HOW TO CREATE USEFUL SOFTWARE PROCESS DOCUMENTATION

Linda Westfall
The Westfall Team

lwestfall@westfallteam.com

3000 Custer Road, Suite 270, PMB 383
Plano, TX 75075

ABSTRACT

Whether our organization is using ISO 9001, the Software Engineering Institutes Capability Maturity Model - Integrated^{SM}, Total Quality Management, Six Sigma or some other quality framework, one of the cornerstones of any of these frameworks is to document our processes. Unfortunately efforts to document our process often end up in volumptus volumes of verbosity that sit on the shelf and gather dust. ***How to Create Useful Software Process Documentation*** introduces the reader to a simple, practical method for defining and documenting software processes that are easy to understand, easy to use and easy to maintain.

WHY IS PROCESS DOCUMENTATION IMPORTANT

One of the cornerstones to any quality program is documented processes. Processes are "codified good habits" [Down-94] that "define the sequence of steps performed for a given purpose" [IEEE-610]. By standarizing and documenting our software processes we can describe and communicate what works best in our organizations. This can help us:

- Ensure that important steps in our processes aren't forgotten
- Facilitate the propagation of lessons learned from one project to the next so we can repeat our successes and stop repeating actions that lead to problems
- Eliminate the need to "reinvent the wheel" with each new project while providing a foundation for tailoring our processes to the specific needs of that project

Documented processes provide the structured basis for creating metrics that can be used to understand our process capabilities and analyze our process results to identify areas for improvement. Standardized software processes are necessary for training, management, review and tools support.

K.I.S.S

There are many different methods for mapping processes. Examples include the IDEF0 modeling language [IEEE-1320] and entity process models [Humphrey-89]. There are also different proprietary tools that are specifically intended for use in defining processes. However, the main reason I use the method described in this paper is that it is very simple. While many larger companies have process experts, most of the small to mid-sized companies that I work with don't have that luxury. The people charged with defining and documenting their processes are the same people that are responsible for getting the product out the door. They don't have the time to learn special techniques or tools (or the money to purchase these tools).

The process documentation method I describe in this paper can be implemented using basic PC based word processing and graphics tools like Microsoft Word and PowerPoint. I have found that these tool are readily available to most people in even the smallest organizations. This allows everyone to easily document and maintain their own processes with minimal additional skills training and expense.

PROCESS DEFINITION OUTLINE

The method I use to document a process is based on the ETVX process defintion format which includes the definition of Entry criteria, Tasks, Verification and eXit criteria for a process. I expand this outline to include a statement of the purpose of the process, a process flow diagram, and lists of its deliverables and quality records. Appendix A includes a template for documenting a process. Example process definitions are included in Appendices B and C:

- ♦ Appendix B - Software System Testing Process
- ♦ Appendix C - Execute System Tests & Report Anomalies Process

Purpose

The process purpose is a statement of the value added reason for the process. It defines what we are attempting to accomplish by executing the steps in the process. For example, the purpose of a Software Testing process might be to validate the software system against the approved requirements and identify product defects before the product is release to the customer.

Entry Criteria

The entry criteria are specific, measurable conditions that must be met before the process can be started. This may include:

- ♦ Work products that must be completed, approved and/or placed under configuration control
- ♦ Tasks and/or verification steps that must be satisfactorily completed
- ♦ Specific measured values that must been obtained
- ♦ Staff with appropriate levels of expertise that are available to perform the process

♦ Other resources that are available and/or ready for use during the process

Process Flow Diagram

A picture really is worth a thousand words. A simple flow diagram of a process can make that process easier to understand by showing the relationships between the various tasks, veritifcation steps and deliverables and by showing who is responsible for each task or verification step.

Roles: The first step in creating a process flow diagram is to define the various roles of the process. These are the individuals or groups responsible for the steps that are part of the process. Their roles are listed in the “swim lanes” along the left side of the process flow diagram as illustrated in Figure 1.

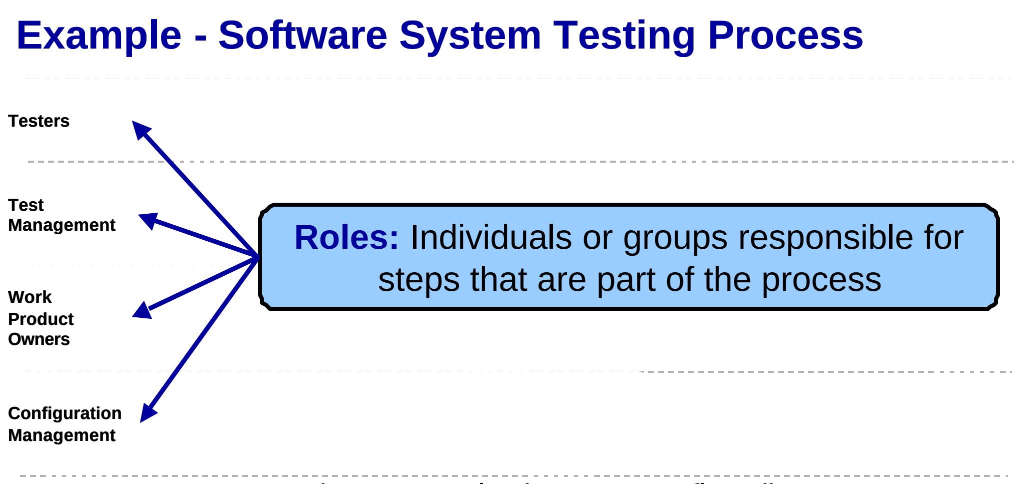

Example - Software System Testing Process
Testers
Test Management
Work Product Owners
Configuration Management
Roles: Individuals or groups responsible for steps that are part of the process

Figure 1 – Roles in a process flow diagram

In order to make our standardized processes as adaptable as possible, the roles should be labeled in generic terms rather than by using specific titles or the actual names of individuals or groups. This keeps us from being forced to update our process documentation every time there is a reorginization or staff turnover. It also allows us to easily tailor our processes to projects of various sizes. For example, on very large projects, an entire team might be assigned to one of the process roles but on very small projects a single individual might be assigned several roles.

Process Flowchart: A process flowchart can then be drawn across the “swim lanes” that illustrates the various tasks, verification steps, decisions and deliverables of the process as illustrated in Figure 2. This method makes it easy for an indivdual assigned to a role to read across their “swim lane” and identify their responsibilities.

If more than one role is responsible for a step, the box for that step spans multiple “swim lanes”. For example, task T1: Requirements Elicitation in Figure 2 is the responsibility of the customer, marketing, product management and systems engineering. Occasionally, a task needs to span more than one “swim lane” but the order of the “swim lanes” does not allow for a single solid box to be used. This can be illustrated using a split box as illistrated in the verification step V1: FRS Technical Review & Approval in Figure 2. It should be noted that sometimes a simple reordering of the roles in the “swim lanes” can illiminate the need for this special notation. For example, in Figure 2, if the marketing and customer roles were swapped, a single solid box could be used.

Each task, verification step and deliverable in the flowchart is labeled sequentially (e.g., T1, T2, …,Tn, V1, V2, …, Vn and D1, D2, …, Dn) so that they can quickly be referenced to their associated descriptive text.

Decisions that need to be made as part of the process can be illustrated as a diamond (the standard flow chart symbol for a decision). These decisions can be part of a step and be included inside a step box (see example in step VI: Conduct Periodic Test Status Reviews in the Software System Testing Process flow diagram in Appendix B) or these decisions can be separate tasks or verification steps in the process.

Along the bottom of the process flow diagram, I document the major deliverables of the process. I use the standard flow chart symbols for documents and data stores to distinguish between documents and electronic files or database items.

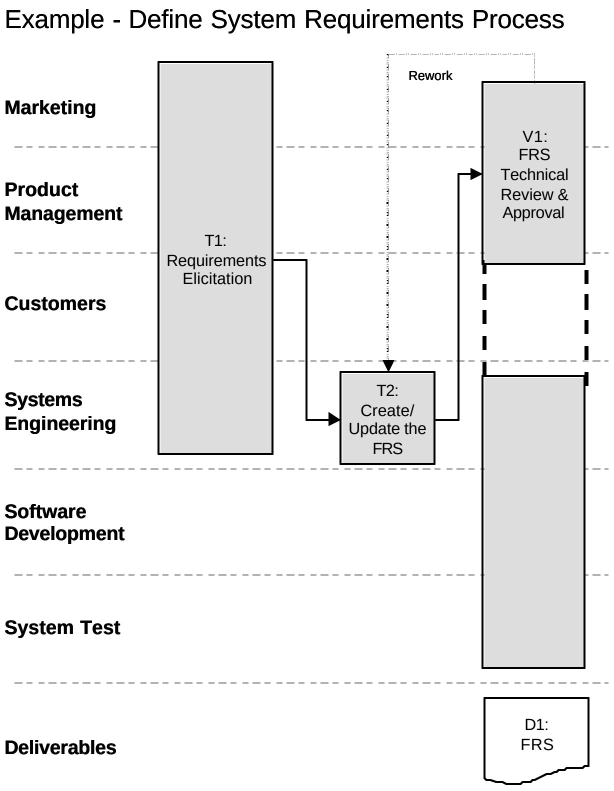

Example - Define System Requirements Process
Marketing
Product Management
Customers
Systems Engineering
Software Development
System Test
Deliverables
T1: Requirements Elicitation
T2: Create/ Update the FRS
V1: FRS Technical Review & Approval
Rework
D1: FRS

Figure 2 – Process Flow Chart

Tasks & Verification Steps

A task is a cohesive, individual unit of work that must be accomplished to complete a process. A verification step is a set of specific actions taken to evaluate the outcome(s) of one or more tasks to determine whether the requirements and/or specifications have been met. For example, as illustrated in Figure 2, there are tasks for elicitiing the requirements and for creating the Functional Requirements Specification (FRS). The verification step V1 then evaluates the FRS document (the outcome of those tasks) through a technical review and approval step.

For each task or verification defined in a Process Flow Diagram, a textual description should be included in the process definition. These descriptions may include:

- Detailed work instructions (or pointer to the work instructions) that describe how to accomplish the task or verification step including specific techniques to be used
- Additional descriptions of specific responsibilities (e.g., the System Engineer is responsible for eliciting the requirements with inputs from Marketing, Product Management and the Customers)

- Pointers to standards to be used when conducting the task or verification step or when creating the output (e.g., verification standards like formal document inspection standards, work product standards like coding standards)
- Pointers to standardized templates for creating the outputs of the task or verification step (e.g., document templates, verification checklists, meeting agenda templates)
- Descriptions of (or pointers to the descriptions of) specific metrics associated with the task or verification step
- Required levels of expertise (or pointers to the descriptions of required levels of expertise) that must be possessed by those responsible for the task or verification step
- Other resources (e.g., tools, hardware) that must be available and/or that are used

Exit Criteria

The exit criteria are specific, measurable conditions that must be met before the process can be completed.

Deliverables

Deliverables are the tangible, physical objects or specific measurable accomplishments that are the outcomes of the tasks or verification steps. I typically show only completed deliverables and not intermediate work products. For example, in Figure 2 I don't show the draft versions of the FRS but only show the final FRS under the verification step that results in its approval/completion.

Quality Records

While deliverables are the direct, intended outputs from the processes, quality records are secondary outputs that provide the evidence that the appropriate activities took place and that the execution of those activities met the required standards. Examples of quality records include: test logs, minutes from meetings, audit reports, engineering notebooks, action item lists, and status reports. For each quality record, I list:

- Its custodian (the role responsible for collecting that record)
- Where that record will be maintained (e.g. project file, specific database)
- The minimum retention period for that record (the minimum length of time that the record will be kept available for access)

HIERARCHY OF PROCESSES

I have found that different levels of detail are required for defining processes at different times. For example, a very high-level view of the entire development process as illustrated in Figure 3 may be appropriate for planning, training, executive management review, or customer discussions. However, when the actual process is being executed, a detailed step-by-step set of work instructions is needed. There may also be times when intermediate level process defintions are appropriate.

This can be accomplished through the creation of process definitions at different levels of detail. If the process definitions are kept online, the linking of these various levels of process documentation can be accomplished easily using hyperlinks. For example, the System Test box in Figure 3 has been hyperlinked to the Software System Test Process definition in Appendix B (click on the System Test Box to jump to the Software System Test Process definition).

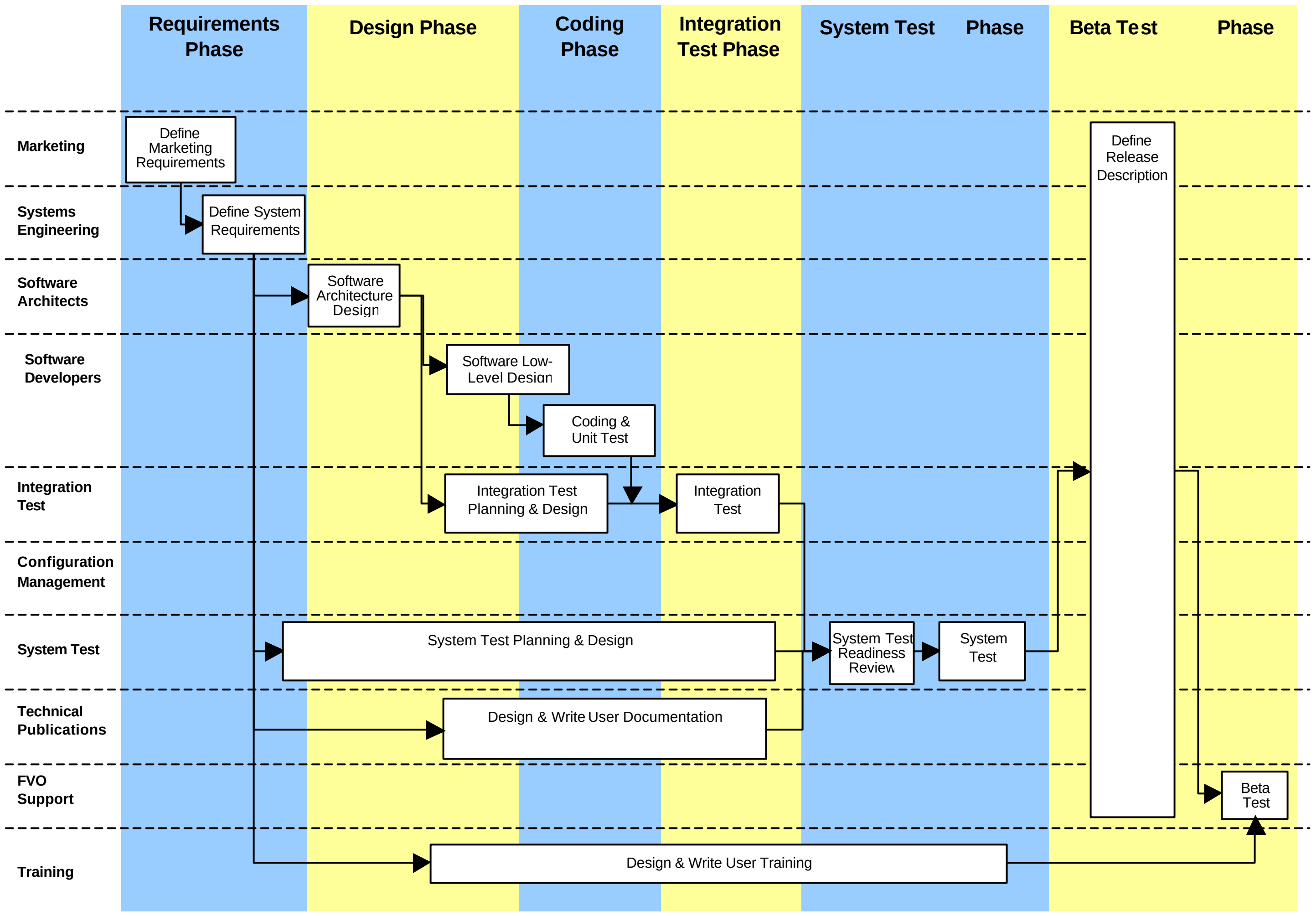


Figure 3 – Process Flow Diagram of Software Development Process

Each box in this process flow diagram could be linked to the associated lower-level descriptions as appropriate. If more detailed process definitions are required, hyperlinks to the next lower-level of process definitions could be created and so on, creating a hierarchy of process definitions. For example, the Execute Tests & Report Anomalies box in the Process Flow Diagram in Appendix B has been hyperlinked to the Execute Tests & Report Anomalies Process definition in Appendix C (click on the Execute Tests & Report Anomalies Box to jump to the Execute Tests & Report Anomalies Process definition). As appropriate, some of the boxes in the Execute Tests & Report Anomalies Process could also be hyperlinked to lower-level processes (e.g., CCB Analysis).

One of the advantages of creating a hierarchical process definition framework is the removal of redundancy. For example, the Execute Test & Report Anomalies Process defined in Appendix C has been generically defined. It could also be linked to from the Integration Test procedure or the Beta Test procedure. This eliminates the need to repeat these instructions and makes it much easier to maintain the documented procedures.

CONCLUSIONS

Having well defined and documented processes is an essential element of a quality system. However, this does not mean that specialized expertise and tools are required. Simple, useful process documentation can be created using the word processing and graphical tools available on almost any PC.

A heirarchical process definition structure can increase the usability of the process documentation while at the same time making them easier to maintain.

ABOUT THE AUTHOR

Linda Westfall is the President of The Westfall Team, which provides Software Quality Engineering and Software Metrics training and consulting services. Prior to starting her own business, Linda was the Senior Manager of Quality Metrics and Analysis at DSC Communications where her team designed and implemented a corporate wide metric program. Linda has more than twenty years of experience in real-time software engineering, quality and metrics. She has worked as a Software Engineer, Systems Analyst, Software Process Engineer and Manager of Production Software.

Very active professionally, Linda Westfall is the Chair of the American Society for Quality (ASQ) Software Division. She has also served as the Software Division's Program Chair and Certification Chair and on the ASQ National Certification Board.

REFERENCES

Down-94: Alex Down, Michael Coleman, Peter Absolon, *Risk Management for Software Projects*, McGraw-Hill Book Company, London 1994.

Humphrey-89: Watts Humphrey and Marc Kellner, *Software Process Modeling: Principles of Entity Process Models*, http://www.sei.cmu.edu/publications/documents/89.reports/89.tr.002.html

IEEE-1320: IEEE Standards Software Engineering, Volume 4, IEEE Standard for Functional Modeling Language – Syntax and Semantics for IDEF0, EEE Std. 1320.1-1998, The Institute of Electrical and Electronics Engineers, 1999, ISBN 0-7381-1562-2.

IEEE-610: IEEE Standards Software Engineering, Volume 1, *IEEE Standard Glossary of Software Engineering Terminology*, IEEE Std. 610-1990, The Institute of Electrical and Electronics Engineers, 1999, ISBN 0-7381-1559-2.

Appendix A – Process Documentation Template

<Process Name>

PURPOSE

- <List of the key value added reasons for executing this process>
-
-

ENTRY CRITERIA

- <List of the specific measurable conditions that must be true before this process can be started>
-
-

PROCESS FLOW DIAGRAM

<Role>

<Role>

<Role>

<Role>

Deliverables

TASKS

T1. <A sequenced list of the specific steps and their descriptions that must be performed to execute the process>
T2.
T3.

VERIFICATION

V1. <A sequenced list of verification steps that are performed during the process as checkpoints to ensure that the process is being performed as required and that the products of the process meet the required quality levels>
V2.
V3.

EXIT CRITERIA

- <List of the specific measurable conditions that must be true before this process can be completed>
-
-

DELIVERABLES

D1. <A list of the major deliverables of the process and their descriptions>
D2.
D3.

QUALITY RECORDS

Required Record	**Custodian**	**Retention Period**
<quality record name>	<role responsible for collecting/maintaining the quality record>	<retention period for the quality record>

Appendix B – Example Process Definition: Software System Test Process

Software System Testing Process

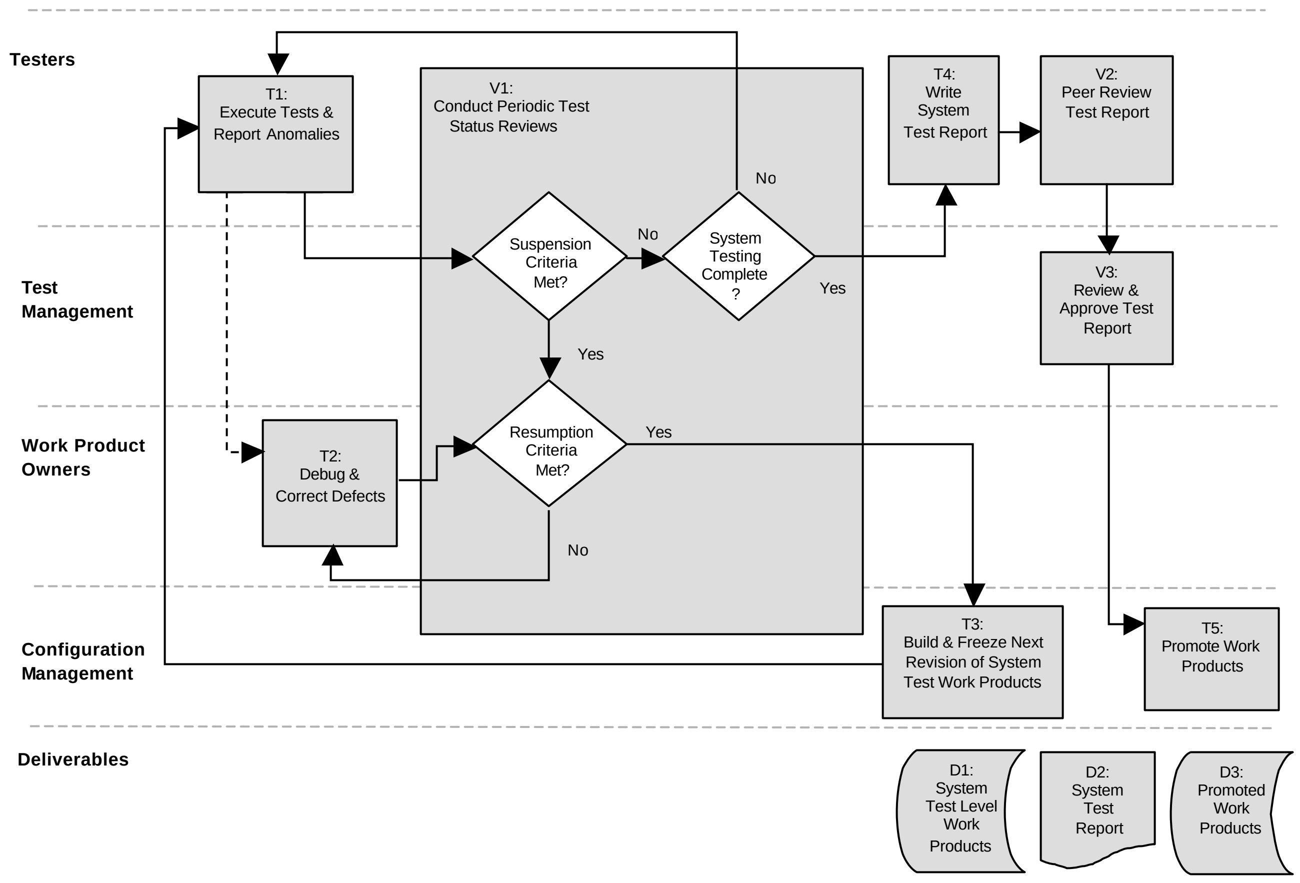
PURPOSE

- To validate the software system against the approved requirements
- To identify product defects before the product is released to the customer

ENTRY CRITERIA

- System Test Plan approved & under CM control
- System Test Cases and System Test Procedures approved & under CM control
- System Test Lab ready and available for use
- Integration Test complete
- Software Load promoted to System Test state
- User documentation & software installation procedures approved & under CM control
- Testing staff available with the appropriate levels of expertise

PROCESS FLOW DIAGRAM


Testers
Test Management
Work Product Owners
Configuration Management
Deliverables
T1: Execute Tests & Report Anomalies
V1: Conduct Periodic Test Status Reviews
Suspension Criteria Met?
No
Yes
System Testing Complete ?
No
Yes
Resumption Criteria Met?
Yes
No
T2: Debug & Correct Defects
T4: Write System Test Report
V2: Peer Review Test Report
V3: Review & Approve Test Report
T3: Build & Freeze Next Revision of System Test Work Products
T5: Promote Work Products
D1: System Test Level Work Products
D2: System Test Report
D3: Promoted Work Products

TASKS

T1. Execute System Tests & Report Anomalies: the system tester will execute a selected set of test cases for each system test load. If system test is suspended and restarted, these selected test cases will include cases to test all corrections and regression test the software. Any anomalies identified during system test are reported by the tester in accordance with the Anomaly Reporting process.

T2. Debug & Correct Defects: the owner (e.g., software development, technical publications) of each work product that is suspected to have caused the anomaly debugs that work product and corrects any identified defect(s) in accordance with the Problem Resolution process.

T3. Build & Freeze Next Revision of System Test Work Products: configuration management builds any updated revision of the software product(s) (e.g., software load, users manual, and installation instructions) that includes the identified corrected comp onents in accordance with the Software Build Process.

T4. Write System Test Report: At the end of the final cycle of System Test execution, the tester writes a System Test Report that includes a summary of the results from all of the System Test cycles.

T5. Promo te the Work Products: configuration management promotes the system tested work products for use in beta testing.

VERIFICATION

V1. Conduct Periodic Test Status Reviews: system test status review meetings are held on a periodic basis (as specified in the system test plan) during system test. If at any time it is determined that the suspension criteria (as specified in the system test plan) are met, system test execution is halted until the resumption criteria (as specified in the system test plan) are met and a new revision of the software load is built.

V2. These meetings are also used to determine when system test is complete based on the system test completion criteria (as specified in the system test plan).

V3. Peer Review Test Report: the testers peer review the system test report in accordance with the Peer Review process.

V4. Review & Approve Test Report: test management reviews and approves the final test report for distribution.

EXIT CRITERIA

- System test completion criteria are met (as specified in the system test plan).
- System Test Report is approved by test management.
- Final System Test software work products are promoted to beta test status.

DELIVERABLES

D1. System Test Level Software Loads: one or more intermediate System Test level software loads may be built to include the corrections to defects identified during System Test (note that the initial System Test level software load is the load promoted from Integration Test).

D2. System Test Report (see System Test Report Template)

D3. Promoted Software Load: at the completion of System Test, the final System Test level software load is promoted to become the initial Beta Test level software load.

QUALITY RECORDS

Required Record	**Custodian**	**Retention Period**
System Test log	System Testers (project file)	1 year minimum
Minutes from all System Test Status Review meetings	System Test Manager (project file)	1 year minimum
Minutes from any Product Build Verification meetings held	Configuration Management (project file)	1 year minimum
Anomaly Reports of anomalies found during System Test	System Testers (Change Request Database)	1 year minimum
System Test report peer review minutes	System Testers (project file)	1 year minimum

Appendix C – Example Process Definition: Execute Tests & Report Anomalies

Execute Tests & Report Anomalies

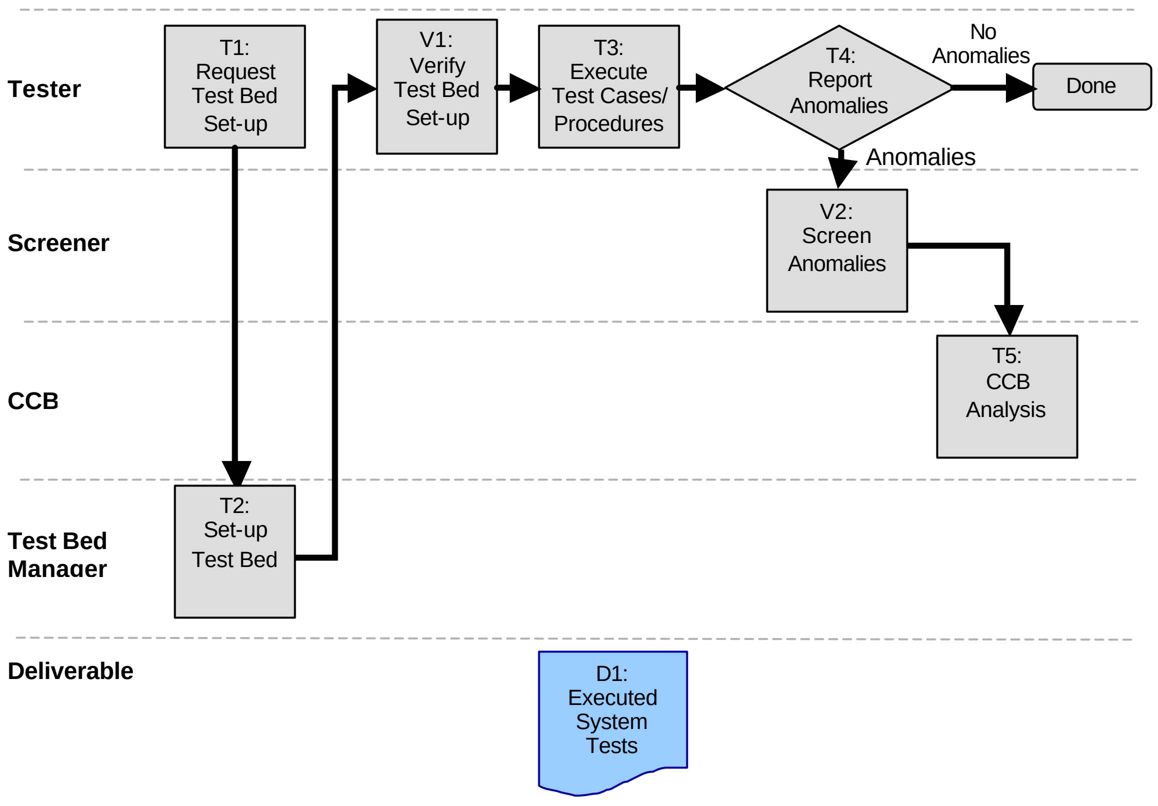
PURPOSE

- To execute a specific set of software tests
- To report and screen anomalies discovered during the execution of those tests

ENTRY CRITERIA

- Test Cases and System Test Procedures for the tests to be executed are approved & under CM control
- Test bed scheduled and available for use

PROCESS FLOW DIAGRAM



TASKS

T1. Request Test Bed Set-up: the tester completes a test bed configuration request form to reflect the test bed software, test database and hardware configuration required to conduct the test as specified in the test cases/procedures. This form is then submitted to the test bed manager.

T2. Set-up the Test Bed – the test bed manager configures the test bed as requested on the configuration request form.

T3. Execute Test Cases /Procedures: the tester executes the test cases/procedures and records the results of that execution in the test database (i.e., passed, failed, blocked). The tester records the test execution in the Test Log including the following information:

- Start/end time
- Tester(s) and observer(s)
- Environment (tools, test beds, simulators)
- Configuration (software & hardware)
- Cases/Procedures executed

- Failures and anomalies observed
- Other notes and observations

T4. Report Anomalies: if there are any anomalies to report, the tester reports those anomalies in accordance with the Creating an Anomaly Report process. The tester should attempt to reproduce any anomalies discovered and report the results of those attempts as part of the description of the anomaly.

T5. CCB Analysis: the Change Control Board (CCB) then analyzes any reported anomalies in accordance with the CCB Analysis process.

VERIFICATION

V1. Verify the Test Bed Set-up: the tester uses the test bed configuration request form as a checklist and verifies that the test bed has been appropriately configured.

V2. Screen Anomalies: the screener evaluates the anomaly report for completeness and resolves any issues with the tester. This includes checking to ensure that the:

- Anomaly title is a clear, concise summary of the anomaly
- Description of the anomaly is complete and understandable and includes information about attempts to reproduce the anomaly
- Steps to reproduce the anomaly are described to an appropriate level of detail so that the developer can reproduce the anomaly during debugging
- Test bed configuration and hardware/software environment have been specified
- Test cases/procedures being run when the anomaly occurred are specified
- Severity of the anomaly has been appropriately specified

EXIT CRITERIA

- All test cases/procedures have been executed or are blocked
- Test logs have been completed to record the test execution
- Anomaly reports have been created for all anomalies detected and those reports have been analyzed by the CCB

DELIVERABLES

D1. Executed Tests: the set of test cases/procedures that are either executed or documented as blocked

QUALITY RECORDS

Required Record	Custodian	Retention Period
Completed Test Bed Configuration form	Testers (included as part of the System Test Log)	1 year minimum
Test log	Testers (project file)	1 year minimum
Anomaly Reports of anomalies found during the execution of the tests	Testers (Change Request Database)	1 year minimum