PAS

Plan de Acción de

Sistemas

Plan de Gestión de Pruebas

PAS_PLAN DE GESTION DE PRUEBA_V1.0.doc

Versión 1.0

Acceso Norte Ruta Nacional Nº3 Caleta Olivia (9011) Santa Cruz TEL. 0297 4854888 Int 114
e-mail: pas@uaco.unpa.edu.ar

Este documento cumple con las pautas establecidas por el Plan de Acción de Sistemas para la generación de documentación según el estándar EST_DOCUMENTACION_V2.1.

La información contenida en este documento esta sujeta a modificaciones.

Equipo de Trabajo

Díaz Vivar, Miriam Rosana
Miranda, María Gabriela
Molina, Sonia Elizabeth

Contenido

1. Introducción

El PAS es el área de desarrollo y mantenimiento de Sistemas del PSTI. Su misión es proveer información oportuna y confiable para la toma de decisiones, brindar soportes confiables para la administración de la información generada en la Gestión Universitaria y garantizar un marco de calidad para el desarrollo e implantación de sistemas informáticos.

Es por esto la importancia de definir un proceso de pruebas para guiar el diseño, ejecución y documentación de las mismas al fin de proveer sistemas confiables que respondan a las necesidades institucionales

El presente documento se encuentra organizado de la siguiente manera:

En la sección 1 se muestra el propósito del presente documento, en la sección 2 se detalla específicamente la gestión de las pruebas, los artefactos relacionados y los roles involucrados según el proceso del área, en la sección 3 se describe el proceso de prueba propiamente dicho, la sección 4 contiene las metodologías que pueden utilizarse al momento de planificar las pruebas, en la sección 5 se detallan los recursos tanto humanos como materiales que participarán del proceso de pruebas, la sección 6 se muestra el calendario asociado a las pruebas y por último la sección 7 presenta como se documentaran los resultados de las pruebas.

1.1 Propósito del Documento

El propósito del plan de pruebas es explicitar el alcance, enfoque, recursos requeridos, calendario y responsables del proceso de pruebas.

El objetivo principal de este documento es guiar el proceso de pruebas de los distintos proyectos del área y ser una herramienta que sirva para:

- Brindar información que será de utilidad para definir el plan de pruebas de cada sistema concerniente al área PAS.
- Guiar en el diseño de los casos de prueba, definición del ambiente de prueba a utilizar, aspectos a probar del software, condiciones de finalización, suspensión o repetición de las mismas, artefactos que se confeccionaran y responsables de la creación y revisión de los mismos.
- Guiar a la Gerencia Colaborativa en el seguimiento y control de los proyectos desde los aspectos de las pruebas.

1.2 Alcance

Los sistemas, aplicativos, personalizaciones o scripts que se implanten o desarrollen desde el área se someterán al proceso de prueba que describe este documento, el cual se adaptará dependiendo si el artefacto a probar corresponde a código cerrado o abierto, el enfoque de las pruebas puede corresponder a alguna de las siguientes categorías de pruebas:

- Funcionalidad: Para asegurar que el sistema realiza sus funciones normales de manera correcta.
- Recuperación: Para corroborar que el sistema pueda recuperarse adecuadamente de diversos tipos de fallas, como: fallas de hardware, fallas de corriente, fallas en el sistema operativo, etc.
- Conectividad/Performance: Para chequear los tiempos de respuesta y de acceso al sistema de acuerdo a la arquitectura del sistema.

- Seguridad: Acceso remoto y local al sistema haciendo uso de nombre de usuario y contraseña.

- Convivencia con otros sistemas: Para asegurar que podrá instalarse y funcionar correctamente compartiendo hardware otros sistemas, importando, exportando información hacia otros sistemas relacionados.

- Eficiencia: El propósito es verificar que el producto usa los recursos de manera eficiente.

- Correctitud: Para verificar que el producto se comporta de acuerdo a la especificaciones funcionales que debería proveer. Es una propiedad absoluta: cualquier desviación implica un software no−correcto.

- Confiabilidad: Para medir la probabilidad de ocurrencia de fallas, esta es una medida relativa: ya que un software puede ser confiable si la consecuencia de un error no es seria; o si la cantidad de errores por unidad de tiempo no es alta.

- Robustez: Un programa es robusto si se comporta razonablemente aun en circunstancias que no fueron anticipadas en los requerimientos. Si se puede pensar en acontecimientos imprevistos, entonces hay que incluirlos en la especificación y se habla de corrección.
  'Si especificado y verifica es correcto.
  'Si no especificado y verifica es robusto.

- Escalabilidad: Un software es escalable si permite cambios que lo hacen capaz de satisfacer nuevos requerimientos. Esta cualidad se logra mediante un buena modularización.

1.3 Referencias

- PAS_ Documento de Definición del Proceso_ V1.0
- Ingeniería de Software, Un enfoque práctico (4ta. Ed.) Roger S. Presman.
- Managing the software (Watt Humprey): Capítulos 4, 9, 13 y 15.

2. Gestión de pruebas

La gestión de las pruebas estará a cargo del Gerente de Pruebas, el cual tiene la responsabilidad de guiar el proceso de pruebas en los distintos proyectos del área.

A continuación se detallan las responsabilidades de cada uno de los roles intervinientes:

2.1 Roles Involucrados

Dadas las características de los distintos Proyectos del Área PAS se acotaron los roles relacionados a las pruebas de acuerdo al proceso vigente en el Área.

2.1.1 Analista de Sistemas

Este rol es responsable de conocer el dominio del problema, analizar minuciosamente la funcionalidad del sistema, el modelo de datos y los defectos potenciales del sistema / versión del sistema / modulo u operación objeto de estudio para elaborar el Documento de Casos de Prueba.

También es el responsable de generar los lotes de prueba generales, en el caso en que sea necesario diseñar procedimientos (scripts, consultas SQL, stored procedure) para probar el código.

En caso de tratarse de una actualización o conversión del sistema deberá estar al tanto de todas las mejoras incorporadas y el objetivo de los mismas para poder definir mejor los ítems a probar ya sean circuitos, procedimientos, etc.

Deberá definir y documentar los ambientes de prueba, entendiendo por ambiente el Hardware/Software en el que se va a probar el artefacto a ser evaluado, configuraciones del mismo, parámetros y configuración del software relacionado.

2.1.2 Tester

Este rol es responsable de ejecutar las pruebas y registrar los resultados obtenidos

En los casos en que se utilice una herramienta para la automatización de las pruebas, será el tester el responsable de adquirir las habilidades necesarias para su uso. Deberá ser capaz de analizar y proponer nuevos y posibles casos de pruebas.

2.1.3 Gerente de pruebas

El Gerente de pruebas es responsable de identificar y definir los requerimientos de prueba, supervisando detalles del progreso de las mismas y resultados en cada ciclo de prueba, evaluando la calidad global experimentada como resultado de las actividades de prueba. Además evalúa las pruebas planificadas por los líderes, el diseño y la ejecución de las mismas. También selecciona y describe los procedimientos de prueba que se necesitan, y es responsable de la evaluación de las pruebas de integración y de sistema cuando estas se ejecutan.

2.1.4 Implementador

En el caso de detectarse errores o fallas en los artefactos probados será el implementador el responsable de corregirlos en los casos de aplicaciones de código abierto, también quien documente en el Reporte de Errores en la columna Acción tomada acerca de como se resolvió la situación.

2.1.5 Líder del Proyecto

Es el responsable de definir junto con el Gerente de Pruebas que tipo de pruebas se llevaran a cabo, desarrollar los planes de prueba que correspondan y generar el informe de resultados obtenidos

Las tareas a desempeñar por los roles antes mencionados se encuentran especificadas con mayor detalle en el proceso definido para el área PAS_Definicion de Proceso vigente.

2.2 Entregables y criterios de aceptación

A continuación se detallan los artefactos que resultan del proceso de pruebas.

Artefacto	Responsable del Artefacto	Criterios de aceptación
Plan de Pruebas del Área PAS	Gerente de Pruebas	El artefacto será aprobado por la Gerencia Colaborativa y Gerencia General
Plan de Pruebas de los sistemas	Líder del Proyecto	El artefacto será aprobado por la Gerencia Colaborativa[1].
Casos de Prueba	Analista de Sistemas de cada proyecto	El artefacto será aprobado por el Líder del proyecto y Gerente de Pruebas.
Reporte de errores	Analista de Sistemas de cada proyecto	El artefacto será aprobado por el Líder del Proyecto y el Gerente de Pruebas.
Informe de Resultados obtenidos.	Líder del Proyecto	El artefacto será aprobado por la Gerencia Colaborativa.

Tabla 1: Entregables y criterios de aceptación

Plan de Prueba del Sistema: Para cada sistema el Líder del proyecto instanciará el plan de pruebas del área, se desarrollará una vez aprobado el plan de proyecto por parte del cliente, y en el caso de realizarse acuerdos de trabajo intermedios que involucren la prueba de nuevas aplicaciones, módulos, operaciones,personalizaciones, etc. En el caso de testear varias versiones de un sistema en un año se actualizara el Plan de pruebas del sistema. La plantilla a utilizar es FOR015_PLAN DE PRUEBAS_V2.0.dot

Casos de Prueba: El Analista de Sistemas de cada proyecto desarrollará el documento de Casos de Prueba, el cual deberá contemplar todas las operaciones a probar e incluir la mayor cantidad de casos que permitan encontrar potenciales errores. La plantilla a utilizar es FOR074_CASOS DE PRUEBAS_V1.0.dot

[1] Existirá un Plan de prueba por sistema.

Reporte de errores: Este artefacto es de características técnicas y describirá todos los errores, fallas y defectos encontrados en los productos testeados, ya sean de código abierto, o cerrado. Desarrollarlo es responsabilidad del Analista de Sistemas, y lo hace basándose en los Documentos de Casos de Pruebas ejecutados por los testers. Esta dirigido al equipo de implementación para la posterior depuración de los mismos.
La plantilla a utilizar es FOR075_REPORTE DE ERRORES ENCONTRADOS.dot

Informe de Resultados Obtenidos: Este artefacto organiza y muestra un análisis de los resultados de las pruebas, puede contener un enunciado general de la calidad relativa al sistema que se esta probando y ofrecer recomendaciones para futuros procesos de prueba. Este informe se desarrolla tanto en sistemas de código abierto como de código cerrado. El responsable de generar este artefacto es el Líder de Proyecto. La plantilla a utilizar es FOR073 80_INFORME DE RESULTADOS OBTENIDOS.dot

2.3 Ciclo de vida de las pruebas

La ejecución de pruebas de un sistema involucra las etapas que a continuación se detallan:

- Planificación de pruebas.
- Diseño y construcción de los Casos de Prueba.
- Preparación del ambiente de pruebas y generación de datos de prueba.
- Ejecución de las pruebas.
- Registro de errores encontrados.
- Registración de resultados obtenidos.
- Depuración de los errores.
- Informe de los resultados obtenidos.

En el marco del proceso definido para el área, las etapas mencionadas forman el ciclo de vida de las pruebas el cual se integrara al proceso de desarrollo de la aplicación[2]

De manera que:

- Durante la fase de Elaboración se defina el Plan de Pruebas.
- Durante la fase de Elaboración se definan los Casos de Prueba.
- Durante la fase de Construcción, se construyan los Casos de Prueba y se genere las bases de datos de prueba.
- Posteriormente se ejecuten todos los scripts y programas de pruebas anteriormente desarrollados y se realice un adecuado seguimiento.

[2] Para el caso de desarrollo de una nueva aplicación, sistema, o modulo de un sistema existente.

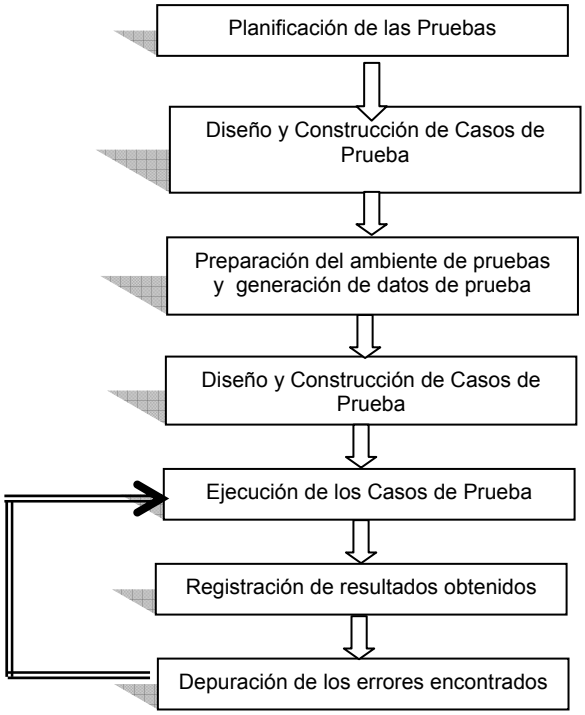


Figura 1: Ciclo de vida de las pruebas

3. Proceso de Prueba

El proceso de prueba se compone de los siguientes pasos:

1. El circuito comienza con la generación del Plan de Pruebas por parte del **Líder del proyecto** en la fase de Elaboración. Dicho plan se generará una vez aprobado el Plan de Proyecto para el área[3] al cual pertenece el sistema a probar.
2. Luego en la fase de Construcción, el **Analista de Sistemas** del Proyecto diseñará las pruebas basándose en la documentación y conocimiento del software a probar, esto dará como resultado tantos Documentos de Casos de Prueba como testers sean designados.

   Esta actividad dependerá de la situación:

   - Si el sistema esta en producción y se prueba una nueva versión o actualización del mismo, en el caso en que exista, se utilizará el Documento General de Casos de Prueba[4].
   - En la prueba del sistema o aplicación existirá un documento de Casos de Prueba por **tester**, en éste el **Analista de Sistemas** completará el ambiente de prueba a utilizar, cuales serán los datos de entrada y los resultados esperados.

[3] Área: Proyecto dentro del Plan de Acción de Sistemas.

[4] Este documento reúne pruebas de rutina de todos los módulos y operaciones del sistema y se va generando una nueva versión del mismo por cada versión o actualización del sistema en cuestión.

- Si se esta probando un sistema por primera vez, a parte de los documentos de Casos de Prueba que se generen por tester, se deberá crear un documento General de Casos de Prueba que reúna todos los eventos de prueba, para reutilizar en futuras pruebas.
- Si se esta probando una nueva operación, se adicionaran tantos casos de prueba como sean necesarios al final del Documento General de Casos de prueba, donde se especificará la situación por la cual se generaron dichos casos.
- Si se esta probando (un script ya sea de consulta, actualización, procedimiento almacenado, etc) se deberá generar un documento de Caso de Prueba que contemple la funcionalidad e impacto del mismo en el sistema.

3. Una vez que el Documento de Casos de Prueba fue aprobado, inicialmente por el **Líder del Proyecto**, éste pasará al **Tester**.
4. El **tester** previamente verificará que el ambiente de prueba se cumpla y ejecutará los casos con los datos de prueba proporcionados, documentando los resultados obtenidos y conclusiones inferidas en el mismo documento de Casos de Prueba.
5. A partir de los resultados obtenidos, y en el caso de que existieran errores el **Analista de Sistemas** compilará los errores registrados en los distintos Documentos de Casos de Prueba en un único Reporte de Errores, en éste se dejará constancia de todos los defectos, errores y fallos detectados, este artefacto es de características técnicas, dirigido principalmente al **Implementador** con el objetivo de que se corrijan los errores. El Reporte de Errores estará asociado a varios documentos de Casos de Prueba, será actualizado por el Implementador en los casos en que corresponda.
   - En los sistemas de código cerrado el reporte de errores también será generado por el **Analista de Sistemas** con el objetivo de enviar al SIU o a la entidad que corresponda y si existiera una solución (proporcionada por el SIU, etc.) será registrada por él en el mismo documento.
   - En los sistemas o aplicaciones de código abierto, a partir de la detección se realizará la depuración (localización y corrección de defectos si correspondiere).
6. La depuración estará a cargo del **Implementado**r, el cual puede o no[5] corregir los errores. Es su responsabilidad completar la acción tomada en el documento Reporte de Errores, columna Acción tomada.
7. Si se corrige un error, se debe volver a probar el software para comprobar que el problema este resuelto y cumple con la funcionalidad requerida, en este caso el Reporte de errores completo y el producto a evaluar volverá al **Analista de Sistemas** quien definirá un nuevo documento de Casos de Prueba, en el que se menciona si se trata de una ejecución o re-ejecución, este va al tester quien en el caso en que persistan los errores (los mismos o nuevos) los registrará en el mismo reporte de errores inicial, a partir de acá el proceso de repite para todas las situaciones en las que se continúen encontrando errores.
8. Una vez finalizadas las pruebas y depurados los errores encontrados el **Líder del Proyecto** generará el Informe de Resultados Obtenidos el cual va dirigido, a la **Gerencia Colaborativa**, **Gerencia General** y **cliente** según corresponda, el mismo

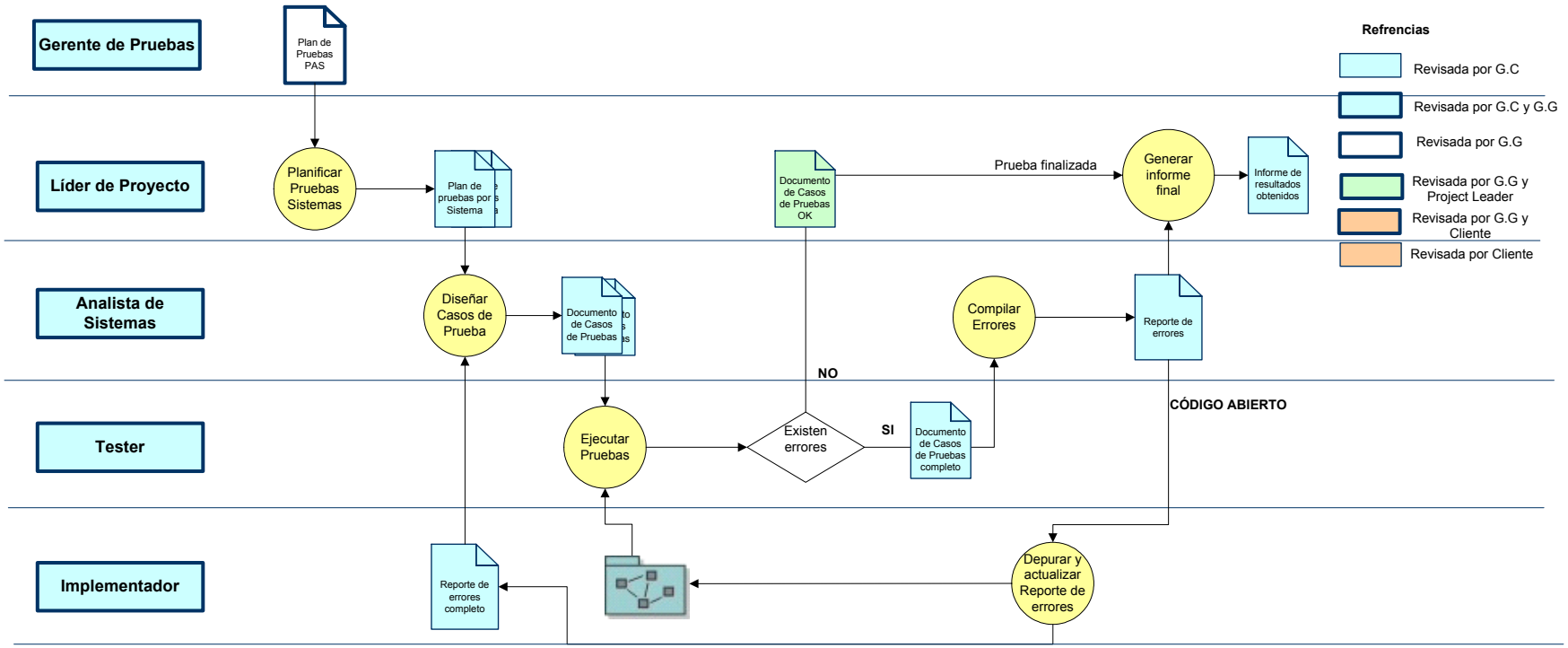
[5] En el caso de no corregir los errores deberá informar las causas: Ej Desconocimiento del origen del error, etc.

contiene un resumen de los resultados de las pruebas ejecutadas, aporta una evaluación del software basada en dichos resultados y una descripción del análisis de errores encontrados el cual servirá para realizar predicciones de la fiabilidad del software, detectar las causas más habituales de error y mejorar los procesos de desarrollo y prueba.

9. Una vez revisado volverá al **Líder de Proyecto** para que este lo almacene en el directorio correspondiente del espacio de trabajo.

3.1

Proceso de Pruebas



Página 1

4. Metodología de pruebas

4.1 Tipos de prueba

Si bien el área de estudio de las pruebas es muy amplia, a continuación de describen un conjunto de los tipos de pruebas que pueden ejecutarse en cada proyecto, las mismas se clasifican en las siguientes categorías:

4.1.1 Pruebas de Caja Blanca o prueba lógica

Las pruebas de caja blanca normalmente se denominan pruebas de cobertura o pruebas de caja transparente, al total de pruebas de caja blanca se le llama cobertura, la cobertura es un número porcentual que indica cuanto código del programa se ha probado.

Básicamente la idea de pruebas de cobertura consiste en diseñar un conjunto de pruebas en las que se va ejecutando sistemáticamente el código hasta que se haya ejecutado todo o la gran mayoría de él, si bien en algunos casos puede ser complicado una vez realizada se tiene la seguridad del funcionamiento de todos los módulos.

Las pruebas de caja blanca no reemplazan, solo complementan a las de caja negra y de aceptación, deben ser realizadas una vez terminado el software y no deben ser confundidas con las pruebas informales que realiza el programador durante el desarrollo, dado que si bien estas van cubriendo distintos fragmentos de código, no tienen un diseño apropiado.

En el plan de pruebas del sistema deberá especificarse si se utilizara este tipo de pruebas, en que módulos y la forma en que se llevaran a cabo.

Dentro de este tipo de pruebas se encuentran:

- **Prueba de caminos Básicos**

En este tipo de pruebas se realiza un análisis sobre la representación gráfica de un programa, denominada grafo de control. En este grafo, los nodos representan bloques de instrucciones de un programa y los flujos de ejecución para dichas instrucciones se representan por medio de aristas. A partir de este grafo es posible identificar un conjunto básico de caminos de ejecución, sobre el cual se pueden realizar pruebas con el propósito de ejercitar el flujo de ejecución de de los caminos en una unidad.

- **Pruebas de condiciones**

Estas pruebas se basan también en un grafo de control, pueden generarse casos de prueba para elementos individuales de expresiones lógicas. De esta forma se pretende probar cada condición con todas sus posibles alternativas

- **Pruebas de flujos de datos**

Estas pruebas son realizadas con el objetivo de detectar posibles contradicciones o redundancias en la definición de los datos utilizados en el software. Para lograrlo se efectúa un análisis del comportamiento de cada uno de los datos o de cada uno de os flujos de ejecución.

- **Pruebas de bucles**

A partir del grafo de control, es posible generar casos de prueba para las iteraciones definidas en los programas, con el propósito si se realizan de modo correcto.

Las pruebas que se incluyen dentro de la categoría de Caja Blanca se ejecutan solo en los casos en que se trate de software de código abierto.

4.1.2. Pruebas de Caja Negra

Sinónimos de este tipo de pruebas son pruebas funcionales, pruebas de entrada/salida o pruebas inducidas por los datos.

Las pruebas de caja negra se centran en lo que se espera de un módulo, intentan encontrar casos en que el módulo no se ajusta a su especificación. Por ello se denominan pruebas funcionales, y el tester se limita a suministrarle datos como entrada y estudiar la salida, sin preocuparse de lo que pueda estar haciendo el módulo por dentro.

Las pruebas de caja negra están especialmente indicadas en aquellos módulos que van a ser interfaz con el usuario.

Las pruebas de caja negra se apoyan en la especificación de requisitos del módulo. Se utiliza el termino "cobertura de especificación" para dar una medida del número de requisitos que se han probado, lo mas recomendable será conseguir una alta cobertura en esta línea.

Dentro de esta categoría se encuentran:

- **Partición equivalente :**

La idea de esta técnica es dividir los valores válidos y no válidos para entradas y salidas en un número de reducido de particiones, de forma tal que el comportamiento del software sea el mismo para cualquier valor contenido en una partición particular. El propósito principal de una partición es reducir la cantidad de casos de prueba generados en el proceso.

- **Análisis de los valores limites:**

La generación de casos de prueba en esta técnica se enfoca en los valores limite, bajo la consideración de que existe una tendencia a fallar precisamente cuando el software trabaja con valores extremos de las variables de entrada.

- **Pruebas según la experiencia :**

Este tipo de prueba se realiza a partir de la intuición y la experiencia. La idea básica es redactar una lista de las posibles fallas o de las posibles situaciones en las cuales suele ocurrir algún problema y así desarrollar casos de pruebas basados en la información contenida en esas listas.

- **Tablas de decisión:**

Permiten describir el comportamiento de un programa a partir de un conjunto de acciones que este realiza cuando opera en determinadas condiciones. En este enfoque las condiciones pueden ser interpretadas como entradas de un programa, y las acciones como las salidas producidas.

4.2 Estrategias de prueba

Las estrategias de software intentan agrupar diversos tipos y técnicas de prueba, a fin de asegurar que se hayan probado todos los niveles del software.

Utilizaremos los siguientes niveles para clasificar las pruebas:

4.2.1 Nivel de Unidad

En este nivel se prueba cada unidad o módulo con el objetivo de eliminar errores en la interfaz y en la lógica interna. Esta actividad utiliza caja negra y caja blanca, según convenga para lo que se desea probar.

Se deben diseñar pruebas para descubrir los siguientes errores:

- Tipificación impropia o inconsistente.
- Inicialización o valores implícitos erróneos.
- Nombres de variables incorrectos.
- Tipos de datos inconsistentes.
- Errores de cálculos: Dentro de este tipo de error se encuentran los errores de:
    - Precedencia aritmética incorrecta o mal interpretada.
    - Inicializaciones incorrectas.
    - Falta de precisión.
    - Incorrecta representación simbólica de una expresión.
- Comparaciones entre tipos de datos distintos.
- Operadores lógicos o de precedencia incorrectos.
- Igualdad esperada cuando los errores de precisión la hacen poco probable.
- Variable o comparaciones incorrectas.
- Terminación de bucles inapropiada o inexistente.
- Fallo de salida cuando se encuentra una iteración divergente.
- Bucles que manejan variables modificadas en forma inapropiada.

4.2.2 Nivel de Integración

El objetivo de estas pruebas es tomar los módulos probados en unidad y construir una estructura de programa que este de acuerdo con lo que dicta el diseño.

En este tipo de prueba se observa como se acoplan los distintos módulos, ya que lo que resulta de un módulo puede no ser lo que espera otro. Para esto existen dos enfoques fundamentales: uno ascendentes y otro descendente.

- Ascendentes: Empieza a agrupar desde los módulos de mayor jerarquía, avanzando progresivamente hacia los que poseen menor jerarquía.

- Descendente: Agrupa los módulos de menor jerarquía en racimos y avanza hacia los de menor jerarquía.

Independientemente del enfoque que se utilice, lo mas importante es realizar la integración de manera ordenada y, en caso de encontrar un error corregirlo y aplicar el concepto de regresión.

Este concepto expresa que ante la aparición de un error, se lo debe solucionar y luego realizar a ese modulo una nueva prueba de unidad.

4.2.3 Nivel de Sistema (En algunos casos pruebas de usuario)

Tras la culminación de la prueba de integración, el sistema estará completamente ensamblado y ya se corrigieron los errores de las pruebas unitarias y de integración.

El objetivo de estas pruebas es verificar que el sistema cumpla con los requerimientos funcionales y no funcionales definidos por el usuario, se deben observar aspectos con la recuperación del sistema, la seguridad, la resistencia, la escalabilidad y el rendimiento estándar.

- **Requerimientos Funcionales**

La validación del software se consigue mediante una seria de pruebas de caja negra que demuestran conformidad con los requerimientos. Estas pruebas pueden ser ejecutadas por el usuario apoyado y dirigido por el equipo de pruebas.

Las técnicas a utilizar son:

- Pruebas Alfa

Son conducidas por el usuario en el ambiente de prueba preparado para tal fin. El usuario utiliza el sistema en forma natural bajo la vigilancia del tester quien ira registrando de los errores y problemas encontrados.

- Pruebas Beta

Son realizadas por distintos usuarios finales en su lugar de trabajo. La prueba Beta es una ejecución del sistema en un ambiente controlado, aunque sin la presencia del equipo de prueba. El usuario registra todos los problemas detectados, reales y/o imaginarios y los informa al equipo de prueba en los intervalos definidos en el Plan de Prueba.

- **Requerimientos No funcionales**

Teniendo en cuenta el nivel de detalle técnico de estas pruebas no se requiere la participación del usuario durante la ejecución de las mismas.

Las pruebas a realizar son las siguientes:

- Prueba de recuperación:

Los sistemas a probar deben tener la capacidad de recuperarse ante cualquier contingencia, de acuerdo con los requerimientos no funcionales fijados por el usuario. En este tipo de pruebas se fuerza el fallo del software de muchas formas y se verifica que la recuperación se lleva a cabo apropiadamente.

Si la recuperación es automática hay que evaluar la correcta reinicialización de:

- Los mecanismos de recuperación del estado del sistema
- La recuperación de datos
- El proceso de re-arranque del sistema.
- Si la recuperación de la falla requiere de la intervención humana, hay que evaluar los tiempos de corrección para determinar si están dentro de los límites especificados.

- Pruebas de Seguridad y Control de Acceso

La prueba de seguridad debe asegurar razonablemente el cumplimiento de los requerimientos no funcionales relacionados con los objetivos de confidencialidad, disponibilidad, integridad y exactitud de la información.

Con estas pruebas se verificarán los mecanismos de protección incorporados en el sistema que lo protegerán de accesos ilegales

En cuanto a los mecanismos propios de la aplicación deberá constatarse que

- Las contraseñas no sean visibles al ser ingresadas.
- Se encuentren encriptadas y almacenadas fuera del acceso publico.
- Caducidad de sesiones de usuarios

Debe verificarse además que el esquema de copias de respaldo sea el adecuado. Se debe prever, en el marco de un simulacro de contingencia, la restauración de los datos respaldados y como registra el sistema el log de transacciones

- Prueba de Stress

El objetivo de esta prueba es testear el funcionamiento del sistema bajo condiciones extremas de demandas de recursos.
La prueba de stress debe ejecutar a la aplicación de manera tal que demande recursos en cantidad, frecuencia, volúmenes anormales.
Por Ejemplo:

- Diseñar pruebas que generen una gran cantidad de transacciones por segundo.
- Diseñar pruebas que incrementen en un orden de magnitud los volúmenes de datos almacenados para comprobar los tiempos de respuesta
- Ejecutar casos de prueba que requieren el máximo de memoria o de otros recursos

- Pruebas de Performance

Esta prueba esta diseñada para probar el rendimiento del software en tiempo de ejecución dentro del contexto de un sistema integrado
La validez de estas pruebas esta sujeta a poder simular todas las condiciones de funcionamiento del ambiente de producción, entre otros factores plataforma de hardware, líneas de comunicaciones y concurrencia de usuarios.

- Pruebas de Volumen

Verifica el comportamiento de la base del sistema frente a grandes cantidades de datos. En este tipo de pruebas se deberá especificar por ejemplo cual es la cantidad normal de registros con los que trabaja la aplicación y con que volumen se realizara las pruebas, si esa información estará centralizada o distribuida, etc.

4.2.4 Nivel de Aceptación

Las pruebas de aceptación son el último testeo que se le realiza a la aplicación antes de que ingrese en producción. El subconjunto de pruebas que se lleven a cabo es fijado por el equipo de pruebas, de acuerdo a lo requerido por los usuarios claves, ya que son estos quienes en base a los resultados de las pruebas darán su aprobación.

4.3 Las Pruebas dentro del Proceso de Desarrollo

A continuación se describe como se relaciona cada estrategia de prueba con los artefactos generados dentro del proceso de desarrollo adoptado por el área, cada una de las pruebas tendrá que tener como entrada uno o varios artefactos correspondientes al proceso para que el resultado de las pruebas sea exitoso, las correcciones trazeables y la documentación sea consistente.

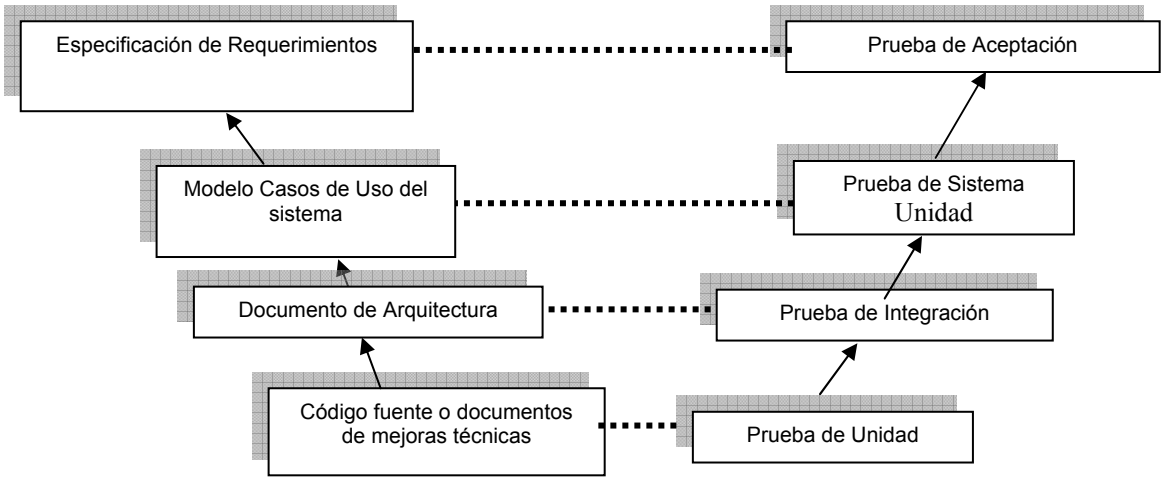


Figura Nº 2: Las pruebas dentro del proceso de desarrollo

5. Recursos

En esta sección se presentan los recursos que estarán afectados para las pruebas de los distintos sistemas, aplicativos, scripts, etc. durante el período para el cual se desarrollo este plan detallando para cada uno el rol asociado, es importante tener en cuenta que una persona puede cumplir mas de un rol en determinados momentos.

5.1 Recursos Humanos

Nombre del Recurso	Responsable
Gerencia Colaborativa	MMI-SMO-MDI
Gerente de Pruebas	MDI
Líder proyecto	PVI – Proyecto Hacienda LCA, JCV - CIS FAR – Proyecto Académica
Analista de Sistemas	- Proyecto : Hacienda CGA , PVI , MCH - Proyecto : Académica DMA, FAR, GQU - CIS: LCA , JVA
Implementador	GQU MCH -CGA
Testers	- Proyecto : Hacienda CGA , PVI , MCH - Proyecto : Académica DMA, FAR, GQU - CIS: LCA, JVA

Tabla 2: Entregables y criterios de aceptación

5.2 Recursos Materiales

Esta sección describe todos los recursos necesarios para ejecutar el proceso de pruebas, tanto las herramientas de automatización, las plantillas, etc.

Nombre del Recurso	Ubicación
- Software de automatización de las pruebas	
– OpenLink ODBC Benchmark Utility	Zeppelín\CIS
– Benchmark Factory for Database 4.7.1 for Windows	Zeppelín\CIS
Plantillas	
– FOR074_CASOS DE PRUEBA_V1.0.dot	Zeppelín\PRODUCCION\FORMULARIOS
– FOR075_REPORTE DE ERRORES ENCONTRADOS_V1.0.dot	Zeppelín\PRODUCCION\FORMULARIOS
– FOR073 80_INFORME DE RESULTADOS OBTENIDOS_V1.0.dot_	Zeppelín\PRODUCCION\FORMULARIOS
– FOR015_PLAN DE PRUEBAS_V2.1.2.0 dot	Zeppelín\PRODUCCION\FORMULARIOS

Tabla 3: Recursos utilizados en el proceso de pruebas

6. Calendario

El calendario de pruebas se describirá en los planes de pruebas de cada sistema, en el cual se mencionara: El responsable de la prueba, el artefacto generado y el responsable del mismo.

7. Resultados de las pruebas

Para todas las pruebas realizadas se deberá registrar el resultado en el documento de Casos de Prueba, este servirá para cotejar las diferencias con los resultados esperados, los errores encontrados se describirán en el *Reporte de Errores Encontrados*, la manera de completar este reporte se detalla en línea Guía correspondiente.

Si el resultado de la prueba no es satisfactorio, en caso de ser un producto del área volverá al implementador y en el caso de ser un producto del SIU se analizaran las posibles alternativas: personalización, cambio de versión y estos resultados quedaran registrados en el Informe de resultados obtenidos.

Estos documentos quedarán almacenados en el directorio Testing correspondiente al proyecto y sistema en cuestión, de acuerdo al ambiente que corresponda.

8. Apéndices

9. Glosario

Error (error):
Es una equivocación cometida por el desarrollador. Algunos ejemplos de errores son: un error de tipeo, una mal interpretación de un requerimiento o de la funcionalidad de un método.

Defecto (fault, defect):
Un error puede conducir a uno o más defectos. Un defecto se encuentra en un artefacto y puede definirse como una diferencia entre la versión correcta del artefacto y una versión incorrecta. Coincide con la definición de diccionario, "imperfección". Por ejemplo, un defecto es haber utilizado el operador "<" en vez de "<=".

Falla (failure):
En terminología IEEE, una falla es la discrepancia visible que se produce al ejecutar un programa con un defecto. En general, un proceso (incluido el de ejecución) puede producir fallas, en la medida que no cumple con su cometido.

Depuración
El proceso de analizar y corregir los defectos que se sospecha que contiene el software.

Código muerto: Aquellas funciones y/o procedimientos que hemos incluido por encontrarse en recopilaciones pero que estas nunca son ejecutadas por el programa, estas funciones no necesariamente deberán ser removidas pero si probadas por si algún día en revisiones futuras son incluidas.

Pruebas (test): una actividad en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas, los resultados se observan y registran y se realiza una evaluación de algún aspecto

Caso de prueba (test case): un conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para un objetivo particular

Verificación: El proceso de evaluación de un sistema (o de uno de sus componentes para determinar si los productos de una fase dada satisfacen las condiciones impuestas al comienzo de dicha fase

Validación: El proceso de evaluación de un sistema o de uno de sus componentes durante o al final del proceso de desarrollo para determinar si satisface los requisitos marcados por el usuario.