<AdminPi>

<Iteration/ Master> Test Plan

Version <1.0>

[Note: The following template is provided for use with the Rational Unified Process. Text enclosed in square brackets and displayed in blue italics (style=InfoBlue) is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).]

[To customize automatic fields in Microsoft Word (which display a gray background when selected), select File>Properties and replace the Title, Subject and Company fields with the appropriate information for this document. After closing the dialog, automatic fields may be updated throughout the document by selecting Edit>Select All (or Ctrl-A) and pressing F9, or simply click on the field and press F9. This must be done separately for Headers and Footers. Alt-F9 will toggle between displaying the field names and the field contents. See Word help for more information on working with fields.]

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <01/abril/2019> | <1.0> | Primera versión de para el plan de estudio para las pruebas del proyecto | Piceno Roque Adán Oswaldo> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 3

1.1 Purpose 3

1.2 Scope 3

1.3 Intended Audience 3

1.4 Document Terminology and Acronyms 3

1.5 References 3

1.6 Document Structure 3

2. Evaluation Mission and Test Motivation 3

2.1 Background 3

2.2 Evaluation Mission 3

2.3 Test Motivators 3

3. Target Test Items 3

4. Outline of Planned Tests 3

4.1 Outline of Test Inclusions 3

4.2 Outline of other candidates for potential inclusion 3

4.3 Outline of Test Exclusions 3

5. Test Approach 3

5.1 Initial Test-Idea Catalogs and other reference sources 3

5.2 Testing Techniques and Types 3

5.2.1 Data and Database Integrity Testing 3

5.2.2 Function Testing 3

5.2.3 Business Cycle Testing 3

5.2.4 User Interface Testing 3

5.2.5 Performance Profiling 3

5.2.6 Load Testing 3

5.2.7 Stress Testing 3

5.2.8 Volume Testing 3

5.2.9 Security and Access Control Testing 3

5.2.10 Failover and Recovery Testing 3

5.2.11 Configuration Testing 3

5.2.12 Installation Testing 3

6. Entry and Exit Criteria 3

6.1 Test Plan 3

6.1.1 Test Plan Entry Criteria 3

6.1.2 Test Plan Exit Criteria 3

6.1.3 Suspension and resumption criteria 3

6.2 Test Cycles 3

6.2.1 Test Cycle Entry Criteria 3

6.2.2 Test Cycle Exit Criteria 3

6.2.3 Test Cycle abnormal termination 3

7. Deliverables 3

7.1 Test Evaluation Summaries 3

7.2 Reporting on Test Coverage 3

7.3 Perceived Quality Reports 3

7.4 Incident Logs and Change Requests 3

7.5 Smoke Test Suite and supporting Test Scripts 3

7.6 Additional work products 3

7.6.1 Detailed Test Results 3

7.6.2 Additional automated functional Test Scripts 3

7.6.3 Test Guidelines 3

7.6.4 Traceability Matrices 3

8. Testing Workflow 3

9. Environmental Needs 3

9.1 Base System Hardware 3

9.2 Base Software Elements in the Test Environment 3

9.3 Productivity and Support Tools 3

9.4 Test Environment Configurations 3

10. Responsibilities, Staffing and Training Needs 3

10.1 People and Roles 3

10.2 Staffing and Training Needs 3

11. Iteration Milestones 3

12. Risks, Dependencies, Assumptions and Constraints 3

13. Management Process and Procedures 3

13.1 Measuring and Assessing the Extent of Testing 3

13.2 Assessing the deliverables of this Test Plan 3

13.3 Problem Reporting, Escalation and Issue Resolution 3

13.4 Managing Test Cycles 3

13.5 Traceability Strategies 3

13.6 Approval and Signoff 3

<Iteration/ Master> Test Plan

# Introduction

## Purpose

El propósito del plan de prueba de iteración es reunir toda la información necesaria para planear y controlar el esfuerzo de prueba para una iteración determinada. Describe el enfoque para probar el software, y es el plan de nivel superior generado y utilizado por los gerentes para dirigir el esfuerzo de prueba. This *Test Plan* for the

<Project Name> supports the following objectives:

• [Identifica los elementos que deben ser dirigidos por las pruebas.

• Identifica la motivación y las ideas detrás de las áreas de prueba a ser cubiertas.

• Describe el enfoque de prueba que se utilizará.

• Identifica los recursos requeridos y proporciona una estimación de los esfuerzos de prueba.

• Enumera los elementos entregables del proyecto de prueba.]

## Scope

Dentro de los alcances de pruebas se analizará los siguiente:

1. Conexiones a las bases de datos
2. Las pruebas unitarias
3. Validación de datos
4. Funcionalidad
5. Usabilidad
6. Seguridad

Cada uno se tendrá que contemplar para garantizar la eficacia y la calidad de software así como deberá cumplir los requerimientos que se analizaron en los casos de uso, así como

## Intended Audience

El documento esta proporcionado para los desarrolladores y líderes del proyecto, así como los diseñadores de las interfaces, los testers.

Los líderes del proyecto tendrán en cuenta de los que se va a contemplar para garantizar la funcionalidad del proyecto.

## Document Terminology and Acronyms

No aplica (N/A)

## References

No aplica (N/A)

# 

# Evaluation Mission and Test Motivation

Cada prueba que se analizará en este documento es lo que permitirá garantizar la funcionalidad de cada uno de los puntos que se describen en la estructura de diseño del proyecto. Cada punto tiene un objetivo y conforme a los resultados se visualizará si el proyecto va por buen camino o debe de haber cambios ya sea en la forma en la que se está trabajando en el desarrollo del software o en toma de los requerimientos, tener una mejora continua (mantenimiento y actualizaciones), y llevar un control total durante y después del desarrollo del software.

## Background

Por cada desarrollo de software siempre se encontrarán fallas, adminPI deberá garantizar dentro de sus pruebas, las de seguridad entre la usabilidad del sistema por parte de los usuarios, garantizar la integridad de la información con la que se está trabajando, deberá tener congruencia, ser rápido en sus tiempos de respuesta, y que los resultados sean los esperados.

Cada prueba que se realice permitirá ver el avance del proyecto abarcado los puntos mencionados entre otros.

Por cada caso de uso abarcado se realizará los test para ver que puntos si fueron cubiertos y cuales se ocupan volver a analizar y planificar o si es necesario excluir.

Para ver el plan del proyecto, véase los documentos de casos de uso y planificación del proyecto

## Evaluation Mission

[Provide a brief statement that defines the mission for the evaluation effort in the current iteration. This statement might incorporate one or more concerns including:

* find as many bugs as possible
* find important problems, assess perceived quality risks
* advise about perceived project risks
* certify to a standard
* verify a specification (requirements, design or claims)
* advise about product quality, satisfy stakeholders
* advise about testing
* fulfill process mandates
* and so forth

Each mission provides a different context to the test effort and alters the way in which testing should be approached.]

## Test Motivators

Entre los problemas que podemos encontrar y se deben de tomar en cuenta:

* Los riesgos de calidad: el como este estructurado el sistema y su funcionalidad como su tiempo de vida, etc.
* Casos de uso: averiguar que casos de uso son factibles y su nivel de prioridad para poder trabajar en ello, si funciona correctamente o se necesita volver a planificar.
* Diseño de interfaces: la navegación del sistema, como los usuarios entienden la funcionalidad del sistema, averiguar posibles fallos.
* Adaptación: En que equipos funcionan, cuales fallan, etc,

# Target Test Items

Entre las áreas en la cual se deberá realizar las pruebas de funcionalidad son:

* Sistema Operativo: Windows, Mac.
* Hardware: equipos de cómputo (pc, laptop), conexión con impresora.
* Conexión en red: manejar el protocolo TCP/IP para la DB y entre equipos (LAN) según lo que se requiera

# Outline of Planned Tests

[This section provides a high-level outline of the testing that will be performed. The outline in this section represents a high level overview of both the tests that will be performed and those that will not.]

## Outline of Test Inclusions

Entre las pruebas principales tenemos:

La seguridad entre los roles de los usuarios: que cada uno de los derechos estén bien asignados, para evitar inyecciones de sql, cambios en la integridad de los datos, que cada usuario cumpla con sus respectivas acciones.

Pruebas con la conexión de DB mediante un servidor: por medio del protocolo TCP/IP si este es requerido, tener en cuenta los riesgos y demás al instalar el servidor, así mismo si se requerirán medios de seguridad u otro requerimiento de seguridad que deba estar enlazado con el software.

## Outline of Other Candidates for Potential Inclusion

No aplica (N\A)

## Outline of Test Exclusions

No aplica (N\A)

# Test Approach

## Initial Test-Idea Catalogs and Other Reference Sources

Pruebas unitarias

Pruebas de rendimiento

[Provide a listing of existing resources that will be referenced to stimulate the identification and selection of specific tests to be conducted. An example Test-Ideas Catalog is provided in the examples section of RUP.]

## Testing Techniques and Types

### Data and Database Integrity Testing

[The databases and the database processes should be tested as an independent subsystem. This testing should test the subsystems without the target-of-test’s User Interface as the interface to the data. Additional research into the DataBase Management System (DBMS) needs to be performed to identify the tools and techniques that may exist to support the testing identified in the following table.]

|  |  |
| --- | --- |
| Technique Objective: | El objetivo general del plan es establecer la cronología y condiciones para la aplicación de las pruebas de manera de obtener, un sistema que pueda ser completado con una recepción total de los interesados y entrar en operación con la totalidad de las funcionalidades requeridas para su funcionamiento.  Modulos del Sistema a ser probados:  -  -  - |
| Technique: | ***En estos módulos se realizarán pruebas para validar:***   * La visualización de los datos, ingresados o modificados. * La operación de los servicios, confeccionados para dar respuesta a los productos del sistema. * La respuesta y realización de las transacciones de cada módulo. * Que los estados de las actividades y documentos generados en el sistema se reflejen de acuerdo a la secuencia lógica requerida por el usuario. * La secuencia lógica de las funcionalidades y transacciones. |
| Oracles: |  |
| Required Tools: | [The technique requires the following tools:   * Test Script Automation Tool * base configuration imager and restorer * backup and recovery tools * installation-monitoring tools (registry, hard disk, CPU, memory, and so forth) * database SQL utilities and tools * Data-generation tools] |
| Success Criteria: | * Aceptación del plan de pruebas . Revisión y aceptación del documento que contiene los casos de pruebas del proyecto. * Aceptación de paquetes . Revisión y aceptación de los paquetes de desarrollo, y que este cumpla con las condiciones de aceptación. * Aceptación de ambiente . Revisión y aceptación del ambiente de certificación, y que este cumpla con las condiciones de aceptación. |
| Special Considerations: | * [Testing may require a DBMS development environment or drivers to enter or modify data directly in the databases. * Processes should be invoked manually. * Small or minimally sized databases (limited number of records) should be used to increase the visibility of any non-acceptable events.] |

### Function Testing

En esta sección se presentan los casos de pruebas generales para usarlos con la herramienta ERMT. Cada cuadro está asociado a un caso de Uso, desde ahí se desglosa en los diferentes módulos involucrados para el funcionamiento y se evalúa el resultado obtenido.

|  |  |
| --- | --- |
| Technique Objective: | En esta sección se exponen los tipos de pruebas a utilizar para el proyecto, cada una de ellas presenta un formato, el cual se va registrar los resultados. |
| Technique: | **Pruebas Unitarias**  Pruebas por cada unidad, en este caso una unidad es equivalente a un requerimiento. El requerimiento es aprobado y aprobado si este cumple con lo que está escrito en la especificación de requerimientos.  **Pruebas de Frontera**  Pruebas frontera, son las que toman en cuenta valores límite, para verificar el comportamiento de la herramienta en esos casos.  **Pruebas de Integración** Las pruebas de integración, como su nombre lo indica, son pruebas hechas a un conjunto de requerimientos, en este caso se distribuyen en los tipos de requerimientos que se definieron.  **Pruebas de Sistema**  Las pruebas de sistema son pruebas realizadas a la herramienta como un conjunto, que casos de uso cumple a cabalidad, con rutas de éxito y fallo, que han sido definidas en el documento de Casos de Uso. |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | • Netbeans |
| Success Criteria: | [The technique supports the testing of:  • all key use-case scenarios  • all key features] |
| Special Considerations: | [Identify or describe those items or issues (internal or external) that impact the implementation and execution of function test.] |

### Business Cycle Testing

[Business Cycle Testing should emulate the activities performed on the <Project Name> over time. A period should be identified, such as one year, and transactions and activities that would occur during a year’s period should be executed. This includes all daily, weekly, and monthly cycles, and events that are date-sensitive, such as ticklers.]

|  |  |
| --- | --- |
| Technique Objective: | [Exercise target-of-test and background processes according to required business models and schedules to observe and log target behavior.] |
| Technique: | [Testing will simulate several business cycles by performing the following:   * The tests used for target-of-test’s function testing will be modified or enhanced to increase the number of times each function is executed to simulate several different users over a specified period. * All time or date-sensitive functions will be executed using valid and invalid dates or time periods. * All functions that occur on a periodic schedule will be executed or launched at the appropriate time. * Testing will include using valid and invalid data to verify the following:   + The expected results occur when valid data is used.   + The appropriate error or warning messages are displayed when invalid data is used.   + Each business rule is properly applied.] |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the following tools:   * Test Script Automation Tool * base configuration imager and restorer * backup and recovery tools * Data-generation tools] |
| Success Criteria: | [The technique supports the testing of all critical business cycles.] |
| Special Considerations: | • [System dates and events may require special support activities.  • A business model is required to identify appropriate test requirements and procedures.] |

### User Interface Testing

[User Interface (UI) testing verifies a user’s interaction with the software. The goal of UI testing is to ensure that the UI provides the user with the appropriate access and navigation through the functions of the target-of-test. In addition, UI testing ensures that the objects within the UI function as expected and conform to corporate or industry standards.]

|  |  |
| --- | --- |
| Technique Objective: | [Exercise the following to observe and log standards conformance and target behavior:   * Navigation through the target-of-test reflecting business functions and requirements, including window-to-window, field-to- field, and use of access methods (tab keys, mouse movements, accelerator keys). * Window objects and characteristics can be exercised–such as menus, size, position, state, and focus.] |
| Technique: | [Create or modify tests for each window to verify proper navigation and object states for each application window and object.] |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the Test Script Automation Tool.] |
| Success Criteria: | [The technique supports the testing of each major screen or window that will be used extensively by the end user.] |
| Special Considerations: | [Not all properties for custom and third-party objects can be accessed.] |

### Performance Profiling

[Performance profiling is a performance test in which response times, transaction rates, and other time-sensitive requirements are measured and evaluated. The goal of Performance Profiling is to verify performance requirements have been achieved. Performance profiling is implemented and executed to profile and tune a target-of-test's performance behaviors as a function of conditions such as workload or hardware configurations.

**Note**: Transactions in the following table refer to “logical business transactions”. These transactions are defined as specific use cases that an actor of the system is expected to perform using the target-of-test, such as add or modify a given contract.]

|  |  |
| --- | --- |
| Technique Objective: | [Exercise behaviors for designated functional transactions or business functions under the following conditions to observe and log target behavior and application performance data:  • normal anticipated workload  • anticipated worst-case workload] |
| Technique: | • [Use Test Procedures developed for Function or Business Cycle Testing.  • Modify data files to increase the number of transactions or the scripts to increase the number of iterations that occur in each transaction.  • Scripts should be run on one machine (best case to benchmark single user, single transaction) and should be repeated with multiple clients (virtual or actual, see Special Considerations below).] |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the following tools:   * Test Script Automation Tool * an application performance profiling tool, such as Rational Quantify * installation-monitoring tools (registry, hard disk, CPU, memory, and so on * resource-constraining tools; for example, Canned Heat] |
| Success Criteria: | The technique supports testing:  • Single Transaction or single user: Successful emulation of the transaction scripts without any failures due to test implementation problems.]  • Multiple transactions or multiple users: Successful emulation of the workload without any failures due to test implementation problems.] |
| Special Considerations: | [Comprehensive performance testing includes having a background workload on the server.  There are several methods that can be used to perform this, including:  • “Drive transactions” directly to the server, usually in the form of Structured Query Language (SQL) calls.  • Create “virtual” user load to simulate many clients, usually several hundred. Remote Terminal Emulation tools are used to accomplish this load. This technique can also be used to load the network with “traffic”.  • Use multiple physical clients, each running test scripts, to place a load on the system.  Performance testing should be performed on a dedicated machine or at a dedicated time. This permits full control and accurate measurement.  The databases used for Performance Testing should be either actual size or scaled equally.] |

### Load Testing

[Load testing is a performance test that subjects the target-of-test to varying workloads to measure and evaluate the performance behaviors and abilities of the target-of-test to continue to function properly under these different workloads. The goal of load testing is to determine and ensure that the system functions properly beyond the expected maximum workload. Additionally, load testing evaluates the performance characteristics, such as response times, transaction rates, and other time-sensitive issues).]

[**Note**: Transactions in the following table refer to “logical business transactions”. These transactions are defined as specific functions that an end user of the system is expected to perform using the application, such as add or modify a given contract.]

|  |  |
| --- | --- |
| Technique Objective: | [Exercise designated transactions or business cases under varying workload conditions to observe and log target behavior and system performance data.] |
| Technique: | • [Use Transaction Test Scripts developed for Function or Business Cycle Testing as a basis, but remember to remove unnecessary interactions and delays.  • Modify data files to increase the number of transactions or the tests to increase the number of times each transaction occurs.  • Workloads should include (for example, Daily, Weekly, Monthly and so forth) Peak loads.  • Workloads should represent both Average as well as Peak loads.  • Workloads should represent both Instantaneous and Sustained Peaks.  • The Workloads should be executed under different Test Environment Configurations.] |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the following tools:   * Test Script Automation Tool * Transaction Load Scheduling and control tool * installation-monitoring tools (registry, hard disk, CPU, memory, and so on) * resource-constraining tools (for example, Canned Heat) * Data-generation tools] |
| Success Criteria: | [The technique supports the testing of Workload Emulation, which is the successful emulation of the workload without any failures due to test implementation problems.] |
| Special Considerations: | • [Load testing should be performed on a dedicated machine or at a dedicated time. This permits full control and accurate measurement.  • The databases used for load testing should be either actual size or scaled equally.] |

### Stress Testing

[Stress testing is a type of performance test implemented and executed to understand how a system fails due to conditions at the boundary, or outside of, the expected tolerances. This typically involves low resources or competition for resources. Low resource conditions reveal how the target-of-test fails that is not apparent under normal conditions. Other defects might result from competition for shared resources, like database locks or network bandwidth, although some of these tests are usually addressed under functional and load testing.]

[**Note**: References to transactions in the following table refer to logical business transactions.]

|  |  |
| --- | --- |
| Technique Objective: | [Exercise the target-of-test functions under the following stress conditions to observe and log target behavior that identifies and documents the conditions under which the system **fails** to continue functioning properly  • little or no memory available on the server (RAM and persistent storage space)  • maximum actual or physically capable number of clients connected or simulated  • multiple users performing the same transactions against the same data or accounts  • “overload” transaction volume or mix (see Performance Profiling above)] |
| Technique: | • [Use tests developed for Performance Profiling or Load Testing.  • To test limited resources, tests should be run on a single machine, and RAM and persistent storage space on the server should be reduced or limited.  • For remaining stress tests, multiple clients should be used, either running the same tests or complementary tests to produce the worst-case transaction volume or mix. |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the following tools:   * Test Script Automation Tool * Transaction Load Scheduling and control tool * installation-monitoring tools (registry, hard disk, CPU, memory, and so on) * resource-constraining tools (for example, Canned Heat) * Data-generation tools] |
| Success Criteria: | The technique supports the testing of Stress Emulation. The system can be emulated successfully in one or more conditions defined as stress conditions and an observation of the resulting system state during and after the condition has been emulated can be captured.] |
| Special Considerations: | • [Stressing the network may require network tools to load the network with messages or packets.  • The persistent storage used for the system should temporarily be reduced to restrict the available space for the database to grow.  • Synchronize the simultaneous clients accessing of the same records or data accounts.] |

### Volume Testing

[Volume testing subjects the target-of-test to large amounts of data to determine if limits are reached that cause the software to fail. Volume testing also identifies the continuous maximum load or volume the target-of-test can handle for a given period. For example, if the target-of-test is processing a set of database records to generate a report, a Volume Test would use a large test database, and would check that the software behaved normally and produced the correct report.]

|  |  |
| --- | --- |
| Technique Objective: | [Exercise the target-of-test under the following high volume scenarios to observe and log target behavior:  • Maximum (actual or physically-capable) number of clients connected, or simulated, all performing the same, worst case (performance) business function for an extended period.  • Maximum database size has been reached (actual or scaled) and multiple queries or report transactions are executed simultaneously.] |
| Technique: | • [Use tests developed for Performance Profiling or Load Testing.  • Multiple clients should be used, either running the same tests or complementary tests to produce the worst-case transaction volume or mix (see Stress Testing) for an extended period.  • Maximum database size is created (actual, scaled, or filled with representative data) and multiple clients are used to run queries and report transactions simultaneously for extended periods.] |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the following tools:   * Test Script Automation Tool * Transaction Load Scheduling and control tool * installation-monitoring tools (registry, hard disk, CPU, memory, and so on) * resource-constraining tools (for example, Canned Heat) * Data-generation tools] |
| Success Criteria: | [The technique supports the testing of Volume Emulation. Large quantities of users, data, transactions, or other aspects of the system use under volume can be successfully emulated and an observation of the system state changes over the duration of the volume test can be captured.] |
| Special Considerations: | [What period of time would be considered an acceptable time for high volume conditions, as noted above?] |

### Security and Access Control Testing

[Security and Access Control Testing focuses on two key areas of security:

• Application-level security, including access to the Data or Business Functions

• System-level Security, including logging into or remotely accessing to the system.

Based on the security you want, application-level security ensures that actors are restricted to specific functions or use cases, or they are limited in the data that is available to them. For example, everyone may be permitted to enter data and create new accounts, but only managers can delete them. If there is security at the data level, testing ensures that “user type one” can see all customer information, including financial data, however, “user two” only sees the demographic data for the same client.

System-level security ensures that only those users granted access to the system are capable of accessing the applications and only through the appropriate gateways.]

|  |  |
| --- | --- |
| Technique Objective: | [Exercise the target-of-test under the following conditions to observe and log target behavior:   * Application-level Security: an actor can access only those functions or data for which their user type is provided permissions. * System-level Security: only those actors with access to the system and applications are permitted to access them. |
| Technique: | * [Application-level Security: Identify and list each user type and the functions or data each type has permissions for.]   + Create tests for each user type and verify each permission by creating transactions specific to each user type.   + Modify user type and re-run tests for same users. In each case, verify those additional functions or data are correctly available or denied. * System-level Access: [See Special Considerations below] |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the following tools:   * Test Script Automation Tool * “Hacker” security breach and probing tools * OS Security Admin Tools] |
| Success Criteria: | [The technique supports the testing of for each known actor type the appropriate functions or data affected by security settings can be tested.] |
| Special Considerations: | [Access to the system must be reviewed or discussed with the appropriate network or systems administrator. This testing may not be required as it may be a function of network or systems administration.] |

### Failover and Recovery Testing

[Failover and recovery testing ensures that the target-of-test can successfully failover and recover from a variety of hardware, software or network malfunctions with undue loss of data or data integrity.

For those systems that must be kept running failover testing ensures that, when a failover condition occurs, the alternate or backup systems properly “take over” for the failed system without any loss of data or transactions.

Recovery testing is an antagonistic test process in which the application or system is exposed to extreme conditions, or simulated conditions, to cause a failure, such as device Input/Output (I/O) failures, or invalid database pointers and keys. Recovery processes are invoked, and the application or system is monitored and inspected to verify proper application, or system, and data recovery has been achieved.]

|  |  |
| --- | --- |
| Technique Objective: | [Simulate the failure conditions and exercise the recovery processes (manual and automated) to restore the database, applications, and system to a desired, known, state. The following types of conditions are included in the testing to observe and log target behavior after recovery:  • power interruption to the client  • power interruption to the server  • communication interruption via network servers  • interruption, communication, or power loss to DASD (Dynamic Access Storage Devices) and DASD controllers  • incomplete cycles (data filter processes interrupted, data synchronization processes interrupted)  • invalid database pointers or keys  • invalid or corrupted data elements in database] |
| Technique: | [The tests already created for Function and Business Cycle testing can be used as a basis for creating a series of transactions to support failover and recovery testing, primarily to define the tests to be run to test that recovery was successful.  • Power interruption to the client: power the PC down.  • Power interruption to the server: simulate or initiate power down procedures for the server.  • Interruption via network servers: simulate or initiate communication loss with the network (physically disconnect communication wires or power down network servers or routers).  • Interruption, communication, or power loss to DASD and DASD controllers: simulate or physically eliminate communication with one or more DASDs or controllers.  Once the above conditions or simulated conditions are achieved, additional transactions should be executed and, upon reaching this second test point state, recovery procedures should be invoked.  Testing for incomplete cycles uses the same technique as described above except that the database processes themselves should be aborted or prematurely terminated.  Testing for the following conditions requires that a known database state be achieved.  Several database fields, pointers, and keys should be corrupted manually and directly within the database (via database tools). Additional transactions should be executed using the tests from Application Function and Business Cycle Testing and full cycles executed.] |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the following tools:   * base configuration imager and restorer * installation monitoring tools (registry, hard disk, CPU, memory, and so on) * backup and recovery tools] |
| Success Criteria: | The technique supports the testing of:   * One or more simulated disasters involving one or more combinations of the application, database, and system. * One or more simulated recoveries involving one or more combinations of the application, database, and system to a known desired state.] |
| Special Considerations: | * [Recovery testing is highly intrusive. Procedures to disconnect cabling (simulating power or communication loss) may not be desirable or feasible. Alternative methods, such as diagnostic software tools may be required. * Resources from the Systems (or Computer Operations), Database, and Networking groups are required. * These tests should be run after hours or on an isolated machine.] |

### Configuration Testing

[Configuration testing verifies the operation of the target-of-test on different software and hardware configurations. In most production environments, the particular hardware specifications for the client workstations, network connections, and database servers vary. Client workstations may have different software loaded⎯for example, applications, drivers, and so on⎯and, at any one time, many different combinations may be active using different resources.]

|  |  |
| --- | --- |
| Technique Objective: | [Exercise the target-of-test on the required hardware and software configurations to observe and log target behavior under different configurations and identify changes in configuration state.] |
| Technique: | * [Use Function Test scripts. * Open and close various non-target-of-test related software, such as Microsoft Excel and Word applications, either as part of the test or prior to the start of the test. * Execute selected transactions to simulate actors interacting with the target-of-test and the non-target-of-test software. * Repeat the above process, minimizing the available conventional memory on the client workstation.] |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the following tools:   * base configuration imager and restore * installation monitoring tools (registry, hard disk, CPU, memory, and so on)] |
| Success Criteria: | [The technique supports the testing of one or more combinations of the target test items running in expected, supported deployment environments.] |
| Special Considerations: | * [What non-target-of-test software is needed, is available, and what is accessible on the desktop? * What applications are typically used? * What data are the applications running; for example, a large spreadsheet opened in Excel or a 100-page document in Word? * The entire system’s netware, network servers, databases, and so on, also needs to be documented as part of this test.] |

### Installation Testing

[Installation testing has two purposes. The first is to ensure that the software can be installed under different conditions⎯such as a new installation, an upgrade, and a complete or custom installation⎯under normal and abnormal conditions. Abnormal conditions include insufficient disk space, lack of privilege to create directories, and so on. The second purpose is to verify that, once installed, the software operates correctly. This usually means running a number of the tests that were developed for Function Testing.]

|  |  |
| --- | --- |
| Technique Objective: | [Exercise the installation of the target-of-test onto each required hardware configuration under the following conditions to observe and log installation behavior and configuration state changes:   * new installation: a new machine, never installed previously with <Project Name> * update: a machine previously installed <Project Name>, same version * update: a machine previously installed <Project Name>, older version] |
| Technique: | * [Develop automated or manual scripts to validate the condition of the target machine.   + new: never installed   + same or older version already installed * Launch or perform installation. * Using a predetermined subset of Function Test scripts, run the transactions.] |
| Oracles: | [Outline one or more strategies that can be used by the technique to accurately observe the outcomes of the test. The oracle combines elements of both the method by which the observation can be made and the characteristics of specific outcome that indicate probable success or failure. Ideally, oracles will be self-verifying, allowing automated tests to make an initial assessment of test pass or failure, however, be careful to mitigate the risks inherent in automated results determination.] |
| Required Tools: | [The technique requires the following tools:   * base configuration imager and restorer * installation monitoring tools (registry, hard disk, CPU, memory, and so on)] |
| Success Criteria: | [The technique supports the testing of the installation of the developed product in one or more installation configurations.] |
| Special Considerations: | [What <Project Name> transactions should be selected to comprise a confidence test that <Project Name> application has been successfully installed and no major software components are missing?] |

# Entry and Exit Criteria

## Test Plan

### Test Plan Entry Criteria

* Se deberá dividir las secciones de los test a quienes corresponda.
* Evitar que el o los desarrolladores realicen los test si este no es necesario, preferentemente que exista uno o varios tester que se encarguen de realizar las pruebas.
* Tener en mano todo documento o herramienta que ayude a realizar las pruebas.
* Tener bien especificado en un cronograma el tiempo estimado para el desarrollo y las pruebas.
* Si los usuarios realizaran las pruebas beta, que contemplen un formato de reporte.

[Specify the criteria that will be used to determine whether the execution of the **Test Plan** can begin.]

### Test Plan Exit Criteria

* Mantener en un marco o documento nuevo donde se especifique el transcurso de la prueba.
* Contemplar quien realiza los test y quien se encarga del desarrollo.
* Llevar a cabo un reporte de las fallas encontradas y las soluciones.

[Specify the criteria that will be used to determine whether the execution of the **Test Plan** is complete or that continued execution provides no further benefit.]

### Suspension and Resumption Criteria

* Especificar el error o problema que evita realizar la prueba.
* Evaluar el estado crítico por la cual se debe llevar a cabo la prueba en un futuro.
* Quién lo autoriza y quienes lo aprueban.

[Specify the criteria that will be used to determine whether testing should be prematurely suspended or ended before the plan has been completely executed, and under what criteria testing can be resumed.]

## Test Cycles

### Test Cycle Entry Criteria

* Tener en mano los reportes anteriores
* Quien fue el encargado de solucionar dichos errores.
* Hay que especificar que se analizara.

[Specify the criteria to be used to determine whether the test effort for the next Test Cycle of this **Test Plan** can begin.]

### Test Cycle Exit Criteria

* Si las especificaciones en los casos de uso y en los requerimientos cumplen mínimo con un 90% de funcionalidad, esto es, que el rendimiento y la estimación de encontrar errores sea mínima.
* Que cada reporte de error o fallas en el sistema se haya contemplado, no obstante que un 80% de los reportes se debió dar solución.

[Specify the criteria that will be used to determine whether the test effort for the current Test Cycle of this **Test Plan** is deemed sufficient.]

### Test Cycle Abnormal Termination

* Especificar el error o problema que evita realizar la prueba.
* Evaluar el estado crítico por la cual se debe llevar a cabo la prueba en un futuro.
* Quién lo autoriza y quienes lo aprueban.

[Specify the criteria that will be used to determine whether testing should be prematurely suspended or ended for the current test cycle, or whether the intended build candidate to be tested must be altered.]

# Deliverables

[In this section, list the various artifacts that will be created by the test effort that are useful deliverables to the various stakeholders of the test effort. Don’t list all work products; only list those that give direct, tangible benefit to a stakeholder and those by which you want the success of the test effort to be measured.]

## Test Evaluation Summaries

[Provide a brief outline of both the form and content of the test evaluation summaries, and indicate how frequently they will be produced.]

## Reporting on Test Coverage

[Provide a brief outline of both the form and content of the reports used to measure the extent of testing, and indicate how frequently they will be produced. Give an indication as to the method and tools used to record, measure, and report on the extent of testing.]

## Perceived Quality Reports

[Provide a brief outline of both the form and content of the reports used to measure the perceived quality of the product, and indicate how frequently they will be produced. Give an indication about to the method and tools used to record, measure, and report on the perceived product quality. You might include some analysis of Incidents and Change Request over Test Coverage.]

## Incident Logs and Change Requests

[Provide a brief outline of both the method and tools used to record, track, and manage test incidents, associated change requests, and their status.]

## Smoke Test Suite and Supporting Test Scripts

[Provide a brief outline of the test assets that will be delivered to allow ongoing regression testing of subsequent product builds to help detect regressions in the product quality.]

## Additional Work Products

[In this section, identify the work products that are optional deliverables or those that should not be used to measure or assess the successful execution of the **Test Plan**.]

### Detailed Test Results

[This denotes either a collection of Microsoft Excel spreadsheets listing the results determined for each test case, or the repository of both test logs and determined results maintained by a specialized test product.]

### Additional Automated Functional Test Scripts

[These will be either a collection of the source code files for automated test scripts, or the repository of both source code and compiled executables for test scripts maintained by the test automation product.]

### Test Guidelines

[Test Guidelines cover a broad set of categories, including Test-Idea catalogs, Good Practice Guidance, Test patterns, Fault and Failure Models, Automation Design Standards, and so forth.]

### Traceability Matrices

[Using a tool such as Rational RequisistePro or MS Excel, provide one or more matrices of traceability relationships between traced items.]

# Testing Workflow

[Provide an outline of the workflow to be followed by the Test team in the development and execution of this **Test Plan**.]

The specific testing workflow that you will use should be documented separately in the project's Development Case. It should explain how the project has customized the base RUP test workflow (typically on a phase-by-phase basis). In most cases, we recommend you place a reference in this section of the **Test Plan** to the relevant section of the Development Case. It might be both useful and sufficient to simply include a diagram or image depicting your test workflow.

More specific details of the individual testing tasks are defined in a number of different ways, depending on project culture; for example:

* defined as a list of tasks in this section of the **Test Plan**, or in an accompanying appendix
* defined in a central project schedule (often in a scheduling tool such as Microsoft Project)
* documented in individual, "dynamic" to-do lists for each team member, which are usually too detailed to be placed in the **Test Plan**
* documented on a centrally located whiteboard and updated dynamically
* not formally documented at all

Based on your project culture, you should either list your specific testing tasks here or provide some descriptive text explaining the process your team uses to handle detailed task planning and provide a reference to where the details are stored, if appropriate.

For Master Test Plans, we recommend avoiding detailed task planning, which is often an unproductive effort if done as a front-loaded activity at the beginning of the project. A Master Test Plan might usefully describe the phases and the number of iterations, and give an indication of what types of testing are generally planned for each Phase or Iteration.

**Note**: Where process and detailed planning information is recorded centrally and separately from this Test Plan, you will have to manage the issues that will arise from having duplicate copies of the same information. To avoid team members referencing out-of-date information, we suggest that in this situation you place the minimum amount of process and planning information within the Test Plan to make ongoing maintenance easier and simply reference the "Master" source material.]

# Environmental Needs

[This section presents the non-human resources required for the **Test Plan**.]

## Base System Hardware

The following table sets forth the system resources for the test effort presented in this *Test Plan*.

[The specific elements of the test system may not be fully understood in early iterations, so expect this section to be completed over time. We recommend that the system simulates the production environment, scaling down the concurrent access and database size, and so forth, if and where appropriate.]

[**Note**: Add or delete items as appropriate.]

| **System Resources** | | |
| --- | --- | --- |
| **Resource** | **Quantity** | **Name and Type** |
| Database Server |  |  |
| —Network or Subnet |  | TBD |
| —Server Name |  | TBD |
| —Database Name |  | TBD |
| Client Test PCs |  |  |
| —Include special configuration requirements |  | TBD |
| Test Repository |  |  |
| —Network or Subnet |  | TBD |
| —Server Name |  | TBD |
| Test Development PCs |  | TBD |

## Base Software Elements in the Test Environment

The following base software elements are required in the test environment for this *Test Plan*.

[Note: Add or delete items as appropriate.]

| **Software Element Name** | **Version** | **Type and Other Notes** |
| --- | --- | --- |
| NT Workstation |  | Operating System |
| Windows 2000 |  | Operating System |
| Internet Explorer |  | Internet Browser |
| Netscape Navigator |  | Internet Browser |
| MS Outlook |  | eMail Client software |
| Network Associates McAfee Virus Checker |  | Virus Detection and Recovery Software |

## Productivity and Support Tools

The following tools will be employed to support the test process for this *Test Plan*.

[Note: Add or delete items as appropriate.]

| **Tool Category or Type** | **Tool Brand Name** | **Vendor or In-house** | **Version** |
| --- | --- | --- | --- |
| Test Management |  |  |  |
| Defect Tracking |  |  |  |
| ASQ Tool for functional testing |  |  |  |
| ASQ Tool for performance testing |  |  |  |
| Test Coverage Monitor or Profiler |  |  |  |
| Project Management |  |  |  |
| DBMS tools |  |  |  |

## Test Environment Configurations

The following Test Environment Configurations needs to be provided and supported for this project.

| **Configuration Name** | **Description** | **Implemented in Physical Configuration** |
| --- | --- | --- |
| Average user configuration |  |  |
| Minimal configuration supported |  |  |
| Visually and mobility challenged |  |  |
| International Double Byte OS |  |  |
| Network installation (not client) |  |  |

# Responsibilities, Staffing, and Training Needs

[This section presents the required resources to address the test effort outlined in the **Test Plan**—the main responsibilities, and the knowledge or skill sets required of those resources.]

## People and Roles

This table shows the staffing assumptions for the test effort.

[**Note**: Add or delete items as appropriate.]

| **Human Resources** | | |
| --- | --- | --- |
| **Role** | **Minimum Resources Recommended**  **(number of full-time roles allocated)** | **Specific Responsibilities or Comments** |
| Test Manager |  | Provides management oversight.  Responsibilities include:   * planning and logistics * agree mission * identify motivators * acquire appropriate resources * present management reporting * advocate the interests of test * evaluate effectiveness of test effort |
| Test Analyst |  | Identifies and defines the specific tests to be conducted.  Responsibilities include:   * identify test ideas * define test details * determine test results * document change requests * evaluate product quality |
| Test Designer |  | Defines the technical approach to the implementation of the test effort.  Responsibilities include:   * define test approach * define test automation architecture * verify test techniques * define testability elements * structure test implementation |
| Tester |  | Implements and executes the tests.  Responsibilities include:   * implement tests and test suites * execute test suites * log results * analyze and recover from test failures * document incidents |
| Test System Administrator |  | Ensures test environment and assets are managed and maintained.  Responsibilities include:   * administer test management system * install and support access to, and recovery of, test environment configurations and test labs |
| Database Administrator, Database Manager |  | Ensures test data (database) environment and assets are managed and maintained.  Responsibilities include:   * support the administration of test data and test beds (database). |
| Designer |  | Identifies and defines the operations, attributes, and associations of the test classes.  Responsibilities include:   * defines the test classes required to support testability requirements as defined by the test team |
| Implementer |  | Implements and unit tests the test classes and test packages.  Responsibilities include:   * creates the test components required to support testability requirements as defined by the designer |

## Staffing and Training Needs

This section outlines how to approach staffing and training the test roles for the project.

[The way to approach staffing and training will vary from project to project. If this section is part of a Master Test Plan, you should indicate at what points in the project lifecycle different skills and numbers of staff are needed. If this is an Iteration Test Plan, you should focus mainly on where and what training might occur during the Iteration.

Give thought to your training needs, and plan to schedule this based on a Just-In-Time (JIT) approach—there is often a temptation to attend training too far in advance of its usage when the test team has apparent slack. Doing this introduces the risk of the training being forgotten by the time it's needed.

Look for opportunities to combine the purchase of productivity tools with training on those tools, and arrange with the vendor to delay delivery of the training until just before you need it. If you have enough headcount, consider having training delivered in a customized manner for you, possibly at your own site.

The test team often requires the support and skills of other team members not directly part of the test team. Make sure you arrange in your plan for appropriate availability of System Administrators, Database Administrators, and Developers who are required to enable the test effort.]

# Iteration Milestones

[Identify the key schedule milestones that set the context for the Testing effort. Avoid repeating too much detail that is documented elsewhere in plans that address the entire project.]

| **Milestone** | **Planned Start Date** | **Actual Start Date** | **Planned End Date** | **Actual End Date** |
| --- | --- | --- | --- | --- |
| Iteration Plan agreed |  |  |  |  |
| Iteration starts |  |  |  |  |
| Requirements baselined |  |  |  |  |
| Architecture baselined |  |  |  |  |
| User Interface baselined |  |  |  |  |
| First Build delivered to test |  |  |  |  |
| First Build accepted into test |  |  |  |  |
| First Build test cycle finishes |  |  |  |  |
| [Build Two will not be tested] |  |  |  |  |
| Third Build delivered to test |  |  |  |  |
| Third Build accepted into test |  |  |  |  |
| Third Build test cycle finishes |  |  |  |  |
| Fourth Build delivered to test |  |  |  |  |
| Fourth Build accepted into test |  |  |  |  |
| Iteration Assessment review |  |  |  |  |
| Iteration ends |  |  |  |  |

# Risks, Dependencies, Assumptions, and Constraints

[List any risks that may affect the successful execution of this **Test Plan**, and identify mitigation and contingency strategies for each risk. Also indicate a relative ranking for both the likelihood of occurrence and the impact if the risk is realized.]

| **Risk** | **Mitigation Strategy** | **Contingency (Risk is realized)** |
| --- | --- | --- |
| Prerequisite entry criteria is not met. | <Tester> will define the prerequisites that must be met before Load Testing can start.  <Customer> will endeavor to meet prerequisites indicated by <Tester>. | * Meet outstanding prerequisites * Consider Load Test Failure |
| Test data proves to be inadequate. | <Customer> will ensure a full set of suitable and protected test data is available.  <Tester> will indicate what is required and will verify the suitability of test data. | * Redefine test data * Review Test Plan and modify * components (that is, scripts) * Consider Load Test Failure |
| Database requires refresh. | <System Admin> will endeavor to ensure the Database is regularly refreshed as required by <Tester>. | * Restore data and restart * Clear Database |

[List any dependencies identified during the development of this **Test Plan** that may affect its successful execution if those dependencies are not honored. Typically these dependencies relate to activities on the critical path that are prerequisites or post-requisites to one or more preceding (or subsequent) activities You should consider responsibilities you are relying on other teams or staff members external to the test effort completing, timing and dependencies of other planned tasks, the reliance on certain work products being produced.]

| **Dependency between** | **Potential Impact of Dependency** | **Owners** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

[List any assumptions made during the development of this **Test Plan** that may affect its successful execution if those assumptions are proven incorrect. Assumptions might relate to work you assume other teams are doing, expectations that certain aspects of the product or environment are stable, and so forth].

| **Assumption to be proven** | **Impact of Assumption being incorrect** | **Owners** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

[List any constraints placed on the test effort that have had a negative effect on the way in which this **Test Plan** has been approached.]

| **Constraint on** | **Impact Constraint has on test effort** | **Owners** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

# Management Process and Procedures

[Outline what processes and procedures are to be used when issues arise with the **Test Plan** and its enactment.]

## Measuring and Assessing the Extent of Testing

[Outline the measurement and assessment process to be used to track the extent of testing.]

## Assessing the Deliverables of this Test Plan

[Outline the assessment process for reviewing and accepting the deliverables of this **Test Plan**]

## Problem Reporting, Escalation, and Issue Resolution

[Define how process problems will be reported and escalated, and the process to be followed to achieve resolution.]

## Managing Test Cycles

[Outline the management control process for a test cycle.]

## Traceability Strategies

[Consider appropriate traceability strategies for:

* Coverage of Testing against Specifications — enables measurement the extent of testing
* Motivations for Testing — enables assessment of relevance of tests to help determine whether to maintain or retire tests
* Software Design Elements — enables tracking of subsequent design changes that would necessitate rerunning tests or retiring them
* Resulting Change Requests — enables the tests that discovered the need for the change to be identified and re-run to verify the change request has been completed successfully]

## Approval and Signoff

[Outline the approval process and list the job titles (and names of current incumbents) that initially must approve the plan, and sign off on the plans satisfactory execution.]