal specifications: a theory and a tool. Softw.Eng. J., 6(6):387–405, 1991.
[7] A. Bertolino. ISSTA 2002 Panel: is ISSTA research relevant to industrial users? In Proc. ACM/SIGSOFT Int. Symp. onSoftware Testing and Analysis, pages 201–202. ACM Press, 2002.
[8] A. Bertolino and E. Marchetti. Software testing (chapt.5). In P. Bourque and R. Dupuis, editors, Guide to the Software Engineering Body of Knowledge SWEBOK, 2004 Version, pages 5–1–5–16. IEEE Computer Society, 2004.
[9] A. Bertolino, E. Marchetti, and H. Muccini. Introducing a reasonably complete and coherent approach for modelbased testing. Electr. Notes Theor. Comput. Sci., 116:85–97, 2005.
[10] A. Bertolino and A. Polini. The audition framework for testing web services interoperability. In Proc. ROMICRO'05, pages 134–142. IEEE, 2005.
[11] A. Bertolino, A. Polini, P. Inverardi, and H. Muccini. Towards anti-model-based testing. In Proc. DSN 2004 (Ext.abstract), pages 124–125, 2004.
[12] S. Biffl, A. Aurum, B. Boehm, H. Erdogmus, and P. Gruenbacher, editors.Value-Based Software Engineering.Springer-Verlag, Heidelberg, Germany, 2006.
[13] S. Biffl, R. Ramler, and P. Gruenbacher. Value-based management of software testing. In [12].
[14] R. V. Binder. Testing Object-Oriented Systems Models, Patterns,and Tools. Addison Wesley Longman, Inc., Reading,MA, 2000.