

# QA Framework Proposal

Prepared by:



**Proposal Number: 2005.010**

**Version: 1.0**

**29-JUL-2005**

---

## Table of Contents

1.	Executive Summary	4
2.	Acronyms	4
3.	Thatavarti Solution	5
4.	Project Plan	16
5.	Risk, Contingencies, Assumptions, Dependencies and Constraints	17
6.	Version Management Procedure	18
7.	Training	19
8.	Team Structure	19
9.	Processes and Procedures	20
10.	Communication Channel	21
11.	Status Reporting	22
12.	Knowledge Management	22
13.	Case Studies	22
14.	Retention Policy	22
15.	Why Thatavarti?	23
16.	Glossary	23
17.	Test Metrics	23
18.	Templates and Formats	23

- Appendix 01: Test Life Cycle**
- Appendix 02: Installation testing checklist**
- Appendix 03: Review document**
- Appendix 04: System study procedure**
- Appendix 05: Doubts & Issues procedure**
- Appendix 06: Test data procedure**
- Appendix 07: Test case template**
- Appendix 08: Test scenario template**
- Appendix 09: Test plan template**
- Appendix 10: Test metrics**
- Appendix 11: Glossary**
- Appendix 12: Thatavarti Case studies**
- Appendix 13: Trace ability matrix**
- Appendix 14: Defect report**
- Appendix 15: Test Summary report**

## 1. Executive Summary

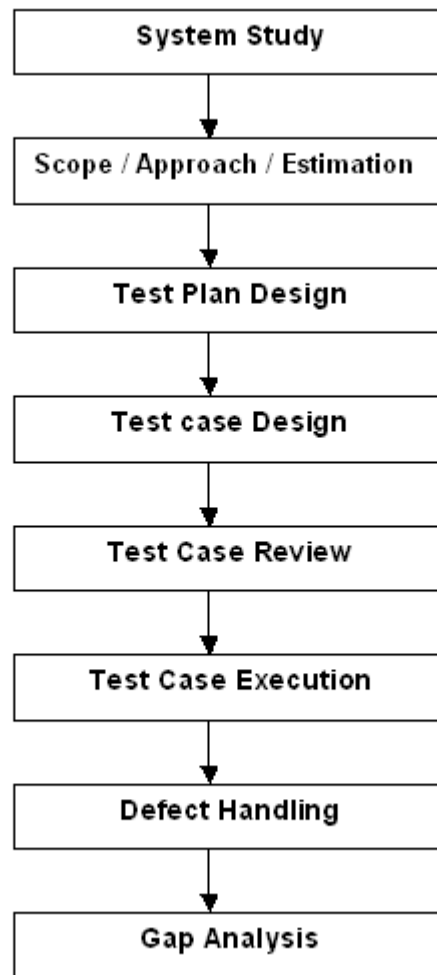
Thatavarti Technologies is an Independent Testing Vendor serving development companies to deliver the bug free software. At the time of Inception TT strength was 10 and now we have grown to 73 serving for 10 different clients. *This engagement with is an opportunity for TT to prove its Testing capabilities.*

## 2. Acronyms

Abbreviation	Description
SME	Subject Matter Expert
KT	Knowledge Transfer
TT	Thatavarthi Technologies
T	Testing
TC	Test Case
TS	Test Scenario
TP	Test Plan
w.r.t.	With respect to
SPOC	Single Point of Contact
4L	4 Layered
TM	Test Management
MPP	Microsoft Project Plan

### 3. Thatavarti Solution

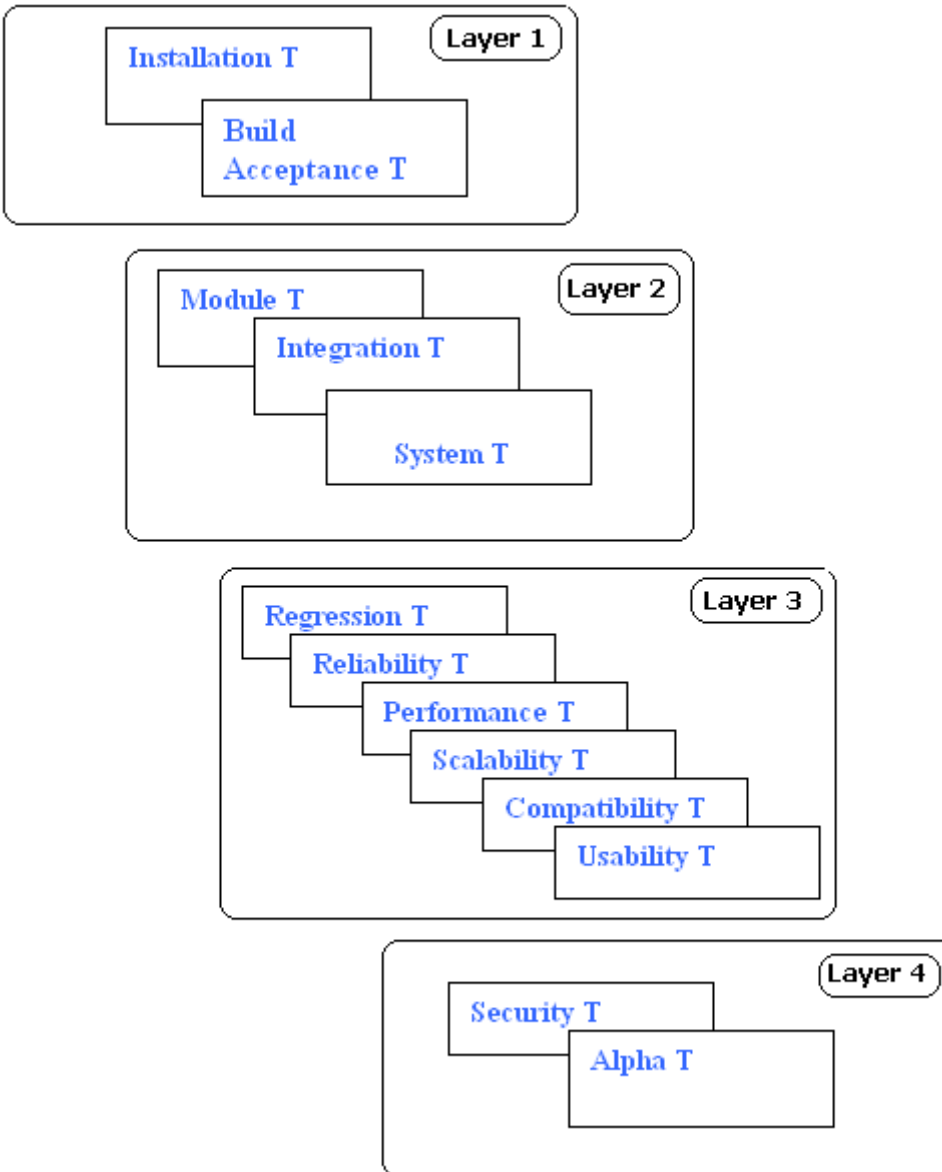
TT will implement 8-step approach for execution.



Appendix 01: Test Life Cycle

### 3.1 Test Strategy:

TM is a part of Test Plan. TT implements **4L** Product Testing Methodology to test the CLIENT products.



4-Layer Product Testing Architecture

**3.1.1 Installation Testing**

Installation testing is defined as testing that occurs outside the development environment. This will occur on the computer system, the software product will eventually be installed on. Installation tests will check the installation and configuration procedure as well as any missing dependencies.

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
Installation source of the application should be available.	Execute the heuristic Checklist (Check List checks for all possible defects in the Installation.)  Please find the check list in Appendix 02.	Status of the filled Check List.	Check the Proof of execution in Check List. Successful installation message.	Successful installation of software.

**Appendix 02: Installation Testing Checklist**

**3.1.2 Build Acceptance Test:**

The build acceptance test is a simplistic check of a product's functionality in order to determine if the product is capable of being tested to a greater extent. Every new build should undergo a build acceptance test to determine if further testing can be executed.

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
New build received with release notes.	Build acceptance test case execution	Execution results	Review, Execution result documents	Stable build, Pass status for all test cases.

### 3.1.3 Unit / Module Testing

Unit testing is a verification effort on the smallest unit of the software design – the software component or module. Unit testing is white-box oriented, and this can be conducted in parallel for multiple components.

Unit testing is mainly focused on following areas

**“White box”**

- Statement testing
- Branch / Decision testing
- Data flow testing
- Branch condition testing
- Branch condition combination testing
- Modified condition decision testing

**“Black box”**

- Equivalence partitioning
- Boundary value analysis
- State transition testing
- Cause-effect graphing

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
Availably of modules. Stable build	Preparation of Unit Test Plan. Preparation of Unit Test cases. ‘Preparation’/ ‘CLIENT shall Provide’ Test data for Unit test cases Execution of test cases.	Unit test plan Document Unit Test Case Document. Test data document for Unit Test cases Test Status report.	Review of Unit Test Plan Document Traceability Matrix for the all Unit test cases. Review of Test data Document Success messages for passed test cases and screen shots for failed test cases.	Sign off Unit Test plan Document. Sign off Unit test Case Document. Frozen Test data Document 2 rounds of test execution.

### 3.1.4 Integration Testing

The primary objective of integration testing is to discover errors in the interfaces between Modules / Sub-Systems.

Techniques/Approaches for Integration testing:

**Top-Down approach:** is an incremental approach to testing of the program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main control module, this could be done as depth- first or breadth-first manner.

**Bottom-up approach:** as the name implies, begins construction and testing with atomic modules i.e., from the components at the lowest levels in the program structure

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
Unit testing of all Modules is Signed off	Preparation of Integration Test Plan.  Preparation of Integration Test Scenarios.  Execution of test Scenarios.	Integration Test Plan Document  Integration Test scenarios Document.  Test Status report.	Review of Integration Test Plan  Traceability Matrix for the all Integration test Scenarios  Success messages for passed test scenarios and screen shots for failed test scenarios	Sign off Integration Test plan Document.  Sign off Integration test Scenarios Document.  Sign off Integration testing.

**3.1.5 System Testing**

The primary objective of system testing is to discover errors when the system is tested as a whole. System testing is also called as End-To-End Testing.

System testing is mainly focused on following areas

Identifying the End-To-End / Business Life Cycles.

Design the test and data.

Optimize the End-End / Business Life Cycles.

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
Successful completion of integration testing.	Preparation of System Test Plan.  Preparation of System Test cases.  'Preparation'/ 'CLIENT shall provide test data for System test cases  Execution of test cases.	System Test Plan Document  System Test Case Document.  Test data document for System Test cases  Test Status report.	Review of system test plan document  Traceability matrix for the all System test cases.  Review of test data document for system testing  Success messages for passed test cases and screen shots for failed test cases	Sign off system test plan document.  Sign off System test scenarios document.  Frozen test data Document  Sign off system testing.

**3.1.6 Regression Testing:**

Testing the new version of product/ new build to ensure enhancements implemented correctly and existing functionalities are stable and not corrupted.

**Regression Testing Methods**

Regression testing can be done either manually or by automated testing tools.

*Manual testing* will be done for small systems, where investing in automated tools might not be feasible enough.

*Automated testing:* One class of these tools is called as *Capture-playback* tool. This is very helpful in situations where the system undergoes lots of version changes.

Regression testing are basically repetitive tests for which automation may be a good approach

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
Testing of change requirements are signed off	Executing the Regression Test cases	Regression Test and Defect Report	Test case review document. Message of pass test cases and screen shots for fail scenarios.	All Regression Test Cases are executed Status report Document

**3.1.7 Reliability Testing**

**Reliability Testing** is a property, which defines how well the software meets its requirements. Reliability is considered as the probability of failure-free operation for a specified time in a specified environment for a given purpose

Objective is to find Mean Time between failure/time available under specific load pattern. Mean time for recovery.

**Reliability Testing helps you to confirm:**

- Business logic performs as expected
- Active buttons are really active
- Correct menu options are available
- Reliable hyper links

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
Functionality testing is signed-off.	Preparation of Reliability test plan. Preparation of Reliability Test cases. Execution of test cases.	Test Plan document. Test Case document. Test Execution Report.	Test Plan review document Traceability Matrix and Test Case review document Success messages for passed test cases. And screen shots for failed test cases.	Testing baselines, schedules are planned and test plan document is frozen. Business critical test cases are Signed-off. 85% stability of the application.

### 3.1.8 Performance Testing

**Definition:** Demonstrating system functions to specifications with acceptable response time while processing the required transaction volume on a production sized database.

#### Performance Test Plan

Entry Criteria	Activities	Deliverables	Proof of Execution	Entry Criteria
Stable Architecture System testing is completed	Preparation Of Test plan Document	Test Plan Document	Test Plan Review document	Performance baselines identified. Architecture critical scenarios are identified.

**Smoke Testing:** Initial testing implemented on the Application to check the performance before going to exploratory test.

#### Performance Smoke Test

Entry Criteria	Activities	Deliverables	Proof of Execution	Entry Criteria
Performance test plan is ready. Availability of scripts for Execution	Testing the application with 2 users	Transaction per second, No. of transactions passed/Failed	Summary Report containing Date, Start and Stop time of test and duration of test.	Completion of Single iteration of test

**Load Testing / Exploratory :** Testing application with the load the customer wants to have on his application

#### Performance Load/Exploratory Test

Entry Criteria	Activities	Deliverables	Proof of Execution	Entry Criteria
Successful Execution of Smoke test	Testing the application with number of user as per Client Requirement	Analysis, Summary Report, Reports Graphs.	Summary Report containing Date, Start and Stop time of test and duration of test.	Completion of script execution

**Stress Testing:** Testing Application with an intention to find the break point by implementing heavy load  
**Performance Stress Testing**

Entry Criteria	Activities	Deliverables	Proof of Execution	Entry Criteria
Successful Execution of Load / Exploratory test	Testing the application with Heavy load of user, to find Application break point.	Analysis, Summary Report, Graphs.	Summary Report containing Date, Start and Stop time of test and duration of test.	Completion of script execution.

**3.1.9 Scalability Testing:**

**Objective is to find the maximum number of user system can handle.**

A test that applies increasing workloads to determine a system's ability to scale. This test answers the question, "Given an increase from load x to load y, how will the system behave at load y versus at load x?"

**Classification:**

**Network Scalability**

**Server Scalability**

**Application Scalability**

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
After successful completion of functionality testing and all required non-functional testing.	Preparation of Scalability Test Plan.  Identify scalability test scenarios.  Execute the application on different Networks and servers increasing the number of users with adequate and inadequate resources.	Test Plan Document.  Test case Document.  Test Status report.	Test Plan Review Document  Test scenarios  System behavior of different loads.	Test Plan Document signoff.  All Test scenarios are executed  Status report Document

**3.1.10 Compatibility Testing**

Compatibility testing provides a basic understanding of how a product will perform over a wide range of hardware, software & network configuration and to isolate the specific problems. Compatibility testing verifies that a product looks and functions the same across all supported environments.

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
After successful completion of functionality testing and all required non-functional testing.	Preparation of Compatibility Test Plan. Preparation of Compatibility Test cases. Execute the application on different browsers and operating Systems Execute the application with <b>Test Bed Creation</b> i.e. 1) Partition of the hard disk 2) Creation of Base Image	Test Plan Document.  Test case Document.  Test Status report.	Test Plan Review Document  Test Case review document  Success message for pass cases and screenshots for fail cases.	Test Plan Document signoff. All Test Cases are executed  Status report Document

**3.1.11 Usability Testing:**

Usability Testing is defined as the effectiveness, efficiency, and satisfaction with which specified users achieve specified goals in particular environments. Evaluates the ease of Using, learning the system, system user documents, effectiveness of system functioning in supporting User tasks by the end users.

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
CLIENT shall make the application available	To test the, User friendliness of the product includes, Test case execution.	Usability Test case Document,  Test Status Report	Test case Review Document  Not Applicable	Test Case Document  Product should be user friendly

**3.1.12 Security Testing:**

To test the application and system level accessibility correctness.

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
Product is stable in terms of Functionality and Performance	Verify that an actor can access only those functions or data for which their user type is provided permissions. System-level Security: Verify that only those actors with access to the system and applications are permitted to access them	Authorization of Authentication Test Case execution reports.	Review Documents.  Screenshots of Pass and Fail results.	All Test Cases pass.

**3.1.13 Alpha Testing:**

- Alpha Testing is carried out on the developer's premises itself in a **controlled** environment.
- Generally, the Quality Assurance cell is the body that is responsible for conducting the Alpha test.
- On successful completion of this phase, the software is ready to migrate outside the developer's premises for more rigorous and unplanned testing that takes place at the next level of testing, i.e., Beta testing.

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
System testing is signed-off and completion of required non-functional testing of the product is complete.	Test Plan creation Test scenario identification for major business functionalities Preparation of test cases for major business requirements. Execution of test cases.	Test plan document. Business Test scenarios document.  Test case document. Test Execution Report.	Test Plan review document Test Scenario review document Traceability Matrix and Test case review document Success messages for passed test cases. And screen shots for failed test cases.	Testing baselines and schedules are planned and Test plan document is frozen.  Business Critical Test Scenarios are signed-off.  Business critical test cases are signed-off.  Alpha testing signed off for Beta testing.

**3.1.14 Ad-hoc Testing:** Ad-hoc testing will be carried out with out having any specific objective; test resource will test the application with the domain knowledge and testing experience.

Test Scenarios:

1. Test scenarios will be identified for each module and product.
2. Test scenarios will be associated with both positive and negative data
3. Optimization of test scenarios will be taken care as the test cycles increase

Test Cycle:

1. Test cases and Test scenarios will be base lined for each test cycle
2. Test execution reports and metrics will be maintained independently for each cycle

### **3.1.15 Test Automation:**

Test automation is code or script, which executes tests without human intervention.

Software testing that utilizes a variety of tools to automate the testing process and when the importance of having a person manually testing is diminished. Automated testing still requires a skilled quality assurance professional with knowledge of the automation tool and the software being tested to set up the tests.

#### Why Automation:

- Avoid the errors that humans make when they get tired after multiple repetitions. The test program won't skip any test by mistake
- Each future test cycle will take less time and require less human intervention
- Required for Regression Testing

Life Cycle of Automation:

- Analyze the Application
- Select the Tool
- Identify the Scenarios
- Design/Record the Script
- Modify the Script
- Run the Test Script
- Reporting the Defects

#### Benefits of Test Automation:

- Allows more testing to happen
- Tightens / Strengthen Test Cycle
- Testing is consistent, repeatable
- Useful when new patches released
- Makes configuration testing easier
- Test battery can be continuously improved.

Entry Criteria	Activities	Deliverables	Proof of Execution	Exit Criteria
Application should be 80% stable	Analyze the Application Select the Tool Identify the Scenarios Develop the Test Framework Design/Record the Script Modify the Script Run the Test Script	Automation Framework  Automation Scripts  Test Logs	Review of Automation Framework  Review of Test Scripts  Test Logs	Developed Framework  Executable Scripts  Test logs

#### 4. Project Plan

Project plan will be designed in MPP with the help of the following:  
Defined activities.

**5. Risk, Contingencies, Assumptions, Dependencies and Constraints**

#	Risk	Mitigation/Contingency Plan	Impact on Schedules and Cost
1	Acceptance plan may not be available on time from client.	Project managers to plan and follow up with Client project coordinator	Delay in providing this might result in stakeholder conflicts.
2	Required information may not be provided by client to set up the functional environment (Hardware, Software and installation Licenses, etc).	Project Managers to plan and follow up with Client project coordinator	Delay in the starting up the project resulting in schedule and cost over runs
3	Changes to original scope/agreed design/test cases.	Project Managers to plan and follow up with Client project coordinator	Delay in the overall project schedule and cost over runs
4	Changes to the application and test environment may be frequent.	Project Managers to plan and follow up with Client project coordinator	Delay in the overall project schedule and cost over runs
5	Knowledge transfer may not happen as planned.	Project Managers to plan and follow up with Client project coordinator	Delay in the overall project schedule and cost over runs
6	Delay in formal sign-off from the client at agreed major milestones.	Project Managers to plan and follow up with Client project coordinator	Might result in major re-work
7	Client may not provide committed resources on time.	Project Managers to plan and follow up with Client project coordinator	Onsite project manager to plan and follow up with Client project coordinator
8	Installation of application may cause issues.	Onsite project manager to manage this through formal change request procedure	Might result in re-work which can lead to schedule and cost over runs

**5.1 Assumptions**

- ❖ CLIENT will provide the independent environment for testing
- ❖ Application Developer(s) and Business Analysts will review the System Test Plan and provide feedback and help identify scenarios covering new functional and design requirements.
- ❖ A stable QA environment is available – which mirrors the production environment on which the application will run.
- ❖ Build verification testing is performed on all iterations of the code, prior to full-scale regression testing.
- ❖ Subsequent iterations of code will be regression tested against the previous build.
- ❖ The Developer(s)/Business Analysts will review the Test Scenario document and suggest inclusion of any additional test scenarios to achieve full coverage of the application functionality.
- ❖ All the new builds will be deployed on testing environment
- ❖ CLIENT will share existing test suite

Man efforts estimated for each activity.

Resources

Appendix 09: Detail Project Plan (Test Plan)

## 5.2 Dependencies

This table lists the identified system dependencies and respective owners for testing of product. The logical owner for investigation, tracking and reporting of these is the assigned QA resource.  
<List the system dependencies. For example is data loaded from another application or fed into another application that will need consideration in the test planning The following table should be completed to show what the dependencies are and who owns them. For example for a feed to an external system, what is the system name and who is the point of contact for configuration issues. These dependencies should be inserted in the Test Director test labs in the details section as pre-conditions of execution.>

Dependency	Owner
Network Connection.	<b>Technical team of CLIENT</b>
Data base server	Database tam, CLIENT
Development completion	Dev team CLIENT
Unit testing should be complete to start integration testing.	Testing team TT.

## 5.3 Constraints

- a) Complete test data should be made available before testing

## 6. Version Management Procedure

### Version Management:

Version management is storage of complete history of the versions of each component. Supports a concept of workspaces as directories and which files are extracted for modification or viewing purposes.

- Ensures old versions are not lost.
- Ensures developers do not overwrite each other changes.
- Allow you to examine the differences between versions.
- Provides the essential information of Who, What, When and Why.
- Provides permissions to particular users allow to allocated projects.
- All kinds of document tracking

### Quality point of view:

- The purpose of Version management is to establish and maintain the integrity of the products of the software project throughout the software life cycle.

### Procedure:

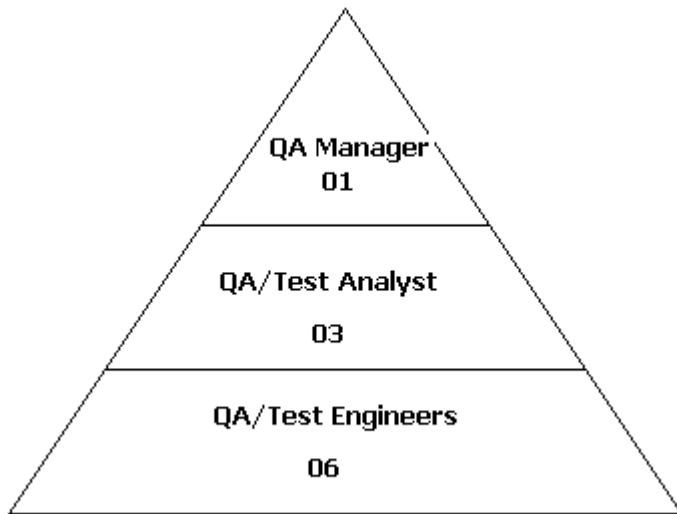
1. Identify the software that needs version control.
2. Identification of the configuration items (Individual program or Any document that needs version control)
3. Identifying the users and granting the privilege to access the version management folders.
4. Defining the tree structure of folders for the project.
5. Define the naming convention for the child items in the tree hierarchy.
6. Naming conventions for the configuration items (files).
7. Maintaining all documents in respective folders with increasing version numbers in as part of name.
8. Should maintain baseline documents.
9. Shared access to be granted to users if a folder and its items are shared amongst different projects.

**7. Training**

TT will train the CLIENT resource in the Inception stage of the Project.  
 TT will share the best practices, case studies and learning's with the CLIENT resources.  
 TT will conduct the trainings to CLIENT resource monthly and need basis.

**8. Team Structure**

This section presents the recommended number of resources for the CLIENT Assignment, their main responsibilities, and their knowledge or skill set.



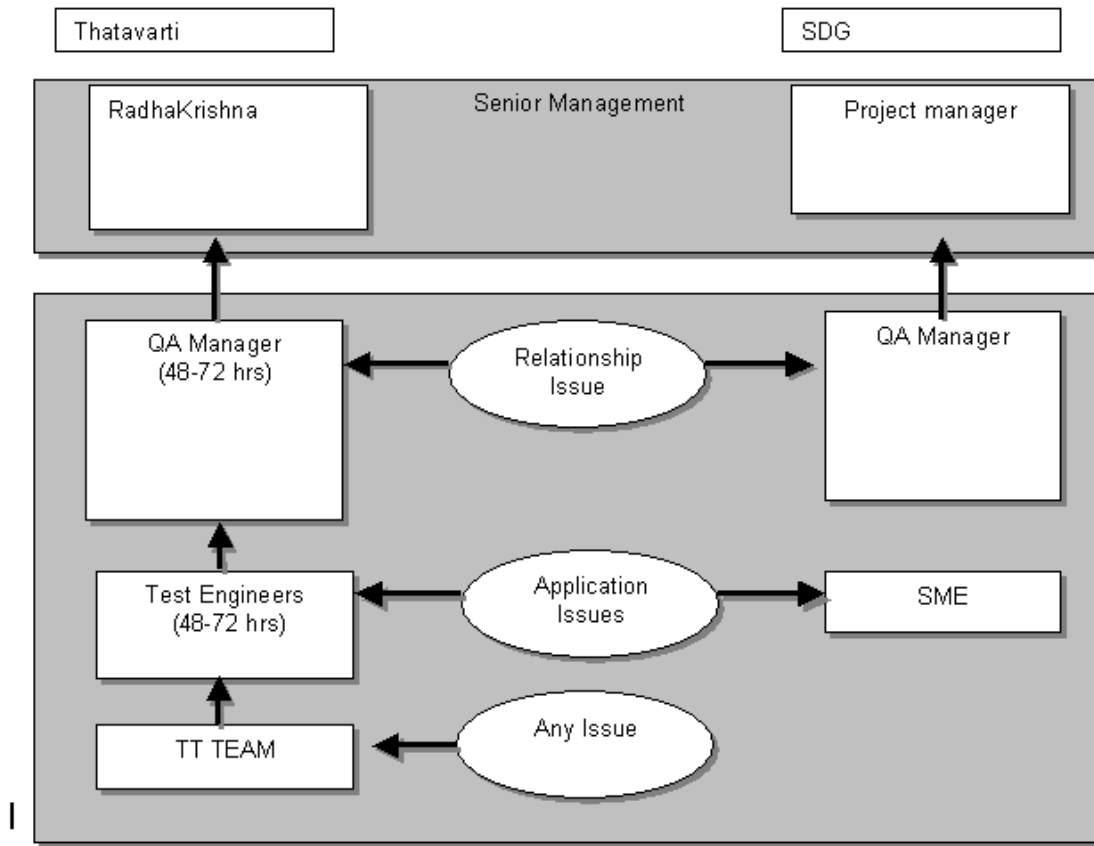
<p>QA/Test Manager  <a href="#">Mail ID</a></p>	<ul style="list-style-type: none"> <li>Frame work Implementation</li> <li>Maintaining multiple projects</li> <li>Test planning</li> <li>Update the status to CLIENT manager</li> <li>Mentor test leads and test members</li> <li>Design and maintain metrics</li> <li>Raise issues and variance</li> <li>Understand and create service level agreements</li> <li>Project reviews and execution reviews</li> <li>Propose the new techniques</li> <li>Define the automation strategy</li> <li>Fill time sheets</li> <li>Update status to Thatavarti</li> </ul>
---	--

QA/Test Analyst <a href="#">Mail ID</a>	Implement Test Plan System study Design Test Cases Participate in test reviews Test case execution Identify test scenarios Defect management Update project metrics Prepare test data Guide the test members Define the scope Raise the issues Fill time sheets
QA/Test Engineer <a href="#">Mail ID</a>	System study Test case design Test case execution Submit defects Track defects Raise issues Prepare test data Peer reviews Update status Fill time sheets

## 9. Processes and Procedures

- 9.1 Appendix 04: System Study Procedure
- 9.2 Appendix 05: Issues & Doubts Procedure.

9.3 Escalation Procedure



9.4 Appendix 01: Test Case Design Procedure

9.5 Appendix 01: Review Procedure

9.6 Appendix 01: Defect Tracking Procedure

9.7 Appendix 06: Test Data procedure

**10. Communication Channel**

**A. Offshore (TT, Hyderabad)/ Onsite(CLIENT) Team Collaboration**

Thatavarti Technologies lays heavy emphasis on communication among global team members. It has been Thatavarti's experience that effective communication is essential for the success of the project

**B. Communication Practices**

To ensure effective communication between offshore and onsite teams, Thatavarti Technologies adopts multiple tools and processes, such as regular account calls and status review calls though voice conferencing. The knowledge transfer activities are given most importance during the inception phase as well as during subsequent phases to ensure that the offshore and onsite teams are completely synchronized. This is supported by activities such as onsite visits, joint workshops with the client teams etc.

Typical Project Communication modes are:

- Monthly account meetings attended by personnel from sales, delivery, and business unit head and senior management representative. These meetings review progress and issues related to the project.
- Daily project review meetings attended by project manager leads and key engineers to discuss day-to-day progress and issues.

## 11. Status Reporting

Thatavarti Technologies will provide the status reports frequently to CLIENT. The status report will contain but is not limited to the following items:

- Daily status reports will be provided with the progress of task.
- Weekly status report will be provided with the Planned/Actual/Pending tasks and also forecast for the coming week.
- Monthly status report will be maintained to understand the following things:
  - Planned task progress
  - Issues related to the task
  - Risks involved in the execution of the task
  - Need for the allocation/de-allocation of resource

## 12. Knowledge Management

TT does the knowledge management of domain, technology and execution of the project on Weekly, Monthly and Quarterly basis using the following techniques.

Team will deliver the presentations on the understanding of domain knowledge of the application.

Team will share all the functional problem areas and team issues.

Team will develop and share the best practices.

TT will deploy a new back-up resource for every 20 working days, to retain domain knowledge.

## 13. Case Studies

Appendix 12

## 14. Retention Policy

TT shares 20% equity with the managers and 18% equity with the employees.

TT treats all employees as partners.

TT evaluates resources on quarterly basis and does appraisal

TT encourages all the employees to take the ownership of the project.

## 15. Why Thatavarti?

- ❖ Better Quality will be delivered with Domain/Technology/Tools Expertise
- ❖ Faster turnaround with the help of parallel work on Verification & Validation.
- ❖ Experienced test teams and round the clock testing.
- ❖ Lower cost with Onsite/Offshore combinations and complete offshore facility

## 16. Glossary

Appendix 11

## 17. Test Metrics

Appendix 10

## 18. Templates and Formats

Appendix 07: Test case template

Appendix 08: Test scenario template

Appendix 09: Test plan template

Appendix 13: Traceability matrix

Appendix 14: Defect report

Appendix 15: Test summary report