**Deployment Package**

**Software Testing and Integration**

**Basic Profile**

**Notes:**

This document is the intellectual propriety of its author’s organization. However, information contained in this document is free of use. The distribution of all or parts of this document is authorized for non commercial use as long as the following legal notice is mentioned:

© ParqueSoft Colombia

Commercial use of this document is strictly forbidden. This document is distributed in order to enhance exchange of technical and scientific information.

This material is furnished on an “as-is” basis. The author(s) make(s) no warranties of any kind, either expressed or implied, as to any matter including, but not limited to, warranty of fitness for purpose or merchantability, exclusivity, or results obtained from use of the material.

The processes described in this Deployment Package are not intended to preclude or discourage the use of additional processes that Very Small Enterprises may find useful.

|  |  |
| --- | --- |
| **Author** | Liliana Gómez Arenas, ParqueSoft - Colombia |
| **Editors** | C. Y. Laporte, École de technologie Supérieure, Canada  ANA VAZQUEZ – 5th level, (México) |
| **Creation date** | 07/02/2008 |
| **Last update** | 5 March 2012 |
| **Status** | Draft |
| **Version** |  |

Versions

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Modification** |
| 07/02/2008 | 0.1 | L. GOMEZ A. | Document creation (Spanish) |
| 12/02/2008 | 0.2 | P. HOYOS C. | Comments on document structure |
| 08/03/2008 | 0.3 | L. GOMEZ A. | Update and completion of many sections |
| 10/04/2008 | 0.4 | P. HOYOS C. | Update of graphical representation of steps. |
| 22/05/2008 | 0.5 | L. GOMEZ A. | Translation to English |
| 14/06/2008 | 0.6 | L. GOMEZ A. | Fist part of Update to comments from Berlin meeting to reduce complexity |
| 08/07/2008 | 0.7 | P. HOYOS C. | Second part of Update to comments from Berlin meeting |
| 16/07/2008 | 0.8 | P. HOYOS C. | Final part of Update to comments from Berlin meeting |
| 14/10/2008 | 0.9 | L. GOMEZ A. | Final version – ready for final review (in Spanish) |
| 14/05/2009 | 1.0 | L. GOMEZ A. | Preliminary version after reviews on Mexico. |
| 18/05/2009 | 1.1 | L. GOMEZ A. | Final version after reviews on Mexico. |
| 29/05/2009 | 1.2 | L. GOMEZ A. | Modifications to Abbreviations, Annex B; deletion of Annex about Training material. |
|  |  |  |  |

Abbreviations/Acronyms

|  |  |
| --- | --- |
| **Abre./Acro.** | **Definition** |
| DP | Deployment Package - a set of artefacts developed to facilitate the implementation of a set of practices, of the selected framework, in a Very Small Entity. |
| VSEs | Very Small Entities – enterprises, organizations, departments or projects having up to 25 people. |
| VSE | Very Small Entity |

Table of Contents

[1. Technical Description 4](#_Toc236631684)

[Purpose of this document 4](#_Toc236631685)

[Why Software Testing is Important ? 4](#_Toc236631686)

[2. Definitions 5](#_Toc236631687)

[Generic Terms 5](#_Toc236631688)

[Specific Terms 5](#_Toc236631689)

[3. Relationships with ISO/IEC 29110 7](#_Toc236631690)

[4. Detailed Description of Process, Activities, Tasks, Steps and Products 9](#_Toc236631691)

[Process: 4.2 Project Management 9](#_Toc236631692)

[Activity: PM.1 Project Planning 9](#_Toc236631693)

[Task: Test Planning 9](#_Toc236631694)

[Process: 4.3 Software Implementation 10](#_Toc236631695)

[Activity SI.3 Software Architectural and Detailed Design 10](#_Toc236631696)

[Test Design - Definition of Test cases and Test Procedures 10](#_Toc236631697)

[Task: Test Design - Verification of the *Test Cases and Test Procedures.* 12](#_Toc236631698)

[Activity: SI.5 Software Integration and Tests 13](#_Toc236631699)

[Task: Perform tests using Test Cases and Procedures for Integration 13](#_Toc236631700)

[Roles 16](#_Toc236631701)

[Product Description 17](#_Toc236631702)

[5. Template 18](#_Toc236631703)

[Table of Contents Template – Testing Plan 18](#_Toc236631704)

[Test Specifications Template 18](#_Toc236631705)

[Software Product/Component Functional Breakdown Template 20](#_Toc236631706)

[Tests Cases Matrix Template 20](#_Toc236631707)

[Table of Contents Template – Test Report 20](#_Toc236631708)

[6. Example of Activity lifecycle 21](#_Toc236631709)

[Example of Testing Practices Lifecycle 21](#_Toc236631710)

[7. Checklist 22](#_Toc236631711)

[8. Tool 23](#_Toc236631712)

[9. References to Other Standards and Models 24](#_Toc236631713)

[ISO 9001 Reference Matrix 24](#_Toc236631714)

[ISO/IEC 12207 Reference Matrix 25](#_Toc236631715)

[CMMI Reference Matrix 25](#_Toc236631716)

[10. References 27](#_Toc236631717)

[11. Evaluation Form 28](#_Toc236631718)

# 1. Technical Description

## Purpose of this document

This Deployment Package (DP) supports the Basic Profile as defined in ISO/IEC 29110 Part 5-1: Management and Engineering Guide. A DP is a set of artefacts developed to facilitate the implementation of a set of practices in a Very Small Entity (VSE). A DP is not a process reference model (i.e. it is not prescriptive). The elements of a typical DP are: description of processes, activities, tasks, roles and products, template, checklist, example, reference and reference to standards and models, and tools.

The content of this document is entirely *informative*.

This document has been produced by Liliana Gómez Arenas and her testing team from GreenSQA S.A.– company of ParqueSoft Colombia, responsible for leading the software strategy of the technological development cluster ParqueSoft Colombia, [www.greensqa.com](http://www.greensqa.com) – [www.parquesoft.com](http://www.parquesoft.com), beyond its official participation to ISO JTC1/SC7/WG24.

## Why Software Testing is Important ?

In present times, quality is a determining factor in the success of software projects. The history of software projects that have not been accepted by the client due to the low reliability of the products is the reason why it has become the center of attention of many organisms, which have engaged in production of standards with the purpose of making products that really comply with requirements demanded by the users. To this end, several conventions have been made with the objective of creating attributes that ensure the release of competitive products in the market.

The objective of the integration tests is to verify if the components or subsystems interact correctly through their interfaces, internal as much external, they cover the established functionality, and they adjust to the requirements specified in the corresponding verifications.

# 2. Definitions

In this section, the reader will find two sets of definitions. The first set defines the terms used in all Deployment Packages, i.e. generic terms. The second set of terms used in this Deployment package, i.e. specific terms.

## Generic Terms

***Process:*** set of interrelated or interacting activities which transform inputs into outputs [ISO/IEC 12207].

***Activity:*** a set of cohesive tasks of a process [ISO/IEC 12207].

***Task:*** required, recommended, or permissible action, intended to contribute to the achievement of one or more outcomes of a process[ISO/IEC 12207].

***Sub-Task:*** When a task is complex, it is divided into sub-tasks.

***Step:*** In a deployment package, a taskis decomposed in a sequence of steps.

***Role***: a defined function to be performed by a project team member, such as testing, filing, inspecting, coding. [ISO/IEC 24765]

***Product:*** piece of information or deliverable that can be produced (not mandatory) by one or several tasks. *(e. g. design document, source code)*.

***Artefact:*** information, which is not listed in ISO/IEC 29110 Part 5, but can help a VSE during the execution of a project.

## Specific Terms

Software Quality Assurance: a planned and systematic pattern of all actions necessary to provide adequate assurance that the item or product conforms to established technical requirements.[IEEE Std 730-1998]

Software Testing: The process of operating a system or a component under specified conditions, observing or recording he results, and of conducting an evaluation of a certain aspect of the system or component. [IEEE]

Software Testing Plan: will identify the kinds of testing that are to be performed. The Test Plan also establishes coverage requirements.

Software Tests Cases: criteria for validation of all required software work products (for a specific intended use)

Perform Software Test: conduct test plan using identified techniques, processes, and test Cases against functional and non-functional requirements and quality standards.

Tool for the automation of Software Testing

Software Testing Environment: supported software test environment which satisfies the testing requirements and in which changes must be controlled to allow regression if necessary.

Defects: non-compliances found during any type of software test.

# 3. Relationships with ISO/IEC 29110

This deployment package covers the activities related to requirements analysis of the ISO Technical Report ISO/IEC 29110 Part 5-1 Engineering and Management Guide – Basic VSE Profile.

In this section, the reader will find an overview of Project Management (PM) and Software Implementation (SI) process, activities, tasks and roles that are directly related to software Testing. These topics will be detailed in details in the next section.

* **Process: 4.2[[1]](#footnote-1) Project Management (PM)**
* **Activity: PM.1 Project Planning**
* **Task:**

|  |  |
| --- | --- |
| **Tasks** | **Roles[[2]](#footnote-2)** |
| PM.1.3 Identify the specific tasks to be performed in order to produce the deliverables and their software components identified in the *Statement of Work.* Include verification, validation and reviews with Customer and Work Team tasks to assure the quality of work products. | PM, TL |

* **Process: 4.3[[3]](#footnote-3) Software Implementation (SI)**
* **Activity: SI.3 Software Architectural and Detailed Design**
* **Tasks:**

|  |  |
| --- | --- |
| **Tasks** | **Roles** |
| SI.3.5 Establish or update *Test Cases and Test Procedures* for integration testing based on *Requirements Specification* and *Software Design.*  Customer provides testing data, if needed. | DES |
| SI.3.6 Verification of the *Test Cases and Test Procedures.*  Verify consistency among *Requirements Specification*, *Software Design* and *Test Cases and Test Procedures*. The results found are documented in a *Verification Results* and corrections are made until the document is approved by AN. | DES, AN |
| SI.3.7 Update the *Traceability Record* incorporating the *Test Cases and Test Procedures.* | DES |
| SI.3.8 Incorporate *Test Cases, Test Procedures and Traceability Record* to the *Software Configuration* as part of the baseline*.* | TL |

* **Process:** 4.3[[4]](#footnote-4) Software Implementation (SI)
* **Activity:** SI.5 Software Integration and Tests
* **Tasks:**

|  |  |
| --- | --- |
| **Roles** | **Task** |
| PR | SI.5.2 Understand *Test Cases* *and* *Test Procedures.*  Set or update the testing environment. |
| PR | SI.5.3 Integrates the *Software* using *Software components* and defines or updates Test Cases and Test Procedures for integration. |
| TSTR  CUS | SI.5.4 Perform tests using *Test Cases and Test Procedures* for integration and document results in *Test Report*. |
| PR | SI.5.5 Correct the defects found until successful test (reaching exit criteria) is achieved. |

# 4. Detailed Description of Process, Activities, Tasks, Steps and Products

## Process: 4.2 Project Management

The purpose of the Project Management process is to establish and carry out in a systematic way the tasks of the software implementation project, which allows complying with the project’s objectives in the expected quality, time and costs.

### Activity: PM.1 Project Planning

#### The Project Planning activity documents the planning details needed to manage the project. The activity provides, among others:

* Project quality assurance strategy planning through verification and validation of work products/deliverables.

### Task: Test Planning

PM.1.3 Identify the specific tasks to be performed in order to produce the deliverables and their software components identified in the *Statement of Work.* Include verification, validation and reviews with Customer and Work Team tasks to assure the quality of work products.

|  |  |
| --- | --- |
|  | |
| ***Objectives:*** | To Identify the specific tasks to be performed in order to verify, validate and review with Customer and Work Team tasks to assure the quality of work products. |
| ***Rationale:*** | This allows identifying the required types of tests, the adequate work scheme for the testing procedure, the work methodology, the tools that must be used, the necessary recourses and effort, and the deliverables in the testing procedure. |
| ***Roles:*** | Project Manager |
| Technical Leader |
| ***Artefacts:*** | Test Tasks |
|  |
| ***Steps:*** | Step 1: Identification of functional or integration tests needs |
| Step 2: Estimation of testing effort |
| ***Step Description:*** | **Step 1: Identification of functional or integration tests needs:**  The Technical Leader must identify the necessary types of tests that allow evaluating software product quality. The selection is made intending to cover ISO9126 standard, where quality characteristics are defined. Functional tests are always included within the base testing group. Based on project conditions, integration, performance or security tests are selected.  The software testing procedure strategy must be complemented, identifying tools that will support the testing procedure management as well as the automation of testing execution activities. If the testing strategy includes performance tests, it is fundamental to identify a tool.  **Step 2: Estimation of testing effort:**  With the technical and functional knowledge of the product it is possible to estimate test design and execution efforts. In order to do this, it is recommended to consider the following criteria:   * knowledge of business model (high, medium, low) * expertise of programmer doing testing (high, medium, low) * development cycle of VSE (prototype, iterative, sequential) * service level agreement with the customer   Then project schedule must be updated |

## Process: 4.3 Software Implementation

The purpose of the Software Implementation process is the systematic performance of the analysis, design, construction, integration and tests activities for new or modified software products according to the specified requirements.

### Activity SI.3 Software Architectural and Detailed Design

The software architectural and detail design activity transforms the software requirements to the systems software architecture and software detailed design. In relation with testing, among others, the activity provides Verified Test Cases and Test Procedures for integration testing:

* Definition of Test cases and Test Procedures, Covering task SI 3.5, SI 3.7 of ISO/IEC 29110-5.1 for VSEs
* Verification of Test cases and Test Procedures, Covering task SI 3.6 of ISO/IEC 29110-5.1 for VSEs
* Incorporate the *Test Cases, Test Procedures and Traceability Record* to the *Software Configuration* as part of the baseline*,* Covering task SI 3.8 of ISO/IEC 29110-5.1 for VSEs

### Test Design - Definition of Test cases and Test Procedures

SI.3.5 Establish or update Test Cases and Test Procedures for integration testing based o

Requirements Specification and Software Design. Customer provides testing data, if needed.

SI.3.7 Update the *Traceability Record* incorporating the *Test Cases and Test Procedures.*

|  |  |
| --- | --- |
|  | |
| ***Objectives:*** | To identify testing conditions that will allow executing a testing procedure in an effective and efficient way. |
| ***Rationale:*** | Understanding functional tests as evaluations of the software product functioning result and integration testing as the procedure to verify correct operation of the existing interfaces between the different components and subsystems, this stage is deemed as the most critical one and determining of procedure success. It allows identifying testing conditions that must be executed in the software to have a real perception of the software product quality degree. These testing conditions must be associated with the quality characteristics defined in ISO9126. |
| ***Roles:*** | DES – Designer |
| ***Artefacts:*** | Testing Requirements Matrix |
| ***Steps:*** | Step 1: Analysis and Application of test design techniques |
| Step 2: Identification of tests cases |
| Step 3: Documentation of test design |
| ***Step Description:*** | **Step 1: Analysis and Application of test design techniques:**  Based on the product information analysis, the designer must identify system functional processes, sub-processes and activities that the software product must support. Additionally, the designer identifies software processes that may be subject to non-functional testing.  Then, guided by each of the product functional processes, the designer identifies and classifies software information input. Relevant information classification into expected software results is identified using test design techniques. It is recommended to use functional techniques when knowledge about software code is not yet available. (e.g. Equivalence Partition, Decision Tables, Threshold Analysis).  **Step 2: Identify and establish Tests Cases:**  Guided by the functional breakdown of the solution, and using test design techniques, the designer identifies the set of tests cases, conditions to test, that ensures a higher level of efficiency in the execution phase. For generic components of a software product (e.g. Basic Data Management, Securities Management, Report Generation, Process Auditing), requirements (tests cases) identified for other projects are used.  **Step 3: Documentation of test design:**  The purpose is to build the *Tests Cases Matrices* as the basic testing instrument. A consolidated of Tests Cases. A Test case is like a testing requirement that identifies a specific condition or aspect that needs to be verified or tested over a software unit or set of units developed or modified, to ensure their incorporation to the product has a high level of quality and complies with the specifications agreed with the user or the scope defined for such functionality. The identified test case must have a general description, object to be tested, type of test, and expected result.  Designer updates the Traceability Record incorporating the Test Cases and Test Procedures. |

### Task: Test Design - Verification of the *Test Cases and Test Procedures.*

SI.3.6 Verification of the *Test Cases and Test Procedures.*

SI.3.8 Incorporate the *Software Design, Test Cases, Test Procedures and Traceability Record* to the *Software Configuration* as part of the baseline*.*

|  |  |
| --- | --- |
|  | |
| ***Objectives:*** | To verify consistency among requirements specifications, software design, and test cases and test procedures. |
| ***Rationale:*** | The verification activity must ensure coverage and adequacy of the test design (test cases and test procedures) |
| ***Roles:*** | DES – Designer |
| AN - Analyst |
|  | TL - Technical Leader |
| ***Artefacts:*** | Testing Requirements Matrix |
| ***Steps:*** | Step 1: Review test cases and test procedures |
| Step 2: Document results |
| Step 3: Correct documents |
| ***Step Description:*** | **Step 1: Review and accept test design**  The designer must verify adequacy of the test design (test cases and test procedures) with the analyst. The verification activity must ensure coverage and adequacy of the test design. Design coverage is associated with two criteria:   * the degree of completeness in design processes of the software product and * coverage in quality characteristics of the test cases.   Adequacy of design is associated to veracity of the expected results recorded for the test cases. |
| **Step 2: Document results**  The results found are documented in a verification result document that can be a bug tracking tool or a simple spreadsheet matrix.  **Step 3: Correct documents**  The Designer will make corrections to test case and test procedures until the document is approved by Analyst then, Technical Leader incorporates the *Software Design, Test Cases, Test Procedures and Traceability Record* to the *Software Configuration* as part of the baseline*.* |

### Activity: SI.5 Software Integration and Tests

The software integration and test activity ensures that the integrated software components satisfy the software requirements. In relation with test, the activity provides:

* Understanding of Test cases and Test Procedures and the Integration environment, Covering task SI 5.2 of ISO/IEC 29110-5.1 for VSEs
* Defines or updates Test Cases and Test Procedures for integration, Covering task SI 5.3 of ISO/IEC 29110-5.1 for VSEs
* Perform tests using *Test Cases and Test Procedures* for integration and document results in *Test Report*, Covering task SI 5.4 of ISO/IEC 29110-5.1 for VSEs

### Task: Perform tests using Test Cases and Procedures for Integration

SI.5.2 Understand *Test Cases* *and* *Test Procedures.* Set or update the testing environment.

SI.5.3 Integrates the *Software* using *Software components* and defines or updates Test Cases and Test Procedures for integration.

SI.5.4 Perform tests using *Test Cases and Test Procedures* for integration and document results in *Test Report*.

|  |  |
| --- | --- |
|  | |
| ***Objectives:*** | To identify software product stability through obtained results, exposing the product to test cases designed for the testing procedure. |
| ***Rationale:*** | This allows executing different types of tests and identifying issues that must be corrected by the software development team. According to the testing plan, the different types of tests are executed in different points of time. |
| ***Roles:*** | Customer CUS |
| Tester TSTR |
| ***Artefacts:*** | Test Report |
|  | Test cases and test procedures updated (optional) |
| ***Steps:*** | Step 1: Preparation of software testing environment |
| Step 2: Execution of testing iterations |
| Step 3: Execution of regression tests |
| Step 4: Closure of testing procedure |
| Step 5: Document results in *Test Report* |
| ***Step Description:*** | **Step 1: Preparation of software testing environment**  During this step, conditions of the environment over which tests will be executed, readiness and availability of components (corrected or baselined) to be tested and data that will be used for this execution are prepared. The objective is to define a controlled and independent environment for the tests. Preparing the testing environment includes: Specifying machine configuration, operating system, browser and TCP/IP configuration when it applies; specifying system software, data base engine and testing support engine.  Regarding testing data, when the application is completely new, tests are executed in the sequence necessary to manually load initial data; otherwise, data from a client in operation, i.e., reproduction data, will be reused.  **Step 2: Execution of testing iterations**  Before execution it´s necessary to guarantee that assigned tester understands Test Cases and Test Procedures and if it is necessary, Designer must defines more or update Test Cases and Test Procedures.  A testing iteration corresponds to the execution of tests cases identified in the design phase. Testing iteration is done once the development team has executed unit testing. According to application stability, the tester may decide, in each iteration, executing all identified testing requirements or just a representative subset. The purpose is accomplishing an efficient benefit relationship between the invested effort in the execution and the number of issues.  Through the execution of tests cases, the tester finds deviations to the expected results, cataloged as product ‘Defects’. Product defects identified are typified and recorded in the testing follow-up and control report that may also be a buf tracking tool.  For Defect classification, the following can be taken into account:  Blockers: This type of issue stops a program/component operation of makes it yield results that prevent continuing operation.  Functional: It occurs when a program is executed and its results do not correspond to the expected result.  Presentation: Issues related to product presentation. These must adjust to the defined standards and the grammatical and orthographic rules of the language in which the program is showed.  **Step 3: Execution of regression tests**  These tests are executed based on the test cases identified in the design phase. The objective is to make sure that new or modified code (to provide solution to issues identified during tests) complies with the specified requisites and that non modified code has not been affected by the maintenance activity.  Regression tests initiate their execution as soon as the technical group delivers the issue solution or adds new characteristics to the software and these are applied in the testing controlled environment. To achieve an efficient procedure, it is possible to combine the regression test with a testing iteration.  **Step 4: Closure of testing procedure**  Once defined testing iterations have been completed and identified defects have been solved, an analysis to demonstrate that there is a *sustainable tendency to decrease* in the number of found defect, is carried out. The closure of testing procedure is done elaborating a closure report, which describes the executed procedure and shows conclusions with corresponding recommendations for process and product improvement. These recommendations make part of the lessons learned base of the VSE. The development team decides whether or not to release the software product, according to results obtained in the tests.  **Step 5: Document results in *Test Report***  The tester or programmer must prepare a progress test report for each testing iteration carried out as a minimum. The report must contain product current condition. The report must be reviewed by all members of the project team, and they must establish commitments for the new testing iteration. |

## Roles

This is an alphabetical list of the roles, its abbreviations and competencies description.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Role** | **Abbreviation** | **Definition and Competency** |
| 1 | Customer | CUS | Knowledge of the Customer processes and ability to explain the Customer requirements.  The Customer (representative) must have the authority to approve the requirements and their changes.  The Customer includes user representatives in order to ensure that the operational environment is addressed.  Knowledge and experience in the application domain. |
| 2 | Designer | DES | Knowledge and experience in the software components and architecture design.  Knowledge of the revision techniques and experience on the software development and maintenance.  Knowledge of the editing techniques and experience on the software development and maintenance.  Knowledge and experience in the planning and performance of integration and system tests. |
| 3 | Programmer | PR | Person in charge of the development of the software, and correct defects found during test activities |
| 4 | Tester | TSTR | Person in charge of the run of test cases and test procedures during test activities.  Knowledge of the functional testing techniques and experience on the software testing. |
| 5 | Technical Leader | TL | Knowledge and experience in the software development and maintenance. |

## Product Description

This is an alphabetical list of the input, output and internal process products, its descriptions, possible states and the source of the product

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Name** | **Description** | **Source** |
| 1 | Testing Plan | Document elaborated per specific project, which contains a detailed description of the testing strategy to be used. It includes, among other aspects: work methodology, activities to be executed, testing scope, designated recourses, estimated effort for each activity, working schedule, follow-up and review milestones, testing procedure closure criteria. | XXX |
| 2 | Functional Breakdown | A hierarchical model of the processes, sub-processes and functional activities of the application being tested. |  |
| 3 | Test Cases | A hierarchical model of the testing requirements identified, to ensure the quality of all the functional processes that the product being tested supports. |  |
| 4 | Test Report | Record of testing execution results in a specific environment. In general, the use of an issue traceability tool is recommended. |  |

# 5. Template

The templates provided us un example with this deployment package should be customized for your project.

## Table of Contents Template – Testing Plan

1. Objective …………………………………..
2. Work Scheme …………………………………..
3. Support Tool …………………………………..
4. Testing Strategy …………………………………..
5. Process Scope …………………………………..
6. Total Testing Effort Estimated …………….
7. Working Schedule ……………………….
8. Testing Process Deliverables …………….
9. Mutual Confidentiality Agreement …

## Test Specifications Template

To be used in an Excel sheet structured, for example, as:







## Software Product/Component Functional Breakdown Template

To be used in an Excel sheet structured, for example, as:



## Tests Cases Matrix Template

To be used in an Excel sheet structured, for example, as:



## Table of Contents Template – Test Report

1. Objective ………………………..…
2. Testing Process Current State ………………..
   1. Process Indicators ………………..
   2. Improvement Aspects ………………..
3. Product Current State ………………..
   1. Product Indicators ………………..
   2. Analysis of Issue Causes ………………..
4. Deliverables ………………..
5. Invested Effort Relation ………………..
6. Outstanding Activities ………………..
7. Conclusions ………….………………..

# 6. Example of Activity lifecycle

***Disclaimer***: *This section provides some graphical representations of examples of testing software practices lifecycle. These examples are provided to help the reader in the implementation of his own testing software lifecycle fitting his IT project’s context and constraints.*

### Example of Testing Practices Lifecycle

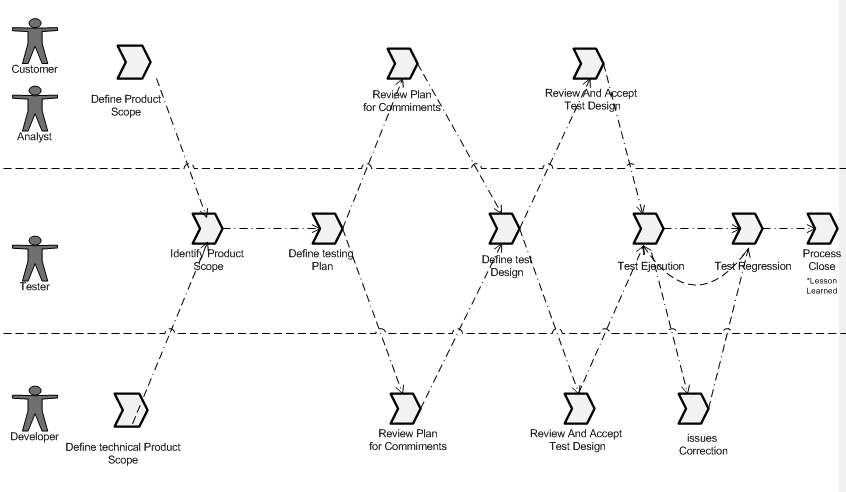


Figure 1 Example of Testing Practices Lifecycle

# 7. Checklist

Example of Checklist for review integration testing

# 8. Tool

To ensure traceability and control of product issues, it is necessary to implement a system, preferably a web system, that allows doing a follow-up of reported issues about a product, and processing requests associated to them.

As key features, this system must allow:

* Automatically assigning issues
* Attaching documents
* Generating automatic notifications via e-mail
* Tracing changes
* Generating simple and interactive queries
* Issue classification by: Type, Cause, Severity.

# 9. References to Other Standards and Models

This section provides references of this deployment package to ISO/IEC Standards and to the Capability Maturity Model IntegrationSM version 1.2 of the Software Engineering Institute (CMMI®[[5]](#footnote-5)).

Notes:

* This section is provided for information purpose only.
* Only tasks covered by this Deployment Package are listed in each table.
* The tables use the following convention:
* Full Coverage = F
* Partial Coverage = P
* No Coverage = N

## ISO 9001 Reference Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| **Title of the Task and Step[[6]](#footnote-6)** | **Coverage**  **F/P/N** | **Clause of ISO 9001** | **Comments** |
| Test Planning  Step 6: Definition of project schedule: | P | 7.3.1 The organization must plan and control the product design and development.  b) appropriate review, verification, and validation for every stage of design and development, and responsibilities and authorities for design and development. |  |
| Test Execution  Step 3: Execution of regression tests | P | 7.3.7 Changes in design and development must be identified and records must be kept. Changes must be reviewed, verified, and validated as appropriate, and approved before implementation. |  |
| Test Execution | F | 7.3.5 Verification must be carried out, according to what was planned (refer to 7.3.1), to ensure that design and development results comply with requirements of design and development input elements. |  |
| Process Feedback  Step 2: Calculation of indicators (Product – Process) | P | 8.2.4 The organization must measure and follow-up product characteristics to verify requirements are being complied with. This must be carried out in the appropriate stages of product execution process, according to the planned arrangements. |  |
| Test Execution  Step 4: Closure of testing process | P | 8.3 The organization must ensure that the product that does not comply with the requirements, is identified and controlled to avoid its non intentional use or deliver. |  |

## ISO/IEC 12207 Reference Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| Task of ISO/IEC 12207 | **Coverage**  **F/P/N** | **Title of the Task and Step** | **Comments** |
| 5.3.7 – Codification and software testing | F | Test Execution  Step 1: Preparation of software testing environment  Step 2: Execution of testing iterations  Step 3: Execution of regression tests |  |
| 5.3.9 – Software scoring testing | F | Test Execution  Step 1: Preparation of software testing environment  Step 2: Execution of testing iterations  Step 3: Execution of regression tests |  |
| 5.3.11 – Software system testing | F | Test Execution  Step 2: Execution of testing iterations  Step 3: Execution of regression tests |  |
| 6.3.2 – Product Quality Assurance Process | P | Test Execution |  |
| 6.4 – Verification | F | Test Execution |  |
| 6.5 - Validation | P | Test Execution |  |

## CMMI Reference Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| Objective/ Practice of CMMI V1.3 | **Coverage**  **F/P/N** | **Title of the**  **Task and Step** | **Comments** |
| Project Planning (PP) | P | **Test Planning**  Step 6:Definition of the project schedule |  |
| Measurement and Analysis (MA) | P | **Process Feedback**  Step 2: Calculation of indicators (Product – Process) |  |
| Process and Product Quality Assurance (PPQA) | P | **Test Execution**  Step 2: Execution of testing iterations |  |
| Verification (VER) | P | **Test Execution**  Step 2: Execution of testing iterations |  |
| Validation (VAL) | P | **Test Execution**  Step 2: Execution of testing iterations |  |

# 10. References

|  |  |
| --- | --- |
| **Key** | **Reference** |
| [CMMI 2010] | Capability Maturity Model Integration ® for Development, Version 1.3, Carnegie Mellon University, 2010. |
| [IEEE Std 730-2002] | IEEE Standard for Software Quality Assurance Plans |
| [ISO/IEC 12207] | ISO/IEC 12207:2008 Systems and software engineering – Software life cycle processes. |
| [ISO/IEC TR 19759] | ISO/IEC TR 19759:2005 Software Engineering – Guide to the Software Engineering Body of Knowledge (SWEBOK). |
| [ISO/IEC 29110] | ISO/IEC TR 29110-5-1-2:2011, Software Engineering—Lifecycle Profiles for Very Small Entities (VSEs) – Part 5-1-2: Management and engineering guide–Generic profile group: Basic profile.  Available at no cost on the following ISO site: <http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html> |
| [ISO/IEC 24765] | ISO/IEC 24765:2010, Systems and Software Engineering Vocabulary. |

# 11. Evaluation Form

|  |
| --- |
| **– Version 0.1**  Your feedback will allow us to improve this package, your comments and suggestions are welcome |
| **1. How satisfied are you with the CONTENT of this deployment package?**   *Very Satisfied*  *Satisfied*  *Neither Satisfied nor Dissatisfied*  *Dissatisfied*  *Very Dissatisfied* |
| **2. The sequence in which the topics are discussed is logical and easy to follow?**   *Very Satisfied*  *Satisfied*  *Neither Satisfied nor Dissatisfied*  *Dissatisfied*  *Very Dissatisfied* |
| **3. How satisfied are you with the APPEARANCE/FORMAT of this deployment package?**   *Very Satisfied*  *Satisfied*  *Neither Satisfied nor Dissatisfied*  *Dissatisfied*  *Very Dissatisfied* |
| **4. Have any unnecessary topics been included? (please describe)** |
| **5. What missing topic would you like to see in this package? (please describe)**   * Proposed topic: * Rationale for new topic |
| 1. **Any error in this deployment package?**    * Please indicate:      + - Description of error :        - Location of error (section #, figure #, table #) : |
| **7. Other feedback or comments:** |
| **8. Would you recommend this Deployment package to a colleague from another VSE?**   *Definitely*  *Probably*  *Not Sure*  *Probably Not*  *Definitely Not* |

**Optional**

* Name:
* e-mail address : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Email this form to** : [lgomez@parquesoft.com](mailto:lgomez@parquesoft.com) or [claude.y.laporte@etsmtl.ca](mailto:claude.y.laporte@etsmtl.ca) or [Avumex2003@yahoo.com.mx](mailto:Avumex2003@yahoo.com.mx)

1. These numbers refer to processes, activities, tasks of ISO/IEC 29110-5.1 [↑](#footnote-ref-1)
2. Roles are defined in a next section. Roles are also defined in ISO/IEC 29110-5.1 [↑](#footnote-ref-2)
3. These numbers refer to processes, activities, tasks of ISO/IEC 29110-5.1 [↑](#footnote-ref-3)
4. These numbers refer to processes, activities, tasks of ISO/IEC 29110-5.1 [↑](#footnote-ref-4)
5. SM CMM Integration is a service mark of Carnegie Mellon University.

   ® Capability Maturity Model, CMMI are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University. [↑](#footnote-ref-5)
6. This is the title of the task from section 3 of this package [↑](#footnote-ref-6)