

Project – TestOut

“Test out your Testing”

System Test Process **– Best practice for** **Business Critical Applications**

Prepared By:
Naveen Lakkur

Version 1.0
Bangalore, India
January 16, 2003

TABLE OF CONTENT

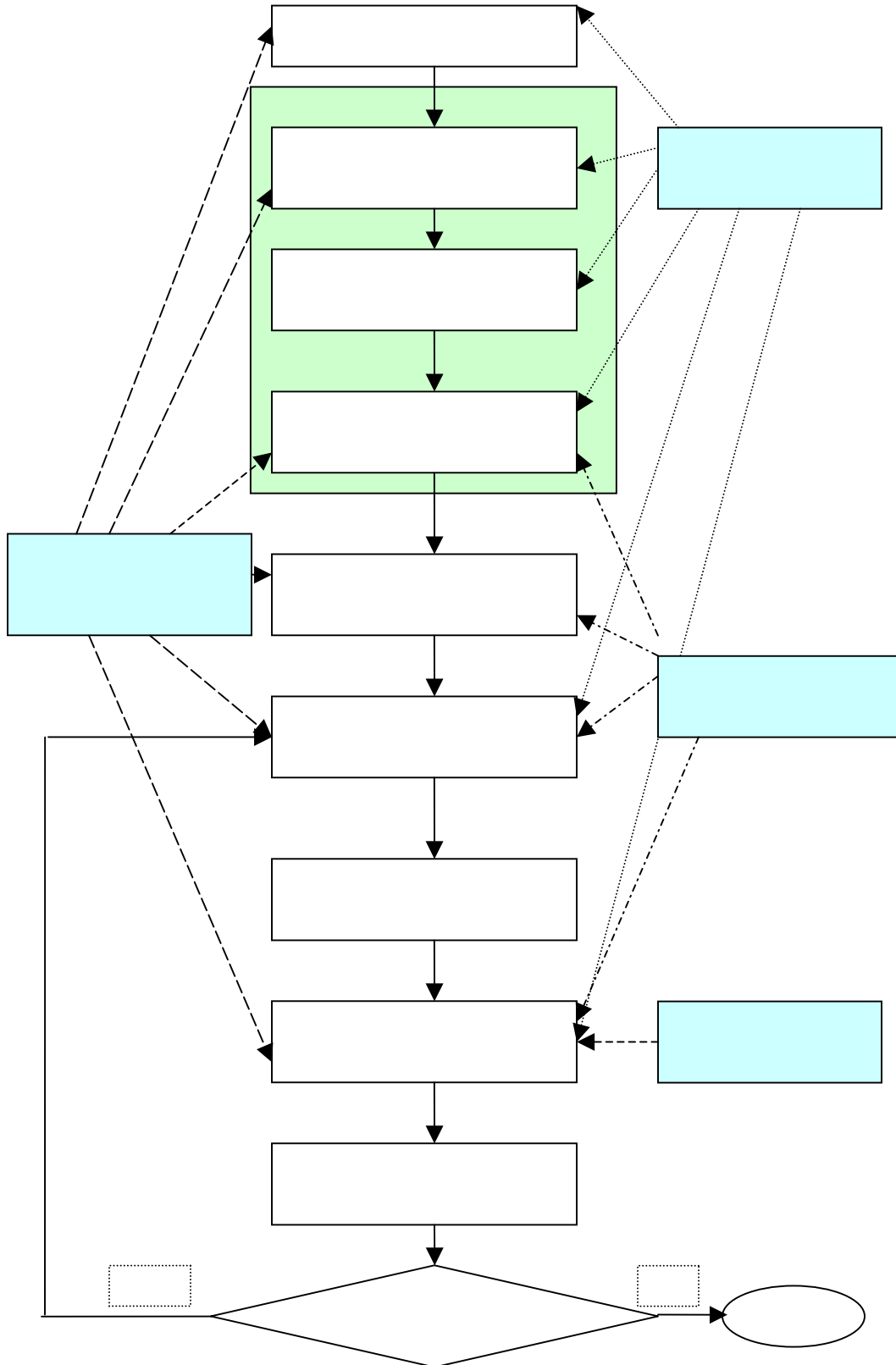
1.0	INTRODUCTION.....	3
2.0	STRATEGY.....	4
3.0	PLANNING	8
4.0	ENVIRONMENT SETUP.....	10
5.0	EXECUTION	12
6.0	DEFECT ASSIGNMENT & FIXING	14
7.0	DEFECT DATA ANALYSIS.....	16
8.0	DEFECT RESOLUTION VERIFICATION	18
9.0	APPENDIX A – SAMPLE DASHBOARD SNAPSHOT	20

1.0 Introduction

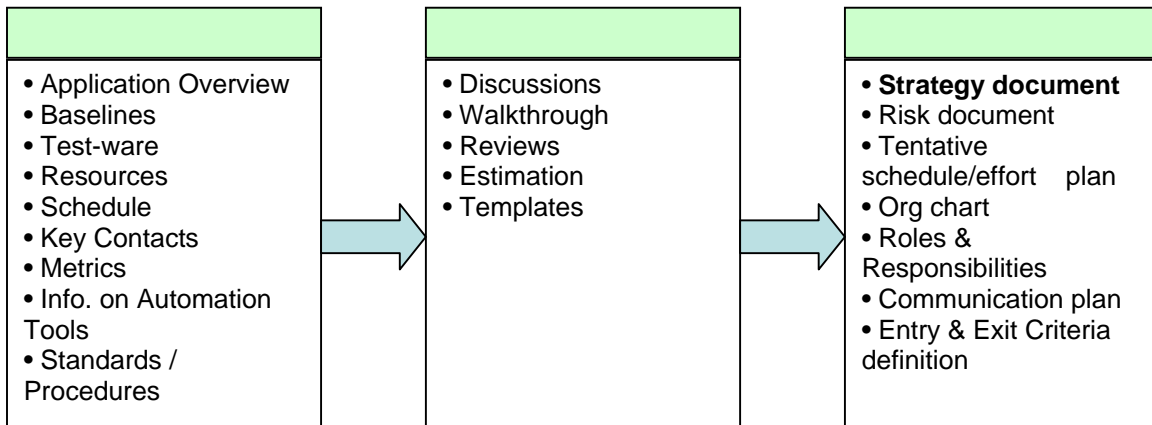
System testing phase definitely plays a major role in the software development life cycle. This phase also takes a lion's share of approximately 30% of the overall effort of the development cycle. It's also is a critical phase, as this phase certifies the conformance to requirements of the customer before releasing the system to the customer.

Any impact on the schedules on the CUT phase, the impact on the testing is also high. Hence the need for an optimized, effective and efficient process to deliver a well tested product and to encounter minimum field errors.

1.1 System Testing - Process Flow



2.0 Strategy



Input

- 2.1 Application Overview – An overview of the application which needs to be tested with respect to its scope, functionality, criticality, complexity, etc.,
- 2.2 Baselines – Documents associated to the application which are base lined such as the Requirement specification document; functional specification document; use-case document; high level design document; low level design document; etc.,
- 2.3 Test-ware – This constitutes of Test Cases (Test condition, which is the representation of a function in a testable state + Data to test conditions); Test Scripts (which could have multiple test cases, which are sequenced and carries information such as Check Points, Instructions & Expected results, enabling easy understanding to layman); Traceability Matrix (which maps functionality to design to test case to test script); Run plan (which address to What need to be done? Who would do? When to do? What are the pre-requisites? What to use?)
- 2.4 Resources - The skill sets of the human resource projected available, time frame they are available; hardware, software resources availability...
- 2.5 Schedule – The time frame by which the testing need to be completed
- 2.6 Key Contacts – Key human resources who could provide or point to information about the project. E.g. Functional consultants, Technical Architects, Module Leads etc.
- 2.7 Metrics – Organizational level/technology level reference metrics
- 2.8 Information on Automation Tools – Information on the availability of the testing associated tools and their applicability to the project
- 2.9 Standards & Procedures – Organizational level / technical level / clients expectations level standards and procedures which need to be adhered to

Tools & Technique

- 2.10 Discussions – Discussions with the key contacts, stakeholders to collect information
- 2.11 Walkthrough – Walkthrough of the application if it is ready or walkthrough of prototype or the story board if available

- 2.12 Reviews – Review of the available baseline documents, test-ware readiness
- 2.13 Estimation – Broad/High level effort estimation for the complete system testing phase
- 2.14 Templates – Verification on the available, prescribed templates associated to the system testing phase

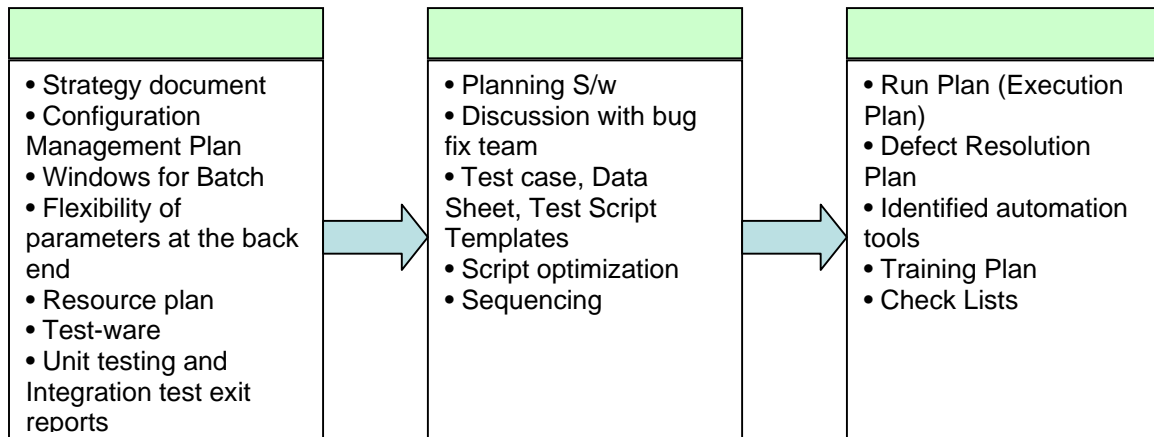
Output

- 2.15 Strategy Document – This document is a live document which gets updated as and when required during the entire lifecycle of the project, this document consists of sections such as,
 - Background, which illustrates the need for the program/project with the problem statement and the solution statement
 - Application Overview, an overview of the application which implements the solution statement proposed
 - Testing Strategy, addressing
 - Scope of testing – The purpose, boundaries (inclusions, exclusions) of the planned testing activity
 - Test Coverage – Elaboration of the inclusions section of the testing scope (in terms of Functional Coverage, Execution Path Coverage, Data Coverage, Benchmark against expected execution, etc.,)
 - Test Execution Methodology – Testing Process by itself (defining the sequence, dependencies, prerequisites, etc.); Types of testing (functional testing, load/performance testing, stress testing, compatibility testing), Rounds of testing proposed (No. of rounds of testing based on the complexity of the system, availability of the application, resources, time, defects identified and its severity),
 - Test Environment – Elaboration of Hardware (Servers, N/w, Clients), Software (O/s, Patches, Application, etc., including tools) requirements to execute the testing activity
 - Test Entry & Exit Criteria – This addresses the conditions on which the System Testing could start (Ex. test environment is setup (including interfaces if any); Completion of unit and integration testing – execution of all unit and integration test scripts; all bugs identified of unit/integration testing to be fixed and closed; the modified application moved to test environment; code baselined; data as per test scripts available in the test environment) And the conditions on which the System Testing could successfully complete (Ex. All test scripts are executed and reasons provided for Invalid and Untested Scripts; There are no showstoppers or critical defects, which are in open status; No major issues/risks are present at the end of testing; maximum no. of open bugs which could be carried forward to the next phase of the project. One other parameter assisting the decision on these criteria is risks associated to the project.
 - Test Organization and Resources
 - Test Organization Management – Identification and Building the organization units to test the system based on the system complexity, technology used, size of the application, etc., to name a few common organization units, testing director, test coordinator (in case the testing is at a program level and/or execution planned from

multi location), test execution manager, test schedule manager testers, test environment manager, bug fix manager)

- Roles & Responsibilities – Definition of roles of responsibilities for each of the org. units identified for the process
- Test Efforts & Estimates – Based on the testing components, coverage, efforts and estimates to be derived
- Test Schedule – High level schedules to cover the system testing of the application
- Risk Management – This caters to Risk Identification (foreseeing of risks both internal and external to project/program), Qualitative analysis of the risks identified, Quantitative analysis of the risks identified, Risk response plan (How, When and Who addresses the respective risk), also the Risk monitoring and control (Communication tools such as Charts, Graphs, should be used during this process) should happen during the entire lifecycle of the project

3.0 Planning



Input

- 3.1 Strategy Document – <Refer section 2.15>
- 3.2 Configuration Management Plan – Management of the code developed, in terms of its versioning, traceability, accessibility & availability of the right version.
- 3.3 Window for batch – Based on the application, its functionality, in case of any batch associated such as nightly cycle, weekly cycle, monthly cycle, quarterly cycle, half yearly cycle, (usually a critical factor in business critical financial processing), this also could be a parameter for deciding the no. of rounds of testing (Ex. If a weekly cycle fails, the same could be tested during the testing of monthly cycle without having to repeat one full round of testing)
- 3.4 Flexibility of parameters setting at the back end – Input about the possibility of accessing the back end data base and changing the values directly to set the pre-requisite data. (Ex. A dependent test script can be executed if pre-requisite data can be set, the same could be tested without having to repeat one full round of testing)
- 3.5 Resource plan – Information about the no. of people, their skill set, hardware, software/software tools along with the available time line, is one of the major inputs to arrive at the schedule plan/execution calendar
- 3.6 Test-ware – Availability/readiness of the test-ware focused more in terms of test cases, data preparation
- 3.7 Exit Report of Unit & Integration testing – This would act as one of the major inputs to determine the health of the application available for system testing

Tools & Technique

- 3.8 Planning Software – Software tools such as MS Project which could be used for schedule planning and tracking would be used.
- 3.9 Discussion with bug fix team – a discussion pertaining to bug fix activity, identification of key members, roles and responsibilities of the team, process of fixing, process for

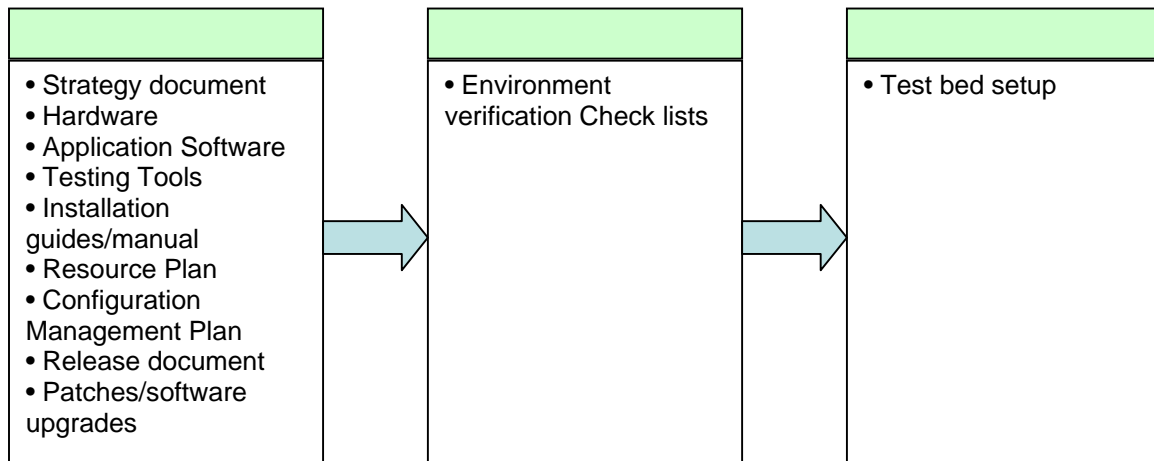
releasing the fixes, defining SLA for each of the severity, would be the core focus of this section

- 3.10 Templates of Test-Cases, Data Sheets, Test Scripts – These templates would guide the preparation of the test scripts , these are used if the test scripts are not available
- 3.11 Script Optimization – Selection of test scripts by identifying and removing the redundant test scripts still maintaining the system coverage would be the core activity
- 3.12 Sequencing – Test script sequencing plays a major role in determining the pass rate of the test scripts. Sequencing is the processing of identification and arranging of the test script in terms of predecessor & successor (also this process address the sequencing of test scripts in terms of Start to Start; Start to Finish; Finish to Start; Finish to Finish).

Output

- 3.13 Run Plan – Also called as execution plan, this carries information such as the schedules for execution of test scripts (optimized in terms of coverage and ordered to handle dependencies) and resources being associated with the schedules for execution
- 3.14 Defect Resolution Plan – This addresses the defect logging and tracking process including bug fixing and releases activity along with roles and responsibility charted, with the key contact person (module wise if required) identified who takes care of assignment of defects identified and who will participate in defect resolution status meeting. As per agreement, SLA for defect resolution is defined here.
- 3.15 Identified Automation Tools – Based on the environment, testing methodology, complexity, available resources, verification of applicability of automation tools identified (Ex. Win Runner; Load Runner; etc.,)
- 3.16 Training Plan – Addressing to the training needs of the testing team with respect to the application to be tested, tools used, defects logging. Training plan also associates to the defect resolution team with respect to the defect tracking tools proposed for usage.
- 3.17 Check Lists – Related checklist for each of the stage in the planning phase could have check list associated which could be used for error minimization. E.g. Environment Readiness checklist, test case coverage/preparation checklist etc.

4.0 Environment Setup



Input

- 4.1 Strategy Document – This would provide the global picture of the system testing execution. <Refer Section 2.3>
- 4.2 Hardware – Based on the strategy of execution of the test scripts for system testing, identification of hardware resources to set up the test bed is essential (Ex. servers, network devices, clients, etc.,)
- 4.3 Application Software – Along with the required software which need to be setup and configured for enabling the system testing (Ex. Operating System; Database setup; Application Server, required drivers, etc.,), even the application which needs to be tested also need to be set up. Database design and list of master/transaction tables are also required
- 4.4 Testing Tools – Identification and set up of automated testing tools (Ex. Win Runner; Load Runner; etc.,)
- 4.5 Installation Guide/Manual – Vendor provided reference manual for installation of hardware or software
- 4.6 Resource Plan – Resource availability with respect to people required for setting up the environment e.g. System Administrator, Database Administrator etc
- 4.7 Configuration Management Plan – Managing the release of the code/application build for testing
- 4.8 Release Document – Application Build release document which addresses to application setup, configuration set up, features available in the release, exceptions etc.
- 4.9 Patches/Software Upgrades – Identification and applying of any software patches to set the environment

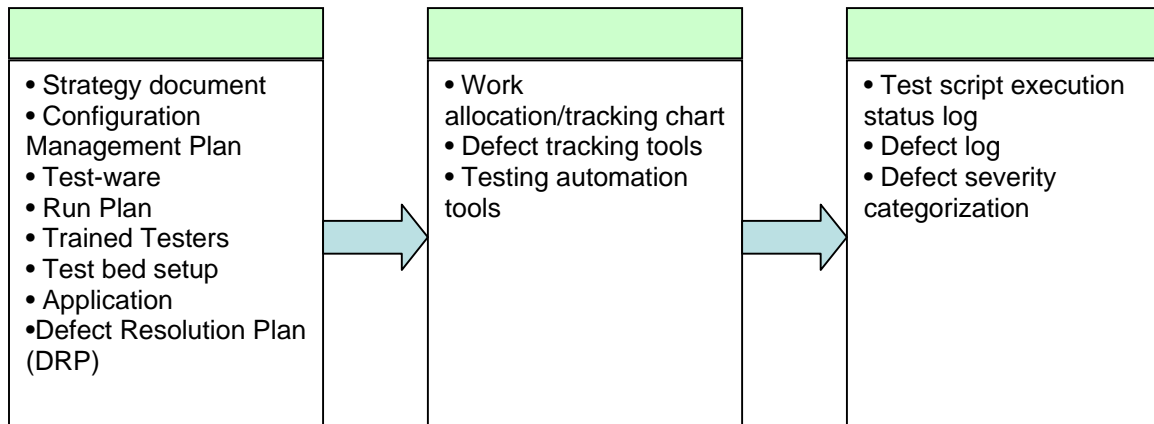
Tools & Technique

- 4.10 Environment Verification Checklist – Guiding the environment setup with check points for each of the sections (Ex. Hardware setup; Database setup; etc.,)

Output

- 4.11 Test Bed Setup – Complete setup of testing environment which could be used for testing the system by executing test scripts.

5.0 Execution



Note: Execution, Defect Assignment & Fixing, Defect data Analysis and Defect Resolution Verification processes happen in parallel

Input

- 5.1 Strategy document – This document is used as reference document which would provide guidelines about the roles & responsibilities, risks and responses for the same, test execution methodology, time frame for execution. <Also refer section 2.15>
- 5.2 Configuration Management Plan – This would help in choosing the right release/build to be tested and defects logged against the same.
- 5.3 Test-ware – Availability of test scripts, which carries Check Points, Instructions & Expected results for better direction, enabling easy understanding to layman <Refer section 2.3>. Automated test scripts in case of automated testing.
- 5.4 Run Plan – Run plan is a part of the test-ware, but it is been illustrated separately in this section as this is the heart of the execution phase of system testing which addresses the schedules for the test scripts execution, sequence of the test script execution (When of it? Is addressed); who needs to execute; where it needs to be executed and what are the pre-requisites for execution.
- 5.5 Trained Tester – As a complement to who executes the test scripts, it is important that the testers are equipped to provide efficient & effective test results (one of the factors which could minimize is the tester errors). The focus on this need to be high because of the reason, defect analysis and correction could be more expensive than defect identification.
- 5.6 Test Bed Setup – Complete environment setup including hardware, software, networking, which readily facilitate execution of test scripts
- 5.7 Application – Application which is tested at the unit and integration level and available for testing at the system level
- 5.8 Defect Resolution Process: This document provides the overall Defect logging and tracking process. <Refer section 3.14>

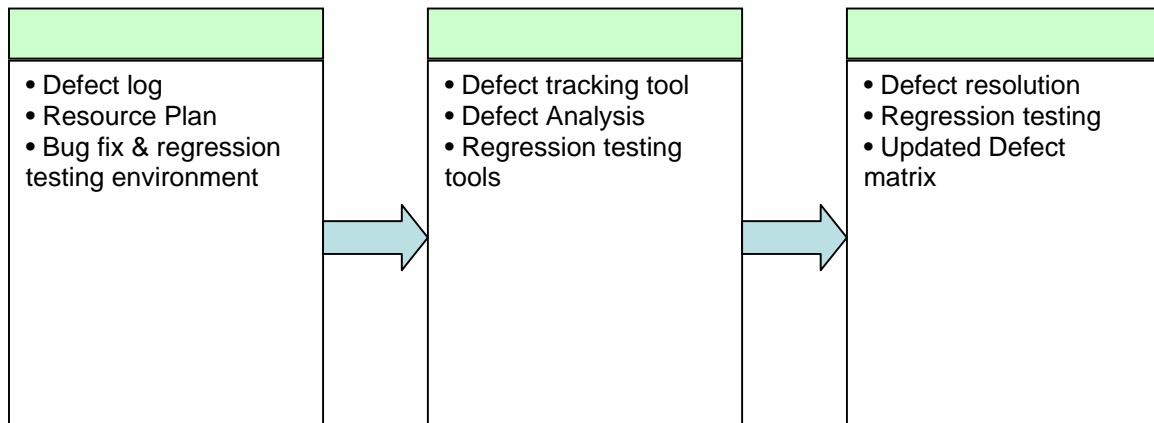
Tools & Technique

- 5.9 Work Allocation/Tracking Chart – As against work plan, this chart facilitates allocation of test scripts to testers and tracking the same to completion. This also would be handy when a defect is fixed and needs re-testing
- 5.10 Defects Tracking Tool – This tool is used for logging the defects identified. This could also be an automated tool. Provision to capture information such as the defect id, defect description, identified by, time stamp of identification, severity of the defect, need to be provided
- 5.11 Testing Automation Tools – Availability of the testing automation tools based on the identification made during the planning phase (Ex. Load Runner, Win Runner, etc.,)

Output

- 5.12 Test Script Execution Status Log – Result after the execution of the test scripts is logged against major/logical steps of the test scripts (if the test script is large in size) else at the test script level itself. Categorization such as Pass, Fail, Work in Progress, On Hold, needs to be logged against each attempted test scripts which depicts the test script execution status.
- 5.13 Defect Log – The defects identified during the execution of test scripts are logged in the defects tracking tool as identified and necessary information regarding the defect is captured and where ever necessary even the screen shots would be captured and made stored as snap shot of the defect which would help defect resolution members in analysis/reproduction of the defect
- 5.14 Defect Severity Categorization – All the defects which are captured should be categorized as Showstopper/Critical/Major/Minor based on its impact on the expected functionality/scope and/or based on the impact on carrying out system testing activity itself.

6.0 Defect Assignment & Fixing



Input

- 6.1 Defect Log – Log of all the defects identified by the testers is the primary source for the defect assignment for fixing.
- 6.2 Resource Plan – This would indicate the availability of resources in the bug fix team and their loading factor. Defects are assigned to the respective members based on these inputs to resolve the same in the prescribed SLA timeframe.
- 6.3 Bug fix & regression testing environment – Based on the criticality of the application and its data, it is important to assess and have a different environment for testing and bug fixing activities as this could be happening in parallel. This is helpful to avoid skewed results while testing and also would avoid influence on the testing schedules.

Tools & Technique

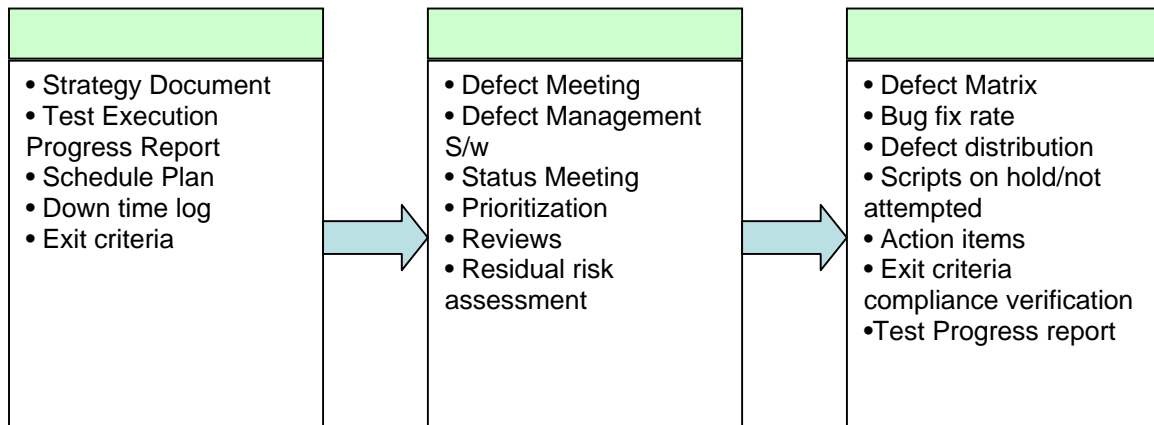
- 6.4 Defect tracking tool – This would be handy to manage the defects identified, logged, till it is confirmed resolved. <Refer section 5.9>. Typical status flow of a defect would be Open, Assigned, Fixed, Closed (usually after the originator certifies by retesting).
- 6.5 Defect analysis – This is a one of the most challenging technique in the whole process of system testing. Typically the defect analysis could lead to any of the following defect categories, namely Code change, Data Related, Duplicate, Environment Related, Tester Error, Modify test case, Known defect, Discussion Item, User training. This list need to be customized based on the project. Based on the outcome of the analysis, necessary action need to be taken to resolve the defect.
- 6.6 Regression testing tools – If in case a regression testing tool such as Win Runner is planned for usage, should be used to test the resolved defect.

Output

- 6.7 Defect Resolution – Action taken by the bug fix team to resolve the defect assigned to them with in the prescribed SLA time frame and maintaining the project quality norms.

- 6.8 Regression testing – Once the assigned defect is fixed this needs regression testing to ascertain the fix plugs the defect. If tools are identified for the purpose, the same should be used.
- 6.9 Update Defect Matrix – Defect matrix would act as one of the major communication tools for the whole system testing activity. Once the bug is fixed, the defect matrix should be updated. The defect status would change from assigned to fixed, which would trigger the process of verification and closure. Updating should be a dynamic process.

7.0 Defect Data Analysis



Input

- 7.1 Strategy Document - <Refer section 2.15>
- 7.2 Test Execution Progress Report – This narrates the status of the execution of the test scripts as against the planned execution
- 7.3 Schedule plan – Provides the timeline planned for the execution of the test scripts
- 7.4 Down time log – Information about all the downtime which was experienced during the test script execution is logged here. Some of the contributors for this is Show Stopper Bugs; Environment Setup issue/Failure; etc.,
- 7.5 Exit Criteria – This is part of the strategy document, but takes a focal light here as compliance to this is the basic requirement to move to the next phase of the project development lifecycle.

Tools & Technique

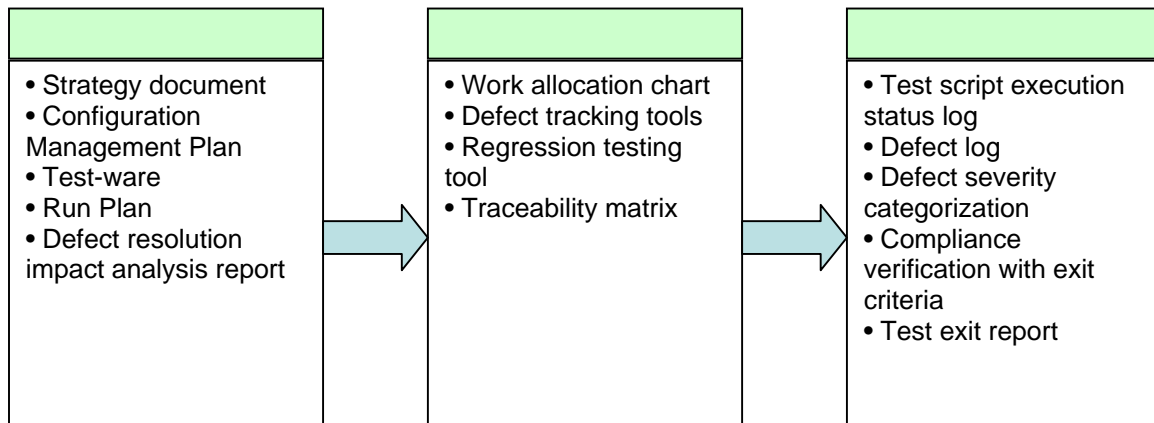
- 7.6 Defect Meeting – This is a very effective technique to have a good turn around time for defect resolution. Key participants for this meeting would be the test lead, bug fix team lead, SQA. The focal point of the meeting would be on the defect classification in terms of severity (Showstopper, Critical, Major, Minor), status (Open, Assigned, Fixed, Re-test, Closed), Deferred, Re-opened), category (Code Change, Data Related, Environment, Tester Error, etc.,) and also on the action items to meet the SLA. All the defects under discussion would be taken up during this meeting and a possible resolution arrived at
- 7.7 Defects Management Software – A tool such as Test Director could be used for management of the defects logged. Suite of features (automated mails to the concerned people, multiple types of reports etc.) available in the software tool enables efficient management of defects and assists in tracking to closure.
- 7.8 Status Meeting – Again a very effective technique to meet SLA for the defect resolution. The focus here would be on the progress reporting, action items planned for meeting the SLA, discussion about the issues along with the action points to counter the same. The participants could be the leads of the project along with the SQA, Project Manager and other management members. This also acts as an escalation platform related to any issues and also helps management in guiding. This should be a periodic affair during the complete lifecycle of the activity.

- 7.9 Prioritization – Pareto Charts is one of the tools which assist in prioritization of the defects and planning of action points, as the 80-20 rule is usually applicable.
- 7.10 Reviews – This is an essential technique to ensure the accuracy of the whole defect management life cycle.
- 7.11 Residual Risk Management – As it is known that risk always exists, it needs to be managed. Process of continuous identification, assessment and response to the same is very essential to avoid issues/crisis/disaster

Output

- 7.12 Defect Matrix – Representation of the classification of the defect essentially constitutes to defect matrix. This holds the core position in the defects dashboard which is a strong communication tool during the complete life cycle of the system testing
- 7.13 Bug fix rate – Computation of turnaround rate, the time taken and the number of defects which as traversed from open status to fixed status
- 7.14 Defect Distribution – Necessary action points could be arrived based on the distribution of defects essentially on the classification such as status, category and severity.
- 7.15 Scripts on hold/not attempted – Analysis and decision making on the scripts which need to be put on hold and which were not attempted because of the dependent scripts being failed.
- 7.16 Action Items – Action items to resolve issues, to adhere to prescribed SLA, to meet exit criteria, to attain/maintain good productivity.
- 7.17 Exit criteria compliance verification – Status verification as against the agreed upon exit criteria of system testing.
- 7.18 Test Progress report – Report summarizing the progress of the testing in terms of Total test scripts planned, attempted, passed, failed, on-hold, not attempted. This holds the core position in the defects dashboard

8.0 Defect Resolution Verification



Input

- 8.1 Strategy document – This document is used as reference document which would provide guidelines about the roles & responsibilities, risks and responses for the same, test execution methodology, time frame for execution. <Also refer section 2.15>
- 8.2 Configuration Management Plan – This would help in choosing the right release/build to be tested and defects logged against the same.
- 8.3 Test-ware – Availability of test scripts, which carries Check Points, Instructions & Expected results for better direction, enabling easy understanding to layman <Refer section 2.3>. Automated test scripts in case of automated testing.
- 8.4 Run Plan – Run plan is a part of the test-ware, but it is been illustrated separately in this section as this is the heart of the defect resolution verification phase of system testing which addresses the schedules for the test scripts execution, sequence of the test script execution (When of it? Is addressed); who needs to execute; where it needs to be executed and what are the pre-requisites for execution.
- 8.5 Defects Resolution Impact Analysis Report – This provides information about the functionalities which had impact during the resolution of identified defects. This acts as pointers to the testers to focus not only on the resolved defects but also on the impacted functionalities.

Tools & Technique

- 8.6 Work Allocation/Tracking Chart – As against work plan, this chart facilitates allocation of test scripts to testers and tracking the same to completion.
- 8.7 Defects Tracking Tool – This tool is used for logging the defects identified. This could also be an automated tool. Provision to capture information such as the defect id, defect description, identified by, time stamp of identification, severity of the defect, need to be provided
- 8.8 Regression testing tool – To verify the fixed defect it may need regression testing to ascertain the fix plugs the defect. If tools are identified for the purpose such as Win Runner, the same should be used.

- 8.9 Traceability Matrix – This is a very important tool which maps from the functionality to the test script level, this assists in identification of the impacted functionalities because of the fix provided for a specific function.

Output

- 8.10 Test Script Execution Status Log – Result after the execution of the test scripts is logged against major/logical steps of the test scripts (if the test script is large in size) else at the test script level itself. Categorization such as Pass, Fail, Work in Progress, On Hold, needs to be logged against each attempted test scripts which depicts the test script execution status.
- 8.11 Defect Log – The defects identified during the execution of test scripts are logged in the defects tracking tool as identified and necessary information regarding the defect is captured and where ever necessary even the screen shots would be captured and made stored as snap shot of the defect which would help defect resolution members in analysis/reproduction of the defect
- 8.12 Defect Severity Categorization – All the defects which are captured should be categorized as Showstopper/Critical/Major/Minor based on its impact on the expected functionality/scope and/or based on the impact on carrying out system testing activity itself.
- 8.13 Compliance verification with exit criteria - Status verification as against the agreed upon exit criteria of system testing. If the test results complies with the required criteria, the system could move to the next phase is the development cycle.
- 8.14 Test Exit Report – This summarizes the whole testing activity process. Exit report essentially carries information such as Execution Methodology; Test Execution Summary covering information such as, Test scripts execution - planned v/s actual, Test scripts/scenario's not attempted or on hold; Defect Summary with respect to # of defects, categorization of defects, distribution of defects in terms of functional and non-functional; Issues faced during the execution of test scripts; Expected objectives of the test round. Exit report is a useful communication tool and acts as a certificate for the system to move to the next phase of the development cycle.

9.0 Appendix A – Sample Dashboard Snapshot

ROUND 3.0		TEST PROGRESS											DEFECTS IDENTIFIED			
TEST REPORT	Scripts - Cumulative						Scripts - Today						Defects (D)	% of Defects to Executed Scripts		
	Total Scripts	Attempted	Hold Dependent	Hold (Others)	Pass (P)	Fail (F)	% of Executed to Total Scripts	Planned	Attempted	Hold Dependent	Hold (Others)	Pass (P)			Fail (F)	% of Executed to Total Scripts
STAND ALONE																
CAYMAN	2					0.0								0.0	0.0	
CAC	3					0.0								0.0	0.0	
VFL Annuities	2					0.0								0.0	0.0	
ANNVFBI	3					0.0								0.0	0.0	
ANNCFBI	3					0.0								0.0	0.0	
PSP Viaterm	1					0.0								0.0	0.0	
PSP EIWL	11					0.0								0.0	0.0	
LTC Concare	3					0.0								0.0	0.0	
TOTAL	28	0	0	0	0	0.0	0	0	0	0	0	0	0	0.0	0	
LPP CORE																
Pre Cycle	661	486	0	0	470	16	73.5	163	163	0	0	155	8	100.0	16	3.3
Post Cycle	203						0.0							0.0		0.0
KAP																
	18					0.0								0.0		0.0
TOTAL	882	486	0	0	470	16	55.1	163	163	0	0	155	8	100.0	16	3.3
REPORTS																
TEST REPORT	Total Scripts	Attempted	Hold	Pending Certification	Pass	Fail	% of Executed to Total Scripts	Planned	Attempted	Performance Hold	Pending Certification	Pass	Fail	% of Executed to Planned	Defects (D)	% of Defects to Executed Scripts
Pre Cycle	122						0.0							0.0		0.0
Post Cycle	24						0.0							0.0		0.0
TOTAL	146	0	0	0	0	0	0.0	0	0	0	0	0	0	0.0	0	0.0
RISKS / ISSUES																

5-Feb-03					
DEFECT MATRIX	Stand Alone	LPP Core	KAP	Reports	Total
DEFECT STATUS					
Open		5			5
Assigned		11			11
Fixed					0
Closed by Testing					0
Closed by Business					0
Released					0
Retest					0
Reopened					0
Deferred					0
Total	0	16	0	0	16
DEFECT CATEGORY					
Code Change					0
Data Related					0
Observation					0
Duplicate					0
Modify Test Case		7			7
Build Maintenance					0
Yet to be Assigned		5			5
New Requirement					0
Environment					0
User Training					0
Data Migration		1			1
Discussion Item		1			1
Tester Error		1			1
Batch Related					0
Known Defect					0
Performance		1			1
Total	0	16	0	0	16
DEFECT SEVERITY					
Showstopper					0
Critical					0
Major		15			15
Minor		1			1
Total	0	16	0	0	16

10.0 Appendix B – Templates

- Risk Management
- Work Plan
- Defect Log



Microsoft Excel
Worksheet