

## **Agility in Testing**

Padmaja Voleti and Manem Sekhar Rao  
Tata Consultancy Services Limited,  
Plot No 1, Survey No. 64/2, Software Units Layout  
Serilingampally Mandal, Madhapur  
Hyderabad - 500034, Andhra Pradesh  
Padmaja.voleti@tcs.com, manem.sekharrao@tcs.com

### **Abstract:**

This paper is based on the experience in the telecom industry particularly in the area of designing and testing of 3G platform product. The scope of testing involves testing fixes to trouble reports (also referred to as production issues), enhancements to existing features or a combination of both.

As with most projects, the schedule for completing the testing varies between two week cycles for major deliverables, which include 20 trouble reports and three day cycles for minor deliverables, which include 5 trouble reports.

Extremely stringent timelines, variance in the quantum of delivery, strong need for delivering a quality product places tremendous pressure on the testing teams. Over a period of time, we have evolved on a pattern of testing and believe our approach is matured and yields desired results.

The purpose of this paper is to highlight the combination of best practices, change in processes and tools evolved and deployed successful in the project.

Optimal use of automation in providing testing functions to overcome redundancy and effective usage of hardware, evolving a mechanism to conduct selective testing (Puzzle deployment), deploying effective technological solutions for knowledge management in areas such as faults information, FAQs to assist re-use of knowledge gained, lazy documentation strongly assist us in providing effective results from the testing process.

In addition, introducing flexibility in the process of deploying the testing services allowed us to evolve the process to further improve results. Leveraging experience from within the testing team and creating expert groups went on to add effectiveness further.

While on the one hand the listed best practices brought in tremendous value from the testing services, we have now initiated an “Early Fault Detection” process that is focused on pre-empting defects before they could affect the production version.

Details on each of the best practices listed above will be shared during the presentation of the paper.

**Keywords:**

Puzzle Deployment  
FST (Fault Slip Through)  
Known Faults and FAQs database  
Review process change  
Resource pool  
System Group  
Automation  
Test Co-ordinator

## **1. Puzzle deployment**

Testing of the 3G platform project involves functional testing and system testing of the major and minor deliveries. In a major delivery we have around 20 trouble reports that are fixed and the deliveries are planned starting with the dates of the content freeze till the delivery to customer. Minor deliveries contain fixes for three to four issues and are delivered, on demand by the customer. All deliveries have stringent timelines. The major delivery cycle consists of two weeks whereas for minor deliveries it is three days. In this duration integration of the code and testing needs to be done. Integration involves 30% of the effort there is a very tight schedule for the testing.

To combat the demanding timelines and the need to deliver a quality product the selective testing methodology is followed which is called as puzzle deployment. The fixes included in the delivery are first analyzed by

the test co-coordinator and the affected components and system areas are identified. For function testing only those test cases which cover the affected components are planned. For system testing pre defined high prioritized test cases are planned.

This methodology of intelligent planning of test cases helps to complete both the system and function testing in the desired time frame. This has helped reduce cost in terms of resources and timelines since test cases in a particular subsystem may not be executed if there are no corrections in that area.

Planning of test cases is done by the test co-coordinator in a tool. In this tool one can store logs and results of the test cases executed. The tool also has the facility to generate reports for the test cases planned for previous deliveries. Periodic analysis of the test cases planned is done by the test co-coordinator and if some test cases have not been executed since a long time they are planned for the next delivery.

Below is the illustration of the test cases executed for the deliveries based on the puzzle deployment:

	<b>SubSystem1</b>	<b>SubSystem2</b>	<b>SubSystem3</b>	<b>SubSystem4</b>
<b>Delivery1</b>	√	√	√	√
<b>Delivery2</b>	√	<b>X</b>	√	<b>X</b>
<b>Delivery3</b>	<b>X</b>	√	<b>X</b>	√
<b>Delivery4</b>	√	√	√	<b>X</b>

√ - Indicates the test cases are executed

X – Indicates the test cases are not executed

**Table 1 Example for Puzzle Deployment for various releases (deliveries)**

## **2. Fault Slip Through (FST) analysis**

Faults if found early are always less costly when compared to the faults found by the end customers. Fault Slip Through (FST) is a “fault which should have been found during system or function testing is actually found

by the customer”. FST analysis is a study driven in the team by the test coordinator on a monthly basis. The analysis includes consolidating all the faults which fall under this category. After consolidation, a study is done for each of the faults to identify the root cause why the fault was not detected during our testing. There may be various reasons attributed to the slip of the fault. It may be due to lack of appropriate hardware or due to missing test cases to detect the fault.

Relevant action is then taken depending on the cause identified. For missing test cases, new test cases are written to cover not only that fault but in a more broader way to cover related components. In this way the existing test suite is enhanced. The test cases written are reviewed internally and by the component owner. After review, the test cases are passed to the automation team to add them to the suite of the test cases. This completes the cycle of FST analysis which enhances the test suite.

For cases for which there is a lack of the desired hardware an order is placed for the required hardware and if approved based on the statistics provided new hardware is procured. These are then used in the nodes and tests are run to help detect faults early.

### **3. Known Faults and FAQs database**

While dealing with a project which involves many subsystems it is of utmost importance to have an inventory of the knowledge. Various tools are developed and deployed in the project to capture Known Faults, FAQ's and project specific documents.

For testing one needs to configure and set up the environment on the nodes on which tests are performed. The entire process involves lot of steps. So documents are very helpful in such cases. The tools are a repository for such kind of documents. There are also lots of common issues which are grouped under FAQs and stored in the tool.

Known faults are those errors that are seen either during testing or environment setup and are acceptable. They are acceptable either because

the fix will be provided late or related to hardware. The original scenario was to write a new Trouble Report (TR) for each error.

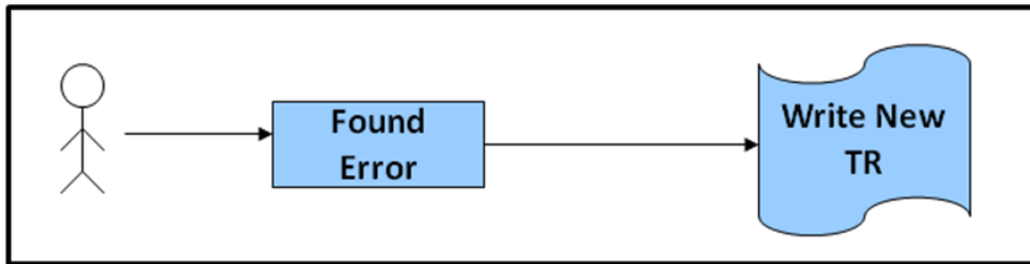


Figure 1. Situation without Known Faults Database

But now one refers to the database before a new TR is written.

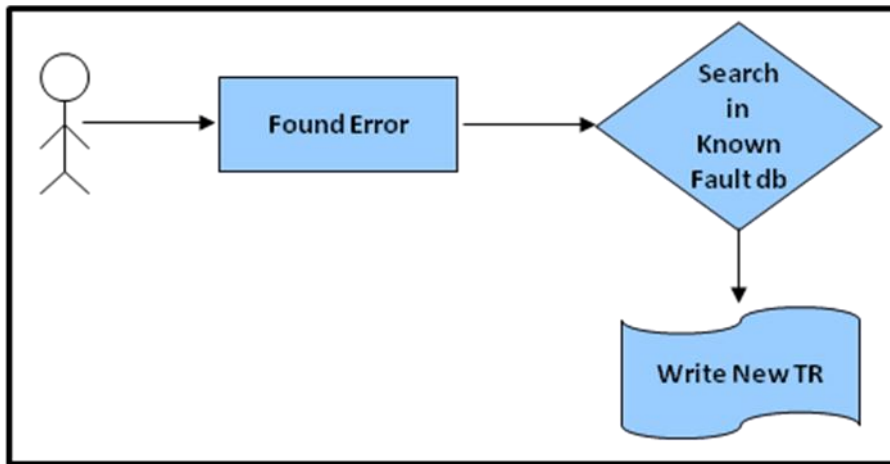


Figure 2. Situation with Known Faults Database

These are segregated based on the subsystem. These tools have an advantage of being a common repository which are useful to both design and test team. The tool provides a search mechanism to retrieve the required information. Hence these tools are the reference point.

## **4. Process change**

### ***4.1. Review strategy***

The testing process in the project goes in three phases. The build contains the trouble report fixes and new enhancements. In the first stage of testing, the verification of the trouble reports and enhancements is done. This is the crucial part of any release made from the project to the client.

As described earlier the entire application is sub divided into different subsystems. For each subsystem functional tests will be executed at the second stage. If any deviation is found in the testing, then a trouble report will be raised for the same deviation.

After the functional testing is completed for all components, regression test at the application level will be performed. This is the third and last level of testing process performed. Trouble report will be raised for any deviations found during this testing.

As shown in the figure below, normally reviews are done for all functional testing, trouble report verification and regression testing just before the delivery. This gives less time for rework in case of any faults found during the reviews.

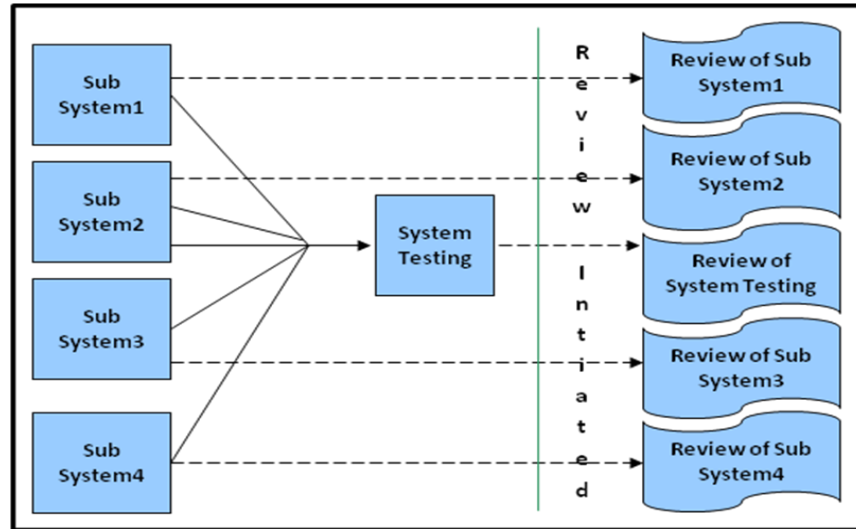


Figure 3. Traditional Review Process

But now reviews are performed as and when the testing for each subsystem is completed. This gives more lead time. It has really proved to be helpful based on past experiences. In case there is a rework for functional testing the testers can redo the tests without slipping the delivery schedule since the mistake is found early.

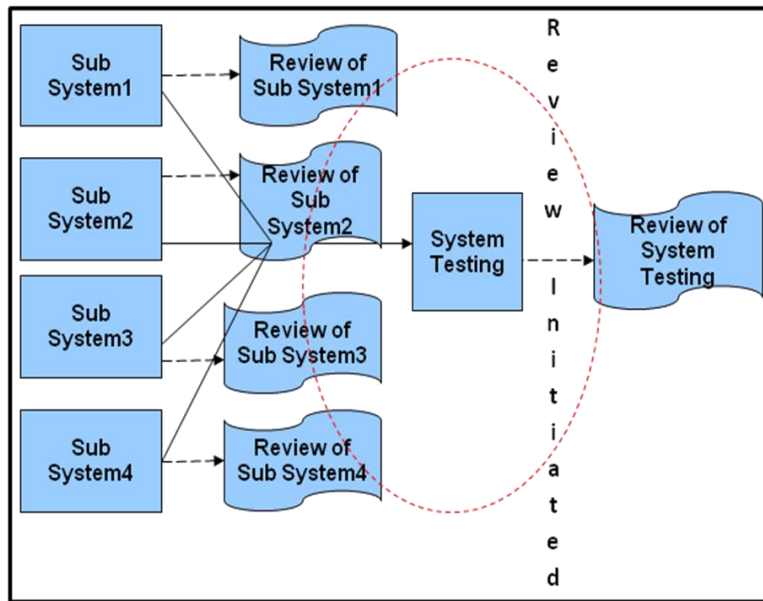


Figure 4. Adapted Review Process

#### 4.2. Resource Pool

The application being tested is very complex so the entire application has been divided into different subsystems. There are different tracks for the application development. Customers at the live site will be at different versions of application in different tracks. Earlier there was one dedicated function tester for one subsystem for each track.

But a resource pool concept is now implemented due to which all function testers are grouped into one pool. All testers in this pool will be aware of the testing methodologies of all tracks. This gives flexibility of working on more deliveries in parallel. This is in contrast to the dedicated tester.

The request for any track and subsystem is now redirected to the pool instead of a particular individual. This has benefited the project in many areas like time sharing, task sharing and back up resource planning.

## 5. System Test Group

The project deals with the maintenance and development of a product which comprises a lot of subsystems. Testing of each of the subsystems needs to be done in various tracks. Each subsystem has many components. As the product becomes more stabilized, the problems tend to be more complex in nature and can span multiple sub-systems. In such a scenario, having test cases particular to one or more components or at sub-system level does not help in testing the solution. The test cases need to be enhanced and improved to span across multiple subsystems.

To resolve this issue a virtual group known as the system group which has representatives of testers from each subsystem who are experts in the subsystem is desirable. They should meet on a regular/need basis to discuss the issues and problems found and share their knowledge and experiences.

This is depicted in the figure below:

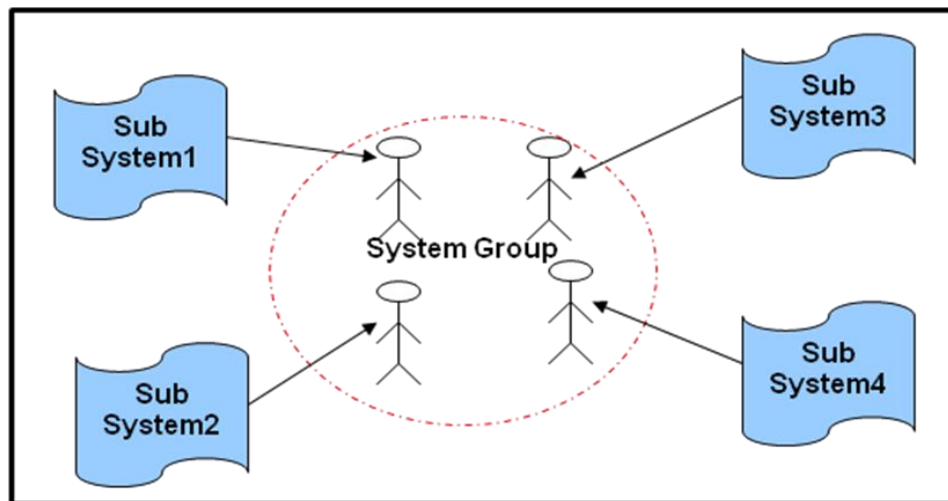


Figure 5. System Group

This group will act as a taskforce and drive the issue and give suggestions for the resolution. This group is the first point of contact in case of any verification, stopping or conflicting issues. This initiative has been

deployed in the design team and we propose to have it in the testing team also.

## **6. Automation**

If manual effort is reduced in the testing process it is always a boon to any project. Almost 90% of the functional and regression test suite are automated using Tcl/Tk.

Due to automation there has been a reduction in the manual effort and also the probability of the human errors that may creep in. The testing is repetitive in nature and hence it is of utmost importance that the suite is automated. Due to automation there is efficient usage of hardware during the week end and overnight. The suite is initiated during the week end and overnight and the logs are analyzed the next day. This has been instrumental in reducing the time for the completion of the testing for each delivery. This has given the edge of reduced timelines for the testing cycle.

## **Conclusion:**

Changes done to processes, effective use of tools and developing methodologies in the testing process has given the edge of delivering with quality and within the time line. This is how agility in the testing has been demonstrated.

## Citations

- **Books**

Perry, William: 2000. Effective methods for Software Testing. : John Wiley & Sons,Inc.

Lewis,E.,William: 2005. Software Testing & Continuous Quality Improvement: New York:Auerbach Publications.

Desikan,Srinivasan & Gopalswamy Ramesh:2007.Software Testing Principles & Practices: New Delhi:Dorling Kindersley Pvt Ltd.

- **Book Chapters**

Perry, William: 2000. Establishing a Software Testing Methodology (pp 63-64). John Wiley & Sons,Inc.

Lewis,E.,William: 2005. Test Project Management(pp 250-251):New York:Auerbach Publications.

Desikan,Srinivasan & Gopalswamy Ramesh:2007.Regression Testing: Best Practices in Regression Testing: (pp 206- 207):New Delhi:Dorling Kindersley Pvt Ltd.

## References

[1] Perry, William: 2000. Effective methods for Software Testing. : John Wiley & Sons,Inc,63-64.

[2] Lewis,E.,William: 2005. Software Testing & Continuous Quality Improvement: New York:Auerbach Publications, 250 – 251

[3] Desikan,Srinivasan & Gopalswamy Ramesh:2007.Software Testing Principles & Practices: New Delhi:Dorling Kindersley Pvt Ltd, 206 – 207.

[4] Test2008: 2008: Conference Proceeding Template.  
<http://test2008.in/icsd-template.doc>