scintel
Technology for a Changing World

Test Suite Re-Engineering Model

1.0 ABSTRACT

This white paper entitled “Test Suite Re-Engineering Model” will provide an understanding as to why and when re-engineering for a test suite is needed. Today agile methodology is dominating the ever famous typical life cycle models, this paper describes a re-engineering model which is Project Model independent Suite and aimed to increase re-usability. The process flow and description of activities in each phase of the model have also been addressed in detail. The paper also discusses the quantifiable parameters to highlight the measurable benefits and thus aid in creating more realistic project Estimation. A few (best) practices have also been detailed in paper.

To support this, a real time case study has been discussed in sequential manner:

a) Challenge
b) Solution
c) Process Followed
d) Measurable Benefits

To support this, a real time case study has been discussed in sequential manner:

- When and why a test suite re- engineering would be needed
- How the re-engineering can be successfully accomplished without affecting the planned schedules of projects.
- Help the test engineers to aptly identify the test suites that may need re-engineering and redesign in order to keep promises on Estimated Schedule, Cost and Effort intact

2.0 RE-ENGINEERING – DOES YOUR TEST SUITE NEED ONE?

A test suite will need to be considered for a re-engineering when one or more of the following criteria are true:

- When there is no standard process for defining, organizing, managing, and documenting the testing activities
- When testing efforts are non-repeatable, non-reusable, and difficult to measure
- When test cases are redundant and repetitive
- When the test results and metrics captured are of little or no value in qualifying the product/application
- When the test suite coverage is inadequate
- When the most important of defects are not uncovered in the initial stages of testing
- When the test suite needs a lot of manual intervention during execution, or when there is scope for automation
- When the application technology becomes obsolete or undergoes change

The objectives of a typical test suite re-engineering and management project should be to:

- Assist in defining, managing, maintaining, and archiving test ware
- Aid in test execution and maintaining the results log over different test runs and builds
- Centralize all testing documentation, information, and access
- Enable test library and test case reuse where-ever applicable
- Improve test engineer's productivity

The test suite after re-engineering should have met the following criterion:

- Contain a stable core set of tests
- Remove tests that are redundant, repetitive or un-needed
- Achieve better testing flow
- Minimize execution time and user interaction
- Maintain and improve dynamic testing ability
- Improve maintainability and portability
- Enhance coverage readability
- Structure enhancements for maintaining parallel and independent test development

3.0 RE-ENGINEERING

3.1 Model

Re-engineering a test suite involves a fundamental rethinking and radical re-design of testing process to achieve improvements in critical measures of performance, such as cost, time, quality and customer satisfaction. The entry criterion for a typical test suite re-engineering project is an existing test suite. The exit criterion is a newly designed and implemented test suite.

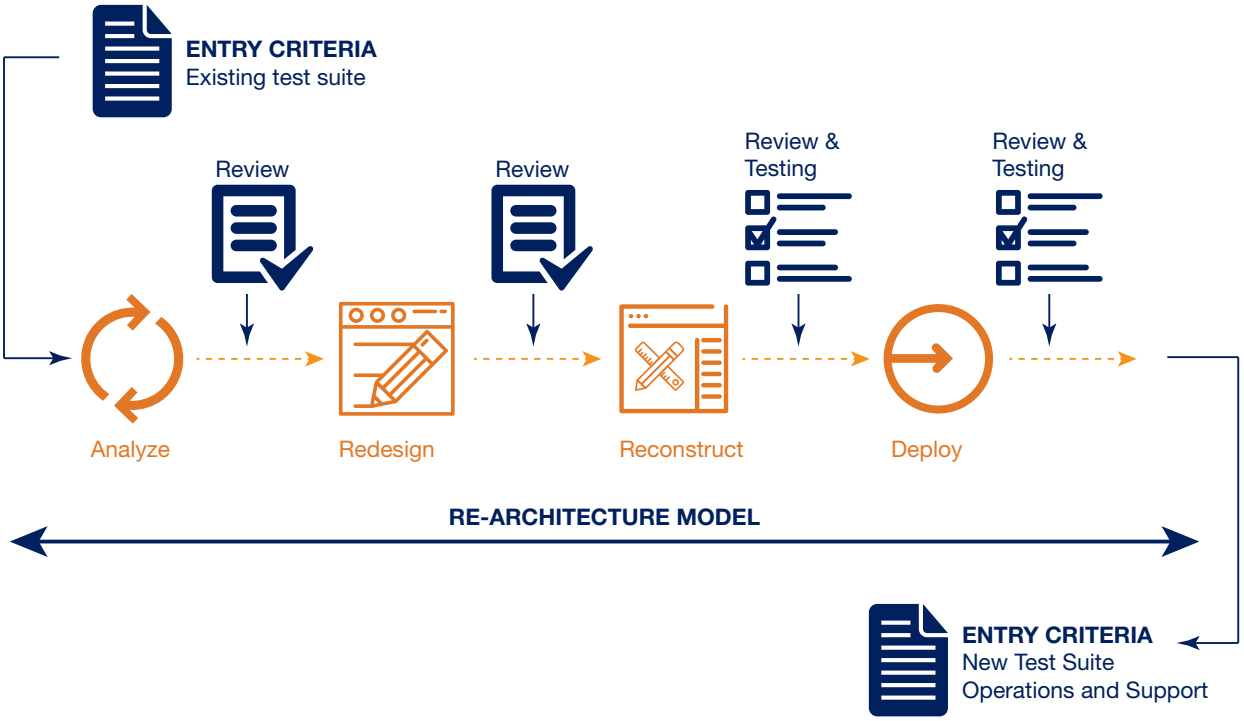

Fig: 1 – Test Suite Re-Engineering Model

This test suite re-engineering model (Fig: 1) defines a four phased approach:

- Phase 1 is the Analysis phase where the re-engineering requirements are gathered.
- Phase 2 is the Redesign phase where the existing test suite is redesigned.
- Phase 3 is the Reconstruct phase where the actual implementation is done.
- Phase 4 is the Deploy phase where the new system replaces the already existing system.

All processes in each phase will be monitored and reviewed from time to time by the review panel through out the project. The new system will be subject to test after the reconstruct and deploy phase. After the Deploy phase, the support/maintenance group takes over the maintenance of the system.

3.2 Process flow diagram

Fig: 2 – Process Flow Diagram

3.3 Process Description

Analysis Phase

This is the first stage of the re-engineering project. All the existing test suites, resources identified as per statement of work and the necessary documentation to execute the activity must be made available to the re-engineering team. Once you have identified an area for re-engineering, the next step is to segregate the existing test suite based on relevance. The tests with similar functionality need to be grouped under a defined directory structure. We also need to perform a gap analysis and make a recommendation for additional test creation or removal of redundant tests as applicable.

A new test plan or test document specification must be chalked out. The tests must be defined with respect to pre-conditions, events and post-conditions to achieve better testing flow. The test document specification may also include templates for procedures that are to be defined and test cases that are to be written. In the Analysis phase, feasibility of certain tasks, such as the extent of automation possible, possible use of six sigma tools such as voice-of-customer (VOC) and other methodologies; should also be addressed.

The re-engineering schedule should be planned such that it does not affect current project schedule until the new set of tests are ready. The re-engineering work also should not take too long and it has to be a short-term project or executed modularly.

In short, Analysis should help you assess:

- The health of the test methodology to determine the need for re-engineering
- How far or near you are from a benchmark
- The existing test suite performance

Redesign Phase

In the redesign phase, the test redesign should be accomplished. The test engineers will chalk out a new design for the test libraries, test cases and other test artifacts required. A description of each test library/test case has to be documented with its scope and objective in the test document specification. Any information that helps illustrate the purpose of a specific test such as requirements documents, functional specifications, etc should also be included.

During the redesign phase, the test engineers will also document detailed test execution steps and define the expected results for each step. This enables standardization and consistency across the testing team. It will also help in linking to the requirements specification to ensure traceability and test coverage. We can define the sequence in which test cases should be executed. This may be based on functional dependencies or some other factors like risk and other priority. Various six sigma tools, bidirectional traceability and Orthogonal Arrays can be made use of in the design stages.

Reconstruct Phase

The test cases, test libraries and other test artifacts are to be re-created during this phase. All the artifacts created have to adhere to the prescribed templates and coding standards. Unit testing has to be done in this phase to uncover any syntax and compilation errors in the test cases/test libraries that comprise the test suite and other cosmetic errors. These defects need to be tracked locally within the project or using an already available defect management tool and fixed.

Deploy Phase

The deploy phase is where the newly designed and constructed test suite replaces the old existing test suite. By the end of the deploy phase, the new test suite must have been completely implemented and ready to be executed. All the necessary infrastructure and set up has to be taken care of. It is at the end of the deploy phase, that the re-engineering project life cycle converges with the testing phase of the project which uses the test suite that is being subject to re-engineering.

If there are any defects that are generated due to errors in the test suite itself and not the product under test, these defects have to be logged and fixed as part of testing in the re-engineering project. It will not be considered as a defect in the product under test. Such defects have to be fixed immediately with high priority and testing of the original project should resume.

3.4 Sub-processes in the Re-Engineering Model

Review

The analysis, redesign, reconstruct and deploy phases should be reviewed and approved from time to time by the review panel which consists of the REAL stake-holders e.g. client and testing team members. Code reviews and Walkthroughs have to be conducted for the test cases and libraries. Once the test artifacts are created, it should be reviewed and approved by the review panel. This will verify the test cases developed by the team and improve them further if required before actual testing begins and the new test suite gets implemented.

Testing

Unit testing will be done during the reconstruct phase. However, after deployment, some amount of pilot testing (Exploratory testing) has to be done compulsorily to ensure that there are no errors generated by the re-architected test suite itself; and that the test framework and set up aligns with the original project's requirements and scope. When the new set of test cases are being executed and exercised in the testing phase of the original project, it supplements the testing phase for the newly architected test suite.

Operations and Support

Every test suite will need to be constantly updated and maintained. There may be new test cases which will need to be added with the number of features that may get added to a product. There may be new scenarios which get generated during exploratory testing, which may find its way into the core set of tests. Such tests may have to created/scripted and added to the test suite. Thus, it is highly recommended to do a gap analysis on a test suite periodically in order to enhance coverage from time to time and identify if it is time yet to consider a face lift for the test suite.

4.0 RE-ENGINEERING - MEASURABLE BENEFITS

Test case metrics compliment defect reports metrics and give a better view of product quality. A few measurable metrics that highlight the benefits of re-architecting a test suite are listed below:

- Time to execute tests – If re-engineering involves automation of the test suite, the time taken to execute the tests should be considerably lesser than the time taken to execute the tests manually. Proper historic data should be available to measure the time taken for execution.

- Number of engineers needed for testing - re-engineering involving automation of the test suite will considerably reduce the need for manual intervention. Thus, the effort in person hours and hence number of engineers needed to execute the test suite will also reduce.

- Tester productivity – With the time taken to execute tests and the number of person hours of efforts decreasing due to the automation involved in re-engineering, the productivity of each test engineer will increase.

- Test coverage – One of the most common reasons for re-engineering of test suites is to increase the test suite coverage. Any re-engineering should definitely include test coverage as one of the most important metrics to know the impact of re-design. Test coverage at the requirements and functional level can be tracked using a simple Requirements traceability matrix. Tracing should be done from the features in the requirements specification, all the way down to test scripts and test cases and vice-versa. When the requirements are managed well, traceability can be established from the source requirement to its lower level requirements and from the lower level requirements back to their source, thus enabling bi-directional traceability.

- Number of defects uncovered – re-engineering involving increased coverage will in-turn lead to the increase in the number of defects uncovered during the early phases of testing.

- Defect Slippage - If D1 be the total number of defects found during Test Phase 1 and D2 be the total number of 'new' defects found during a subsequent phase, Test Phase 2. With respect to Test Phase 1, Defect Slippage ratio would be = D2/ (D1 + D2). This ratio can be converted into a percentage by multiplying the value by 100. The percentage value would be useful in assessing the Test Effectiveness level. Depending on the complexity of the project, acceptable limits for Defect Slippage for each phase of development can be arrived upon by appropriate stakeholders.

- Quality of testing - Post release defect density / field – error rate, Severity of post release defects, are some of the metrics that can be used to measure the quality of testing before and after re-engineering.

- Product quality – A more efficient and effective test suite will help uncover more defects before product release; thereby enhancing the product quality.

5.0 CASE STUDY

Customer –
One of the major personal computer manufacturers

Environment –

Platform – Macintosh
Testing Tools – Client proprietary tools
Duration – 3 Months
Team Size - 17 (peak team size)

Requirement -

- To re-architect one of the test suites used for Point of Sale testing

Objectives -

- Create a stable core set of tests and introduce scheduling by stability of a product which could include the removal of tests that are redundant, repetitive or un-needed
- Improve maintainability and portability
- Achieve better testing flow
- Minimize execution time and user interaction
- Maintain and improve dynamic testing ability
- Introduce ability for on/off testing
- Structure enhancements for maintaining parallel and independent test development
- Introduce test document specification notation for documentation and enhancing coverage readability

Challenge – Create an expert talent pool and impart the application knowledge.

Solution – The team members were ramped up quickly to take on the project

Process followed-

1. **Knowledge sharing**
   - o Informal meetings
   - o Presentations
   - o Video Presentations
   - o E-mails

2. **Training**
   - o Technical presentations
   - o Study of relevant documents
   - o Interactions with the experts including web-ex sessions with experts at other locations.
   - o Understanding of the framework and processes of the existing test suite
   - o Queries via group E-mails

3. **Gathering Domain Knowledge**
   - o Happenings in the related domain were tracked and were updated to the entire team on a regular basis. A weekly newsletter was the success story of this initiative.
   - o Representatives were sent to attend seminars from other related teams to gain expertise

Challenge - Non-availability of a complete test plan or Test Document Specification (TDS)

Solution – Creation of a complete TDS

Process followed –

1. Gathering the requirements

- o VOC - The re-engineering team interacted with team that was using the existing test suite to gather the new set of requirements. This also helped in understanding the real user perceptions, what customer was aiming at.
- o All the Customer Pain areas were identified and broadcasted diligently with the team.

2. Assessing the existing test suite performance

- o Data collection - Metrics and data from the previous runs were gathered
- o Identification of redundant test cases – Creation of traceability matrix helped the team map the test cases to the respective modules and thereby identify the redundant tests.
- o Identification of areas to be enhanced – A root cause analysis for pain areas such as low defect findings and testing effort over run were done. Gap analysis between the existing data and the new set of requirements was conducted to identify the areas which needed improvement.

3. Designing the Test Document Specification (TDS)

- o The team introduced TDS notation for documentation and enhancing coverage readability. Directory structure of tests; test case header, labeling, naming conventions and execution procedure etc were documented.
- o The tests that needed to be re-written or created newly or that could be re-used were identified and categorized using the results of the traceability and gap analysis.
- o Each test was re-defined in terms of pre-conditions, events and post-conditions.
- o Each test was identified by a unique TDS label attached to it which would later help in achieving better testing flow and also in linking it to the TDS.
- o The best platform to execute each test case was also identified and documented – Actual Hardware with manual intervention vs. Test Hardware with no /minimal manual intervention.

Challenge – Minimizing manual intervention

Solution – New test cases created and old test cases modified to run on all supported platforms

Process followed

1. Created a library of Tool Command Language scripts which helped generate new test cases

o Created a framework using scripts to enable and disable a particular functionality testing
o Modified tests to run on the test hardware (which needs no manual intervention) as well as the actual engine.

Other practices followed

- **Reuse**
  - o Code reuse was encouraged
  - o Code was leveraged wherever possible
  - o Reusable components developed within the team were published as reusable assets within the ODC
  - o Reusable libraries and procedures developed within the team were published as reusable assets.
  - o Some ideas were adopted from areas where similar re-engineering had been done earlier
  - o A standard test tool used widely across projects was used

- **Tool Usage**
  - o Client proprietary tools were used in various activities
  - o Presentations on Tool usage were done

- Use of knowledge management and tools repository

- **Code reviews**
  - o Peer review of code
  - o Checklists were created and used for code review
  - o Table reviews and individual reviews were conducted very frequently
  - o The team also had walkthroughs for every library file and test case developed

- **Frequent client interactions and updates**
  - o The team interacted with the client via teleconferences and net meetings twice every week through out the project
  - o All this information was available to the team and to the client on a timely basis and there were no lapses in this regard

Measurable Benefits

- Execution time of the entire regression test suite was decreased by almost 17% due to reduced manual intervention when executed on test automation hardware.
- Manual testing effort for each run was reduced by about 20% when executed on test automation hardware.
- Feature testing time was reduced by about 10% since the tester could select and specify tests that need to be run using one single command and did not need to manually pick and run the tests individually.
- The number of defects that were found by execution of the new test suite was 30% higher than the number uncovered by the existing test suite.
- The core set of tests uncovered almost 40% of the high priority defects while running the basic functionality tests. The rest were uncovered while execution of the regression tests. Earlier these were not found until after the regression tests were run, thereby this reduced defect slippage. The post release severity defects were reduced to almost as little as 1 or 2.

6.0 CONCLUSION

Re-engineering a test suite provides a structured approach to testing, enables test process integration and improvement, and helps institutionalize process adherence. Encouraging the use of a standardized, structured approach to testing will improve the quality of the product released to the business units.

If the test ware is well preserved and maintained in a central repository and is easily accessible whenever required, it is reused throughout the lifecycle. The ability to reuse test ware will improve testing productivity. The value gained by using test management tools increases as the number of software builds grows. It is the iterative use of tools and test cases throughout the project lifecycle that yields the greatest cost and time benefits.

Reuses of test cases will help save in testing time, so that test coverage can increase. This directly contributes to improving the quality of the products. The early completion of testing activities also reduces the total project time and thus creating financial savings with respect to the initial budget allocated.

Test management can also help facilitate in building a knowledge repository and aiding knowledge transfer. Testers new to a project can become productive for the project in minimal ramp up time. The entire test approach for the project will be laid out for them in test management, making it much easier to insert new test cases and execute existing ones. Based on test reports available, one can make informed decisions about the quality of the application under development.

In summary, the re-engineering of test suite or a test management system at the right time helps improve quality, reduce testing cost and time, support decision making, and build a valuable knowledge repository.

ABOUT SCINTEL

Run by CIOs, Scintel partners with clients to understand their business environments, develop innovative applications and customized business processes, manage and enhance existing technology infrastructure and convert raw data into knowledge. Through a deep expertise in IT operations and a sustained focus on best practices, Scintel helps clients around the globe engineer a strategic IT direction, and then applies the knowledge and the expertise to execute with precision. We solve real business problems by developing real business relationships.

QA@scintel.com 1-866-scintel www.scintel.com