

**PLAN DE PRUEBAS DETALLADO**

**SISTEMA DE NOTIFICACIÓN EN LÍNEA**

**PLATAFORMA DE INTEROPERABILIDAD, PDI**

**INTRANET GUBERNAMENTAL**

**© República de Colombia - Derechos Reservados**

Bogotá, D.C., Diciembre de 2008

**FORMATO PRELIMINAR AL DOCUMENTO**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Título: | **PLAN DE PRUEBAS DETALLADO**  **SISTEMA DE NOTIFICACIÓN EN LÍNEA** | | | | | | |
| Fecha elaboración aaaa-mm-dd: | 02 – Diciembre – 2.008 | | | | | | |
| Sumario: | Este documento tiene por objeto establecer el contenido y criterios de aceptación para el entregable “PLAN DE PRUEBAS DETALLADO” para el proyecto SISTEMA DE NOTIFICACIONES Y COMUNICACIONES ELECTRÓNICAS. | | | | | | |
| Palabras Claves: |  | | | | | | |
| Formato: | DOC | | Lenguaje: | | | | Español |
| Dependencia: | Ministerio de comunicaciones: programa “Agenda de Conectividad” –  Proyecto Sistema de Notificaciones y Comunicaciones Electrónicas | | | | | | |
| Código: |  | Versión: | 2.0 | | Estado: | | APROBADO |
| Categoría: |  | | | | | | |
| Autor (es): | Equipo U.T e-Notificaciones | | | Firmas: | |  | |
| Revisó: | Gerente de Proyecto U.T. e-Notificaciones  Álvaro Rueda Zapata | | |  | |
| Aprobó: | Gerente de Proyecto Programa Agenda de Conectividad  Johanna Pimiento Quintero  Gerente de Interventoría  Manuel Gomez Restrepo  Gerente de Proyecto U.T. e-Notificaciones  Álvaro Rueda Zapata | | |  | |
| Información Adicional: |  | | | | | | |
| Ubicación: |  | | | | | | |

**CONTROL DE CAMBIOS**

| **VERSIÓN** | **FECHA** | **No. SOLICITUD** | **RESPONSABLE** | **DESCRIPCIÓN** |
| --- | --- | --- | --- | --- |
| 1.0 | 31/07/2008 |  | Equipo U.T. e-Notificaciones | Creación de documento |
| 1.1 | 29/10/2008 |  | Equipo U.T. e-Notificaciones | Inclusión de las siguientes modificaciones según observaciones del Programa Agenda de Conectividad e Interventoría.   * Inclusión de resultados de la etapa de planificación en el numeral 2.1.1.1. * Fueron incluidas las pruebas de seguridad en el punto 2.3.8. * Fueron incluidas la lista de Dummies a desarrollar para las pruebas de interoperabilidad en el punto 2.3.4. * Fue incluida la tipificación de pruebas en el punto 2.5 * Fueron incluidas las características del ambiente de pruebas en el punto 3.2.1. * Fueron incluidas las técnicas de ejecución de pruebas en el punto 2.6. * Fue incluida la lista de chequeo para pruebas funcionales en el punto 5.2.1. * Fueron descritas las pruebas y las herramientas a utilizar en los puntos 2.5, 2.6, 3.1, 3.2.1. * Para cada herramienta de pruebas de identifico en qué tipo de pruebas será utilizada en el punto 2.3. |
| 1.2 | 06/11/2008 |  | Equipo U.T. e-Notificaciones | * Inclusión del formato para el diseño de las pruebas técnicas en el punto 5.2.3 * Inclusión de matriz de trazabilidad de pruebas técnicas versus requerimientos no funcionales en el punto 5.2.4. * Fueron eliminadas las pruebas del plan piloto por no ser parte de los términos de referencia. |
| 1.3 | 19/11/2008 |  | Equipo U.T. e-Notificaciones | Ajustes al documento de acuerdo a observaciones del Programa Agenda de Conectividad y la Interventoría. |
| 1.4 | 01/12/2008 |  | Equipo U.T. e-Notificaciones | Ajustes al documento de acuerdo a observaciones del Programa Agenda de Conectividad y la Interventoría en reunión del 28/11/2008. |
| 2.0 | 02/12/2008 |  | Equipo U.T. e-Notificaciones | Ajustes al documento de acuerdo a observaciones del Programa Agenda de Conectividad y la Interventoría en reunión del 02/12/2008. |

**TABLA DE CONTENIDO**

[DERECHOS DE AUTOR 6](#_Toc216002462)

[1. INTRODUCCIÓN 7](#_Toc216002463)

[**1.1.** **PROPÓSITO** 7](#_Toc216002464)

[**1.2.** **ALCANCE** 7](#_Toc216002465)

[**1.3.** **DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS** 10](#_Toc216002466)

[**1.4.** **REFERENCIAS** 10](#_Toc216002467)

[**1.5.** **VISTA GENERAL** 11](#_Toc216002468)

[2. ESTRATEGIA DE PRUEBAS 13](#_Toc216002469)

[**2.1.** **TÉCNICAS DE ESPECIFICACIÓN DE LAS PRUEBAS** 13](#_Toc216002470)

[**2.1.1.** **CICLO DE PRUEBAS** 13](#_Toc216002471)

[**2.1.1.1.** **PLANIFICACIÓN** 13](#_Toc216002472)

[**2.1.1.2.** **DISEÑO DE LAS PRUEBAS** 15](#_Toc216002473)

[**2.1.1.3.** **CONFIGURACIÓN** 16](#_Toc216002474)

[**2.1.1.4.** **EJECUCIÓN** 17](#_Toc216002475)

[**2.1.1.5.** **EVALUACIÓN Y CIERRE** 19](#_Toc216002476)

[**2.1.1.6.** **SEGUIMIENTO Y CONTROL** 19](#_Toc216002477)

[**2.2.** **HERRAMIENTAS PARA PRUEBAS** 19](#_Toc216002482)

[**2.2.1.** **JUNIT** 20](#_Toc216002483)

[**2.2.2.** **HTTPUNIT** 20](#_Toc216002484)

[**2.2.3.** **JMETER** 21](#_Toc216002485)

[**2.2.4.** **GRINDER** 21](#_Toc216002486)

[**2.2.5.** **NESSUS** 22](#_Toc216002487)

[**2.2.6.** **XMLBuddy** 22](#_Toc216002488)

[**2.3.** **TIPOS DE PRUEBAS** 23](#_Toc216002489)

[**2.3.1.** **PRUEBAS UNITARIAS** 23](#_Toc216002490)

[**2.3.2.** **PRUEBAS DEL SISTEMA** 24](#_Toc216002491)

[**2.3.3.** **PRUEBAS DE INTEGRACIÓN** 25](#_Toc216002492)

[**2.3.4.** **PRUEBAS DE INTEROPERABILIDAD** 25](#_Toc216002493)

[**2.3.5.** **PRUEBAS DE REGRESIÓN** 26](#_Toc216002494)

[**2.3.6.** **PRUEBAS FUNCIONALES** 27](#_Toc216002495)

[**2.3.7.** **PRUEBAS DE USABILIDAD** 28](#_Toc216002496)

[**2.3.8.** **PRUEBAS DE SEGURIDAD** 29](#_Toc216002497)

[**2.3.9.** **PRUEBAS DE CONFIGURACIÓN** 29](#_Toc216002498)

[**2.3.10.** **PRUEBAS DE RECUPERACIÓN A FALLAS** 30](#_Toc216002499)

[**2.3.11.** **PRUEBAS DE ACEPTACIÓN** 31](#_Toc216002500)

[**2.4.** **ENTREGABLES DE PRUEBAS** 31](#_Toc216002501)

[**2.5.** **MATRIZ DE TIPIFICACIÓN DE PRUEBAS** 32](#_Toc216002502)

[**2.6.** **TECNICAS DE EJECUCIÓN DE PRUEBAS** 32](#_Toc216002503)

[3. RECURSOS DEL PLAN DE PRUEBAS 38](#_Toc216002504)

[**3.1.** **RECURSO HUMANO** 38](#_Toc216002505)

[**3.2.** **RECURSO DEL SISTEMA** 38](#_Toc216002506)

[**3.2.1.** **CONFIGURACION DEL AMBIENTE DE PRUEBAS** 39](#_Toc216002507)

[**3.3.** **HERRAMIENTAS DE REPORTES Y CONTROL DE INCIDENCIAS** 40](#_Toc216002508)

[**3.4.** **ADMINISTRACIÓN DE VERSIONES** 40](#_Toc216002509)

[**3.4.1.** **HERRAMIENTAS** 41](#_Toc216002510)

[4. EVALUACIÓN DE PRUEBAS EJECUTADAS 43](#_Toc216002511)

[**4.1.** **CRITERIOS DE INICIO DE EJECUCIÓN** 43](#_Toc216002512)

[**4.2.** **CRITERIOS DE EVALUACION** 43](#_Toc216002513)

[**4.3.** **CRITERIOS DE TERMINACIÓN** 47](#_Toc216002514)

[**4.4.** **CRITERIOS DE SUSPENSIÓN** 48](#_Toc216002515)

[5. ANEXOS 49](#_Toc216002516)

[**5.1.** **RELEASE NOTES** 49](#_Toc216002517)

[**5.2.** **CASOS DE PRUEBAS** 50](#_Toc216002518)

[**5.2.1.** **FORMATO CASOS DE PRUEBA FUNCIONALES** 50](#_Toc216002519)

[**5.2.2.** **LISTA DE CHEQUEO CASOS DE PRUEBAS FUNCIONALES** 51](#_Toc216002520)

[**5.2.3.** **ENCUESTA PARA PRUEBAS DE USABILIDAD** 52](#_Toc216002521)

[**5.2.4.** **FORMATO CASOS DE PRUEBA TECNICOS** 56](#_Toc216002522)

[**5.2.5.** **MATRIZ CASOS DE USO VS CASOS DE PRUEBA FUNCIONALES** 57](#_Toc216002523)

[**5.2.6.** **MATRIZ REQUERIMIENTOS NO FUNCIONALES VS CASOS DE PRUEBA TÉCNICOS** 58](#_Toc216002524)

[**5.3.** **LISTA DE CHEQUEO** 58](#_Toc216002525)

[**5.4.** **INFORME DE PRUEBAS** 59](#_Toc216002526)

[**5.5.** **PROCEDIMIENTO PARA INCIDENCIAS** 59](#_Toc216002527)

DERECHOS DE AUTOR

A menos que se indique de forma contraria, el copyright del texto incluido en este documento es del gobierno de la República de Colombia. Se puede reproducir gratuitamente en cualquier formato o medio sin requerir un permiso expreso para ello, bajo las siguientes condiciones:

1. El texto particular no se ha indicado como excluido y por lo tanto no puede ser copiado o distribuido.

2. La copia no se hace con el fin de distribuirla comercialmente.

3. Los materiales se deben reproducir exactamente y no se deben utilizar en un contexto engañoso.

4. Las copias serán acompañadas por las palabras "copiado/distribuido con permiso de la República de Colombia. Todos los derechos reservados."

5. El título del documento debe ser incluido al ser reproducido como parte de otra publicación o servicio. Si se desea copiar o distribuir el documento con otros propósitos, debe solicitar el permiso entrando en contacto con el programa Agenda de Conectividad del Ministerio de Comunicaciones de la República de Colombia.

1. INTRODUCCIÓN

El contenido de este documento de plan de pruebas hace parte integral de la metodologia de pruebas que se encuentra en el capitulo 8 del documento general denominado “*Plan de Proyecto*”; el contenido de estos dos (2) documentos se encuentran fundamentados en estándares de calidad que no solo permiten el seguimiento y correcciones a tiempo del software sino que además se encuentra definido por etapas, facilitando el seguimiento y control de los procesos del proyecto en desarrollo y proporcionando a la UNION TEMPORAL E-NOTIFICACIONES garantizar la operatividad y funcionalidad de la solución desarrollada.

Acorde con el enfoque del desarrollo de la solución, el plan de pruebas está basado en la metodología de Rational Unified Process (RUP), lo que hace que este plan de pruebas tenga como propósito establecer las técnicas, herramientas y actividades relacionadas con la ejecución y validación de cada una de las prueas, incluyendo responsabilidades de cada una de las actividades, los recursos y los prerequisitos que deben ser considerados en el esfuerzo de cada una de las pruebas; lo anterior permite garantizar el cumplimiento de los requerimientos planteados en el marco del desarrollo del proyecto denominado “ SISTEMA DE NOTIFICACIONES Y COMUNICACIONES ELECTRÓNICAS”.

* 1. **PROPÓSITO**

Este documento tiene como propósito establecer las técnicas, herramientas y actividades relacionadas con la ejecución y validación del plan de pruebas; incluye responsabilidades de cada una de las tareas, los recursos y los prerequisitos que deben ser considerados en el esfuerzo de cada una de las pruebas, permitiendo garantizar el cumplimiento de los requerimientos planteados en el marco del desarrollo del proyecto denominado “SISTEMA DE NOTIFICACIONES Y COMUNICACIONES ELECTRÓNICAS”.

* 1. **ALCANCE**

Este documento de PLAN DE PRUEBAS DETALLADO, se convierte en una guía para desarrollar de una forma organizada las diferentes actividades que se realizarán en el proceso del plan de pruebas en el desarrollo del proyecto denominado “SISTEMA DE NOTIFICACIONES Y COMUNICACIONES ELECTRÓNICAS”.

La metodologia de pruebas y este documento de plan de pruebas permitirán al equipo profesionales expertos que participan en el frente de pruebas del proyecto denominado “SISTEMA DE NOTIFICACIONES Y COMUNICACIONES ELECTRÓNICAS”, evaluar aspectos como: la lógica estructural, la seguridad, la interconexión, el soporte conceptual, las herramientas de apoyo y sobretodo la independencia de aspectos técnicos del desarrollo de la solución tecnológica contratada, tales como: la plataforma tecnológica o la arquitectura de la solución a probar, sin embargo a continuación se describen las diferentes pruebas a ser aplicadas:

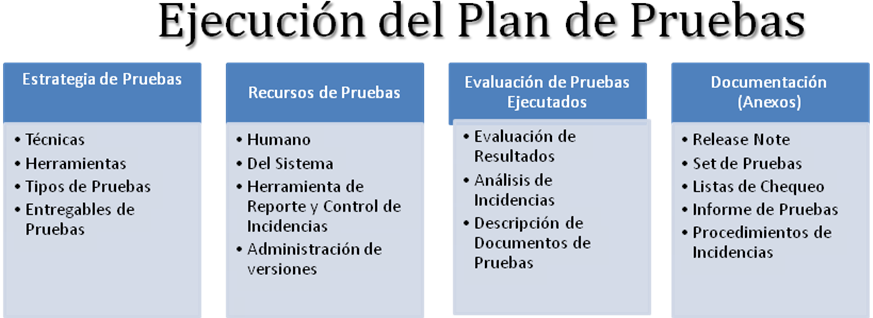
|  |  |  |
| --- | --- | --- |
| **TIPO DE PRUEBA** | **DEFINICIONES** | **FASE DE RUP** |
| UNITARIAS  INTEGRACIÓN | **Unitarias:** Permite verificar la funcionalidad y estructura de cada componente individualmente del sistema una vez que ha sido codificado.  **Integración:** Permite verificar el correcto ensamblaje entre los distintos módulos que componen el sistema desarrollado. | **ELABORACIÓN** |
| SISTEMA:   * Carga * Volumen * Estress * Robustez * Concurrencia, * Interfaz de Usuario Recuperación a Fallas * Rendimiento * Seguridad * Integridad de las BD * Interoperabilidad * Desempeño * Configuración | **Sistema:** Estas pruebas buscan diferencias entre la solución desarrollada y los requerimientos, con el fin de identificar errores que se puedan generar entre la especificación funcional y el diseño del sistema.  **Carga:** Valida aquellos volúmenes de datos máximos especificados en los requerimientos no Funcionales  **Volumen:** Esta prueba somete el software a grandes cantidades de datos para determinar si se alcanzan límites que causen la falla del software  **Estrés:** Valida aquellos volúmenes de datos máximos que resiste el sistema antes de comenzar con errores.  **Robustez:** Valida si el sistema se mantiene estable y consistente después de circunstancias adversas  **Concurrencia:** Valida la capacidad del sistema de atender múltiples solicitudes departe de los usuarios que acceden a un mismo recurso.  **Interfaz de usuario**: Ppermite verificar que la navegación a través de los elementos que se están probando, reflejen las funciones del negocio y los requerimientos funcionales.    **Recuperación a fallas**: Estas pruebas aseguran que el que el software pueda recuperarse a fallas de hardware, software o mal funcionamiento de la red sin pérdida de datos o de integridad de los datos.  **Rendimiento:** Permite validar si la aplicación cumple los criterios de tiempos de respuesta establecidos.  **Seguridad**: Verifica el cumplimiento de las políticas de seguridad acordadas para el sistema.  **Integridad de las bases de datos:** Consiste en asegurar que los métodos y procesos de acceso a la base de datos funcionan correctamente y sin corromper datos.  **Interoperabilidad:** Esta prueba permite verificar todos los artefactos de la solución desarrollada, su arquitectura base, los protocolos de la solución, las interfaces y los módulos del sistema, funcionando en forma conjunta.  **Desempeño:** Este tipo de prueba es un aspecto fundamental en una aplicación, ya que si ésta no responde en el debido tiempo, se pueden perder clientes, o dañar la imagen ante los usuarios.  **Configuración:** Establece y mantiene la integridad de los productos de software a través del ciclo de vida del proceso del mismo. | **CONSTRUCCIÓN** |
| FUNCIONALES | **Funcional:** La prueba funcional es un proceso para procurar encontrar discrepancias entre el programa y la especificación funcional.  **Caja Negra:** Estas pruebas permiten obtener conjuntos de condiciones de entrada que ejecutan todos los requisitos funcionales de un programa.  **Ciclo de Negocio:** Esta prueba tiene por objeto garantizar que el proceso de negocio esta adecuadamente soportado por el software desarrollado y que éste dispone de la funcionalidad adecuada para ejecutar todas las tareas incorporadas en el proceso de negocio.  **Usabilidad:**  Esta prueba permite encontrar problemas de factores humanos, o usabilidad.  **Instalación:** Esta prueba permite verificar la instalación y desinstalación de la aplicación en diferentes entornos de hardware y software |
| ACEPTACIÓN | Es la prueba final basada en las especificaciones del usuario o basada en el uso del programa por el usuario final luego de un periodo de tiempo | **TRANSICIÓN** |
| REGRESIÓN | En esta prueba se valida que el sistema mantenga su correcta funcionalidad debido a la incorporación de un ajuste, corrección o nuevo requerimiento. Es una prueba funcional y técnica que valida que el sistema siga funcionando perfectamente después de que las correcciones sean aplicadas. |

* 1. **DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS**
* **El plan de prueba:** describe todos los métodos que se utilizarán para verificar que el software satisface la especificación del producto y las necesidades del cliente. Incluye los objetivos de calidad, necesidades de recursos, cronograma, asignaciones, métodos, etc.
* **Casos de prueba:** lista los ítems específicos que serán probados y describe los pasos detallados que serán seguidos para verificar el software.
* **Reporte de pruebas:** describen los problemas encontrados al ejecutar los casos de prueba.
* **Herramientas de pruebas y automatización:** documentación de las herramientas empleadas en el proceso de pruebas.
* **Métricas, estadísticas y resúmenes:** indican como ha sido el progreso del proceso de prueba.
  1. **REFERENCIAS**

Algunos documentos del proyecto SISTEMA DE NOTIFICACIONES Y COMUNICACIONES, son base fundamental de este documento inicial de plan de pruebas, que a continuación se relacionan

* Metodología de Pruebas – Punto 8 pagina 67 del Plan de Proyecto.
* Documento de Especificación de Requerimientos.
* Documento de Arquitectura General Detallada
* Términos de referencia del proceso de licitación de la plataforma de interoperabilidad.
  1. **VISTA GENERAL**

Descripción resumida de contenido de cada una de las secciones que siguen, y explicación de la forma en que está organizado el presente documento.

****

**Estrategia de Pruebas:**

En este capítulo se presenta una perspectiva general de la estrategia que se va a seguir para analizar, diseñar, implementar y ejecutar las pruebas del proyecto “SISTEMA DE NOTIFICACIONES Y COMUNICACIONES ELECTRÓNICAS”. Así mismo se definirá qué tipos de pruebas se van a realizar y cómo se ejecutarán.

**Recursos del Plan de Pruebas:**

Este capítulo identifica los recursos humanos y no humanos (hardware, software, herramientas de soporte, configuración de entorno de pruebas, entre otros), necesarios para desarrollar el proceso del plan de pruebas de la solución del Sistema de Notificación en Línea.

**Evaluación de Pruebas Ejecutadas:**

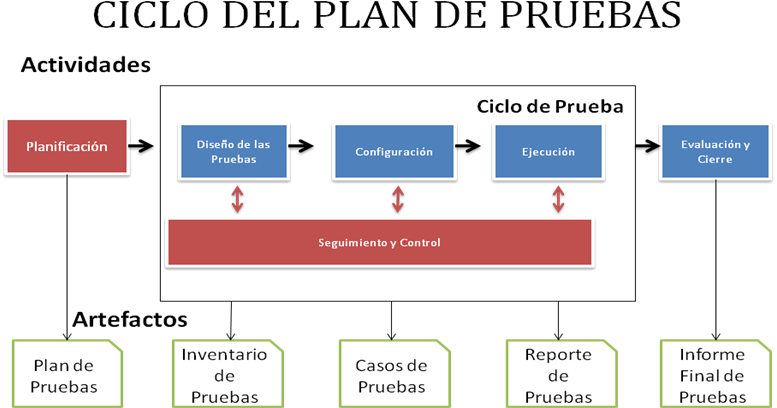
En este capítulo se describe de los métodos de evaluación de las pruebas ejecutadas, de tal forma que permitirá evaluar los grados de aceptación de las pruebas.

**Anexos:**

En este capítulo se describen los documentos anexos que se utilizarán para la especificación y la documentación de la ejecución de las pruebas.

1. ESTRATEGIA DE PRUEBAS
   1. **TÉCNICAS DE ESPECIFICACIÓN DE LAS PRUEBAS**

La estrategia del proceso del plan de pruebas se implementará de acuerdo al esquema de macro-actividades que se presenta en la siguiente gráfica:



* + 1. **CICLO DE PRUEBAS**

El ciclo de pruebas comprende seis actividades las cuales deberán ser desarrolladas de la siguiente manera:

* + - 1. **PLANIFICACIÓN**

Para el desarrollo de la solución del Sistema de Notificación en Línea, se considera de gran importancia la ejecución del plan de pruebas, haciéndose necesario la planificación de las mismas, lo que en consecuencia hace necesario tener claro los siguientes planteamientos:

* Se planifican pruebas personalizando los estándares específicamente para el proyecto de notificaciones.
* Se definen niveles de pruebas a aplicar.
* Se especifican las técnicas a utilizar.
* Se establece el tiempo para la ejecución de cada una de las pruebas.
* Uso de herramientas.
* Criterios de aceptación.
* Recursos involucrados.

En la definición del plan de pruebas, se valorará:

* El alcance de la aplicación.
* La complejidad de sus procesos.
* Plataforma/s en las que se debe probar.
* Conocimientos y formación de quienes ejecutarán las pruebas.
* Normativas legales aplicables.
* Otros recursos involucrados.

Se tendrá en cuenta que:

* Las pruebas estarán presentes a lo largo de todo el ciclo de vida del desarrollo, de la solución.
* Siempre hay errores.
* Probar exhaustivamente el software es imposible.
* No es recomendable que el programador pruebe sus propios programas.
* Se puede disponer de herramientas.
* Se debe considerar la importancia de actualización del plan de pruebas con el fin de reflejar los cambios que se produzcan en los requisitos y/o proceso de desarrollo del producto.

Resultado de la planificación:

* Cronograma detallado de la ejecución de las pruebas; donde se especifica qué prueba se realiza, cuánto tiempo se estima para su ejecución, recursos a utilizar (humanos y tecnológicos); este cronograma se encuentra dentro del cronograma general del proyecto y específicamente en la fase desarrollo (ver plan de construcción)
* Formatos a utilizar para el diseño de las pruebas.
* Formatos a utilizar para el registro y análisis de los resultados de las pruebas.
* Herramientas a utilizar para la gestión de incidencias.
* Procedimientos para el control de cambios.
* Herramientas a utilizar para la ejecución de las pruebas.
  + - 1. **DISEÑO DE LAS PRUEBAS**

Para el diseño de las pruebas, se tendrán en cuenta aspectos que permitirán encontrar defectos en el periodo de desarrollo del software, la realización de pruebas propias de verificación y validación de datos, según se aclara en los siguientes ítems:

1. **Alcance:** El alcance de las pruebas estará dado por el marco del Sistema de Notificación en Línea, que se encuentra en desarrollo, ésta compuesta (Información tomada de los términos de referencia y del documento de Arquitectura General Detallada) por:

* Modelo Conceptual.
* Procesos.
* Descripción de Procesos.
* Vista de Casos de Uso.
* Vista Lógica.
* Diseño de las clases y su organización en paquetes y subsistemas.
* Vista de Datos.
* Vista de Implementación.
* Vista de Despliegue.
* Vista de Integración con Sistemas Externos.
* Vista de Parametrización del Sistema.
* Requerimientos no Funcionales.
* Prototipos del sistema

1. **Inventario de las Pruebas**: En esta sección se especifica el inventario de las pruebas, el cual permitirá:

* Definir y asignar prioridades como; alta, media o baja.
* Establecer un orden de trabajo.
* Decidir qué casos entrarían en una regresión y cuáles no con mayor facilidad.
* Recortar alcance en forma rápida y ordenada.
* Se estima el tiempo en probar cada funcionalidad.
* Evaluar aspectos técnicos del sistema.

**C**

1. **Resultado de la ejecución de las Pruebas:** En este punto se resaltan las entradas fundamentales que son la partida para la ejecución del plan de pruebas.

* Inventario de pruebas priorizado.
* Estimación de esfuerzo de cada funcionalidad.
* Plan de desarrollo del producto.
* Plazos previstos para el proyecto.



1. **Ciclo de la Prueba:** Las actividades de la prueba se realizarán para una determinada versión del producto, sobre la cual se ejecutan las pruebas y se reportan los incidentes encontrados. Para cada versión del producto se realizan alguna o todas las tareas asociadas a las pruebas.



****

El proceso de planificación se ajusta al comenzar cada ciclo debido a posibles:

* Atrasos de desarrollo
* Modificaciones en los requerimientos iníciales
* Cambios en el alcance del producto
* Calidad del producto
  + - 1. **CONFIGURACIÓN**

Este capítulo se enfoca a la definición del proceso de administración de la configuración del proyecto “SISTEMA DE NOTIFICACIONES Y COMUNICACIONES ELECTRÓNICAS”, en el cual se establece el mantenimiento e integridad del software a través del ciclo de vida del proyecto y se proveen contextos de trabajo estables para los posibles cambios antes de ser entregado formalmente en producción.

A continuación se presenta una definición de los conceptos básicos de la disciplina de administración de configuraciones, una descripción de las actividades principales y una propuesta de formatos para facilitar la captura de la información necesaria en las distintas actividades.

**Administración de Configuraciones**: Es el proceso de identificar y definir los elementos o ítems de configuración del sistema, controlando la entrega y el cambio de estos elementos a través del ciclo de vida del sistema, almacenando el estado de los mismos y de las solicitudes de cambio, y verificando la completitud con respecto a los requisitos especificados.

**Configuración**: Conjunto completo (respecto de la Arquitectura del Sistema, es decir que cada componente está representado) y coherente (respecto de que defina una versión estable del sistema, es decir que las versiones de cada componente se correspondan) de Ítems de Configuración que constituyen un producto de software.

**Comité de control de cambios:** Grupo con la autoridad para evaluar, aprobar y/o rechazar la implementación de un cambio. El establecimiento de un **Comité de control de cambios** tiene como objetivo proveer un mecanismo para asegurar que toda solicitud de cambio es direccionada adecuadamente.

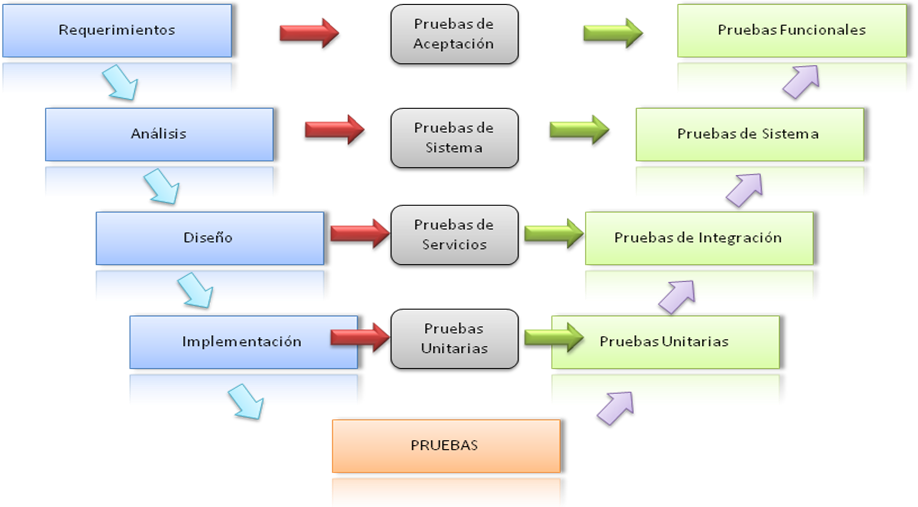
**Ítem de Configuración**: Componente de Software y/o producto de software destinado para ser puesto bajo Administración de Configuraciones.

**Solicitud de Cambio**: Documento a través del cual el equipo técnico autorizado solicita al Grupo de Desarrollo realizar la corrección de un defecto del Sistema de Notificación en Línea o de una mejora sobre la solución antes de salir a producción.

**Versión**: Resultado de la evolución que ha sufrido un Componente de Software en el tiempo.

* + - 1. **EJECUCIÓN**

En el siguiente gráfico se muestra el modelo estándar de ejecución de pruebas:



Como se observa, representa un modelo de pruebas en V, a diferencia de los modelos clásicos, extiende las pruebas a lo largo de todo el ciclo de vida del software.

Mientras se realizan las fases de requerimientos, análisis, diseño e implementación se van diseñando las pruebas del mismo nivel. Al llegar a la etapa de pruebas se inicia la ejecución de lo diseñado desde las pruebas unitarias hasta las pruebas funcionales.

Para cada una de las pruebas se realizará el siguiente procedimiento:



Aquí se tendrán en cuenta las siguientes especificaciones:

* Elementos del sistema, es decir; los módulos y características de la solución que se van a probar.
* Se listarán las especificaciones de cada entrada requerida para ejecutar el caso; incluyendo la sincronizaciones entre cada una de estas.
* Especificaciones de todas las salidas y las características requeridas como el tiempo y la respuesta para los elementos que se van a probar. Estas especificaciones se harán utilizando los formatos establecidos en el numeral 5 de este plan de pruebas.
* Necesidades del entorno del proceso de ejecución del hardware, software y recurso humano.
* Requisitos especiales de procedimiento o restricciones especiales en los procedimientos para ejecutar este caso.
  + - 1. **EVALUACIÓN Y CIERRE**

Para la evaluación y cierre de las pruebas se presentará el informe de pruebas donde se documentará el resultado de cada una de las diferentes pruebas ejecutadas. El contenido de este informe estará compuesto de la manera descrita en la “Propuesta de esquema y contenido del Informé de pruebas”; esto ya que el informe de pruebas es un entregable independiente.

* + - 1. **SEGUIMIENTO Y CONTROL**

Para el seguimiento y control de las pruebas se llevarán a cabo comités técnicos de seguimiento periódico semanal donde se evalúen los siguientes temas.

* Avance de las pruebas según cronograma
* Estado o resultado de las pruebas ejecutadas
* Seguimiento a las incidencias reportadas según la ejecución de pruebas.
* Se presentará plan de contingencia para aquellas incidencias de mayor impacto que sean de riesgo para el proyecto.
  1. **HERRAMIENTAS PARA PRUEBAS**

En este capítulo se señalaran las diferentes herramientas de software que se utilizarán para la ejecución de las pruebas.

Las herramientas que se utilizarán, dependerán del tipo de prueba que se realizará, es decir que por cada tipo de prueba es posible que se utilice una herramienta diferente.

* + 1. **JUNIT**

|  |  |
| --- | --- |
| **Características** | **Tipo de prueba** |
| JUnit es un conjunto de clases (framework) que permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera. Es decir, en función de algún valor de entrada se evalúa el valor de retorno esperado; si la clase cumple con la especificación, entonces JUnit devolverá que el método de la clase pasó exitosamente la prueba; en caso de que el valor esperado sea diferente al que regresó el método durante la ejecución, JUnit devolverá un fallo en el método correspondiente  JUnit es también un medio de controlar las pruebas de regresión, necesarias cuando una parte del código ha sido modificado y se desea ver que el nuevo código cumple con los requerimientos anteriores y que no se ha alterado su funcionalidad después de la nueva modificación.  En la actualidad las herramientas de desarrollo como NetBeans y Eclipse cuentan con plug-ins que permiten que la generación de las plantillas necesarias para la creación de las pruebas de una clase Java se realice de manera automática, facilitando al programador enfocarse en la prueba y el resultado esperado, y dejando a la herramienta la creación de las clases que permiten coordinar las pruebas.  La versión de la herramienta que se utilizará será: release 4.5 | Esta herramienta será utilizada para la ejecución de:   * Pruebas unitarias. * Pruebas de sistema * Pruebas de Integración. |

* + 1. **HTTPUNIT**

|  |  |
| --- | --- |
| **Características** | **Tipo de prueba** |
| Esta herramienta nos permite implementar unidades de test al estilo JUnit, pero donde los módulos a probar son páginas web (servlets, páginas jsp, php, cgi's, etc.). HttpUnit nos permite simular el acceso a un portal web, hacer "click" en los links, consultar las cookies, ejecutar código Java Script, etc.  HttpUnit permite escribir test que simulen a un usuario accediendo a una aplicación basada en Web mediante un navegador. Tiene muchas de las funciones de un navegador: control de cookies por sesiones, análisis de contenido HTML, envío de formularios mediante los métodos GET y POST, autentificación, y otras características. Se puede chequear si hay cierto contenido en la página, enlace por enlace y formulario por formulario, lo que permite verificar que la aplicación devuelve los resultados apropiados  La versión de la herramienta que se utilizará será: release 1.7 | Esta herramienta será utilizada para la ejecución de:   * Pruebas Funcionales. * Pruebas de Sistema * Pruebas de Configuración |

* + 1. **JMETER**

|  |  |
| --- | --- |
| **Características** | **Tipo de prueba** |
| JMeter es una herramienta de carga para llevar acabo simulaciones sobre cualquier recurso de Software.  Inicialmente diseñada para pruebas de estrés en aplicaciones web, hoy en día, su arquitectura ha evolucionado no sólo para llevar a cabo pruebas en componentes habilitados en Internet (HTTP), sino además en Bases de Datos, programas en Perl, requisiciones FTP y prácticamente cualquier otro medio.  Posee la capacidad de realizar desde una solicitud sencilla hasta secuencias de requisiciones que permiten diagnosticar el comportamiento de una aplicación en condiciones de producción.  En este sentido, simula todas las funcionalidades de un Navegador ("Browser"), o de cualquier otro cliente, siendo capaz de manipular resultados en determinada requisición y reutilizarlos para ser empleados en una nueva secuencia.  La versión de la herramienta que se utilizará será: release 1.9 | Esta herramienta será utilizada para la ejecución de:   * Pruebas de Sistema |

* + 1. **GRINDER**

|  |  |
| --- | --- |
| **Características** | **Tipo de prueba** |
| Es un framework libre en Java, para realizar pruebas de carga o estrés utilizando una consola gráfica.  Esta versión hace uso del lenguaje script jhyton, lo que permite a cualquier código Java ser encapsulado como una prueba.  Esta herramienta permite utilizar un ambiente de red para realizar las pruebas de carga de forma totalmente distribuida.  La versión de la herramienta que se utilizará será: The Grinder 3 | Esta herramienta será utilizada para la ejecución de:   * Pruebas de Sistema |

* + 1. **NESSUS**

|  |  |
| --- | --- |
| **Características** | **Tipo de prueba** |
| Nessus es un programa de escaneo de vulnerabilidades en diversos [sistemas operativos](http://es.wikipedia.org/wiki/Sistemas_operativos). Consiste en nessusd, el [daemon](http://es.wikipedia.org/wiki/Daemon) Nessus, que realiza el escaneo en el sistema objetivo, y nessus, el cliente (basado en consola o gráfico) que muestra el avance y reporte de los escaneos. Desde consola nessus puede ser programado para hacer escaneos programados con [cron](http://es.wikipedia.org/wiki/Cron_(unix)).  En operación normal, nessus comienza escaneando los puertos con nmap o con su propio escaneador de puertos para buscar puertos abiertos y después intentar varios  para atacarlo [exploits](http://es.wikipedia.org/wiki/Exploit). Las pruebas de vulnerabilidad, disponibles como una larga lista de plugins, son escritos en NASL (Nessus Attack Scripting Language, Lenguaje de Scripting de Ataque Nessus por sus siglas en inglés), un lenguaje scripting optimizado para interacciones personalizadas en redes.  Opcionalmente, los resultados del escaneo pueden ser exportados en reportes en varios formatos, como texto plano, [XML](http://es.wikipedia.org/wiki/XML), [HTML](http://es.wikipedia.org/wiki/HTML), y [LaTeX](http://es.wikipedia.org/wiki/LaTeX). Los resultados también pueden ser guardados en una base de conocimiento para referencia en futuros escaneos de vulnerabilidades.  Algunas de las pruebas de vulnerabilidades de Nessus pueden causar que los servicios o sistemas operativos se corrompan y caigan. El usuario puede evitar esto desactivando "unsafe test" (pruebas no seguras) antes de escanear. | Esta herramienta será utilizada para la ejecución de:   * Pruebas de Seguridad |

* + 1. **XMLBuddy**

|  |  |
| --- | --- |
| **Características** | **Tipo de prueba** |
| Este es un PlugIn para Eclipse (IDE que será utilizado para el desarrollo). Esta herramienta, entre otras funcionalidades, permite que el desarrollador valide los archivos XML contra el archivo de esquemas XSD correspondiente. | Esta herramienta será utilizada para la ejecución de:   * Pruebas de Interoperabilidad para los archivos XML que genere la aplicación. |

* 1. **TIPOS DE PRUEBAS**

Las pruebas que se realizarán serán aquellas que fueron señaladas como tipos de pruebas en el numeral 8.3 de la metodología de pruebas; en este capítulo solo serán mencionados a manera general los tipos de pruebas.

El objetivo principal de la ejecución de las pruebas esta dado a:

* Descubrir tantos errores como sea posible.
* Notificar acerca de los riesgos percibidos del proyecto
* Identificar falencias funcionales de la aplicación, enmarcadas en grados de usabilidad ya definidos.
* Evaluar la calidad del producto y señalar un indicador de aceptación del mismo.
* Evaluar la calidad técnica del producto y resolver las falencias identificadas en las pruebas de tipo técnico.
* Cumplir con los requerimientos específicos del cliente, en cuanto a la ejecución de las pruebas.
  + 1. **PRUEBAS UNITARIAS**

Las pruebas unitarias tienen como objetivo verificar la funcionalidad y estructura de cada componente individualmente del sistema una vez que ha sido codificado.

Es una Prueba técnica que permitirá:

* Verificar que los módulos del sistema estén libres de errores.
* Que todos los caminos lógicos principales deben ejecutarse correctamente en cada módulo de la aplicación.
* Todas las transacciones deben ser probados.
* Todos los tipos de registro de entrada válidos deben ser procesados
* Todos los tipos de registro de entrada inválidos deben ser procesados correctamente
* Códigos de vuelta no nulos.
* Excepciones a tratamiento normal.
* Todas las salidas válidas son procesadas.
* Rasgos de Control son probados y documentados.

|  |  |
| --- | --- |
| Objetivo de la Prueba: | * Validar las piezas individuales del software como una unidad independiente. |
| Estrategia: | * Se efectúan para los servicios del negocio y para la lógica de beans en capa Web que tengan complejidad alta. * Generar casos de pruebas necesarios que permitan identificar:   + Que al menos cada sentencia o instrucción del programa se ejecute al menos una vez correctamente.   + Que cada condición tenga por lo menos una vez un resultado positivo y/o negativo.   + Que cada bucle del sistema se pueda probar considerando: - ignorar el bucle, pasar una vez, pasar n veces. |
| Herramienta requeridas: | JUNIT |
| Observaciones | La prueba se realizará por Módulo entendiéndose por tal:   * Bloque básico de programa * Implementa función independiente y simple * Puede probarse por separado. |

* + 1. **PRUEBAS DEL SISTEMA**

Las pruebas de sistema buscan diferencias entre la solución desarrollada y los requerimientos, enfocándose en la identificación de los errores que se puedan generar entre la especificación funcional y el diseño del sistema, así como, el negocio objeto de la aplicación.

|  |  |
| --- | --- |
| Objetivo de la Prueba: | * Validar aquellos volúmenes de datos máximos (por lo general las transacciones o informes) que pueden ser completados dentro de un período específico en el tiempo, y con un nivel de concurrencia dado (carga, concurrencia y desempeño). * Validar los requerimientos no funcionales de cada proyecto. |
| Estrategia: | * Realizar Set de Pruebas a partir de los Requerimientos no funcionales. * Realizar pruebas de rendimiento básico. Consiste en probar la aplicación simulando la carga esperada en el entorno de producción. * Realizar las pruebas de concurrencia: verificar el comportamiento de la aplicación en condiciones de sobrecarga de usuarios, que supone permitirá identificar potenciales problemas de rendimiento o cuellos de botella, antes de su pase a producción. * Realizar pruebas de requerimientos no funcionales: Consiste en probar la aplicación con cada uno de los requerimientos no funcionales establecidos en el proyecto. * Identificar posibles cuellos de botella o problemas de rendimiento. * Realizar pruebas de carga: Altos volúmenes de información. |
| Herramienta requeridas: | * JUNIT * HTTPUNIT * JMETER o THE GRINDER 3 |
| Observaciones: | La elección de la herramienta de prueba será a discreción del grupo de pruebas y su elección dependerá de la prueba que se va a realizar, es decir puede que para pruebas de carga y altos volúmenes se utilice THE GRINDER 3 pero para validación de funcionalidad se utilice HTTPUNIT. |

* + 1. **PRUEBAS DE INTEGRACIÓN**

El objetivo de las pruebas de integración es verificar el correcto ensamblaje entre los distintos módulos que componen la solución una vez que han sido probados unitariamente con el fin de comprobar que interactúan correctamente a través de sus interfaces internas y externas, que cubren la funcionalidad establecida y se ajustan a los requisitos no funcionales especificados en las verificaciones correspondientes.

En esta prueba se comprueba la compatibilidad y funcionalidad de los interfaces entre las distintas ‘partes’ que componen el desarrollo de la solución. Estas partes pueden ser módulos, aplicaciones individuales, es decir esta prueba válida la integración entre los diferentes módulos que componen la solución con el fin de garantizar que su operación integrada es correcta, teniendo en cuenta los siguientes temas técnicos:

* El funcionamiento integrado de módulos interdependientes debe estar libre de errores
* Probar todas las dependencias entre módulos
* Probar el flujo de control y el flujo de datos a través de todas las capas

|  |  |
| --- | --- |
| Objetivo de la Prueba: | Validar la integración entre los diferentes módulos que componen la solución con el fin de garantizar que su operación integrada es correcta |
| Estrategia: | Pruebas de Integración Incremental Ascendente   * Combinación de módulos de bajo nivel en grupos que realicen una misma función o subfunción especifica, con el fin de reducir el número de pasos de integración. * Se escribe para cada módulo un módulo impulsor o conductor, con el fin de simular la llamada a los módulos, introducir datos de pruebas y recoger resultados. * Se prueba cada módulo mediante su impulsor. * Se eliminan los módulos impulsores y se sustituyen por los módulos de nivel superior en la jerarquía. |
| Herramienta requeridas: | JUNIT |
| Observaciones: |  |

* + 1. **PRUEBAS DE INTEROPERABILIDAD**

En esta prueba se valida que el sistema se comunique de manera exitosa con los sistemas externos con que se requiera, de acuerdo a los requerimientos no funcionales. Estos sistemas pueden ser sistemas propios de cada entidad, servicios de estampado de tiempo, servicios para la validación de certificados digitales, servicios publicados en el tramitador transaccional, servicios de envío de SMS, servicios de envío de correo electrónico.

|  |  |
| --- | --- |
| Objetivo de la Prueba: | * Validar la interoperabilidad de la solución con sistemas externos. |
| Estrategia: | * Prueba directa con los servicios que se encuentren disponibles en un ambiente de pruebas controlado suministrado por Agenda de conectividad. * Para los servicios que no estén disponibles para prueba se desarrollarán sistemas DUMMIE que garanticen que la integración funcionará. * Para los archivos XML generados por la aplicación se verificará que sean válidos según la definición de los esquemas xsd definidos para cada uno.   La verificación de los archivos GEL- XML la realizará cada desarrollador, aprovechando las características de la herramienta utilizada para estas pruebas.   * La autenticación de los sistemas externos se probará mediante un DUMMIE; el cual llevará el usuario y contraseña de autenticación frente al sistema de notificaciones. Se probarán 3 escenarios uno con el usuario incorrecto, otro con la clave incorrecta y finalmente un escenario con el usuario y clave correctos. * Las pruebas de conexión con el tramitador de gobierno en línea se verificará mediante una prueba técnica utilizando y siguiendo los pasos del Wizard de conexión java con el tramitador de gobierno en línea; de esta prueba se espera que la conexión quede establecida. * Se expondrá un servidor de sFTP externo, con el cual se realizarán transferencias masivas de documentos históricos, se probaran tres escenarios: (Correcto, Incorrecto e incompleto). |
| Herramienta requeridas: | DUMMIES de los servicios que no están construidos o no están disponibles:   * Autenticación * Firma Digital * Estampado de Tiempo * Sistema Emisor   XSD de cada archivo XML generado. XMLBuddy |
| Observaciones: |  |

* + 1. **PRUEBAS DE REGRESIÓN**

En esta prueba se valida que el sistema mantenga su correcta funcionalidad después de la incorporación de un ajuste, corrección o nuevo requerimiento. Es una prueba funcional y técnica que valida que el sistema siga funcionando perfectamente después de que las correcciones sean aplicadas.

|  |  |
| --- | --- |
| Objetivo de la Prueba: | Validar que el sistema siga funcionando perfectamente después de que las acciones correctivas sean aplicadas. |
| Estrategia: | * Repetir las pruebas (unitarias, de integración, funcionales y de carga) que se hicieron antes de corregir defectos o de añadir nuevas funcionalidades, para comprobar que las modificaciones no provocan errores donde antes no los había. |
| Herramienta requeridas: | * Utilizar las mismas herramientas usadas para las pruebas según sea el caso. |
| Observaciones | Los responsables de las Pruebas de Regresión se establecen dependiendo del momento en el que se realicen las modificaciones. |

* + 1. **PRUEBAS FUNCIONALES**

La prueba funcional es un proceso para procurar encontrar discrepancias entre el software desarrollado y la especificación funcional. La prueba funcional normalmente es una actividad de caja negra. Esta prueba permite validar:

* Los procesos y reglas de negocio establecidas,
* Que se cumplan los requerimientos funcionales establecidos

En esta prueba se validan los Casos de Uso que fueron aprobados por el cliente, y a partir de ellos se diseñan y ejecutan los set de pruebas correspondientes. Se deben elaborar los casos de pruebas necesarios que permitan asegurar el funcionamiento de todos los flujos normales y alternos de dichos casos de uso.

|  |  |
| --- | --- |
| Objetivo de la Prueba: | Se asegura el trabajo apropiado de los requisitos funcionales, Incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. |
| Estrategia : | * Validación y ejecución de Set de Pruebas y escenarios definidos, teniendo en cuenta flujo normal y flujos alternativos, usando datos validos e inválidos para verificar lo siguiente:   + Los resultados esperados ocurren cuando se usan datos validos.   + Se despliegan mensajes de error cuando se usan datos inválidos.   + Cada regla de negocio es propiamente aplicada.   + Realizar set de pruebas de los requerimientos mínimos para el adecuado funcionamiento de la aplicación |
| Herramientas Requeridas: | * HTTPUNIT cuando aplique * Formato de casos de prueba funcionales |
| Observaciones: | Para el reporte de incidencias se utilizará una herramienta para el registro y seguimiento. |

* + 1. **PRUEBAS DE USABILIDAD**

Las pruebas de usabilidad son una forma de medir que tan bien puede una persona usar un objeto hecho por el hombre, como puede ser una [página web](http://es.wikipedia.org/wiki/P%C3%A1gina_web), una [interfaz de usuario](http://es.wikipedia.org/wiki/Interfaz_de_usuario), un documento o un dispositivo.

Las pruebas de [usabilidad](http://es.wikipedia.org/wiki/Usabilidad) consisten en seleccionar a un grupo de [usuarios](http://es.wikipedia.org/wiki/Usuario) de una [aplicación](http://es.wikipedia.org/wiki/Aplicaci%C3%B3n_inform%C3%A1tica) y solicitarles que lleven a cabo las tareas para las cuales fue diseñada, en tanto el equipo de diseño, desarrollo y otros involucrados toman nota de la interacción, particularmente de los errores y dificultades con las que se encuentren los usuarios.

No es necesario que se trate de una aplicación completamente terminada, pudiendo tratarse de un [prototipo](http://es.wikipedia.org/wiki/Prototipo)

|  |  |
| --- | --- |
| Objetivo de la Prueba: | * Validar el grado de usabilidad empírico del sistema. * El grado de usabilidad se medirá en tres aspectos clave: * Facilidad de aprendizaje: facilidad con la que nuevos usuarios desarrollan una interacción efectiva con el sistema o producto. * Flexibilidad: relativa a la variedad de posibilidades con las que el usuario y el sistema pueden intercambiar información. * Robustez: es el nivel de apoyo al usuario que facilita el cumplimiento de sus objetivos. |
| Estrategia : | * Se usarán cuatro métricas principales para medir la usabilidad del sistema * [Exactitud](http://es.wikipedia.org/wiki/Exactitud): Número de errores cometidos por los sujetos de prueba y si estos fueron recuperables o no al usar los datos o procedimientos adecuados. * [Tiempo](http://es.wikipedia.org/wiki/Tiempo) requerido para concluir la actividad. * [Recuerdo](http://es.wikipedia.org/wiki/Recuerdo): Qué tanto recuerda el usuario después de un periodo sin usar la aplicación. * Respuesta [emocional](http://es.wikipedia.org/wiki/Emoci%C3%B3n): Cómo se siente el usuario al terminar la tarea (bajo tensión, satisfecho, molesto, etcétera). * Estas métricas será implementadas para cada uno de los aspectos clave señalados en el objetivo de la prueba. * La forma de evaluación será mediante el uso de encuestas; donde cada pregunta evaluará un aspecto clave de usabilidad y aportará valor a una o varias métricas dentro del aspecto clave evaluado. * Las encuestas se realizarán a los usuarios utilizando los prototipos del sistema; para así poder realizar cambios de forma temprana al diseño de la capa de presentación. |
| Herramientas Requeridas: | * Encuesta * Prototipos del sistema. |
| Observaciones: |  |

* + 1. **PRUEBAS DE SEGURIDAD**

Estas pruebas tienen dos enfoques:

* Pruebas de seguridad de la aplicación; donde se verifica que un actor solo pueda acceder a las funciones y datos que su usuario tiene permitido.
* Pruebas de seguridad del sistema; donde se verificar que solo los actores con acceso al sistema y a la aplicación están habilitados para accederla.

|  |  |
| --- | --- |
| Objetivo de la Prueba: | * Que los usuarios están restringidos a funciones específicas o su acceso está limitado únicamente a los datos que está autorizado a acceder. * Que solo aquellos usuarios autorizados a acceder al sistema son capaces de ejecutar las funciones del sistema. * Que el cortafuego oculte apropiadamente la aplicación. * Que los puertos clave solo puedan ser accedidos de la forma establecida para su acceso. * Que los puertos restringidos efectivamente no se encuentren accesibles. |
| Estrategia : | * Identificar cada tipo de usuario y las funciones y datos a los que se debe autorizar. * Crear pruebas para cada tipo de usuario y verificar cada permiso, creando transacciones específicas para cada tipo de usuario. * Modificar tipos de usuarios y volver a ejecutar las pruebas. |
| Herramientas Requeridas: | * Nessus * Pruebas funcionales de seguridad. |
| Observaciones: | Para el reporte de incidencias se utilizará una herramienta para el registro y seguimiento. |

* + 1. **PRUEBAS DE CONFIGURACIÓN**

El propósito de esta prueba es establecer y mantener la integridad de los productos de software a través del ciclo de vida del proceso del mismo. Esta prueba implica la identificación de la Configuración del software en puntos dados en el tiempo, el control sistemático de los cambios en la Configuración y el mantenimiento de la integridad y trazabilidad de la Configuración a través del ciclo de vida del software.

|  |  |
| --- | --- |
| Objetivo de la Prueba: | Validar la integridad de los productos de software. |
| Estrategia : | * Validación y ejecución de Set de Pruebas que representen un ciclo del proceso de negocio principal de principio a fin. * Validación de la integridad de la configuración de todos los sistemas involucrados en puntos datos en el tiempo. * Realizar trazabilidad de los cambios de configuración realizados para puesta a punto. |
| Herramientas Requeridas: | * HTTPUNIT |
| Observaciones: | Para el reporte de incidencias se utilizará una herramienta para el registro y seguimiento. |

* + 1. **PRUEBAS DE RECUPERACIÓN A FALLAS**

Estas pruebas aseguran que el software pueda recuperarse a fallas de hardware, software o mal funcionamiento de la red sin pérdida de datos o de integridad de los datos.

El objetivo de esta prueba es verificar que los procesos de recuperación (manual o automáticos) se realice apropiadamente: las base de datos, las aplicaciones y el sistema a un estado conocido y deseado.

En la prueba se incluyen los siguientes tipos de condiciones:

* Interrupción de energía al cliente
* Interrupción de energía al servidor
* Interrupción de comunicaciones mediante los servidores de la red
* Interrupción de comunicación o pérdida de energía de los discos del servidor o con los controladores
* Ciclos incompletos (procesos de filtro de datos interrumpidos, procesos de sincronización de datos interrumpidos)

|  |  |
| --- | --- |
| Objetivo de la Prueba: | * Validar la capacidad de recuperación a fallas de:   + Hardware   + Software   + Mal funcionamiento de Red. |
| Estrategia : | * Interrumpir la energía del cliente: apagar el PC. * Interrumpir la energía del servidor: simular o iniciar el proceso de apagado del servidor. * Interrupción por medio de los servidores de red: simular o iniciar la pérdida de comunicación con la red * (desconectar físicamente la comunicación o apagar el servidor de red o router * Interrumpir la comunicación o quitar la energía de los discos del servidor o sus controladores: simular o * eliminar físicamente la comunicación con uno o más controladores de disco o los discos.] * Una vez que se lograron o simularon estas condiciones, se deben invocar los procedimientos de * recuperación. |
| Herramientas Requeridas: | * Ninguna |
| Observaciones: | Para el reporte de incidencias se utilizará una herramienta para el registro y seguimiento. |

* + 1. **PRUEBAS DE ACEPTACIÓN**

El objetivo de las pruebas de aceptación es validar que la solución desarrollada cumpla con el funcionamiento esperado y permitir al usuario de dicho sistema determine su aceptación, desde el punto de vista de su funcionalidad y de su rendimiento. Estas pruebas son realizadas por el cliente, donde comprueba que el sistema cumple con lo definido y se obtiene la conformidad del cliente. Esta prueba se realiza mediante el proceso de validación de caja negra.

Estas pruebas corresponden a la ejecución de las siguientes pruebas por parte de los usuarios funcionales o cliente:

* Pruebas Funcionales.
* Pruebas de Usabilidad.
* Pruebas de Configuración
  1. **ENTREGABLES DE PRUEBAS**

De acuerdo al tipo de pruebas ejecutadas puede que el entregable del mismo sea diferente, en el siguiente cuadro se señalan los diferentes entregables por tipo de prueba.

|  |  |
| --- | --- |
| **TIPO DE PRUEBAS** | **ENTREGABLES** |
| Pruebas Unitarias | * Resumen de validación de la prueba. |
| Pruebas de Sistema | * Se entregará un documento de resultados de las pruebas realizadas, que incluya resultados de la ejecución de los scripts de pruebas y análisis de los errores o defectos encontrados durante el proceso de realización de las pruebas. |
| Pruebas de Integración | * Se entregará un documento de pruebas de integración que incluye resultados de la ejecución de los scripts de pruebas y análisis de los defectos encontrados durante el proceso de pruebas |
| Pruebas de Interoperabilidad | * Se entregará un documento de pruebas de interoperabilidad que incluye resultados de la comunicación con los servicios y DUMMIES previstos. |
| Pruebas de Regresión | * Se entregará un documento de pruebas de regresión, que incluye resultados de la ejecución de los scripts de pruebas, análisis de los defectos encontrados durante el proceso de pruebas y solicitud de las correcciones recibidas. |
| Pruebas Funcionales | * Se entregará un documento de pruebas de regresión, que incluye resultados de la ejecución de los scripts de pruebas y análisis de los defectos encontrados durante el proceso de pruebas y solicitud de las correcciones recibidas. |
| Pruebas de Usabilidad | * Indicadores de Usabilidad |
| Pruebas de Seguridad | * Informe de vulnerabilidades * Resultado de pruebas funcionales de seguridad. |
| Pruebas de Configuración | * Resumen de validación de la prueba. |
| Pruebas de Recuperación a Fallas | * Resumen de validación de la prueba. |
| Pruebas de Aceptación | * Resumen de validación de la prueba. |

Los entregables de las pruebas serán elaborados de acuerdo a la estructura del entregable “Informe de Pruebas” solicitados en los términos de referencia para la fase de desarrollo y pruebas.

* 1. **MATRIZ DE TIPIFICACIÓN DE PRUEBAS**

|  |  |
| --- | --- |
| **TIPO DE PRUEBAS** | **TIPO DE PRUEBA** |
| Pruebas Unitarias | Automáticas |
| Pruebas de Sistema | Automáticas |
| Pruebas de Integración | Automáticas |
| Pruebas de Interoperabilidad | Automáticas |
| Pruebas de Regresión | Automáticas y Manuales |
| Pruebas Funcionales | Manuales |
| Pruebas de Usabilidad | Manuales |
| Pruebas de Seguridad | Automáticas y Manuales |
| Pruebas de Configuración | Automáticas y Manuales |
| Pruebas de Recuperación a Fallas | Automáticas y Manuales |
| Pruebas de Aceptación | Manuales |

* 1. **TECNICAS DE EJECUCIÓN DE PRUEBAS**

| **TIPO DE PRUEBAS** | **TECNICA DE EJECUCIÓN** | **HERRAMIENTAS A UTILIZAR** |
| --- | --- | --- |
| Pruebas Unitarias | El uso de JUnit normalmente involucra los siguientes pasos:   1. Crear una subclase de junit.framework.TestCase. 2. Opcionalmente sobrescribir el método setUp() que será invocado en la inicialización de objetos y variables usados por todos los casos de prueba. No todos los casos de uso necesitan esto. Note que setUp() es invocado antes de cada caso particular. 3. Opcionalmente sobrescribir el método tearDown(). Método que será invocado al final de cada caso de prueba y que nos sirve para liberar recursos usados en la prueba o incluso para volver atrás lo probado, por ejemplo cuando el caso de prueba involucre la actualización de datos en una base de datos relacional. 4. Adicionar métodos de prueba a la clase. Note que no necesitamos implementar ninguna interface, pues JUnit usa el paquete reflection del java para detectar automáticamente métodos test. Dichos métodos son reconocidos por su asignatura, la cual debe tener la forma public void test<Descripcion>(), además pueden lanzar cualquier exception. | **JUNIT** |
| Pruebas de Sistema | Las pruebas del sistema tal como están concebidas para el proyecto de notificaciones electrónicas involucra los siguientes pasos:   1. Selección de los casos de uso para los que se probará Carga, volumen, estrés, concurrencia. Estos casos de uso corresponderán al 10% del total de casos de uso y su elección contará con la aprobación del Programa Agenda de Conectividad e Interventoría. 2. Ejecución de las pruebas de Carga, volumen, estrés, concurrencia mediante la herramienta seleccionada. 3. Recopilación de resultados. 4. Reporte de los defectos encontrados según las pruebas. (A través de la herramienta de gestión de incidencias) 5. Asignación de la incidencia. (A través de la herramienta de gestión de incidencias). 6. Corrección de la incidencia. 7. Repetición de la prueba. | **JMETER** |
| Pruebas de Integración | Las pruebas del integración tal como están concebidas para el proyecto de notificaciones electrónicas involucra los siguientes pasos:   1. Selección de los componentes y/o servicios para los que se probará integración con los componentes y/o servicios que tienen relación directa. Estos componentes y/o servicios corresponderán al 10% del total de componentes y/o servicios y su elección contará con la aprobación del Programa Agenda de Conectividad e Interventoría. 2. Ejecución de las pruebas de integración mediante la herramienta seleccionada. 3. Recopilación de resultados. 4. Reporte de los defectos encontrados según las pruebas. (A través de la herramienta de gestión de incidencias) 5. Asignación de la incidencia. (A través de la herramienta de gestión de incidencias). 6. Corrección de la incidencia. 7. Repetición de la prueba. | **GRINDER** |
| Pruebas de Interoperabilidad | Las pruebas de interoperabilidad tal como están concebidas para el proyecto de notificaciones electrónicas involucra los siguientes pasos:   1. Definición de la lista de servicios a probar. 2. Definición de DUMMIES a construir. 3. Construcción de DUMMIES. 4. Ejecución de la prueba de interoperabilidad contra los servicios y DUMMIES definidos. 5. Recopilación de resultados. 6. Reporte de los defectos encontrados según las pruebas. (A través de la herramienta de gestión de incidencias) 7. Asignación de la incidencia. (A través de la herramienta de gestión de incidencias) 8. Corrección de la incidencia. 9. Repetición de la prueba. | * **Servicios del Enrutador transaccional** * **DUMMIES** * **XMLBuddy** |
| Pruebas de Regresión | Las pruebas de regresión tal como están concebidas para el proyecto de notificaciones electrónicas involucra los siguientes pasos:   * 1. Repetición de las pruebas funcionales y de sistema que estén involucradas en los cambios realizados al sistema.   2. Recopilación de resultados.   3. Reporte de los defectos encontrados según las pruebas. (A través de la herramienta de gestión de incidencias)   4. Asignación de la incidencia. (A través de la herramienta de gestión de incidencias)   5. Corrección de la incidencia.   6. Repetición de la prueba. | * **Casos de Prueba** * **Grinder** |
| Pruebas Funcionales | Las pruebas funcionales normalmente involucra los siguientes pasos:   1. Crear los casos de prueba mediante el formato establecido para ellos. 2. Ejecución de los casos de prueba con forme las funcionalidades van siendo liberadas para pruebas. 3. Reporte de los defectos encontrados según las pruebas. (A través de la herramienta de gestión de incidencias) 4. Asignación de la incidencia. (A través de la herramienta de gestión de incidencias) 5. Corrección de la incidencia. 6. Repetición de la prueba. | * **Casos de Prueba** |
| Pruebas de Usabilidad | Las pruebas de usabilidad tal como están concebidas para el proyecto de notificaciones electrónicas (uso de prototipado) involucra los siguientes pasos:   1. Elaboración de prototipos. 2. Elaboración de encuestas de usabilidad. 3. Evaluación de prototipos mediante la aplicación de encuestas de usabilidad a usuarios comunes. 4. Recopilación de datos de la encuesta. 5. Compilación y análisis de resultados de encuestas de usabilidad. | * **Prototipos** * **Encuesta de usabilidad** |
| Pruebas de Seguridad | Las pruebas de seguridad tal como están concebidas para el proyecto de notificaciones electrónicas involucra los siguientes pasos:   1. Elaboración de casos de prueba funcionales de seguridad. 2. Elaboración de casos de prueba automáticos de seguridad. 3. Configuración de ambiente de pruebas. 4. Ejecución de pruebas funcionales de seguridad. 5. Ejecución de casos automáticos de pruebas de seguridad con el uso de las herramientas de evaluación y ataque. 6. Recopilación de resultados. 7. Reporte de los defectos encontrados según las pruebas. (A través de la herramienta de gestión de incidencias) 8. Asignación de la incidencia. (A través de la herramienta de gestión de incidencias) 9. Corrección de la incidencia. 10. Repetición de la prueba. | * **NESSUS** * **Casos de prueba funcionales de seguridad.** |
| Pruebas de Configuración | Dada la metodología RUP, al ser un desarrollo iterativo e incremental se efectuaran pruebas de instalación a medida que sean liberadas las diferentes versiones del aplicativo.  Las pruebas incluirían:   1. Elaboración de listas de chequeo de instalación de sistema operativo, base de datos, servidor de aplicaciones y cualquier otro componente del sistema. 2. Ejecución de la instalación según la lista de chequeo. 3. Recopilación de resultados. 4. Reporte de los defectos encontrados según las pruebas. (A través de la herramienta de gestión de incidencias) 5. Asignación de la incidencia. (A través de la herramienta de gestión de incidencias) 6. Corrección de la incidencia. 7. Repetición de la prueba. | * **Listas de chequeo para instalación.** |
| Pruebas de Recuperación a Fallas | Las pruebas de recuperación a fallas normalmente involucra los siguientes pasos:   1. Crear los casos de prueba de recuperación. 2. Ejecución de los casos de prueba. 3. Reporte de los defectos encontrados según las pruebas. (A través de la herramienta de gestión de incidencias) 4. Asignación de la incidencia. (A través de la herramienta de gestión de incidencias) 5. Corrección de la incidencia. 6. Repetición de la prueba. |  |
| Pruebas de Aceptación | Las pruebas de recuperación a fallas normalmente involucra los siguientes pasos:   1. Ejecución de una muestra de las pruebas funcionales, de carga, de configuración por parte del Programa Agenda de Conectividad e Interventoría. 2. Reporte de los defectos encontrados según las pruebas. (A través de la herramienta de gestión de incidencias) 3. Asignación de la incidencia. (A través de la herramienta de gestión de incidencias) 4. Corrección de la incidencia. 5. Repetición de la prueba. | * **Grinder.** * **Casos de prueba Funcionales.** * **Listas de Chequeo.** |

1. RECURSOS DEL PLAN DE PRUEBAS
   1. **RECURSO HUMANO**

El recurso humano que debe estar disponible para la ejecución de las pruebas varía de acuerdo al tipo de prueba. En el siguiente cuadro se especifica el tipo de perfil necesario por tipo de prueba.

Los perfiles mencionados no necesariamente corresponden a los enunciados en la metodología de pruebas, ya que allí se mencionan perfiles de apoyo al proceso de pruebas y aquí solo se mencionarán los perfiles que van a ejecutar las pruebas o que intervienen directamente en la prueba.

|  |  |
| --- | --- |
| **TIPO DE PRUEBAS** | **PERFIL DEL RECURSO HUMANO** |
| Pruebas Unitarias | * Ingeniero Desarrollador. * Analista de Pruebas. |
| Pruebas de Sistema | * Ingeniero Desarrollador. * Analista de Pruebas. |
| Pruebas de Integración | * Ingeniero Desarrollador. * Analista de Pruebas |
| Pruebas de Interoperabilidad | * Ingeniero Desarrollador. * Analista de Pruebas |
| Pruebas de Regresión | * Ingeniero Desarrollador. * Analista de Pruebas |
| Pruebas Funcionales | * Analista de Pruebas |
| Pruebas de Usabilidad | * Analista de Pruebas * Usuario Funcional. |
| Pruebas de Seguridad | * Ingeniero Desarrollador. * Analista de Pruebas * Arquitecto de Desarrollo. |
| Pruebas de Configuración | * Arquitecto de Desarrollo. |
| Pruebas de Recuperación a Fallas | * Ingeniero Desarrollador. * Analista de Pruebas |
| Pruebas de Aceptación | * Analista de Pruebas. * Usuario Funcional. |

* 1. **RECURSO DEL SISTEMA**

Las pruebas se realizaran en un ambiente controlado y administrado por la UT; a continuación se describen las características de la infraestructura del ambiente de pruebas.

|  |  |  |
| --- | --- | --- |
| **DESCRIPCION** | **FUNCIONALIDAD** | **CANTIDAD** |
| Servidor | Montar ambiente de Pruebas con la solución en proceso de desarrollo | 4 |
| Estaciones de Trabajo | Con acceso al Servidor de Pruebas a través de la red LAN de la UT. | 4 |
| Software: Instalado y configurado | Herramientas Bugzilla, con acceso al equipo técnico del proyecto y del frente de pruebas | 1 |
| Herramientas de pruebas de sistemas | JUNIT, HTTPUNIT, JMETER o THE GRINDER 3. | 1 |
| Share Point | Repositorio central: Acceso a todo el equipo técnico y frente de pruebas. | 1 |

* + 1. **CONFIGURACION DEL AMBIENTE DE PRUEBAS**

El ambiente de pruebas estará implementado en las instalaciones de la UT y su configuración deberá ser la siguiente.

|  |  |  |  |
| --- | --- | --- | --- |
| **COMPONENTE** | **CONFIGURACIÓN** | **SOFTWARE INSTALADO Y CONFIGURADO** | **CANTIDAD** |
| Servidor 1 | **Procesador:** Quad Core Xeon E5405 Porcessor2x6MB Cache, 2,0GHz, 1333MHz FSB, PE2950  **Memoria:** 4GB 667MHz (4x1GB) Daul Ranked Fully Buffered DIMMs  **Tarjeta de Video:** LOM NICs are TOE Ready  **Disco Duro:** 2 dd c/u 300GB, 10K RPM Serial-Attach SCSI 3Gbps 3,5 in HotPlug HardDrive  Regulador de disco duro: PREC6i SAS RAID Controller, 2x4 Connectors, Int, PCIe, 246MB cache, x6 Bkpl  **Red:** 2 Integradas NetXtreme II 5708 Gigabit NICs TOE Capable | * Red Hat Enterprise Linux 4,5ES1 64 * Jboss. | 1 |
| Servidor 2 | **Procesador:** Quad Core Xeon E5405 Porcessor2x6MB Cache, 2,0GHz, 1333MHz FSB, PE2950  **Memoria:** 4GB 667MHz (4x1GB) Daul Ranked Fully Buffered DIMMs  **Tarjeta de Video:** LOM NICs are TOE Ready  **Disco Duro:** 2 dd c/u 300GB, 10K RPM Serial-Attach SCSI 3Gbps 3,5 in HotPlug HardDrive  Regulador de disco duro: PREC6i SAS RAID Controller, 2x4 Connectors, Int, PCIe, 246MB cache, x6 Bkpl  **Red:** 2 Integradas NetXtreme II 5708 Gigabit NICs TOE Capable | * Suse 6 de 64 * Oracle 10DB | 1 |
| Servidor 3 | **Procesador:** Quad Core Xeon E53102x4MB Cache, 1,0GHz, 1066MHz FSB,  **Memoria:** 2GB 667MHz (2x1GB) Daul Ranked Fully Buffered DIMMs  **Tarjeta de Video:** LOM NICs are TOE Ready  **Disco Duro:** 146GB, 10K RPM Serial-Attach SCSI 3Gbps 3,5 in HotPlug HardDrive  **Red:** 2 Integradas NetXtreme II 5708 Gigabit NICs TOE Capable | * Red Hat Enterprise Linux 4,5ES1 64 * Alero * Apache | 1 |
| Servidor 4 | **Procesador:** Quad Core Xeon E53102x4MB Cache, 1,0GHz, 1066MHz FSB,  **Memoria:** 2GB 667MHz (2x1GB) Daul Ranked Fully Buffered DIMMs  **Tarjeta de Video:** LOM NICs are TOE Ready  **Disco Duro:** 146GB, 10K RPM Serial-Attach SCSI 3Gbps 3,5 in HotPlug HardDrive  **Red:** 2 Integradas NetXtreme II 5708 Gigabit NICs TOE Capable | Red Hat Enterprise Linux 4,5ES1 64  bugzilla, | 1 |
| Estaciones de Trabajo | Equipos con mínimo:  Procesador Intel Pentium V  Memoria RAM: 512 MB  RED: Acceso a la red local (Para los equipos dentro del área de trabajo de la UT), Acceso a Internet (Para los equipos fuera del área de trabajo de la UT). | Sistema Operativo: Windows XP  Internet Explorer 6.0, Mozilla 2.0, Office 2003. | 4 |
| FireWall | Publicación de los servidores de pruebas a través de una IP fija.  Pentium IV 3,5GH, 3 GB de RAM, 80GB DD, DVD RW, Fuente redundante | Microsoft Windows 2003 spk 2  Microsoft ISA Server 2004 Spk2 | 1 |

* 1. **HERRAMIENTAS DE REPORTES Y CONTROL DE INCIDENCIAS**

La herramienta que se utilizará para la realización del reporte, seguimiento y control de errores o Bug (Bug Tracking System) es **BUGZILLA**.

**Bugzilla** permite organizar en múltiples formas los defectos de software, permitiendo el seguimiento de múltiples productos con diferentes versiones, a su vez compuestos de múltiples componentes. Permite además categorizar los defectos de software de acuerdo a su prioridad y severidad, así como asignarles versiones para su solución.

Esta herramienta será el apoyo para la metodología de seguimiento y control de incidencias, acordada en el punto 9 del plan de proyecto.

* 1. **ADMINISTRACIÓN DE VERSIONES**

La administración de versiones que se probarán será el mecanismo ideal, para controlar los release de pruebas y los cambios que estos sufrirán en la etapa de corrección de incidencias reportadas.

De acuerdo a lo anterior la administración de versiones contempla las siguientes etapas:

1. **Entrega de la Versión para Pruebas** (Release)
   * La versión debe venir con el Release Note.
   * Lista de Chequeo (si aplica)
2. **Creación de Incidencias en la herramienta**.
   * Se debe especificar a que Release pertenecen las incidencias.
3. **Análisis y Desarrollo de Incidencias**.
   * Se realiza la clasificación de las incidencias.
   * Se empieza el desarrollo de las correcciones.
   * Se integran los desarrollos en el release correspondiente
4. **Pruebas de Correcciones de Incidencias** según el release.
5. **Aprobación del Release**.

Cabe anotar que los puntos 2,3 y 4 hacen parte de nuestra METODOLOGÍA DE SEGUIMIENTO Y CONTROL DE INCIDENCIAS, para mayor referencia, consultar el documento Plan de Proyecto Numeral 9.

* + 1. **HERRAMIENTAS**

**SUBVERSION**

SUBVERSION es un software de sistema de control de versiones, Una característica importante de Subversion es que, a diferencia de CVS, los archivos versionados no tienen cada uno un número de revisión independiente. En cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo

**Ventajas**

* Se sigue la historia de los archivos y directorios a través de copias y renombrados.
* Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
* La creación de ramas y etiquetas es una operación más eficiente.
* Se envían sólo las diferencias en ambas direcciones
* Puede ser publicado mediante Apache, sobre WebDAV/DeltaV. Esto permite que clientes WebDAV utilicen Subversion en forma transparente.
* Maneja eficientemente archivos binarios.
* Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.
* Cuando se usa integrado a Apache permite utilizar todas las opciones que este servidor provee a la hora de autentificar archivos (SQL, LDAP, PAM, etc.).

**MAVEN**

MAVEN una herramienta software para la gestión de proyectos Java, La versión 2 usa un fichero de configuración en XML llamado pom.xml. Su funcionalidad es parecida a Apache Ant de manera que permite compilar, ejecutar test o realizar distribuciones pero con la diferencia que trata de forma automática las dependencias del proyecto.

Una de las más importantes características es su actualización en línea mediante servidores repositorios. Maven es capaz de descargar nuevas actualizaciones de las bibliotecas de las que depende el proyecto y de igual manera subir una nueva distribución a un repositorio de versiones, dejándola al acceso de todos los usuarios.

1. EVALUACIÓN DE PRUEBAS EJECUTADAS

Este capítulo mostrará los criterios de ejecución, evaluación, terminación y suspensión de las pruebas.

* 1. **CRITERIOS DE INICIO DE EJECUCIÓN**

A continuación se señalan las condiciones mínimas que se deben presentar para iniciar la ejecución de las pruebas:

* Se poseen los set de pruebas aprobados con escenarios claros.
* El entorno de pruebas es el adecuado para el tipo de pruebas a iniciar.
* Todos los artefactos requeridos se encuentran disponibles.
* Se recibió la Versión del Software para pruebas con su correspondiente Release Note y Lista de Chequeo cuando esta aplique.
* Todos los recursos humanos y técnicos necesarios se encuentran disponibles.
  1. **CRITERIOS DE EVALUACION**

Los criterios de evaluación estarán dados de forma independiente para cada tipo de pruebas; el siguiente cuadro muestra los criterios de evaluación generales de las pruebas ejecutadas.

|  |  |
| --- | --- |
| **TIPO DE PRUEBAS** | **CRITERIOS DE EVALUACION** |
| Pruebas Unitarias | * Detectar errores en la ejecución de las pruebas. * El 90% de las pruebas realizadas deben ser exitosas. |
| Pruebas de Sistema | * Detectar errores en la ejecución de las pruebas * Que los reportes generados por las herramientas de automatización de las pruebas contengan las mínimas variables que permitan un análisis acertado de cada una de las pruebas realizadas. * Tener en cuenta todos los escenarios posibles. * El 90% de las pruebas realizadas deben ser exitosas. |
| Pruebas de Integración | * La totalidad de los puntos de control probadas debe ser mayor al 75% del total de los componentes que integran la solución. * Detectar errores en la ejecución de las pruebas * El 90% de las pruebas realizadas deben ser exitosas. |
| Pruebas de Interoperabilidad | * Detectar errores en la ejecución de las pruebas * El 90% de las pruebas realizadas deben ser exitosas contra los servicios del tramitador. * El 90% de las pruebas realizadas deben ser exitosas contra los servicios DUMMIES. |
| Pruebas de Regresión | * Para realizar esta prueba se debe tomar como base los criterios de aceptación de las pruebas que se volverán a realizar. |
| Pruebas Funcionales | * El resultado de cada caso de prueba debe ser igual al resultado de salida esperado. * Encontrar fallas al ejecutar los diferentes casos de pruebas. * La aplicación cumple con los requerimientos funcionales especificados en la fase de análisis * La aplicación cumple con los requerimientos mínimos para el funcionamiento |
| Pruebas de Usabilidad | * El resultado de cada caso de prueba debe ser igual al resultado de salida esperado. * Se deben incluir los datos de entrada válidos y esperados como no validos e inesperados Encontrar los errores al ejecutar los diferentes casos de pruebas. * La aplicación debe cumplir con los requerimientos funcionales especificados en la fase de análisis. * La aplicación debe cumplir con los requerimientos mínimos para el funcionamiento. |
| Pruebas de Seguridad | * El resultado de cada caso de prueba debe ser igual al resultado de salida esperado. * La aplicación debe cumplir con los requerimientos mínimos de seguridad. |
| Pruebas de Configuración | * Considerar todos los escenarios posibles. * Qué el sistema funcione bien en el ambiente de pruebas. |
| Pruebas de Recuperación a Fallas | * Considerar todos los escenarios posibles * Qué el sistema funcione de acuerdo a lo esperado después de las pruebas. |
| Pruebas de Aceptación | * Para realizar esta prueba se debe tomar como base los criterios de aceptación de las pruebas que se volverán a realizar. |

Para cada una de las pruebas se tendrá en cuenta:

* **Pruebas Unitarias:** Las pruebas unitarias se evalúan por medio de la siguiente tabla o lista de chequeo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Elemento a Revisar | SI | NO | No Aplica | Observaciones |
| ¿Se realizaron las Pruebas Unitarias con alguna herramienta especializada? |  |  |  |  |
| ¿Con las pruebas realizadas, cuál fue el porcentaje de cobertura del sistema? |  |  |  |  |
| ¿Existe constancia de la realización de las pruebas mencionadas? |  |  |  |  |
| ¿El funcionamiento de la prueba unitaria respeta el diseño establecido? |  |  |  |  |
| ¿Existe un manejo de errores adecuado? |  |  |  |  |
| ¿Se cumplió con la estrategia de ejecución de la prueba? |  |  |  |  |

* **Pruebas del Sistema:** El resultado de las pruebas del sistema se verá reflejado en el siguiente informe:

|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso** | <Identificador del Caso de uso> | **Descripción del escenario** | <Número total de casos de prueba ejecutados de acuerdo al escenario> |
| **Número de pruebas exitosas** | <Del total de pruebas ejecutadas, cuantas pruebas fueron exitosas> | **Número de pruebas Fallidas** | <Del total de pruebas ejecutadas, cuantas pruebas fueron fallidas> |
| **Tiempo máximo de ejecución de una prueba** | <Tiempo máximo que duró en ejecución una Prueba> | **Tiempo Promedio de ejecución de las pruebas** | <Tiempo promedio de ejecución de las pruebas> |
| **Número de peticiones exitosas** | <Número de peticiones http exitosas> | **Número de Peticiones Fallidas** | <Número de peticiones http fallidas> |
| **Número de Errores** | <Numero de errores ocurridos durante las pruebas> | **Tipo de errores** | <Descripción del tipo de errores presentados> |
| **% de Utilización del Procesador** | <Porcentaje de consumo de utilización de CPU durante la ejecución de la prueba> | **Cantidad de Memoria utilizada** | <Cantidad de MB de memoria utilizada en la prueba> |
| **Promedio de bytes enviados** | <Promedio de bytes enviados> | **Promedio de bytes recibidos** | <Promedio de bytes recibidos> |

* **Pruebas de Integración:** El resultado de las pruebas de integración se verá reflejado en el siguiente informe o lista de chequeo:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Elemento a Revisar** | **SI** | **NO** | **No Aplica** | **Observaciones** |
| ¿Se realizaron las pruebas de Integración con alguna herramienta especializada? |  |  |  |  |
| ¿Cuál fue el porcentaje de cobertura de la prueba con relación al sistema total? |  |  |  |  |
| ¿Existe constancia de la ejecución de las pruebas? |  |  |  |  |
| ¿Qué capas o componentes de la arquitectura se cubrió con la ejecución de las pruebas? |  |  |  |  |
| ¿Se estableció un criterio para la ejecución de las pruebas? ¿Cuál? |  |  |  |  |
| ¿Se cumplió la estrategia de ejecución de la prueba? |  |  |  |  |

* **Pruebas de Interoperabilidad:** El resultado de las pruebas de interoperabilidad se verá reflejado en el siguiente informe o lista de chequeo:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Elemento a Revisar** | **SI** | **NO** | **No Aplica** | **Observaciones** |
| ¿Se realizaron las pruebas de Interoperabilidad con alguna herramienta especializada? |  |  |  |  |
| ¿Cuál fue el porcentaje de cobertura de la prueba con relación al sistema total? |  |  |  |  |
| ¿Existe constancia de la ejecución de las pruebas? |  |  |  |  |
| ¿Qué capas o componentes de la arquitectura se cubrió con la ejecución de las pruebas? |  |  |  |  |
| ¿Se estableció un criterio para la ejecución de las pruebas? ¿Cuál? |  |  |  |  |
| ¿Se cumplió la estrategia de ejecución de la prueba? |  |  |  |  |

* **Pruebas de Regresión:** El resultado de las pruebas de regresión se verá reflejado de acuerdo a los tipos de pruebas seleccionados.
* **Pruebas Funcionales:** El resultado de las pruebas funcionales se verá reflejado de acuerdo al formato de set de pruebas, ver anexos.
* **Pruebas de interfaz de Usuario:** El resultado de las pruebas de interfaz de usuario se verá reflejado en el siguiente informe o lista de chequeo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Elemento a Revisar** | **SI** | **NO** | **No Aplica** | **Observaciones** |
| ¿Se realizaron las pruebas de interfaz de usuario con alguna herramienta especializada? |  |  |  |  |
| ¿Cuál fue el porcentaje de cobertura de la prueba con relación al sistema total? |  |  |  |  |
| ¿Existe constancia de la ejecución de las pruebas? |  |  |  |  |
| ¿Qué páginas se cubrió con la prueba? |  |  |  |  |
| ¿Se estableció un criterio para la ejecución de las pruebas? ¿Cuál? |  |  |  |  |
| ¿Se cumplió la estrategia de ejecución de la prueba? |  |  |  |  |

* **Pruebas de Seguridad:** El resultado de las pruebas de seguridad se verá reflejado en el siguiente informe o lista de chequeo:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Elemento a Revisar** | **SI** | **NO** | **No Aplica** | **Observaciones** |
| ¿Se realizaron las pruebas de seguridad con alguna herramienta especializada? |  |  |  |  |
| ¿Cuál fue el porcentaje de cobertura de la prueba con relación al sistema total? |  |  |  |  |
| ¿Existe constancia de la ejecución de las pruebas? |  |  |  |  |
| ¿Qué capas o componentes de la arquitectura se cubrió con la ejecución de las pruebas? |  |  |  |  |
| ¿Se estableció un criterio para la ejecución de las pruebas? ¿Cuál? |  |  |  |  |
| ¿Se cumplió la estrategia de ejecución de la prueba? |  |  |  |  |

* **Pruebas de Configuración:** El resultado de las pruebas de configuración se verá reflejado en el siguiente informe o lista de chequeo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Elemento a Revisar** | **SI** | **NO** | **No Aplica** | **Observaciones** |
| ¿Se realizaron las pruebas de Configuración con alguna herramienta especializada? |  |  |  |  |
| ¿Cuál fue el porcentaje de cobertura de la prueba con relación al sistema total? |  |  |  |  |
| ¿Existe constancia de la ejecución de las pruebas? |  |  |  |  |
| ¿Qué páginas se cubrió con la prueba? |  |  |  |  |
| ¿Se estableció un criterio para la ejecución de las pruebas? ¿Cuál? |  |  |  |  |
| ¿Se cumplió la estrategia de ejecución de la prueba? |  |  |  |  |

* **Pruebas de Recuperación a Fallas:** El resultado de las pruebas de Recuperación a Fallas se verá reflejado en el siguiente informe o lista de chequeo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Elemento a Revisar** | **SI** | **NO** | **No Aplica** | **Observaciones** |
| ¿Se realizaron las pruebas de recuperación a fallas con alguna herramienta especializada? |  |  |  |  |
| ¿Cuál fue el porcentaje de cobertura de la prueba con relación al sistema total? |  |  |  |  |
| ¿Existe constancia de la ejecución de las pruebas? |  |  |  |  |
| ¿Qué páginas se cubrió con la prueba? |  |  |  |  |
| ¿Se estableció un criterio para la ejecución de las pruebas? ¿Cuál? |  |  |  |  |
| ¿Se cumplió la estrategia de ejecución de la prueba? |  |  |  |  |

* **Pruebas de Aceptación:** El resultado de las pruebas de aceptación se verá reflejado de acuerdo a los tipos de pruebas seleccionados.
  1. **CRITERIOS DE TERMINACIÓN**

A continuación se señalan los criterios de terminación de las pruebas a ejecutar.

* Se ejecutaron todas las pruebas del sistema.
* Todas las pruebas se ejecutaron de acuerdo a los criterios de evaluación.
* Las pruebas de carga demuestran que se posee un grado satisfactorio de capacidad operativa y funcional.
* Los incidentes encontrados en las pruebas fueron corregidos y probados.
  1. **CRITERIOS DE SUSPENSIÓN**

Los criterios de suspensión impiden la iniciación y/o continuación de las pruebas ante cualquier situación de improvisto que hace que la ejecución de las pruebas no logre grados satisfactorios de probabilidad de éxito.

* Después de la instalación y configuración del sistema, se evidencia problemas o situaciones anormales en cualquiera de sus componentes.
* Después de la instalación y configuración del sistema, se evidencia que el ambiente de pruebas no es lo suficientemente estable para la ejecución de las pruebas.
* Discrepancia entre la documentación (Set de Pruebas, Casos de Uso) y el sistema.

1. ANEXOS

A continuación se listarán los anexos al plan de pruebas, que básicamente corresponden a todos los documentos, formatos o plantillas que se utilizarán en la especificación, ejecución y documentación de resultados de las pruebas.

* 1. **RELEASE NOTES**

A continuación se presenta el formato que se utilizará como release notes, el cual deberá acompañar cada una de las versiones entregadas para pruebas.

1. **Presentación**
   1. Identificador del Release: <Numero de Release>
   2. Descripción del producto:
2. **Requerimientos de Hardware, Sistema Operativo y Software Base.**

Se deben especificar los requerimientos de Hardware, Sistema Operativo y Software Base que el ambiente de pruebas debe tener instalado y configurado antes de iniciar el proceso de instalación del sistema.

|  |  |
| --- | --- |
| **COMPONENTE** | **REQUERIMIENTO** |
| HARDWARE |  |
| SISTEMA OPERATIVO |  |
| SOFTWARE BASE |  |

1. **Requerimientos del Sistema.**

Aquí se incluyen los requerimientos de instalación y configuración del sistema.

1. **Características Nuevas** (opcional).

Se describen las características nuevas que tiene el release entregado

1. **Características Obsoletas** (opcional).

Se describen las características obsoletas con relación a una versión anterior del release.

1. **Características Eliminadas** (opcional).

Se describen las características eliminadas con relación a una versión anterior del release.

* 1. **CASOS DE PRUEBAS**
     1. **FORMATO CASOS DE PRUEBA FUNCIONALES**

A continuación se presenta el formato que se utilizará como Set de Pruebas funcionales.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **INFORMACIÓN GLOBAL DEL CASO DE PRUEBA** | | | | | | | | | | | |
| **CASO DE PRUEBA No.** | | <Numero del caso de prueba constituido por SP-[numero del caso de uso]-[Numero del caso de prueba]> | | | | **VERSIÓN DE EJECUCIÓN** | | | | | <Versión diligenciado por el analista de pruebas en el momento de ejecutarla. Este número se incrementa de 1 en 1> |
| **FECHA EJECUCIÓN** | | | | | <Fecha de ejecución diligenciado por el analista de pruebas> |
| **CASO DE USO:** | | <Identificación del caso de uso objeto de la prueba> | | | | **MODULO DEL SISTEMA** | | | | | <Nombre del modulo al que corresponde el caso de uso objeto de la prueba> |
| **Descripción del caso de prueba:** | | <Descripción de lo que se pretende probar en el caso de prueba> | | | | | | | | | |
| 1. **CASO DE PRUEBA** | | | | | | | | | | | |
| 1. **Precondiciones** | | | | | | | | | | | |
| <Lista de precondiciones que deben cumplirse para realizar la prueba> | | | | | | | | | | | |
| 1. **Pasos de la prueba** | | | | | | | | | | | |
| <Pasos secuenciales que deben ser ejecutados por el analista de pruebas o usuario, ante el sistema para ejecutar la prueba> | | | | | | | | | | | |
| **DATOS DE ENTRADA** | | | | **RESPUESTA ESPERADA DE LA APLICACIÓN** | | | **COINCIDE** | | **RESPUESTA DEL SISTEMA** | | |
| **CAMPO** | **VALOR** | | **TIPO ESCENARIO** | **SI** | **NO** |
| <Descripción del dato de entrada> | <Valor que debe ser suministrado en la prueba para el dato de entrada> | | <Tipo de escenario que pretende probarse: Correcto/Incorrecto> | <Respuesta que se espera de la aplicación> | | |  |  | <Respuesta que se obtuvo de la aplicación en el momento de la ejecución de la prueba> | | |
|  |  | |  |  | | |  |  |  | | |
|  |  | |  |  | | |  |  |  | | |
|  |  | |  |  | | |  |  |  | | |
|  |  | |  |  | | |  |  |  | | |
|  |  | |  |  | | |  |  |  | | |
| 1. **Post condiciones** | | | | | | | | | | | |
| <Lista de poscondiciones que deben cumplirse después de realizar la prueba> | | | | | | | | | | | |
| 1. **RESULTADOS DE LA PRUEBA** | | | | | | | | | | | |
| **Defectos y desviaciones** | | | | | | | | | | **Veredicto** | |
| <Lista de defectos o desviaciones encontrados por el analista o usuario al ejecutar la prueba> | | | | | | | | | |  | |
| **Observaciones** | | | | | **Probador** | | | | | | |
| <Observaciones generales del analista o usuario sobre la ejecución de la prueba> | | | | |  | | | | | | |
| **Firma:**  **Nombre:**  **Fecha:** | | | | | | |

* + 1. **LISTA DE CHEQUEO CASOS DE PRUEBAS FUNCIONALES**

Con el fin de garantizar que los casos de prueba contemplen el 100% de los escenarios a probar para cada caso de uso; en su construcción deberá tenerse en cuenta la siguiente lista de chequeo.

Cada conjunto de casos de prueba para cada caso de uso deberá contemplar:

|  |  |
| --- | --- |
| **ELEMENTO DEL CASO DE USO** | **CASO DE PRUEBA** |
| **Datos de entrada** | Verificar que los datos de entrada cumplan con:   * Obligatoriedad * Tipo de datos * Longitud * Estructura |
| **Reglas de Negocio** | Validar reglas de negocio que afecten los datos de entrada (Dependencia de datos). |
| Validar reglas de negocio que afecten los flujos. |
| **Flujos Alternos** | Verificar la ejecución de todos los flujos alternos. |
| **Flujos de Excepción** | Verificar la ejecución de todos los flujos de Excepción. |
| **Flujo Básico** | Verificar la ejecución del flujo básico. |
| **Generalidades:** | Los casos de prueba deben especificar exactamente rutas, nombres de archivos, valores para los datos de entrada.  Para asegurar que las rutas y nombres de archivos se cumplan; deberá instalarse una árbol de carpetas predefinido en la estación donde se ejecutará la prueba. |

* + 1. **ENCUESTA PARA PRUEBAS DE USABILIDAD**

Las pruebas de usabilidad se guiaran por la siguiente estructura de encuesta:

|  |  |
| --- | --- |
| **PREGUNTA** | **CRITERIOS DE EVALUACIÓN** |
| **1. ¿Hay términos en idiomas diferentes mezclados?** | 1 = Se encuentran en todo el sistema  2 = Se encuentra en algunas partes del sistema.  3 = No se encuentran en ninguna parte del sistema. |
| **2. ¿Es simple el vocabulario utilizado?** | 1 = El vocabulario es demasiado técnico.  2 = El vocabulario presenta algunas dificultades de comprensión.  3 = El vocabulario es completamente comprensible. |
| **3. ¿Se proporciona tiempo suficiente para realizar las entradas por teclado?** | 1 = El tiempo es muy limitado.  2 = El tiempo es limitado para algunas funcionalidades.  3 = El tiempo es completamente suficiente. |
| **4. ¿Hay algún tipo de asistencia para los usuarios que hacen uso del sistema por primera vez?** | 1 = No existe ninguna ayuda.  2 = Se encuentra ayuda en algunas partes.  3 = Existen ayudas en todo el sistema. |
| **3. ¿El sistema es fácil de operar para alguien que no recibió capacitación en su operación?** | 1 = El sistema es de difícil comprensión.  2 = El sistema es fácil de operar en algunas de sus funcionalidades.  3 = El sistema es completamente fácil de operar. |
| **6. ¿Se entienden la interfaz y su contenido?** | 1 = No se entiende su interfaz.  2 = La interfaz se entiende en algunas partes.  3 = La interfaz es completamente entendible. |
| **7. ¿Resulta fácil identificar un objeto o una acción?** | 1 = Es difícil identificar los objetos o acciones.  2 = Se pueden identificar los objetos y acciones en algunas partes del sistema.  3 = Todos los objetos y acciones son fácilmente identificables. |
| **8. ¿Resulta fácil entender el resultado de una acción?** | 1 = Los resultados de las acciones no son entendibles.  2 = Los resultados de las acciones son entendibles en algunas partes o la mayor parte del sistema.  3 = Todos los resultados de las acciones son entendibles. |
| **9. ¿Está diseñada la interfaz para facilitar la realización eficiente de las tareas de la mejor forma posible?** | 1 = La interfaz es difícil de usar.  2 = La interfaz es difícil de usar en algunas partes del sistema.  3 = La interfaz es completamente sencilla de usar. |
| **10. ¿Son apropiados los mensajes presentado por el sistema?** | 1 = Los mensajes non son apropiados.  2 = Los mensajes son apropiados en algunas partes del sistema.  3 = Todos los mensajes son apropiados y fáciles de comprender. |
| **11. ¿Actúa el sistema en la prevención de errores?** | 1 = El sistema no previene errores del usuario.  2 = El sistema previene algunos o la mayoría de los errores del usuario.  3 = El sistema previene cualquier error que pueda cometer el usuario. |
| **12. ¿El sistema informa claramente sobre los errores presentados?** | 1 = El sistema no informa de manera adecuada sobre los errores cometidos.  2 = El sistema informa de manera adecuada algunos o la mayoría de los errores cometidos por el usuario.  3 = El sistema informa de forma adecuada todos los errores cometidos por el usuario. |
| **13. ¿Se utiliza mensajes y textos descriptivos?** | 1 = Los mensajes de texto no son descriptivos.  2 = La mayoría de los textos son descriptivos o fáciles de interpretar  3 = Todos los textos son descriptivos o fáciles de interpretar. |
| **14. ¿Permite una cómoda navegación dentro del producto y una fácil salida de éste?** | 1 = La navegación no es sencilla.  2 = La navegación presenta algunas dificultades.  3 = La navegación es sencilla, requiere de pocos vínculos para accedes a las funcionalidades del sistema. |
| **13. ¿Se permite al usuario personalizar la interfaz?** | 1 = La interfaz no es personalizable.  2 = La interfaz es personalizable con algunas restricciones.  3 = La interfaz es completamente personalizable. |
| **16. ¿Se proporciona información visual de dónde está el usuario, qué está haciendo y qué puede hacer a continuación?** | 1 = No se presenta ninguna información visual ni otro tipo de ayuda.  2 = Presenta ayudas en algunas partes del sistema.  3 = Las ayudas son apropiadas y están distribuidas a los largo del sistema. |
| **17. ¿Existe atajos del teclado bien hechos?** | 1 = No existen atajos por teclado.  2 = Existen algunos atajos por teclado.  3 = Todas las opciones presentan atajos por teclado. |
| **18. ¿Se presenta al usuario la información que sólo necesita?** | 1 = La información presentadas es más de la que necesita y tiende a ser confusa.  2 = En algunas partes se presenta mayor información a la necesaria.  3 = La información es estrictamente la necesaria según el perfil. |

* + 1. **FORMATO CASOS DE PRUEBA TECNICOS**

A continuación se presenta el formato que se utilizará para documentar las pruebas técnicas; estas pruebas serán documentadas conforme avance el desarrollo de la solución y se tengan versiones liberadas sobre las que se aplicarán estas pruebas.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **INFORMACIÓN GLOBAL DEL CASO DE PRUEBA** | | | | | | |
| **Tipo de Prueba:** | <Descripción del tipo de prueba: Carga, Volumen, Estrés, ETC> | | **Código de la prueba** | | <Codificación de la prueba> | |
| **Descripción de la prueba:** | <Descripción del objetivo de la prueba> | | | | | |
| **Versión de Ejecución** | <Versión o iteración de ejecución de la prueba> | | **Fecha de Ejecución** | | | <Fecha de ejecución en formato AAAA/MM/DD diligenciado por el analista de pruebas al momento de su ejecución> |
| 1. **Prerrequisitos de la prueba** | | | | | | |
| <Lista de los prerrequisitos a tener en cuenta antes de ejecutar la prueba> | | | | | | |
| 1. **Insumos de la prueba** | | | | | | |
| <Lista de Insumos necesarios para ejecutar la prueba> | | | | | | |
| 1. **Lista de chequeo de la prueba** | | | | | | |
| **Pasos a Seguir** | | **Prueba satisfactoria** | | | | **Observaciones** |
| **SI** | | **NO** | |
| <Pasos numerados en orden lógico para la ejecución de la prueba> | |  | |  | |  |
|  | |  | |  | |  |
|  | |  | |  | |  |
|  | |  | |  | |  |
| 1. **Resultados de la prueba** | | | | | | |
| **Defectos y desviaciones** | | | | | | **Veredicto** |
| <Lista de defectos o desviaciones encontrados por el analista o usuario al ejecutar la prueba> | | | | | |  |
| **Observaciones** | | | | | | **Probador** |
| <Observaciones generales del analista o usuario sobre la ejecución de la prueba> | | | | | |  |
| **Firma:**  **Nombre:**  **Fecha:** |

* + 1. **MATRIZ CASOS DE USO VS CASOS DE PRUEBA FUNCIONALES**

A continuación se presenta el formato de matriz de trazabilidad que se llevara para asegurar que sean probados todos los aspectos definidos dentro de los casos de uso.

|  |  |  |
| --- | --- | --- |
| Caso de Uso | Aspecto a Evaluar | Caso de Prueba |
| <Identificación del caso de uso> | **1. Datos Entrada** |  |
| Obligatoriedad | <Identificación del caso de prueba que evalúa Obligatoriedad> |
| Longitud | <Identificación del caso de prueba que evalúa Longitud> |
| Tipo de Dato | <Identificación del caso de prueba que evalúa Tipo de dato> |
| **2. Reglas de Negocios Relacionadas con datos de entrada** |  |
| <Lista de casos de prueba> | <Identificación del caso de prueba que evalúa la regla de negocio> |
| **3. Reglas de Negocios** |  |
| <Lista de casos de prueba> | <Identificación del caso de prueba que evalúa la regla de negocio> |
| **4. Flujos de Excepción** |  |
| <Lista de flujos de excepción> | <Identificación del caso de prueba que evalúa los flujos de excepción |
| **5. Flujos Alternos** |  |
| <Lista de casos de flujos alternos> | <Identificación del caso de prueba que evalúa los flujos alternos.> |
| **6. Flujo Básico** | <Identificación del caso de prueba que evalúa el flujos básico. |

* + 1. **MATRIZ REQUERIMIENTOS NO FUNCIONALES VS CASOS DE PRUEBA TÉCNICOS**

A continuación se presenta el formato de matriz de trazabilidad que se llevara para asegurar que sean probados todos los aspectos técnicos de la solución; en esta matriz se registrará cada caso de prueba técnico y requerimiento no funcional que será verificado. Esta matriz será diligenciada en la medida que las pruebas técnicas sean diseñadas.

|  |  |  |
| --- | --- | --- |
| **CÓDIGO DE LA PRUEBA TÉCNICA** | **REQUERIMIENTO NO FUNCIONAL VERIFICADO** | **OBSERVACIONES** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. **LISTA DE CHEQUEO**

A continuación se presenta el formato que se utilizará para lista de chequeo de las pruebas ejecutadas

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TIPO DE PRUEBA** | **Versión de Ejecución** | **Fecha de Ejecución** | **EJECUTADA** | **CUMPLE** | **NO CUMPLE** | **Observaciones** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

* 1. **INFORME DE PRUEBAS**

Ver el Esquema y Contenido del entregable informe de pruebas de la fase 4.

* 1. **PROCEDIMIENTO PARA INCIDENCIAS**

El procedimiento para el manejo de incidencias esta descrito en el plan de proyecto en el capítulo 9. METODOLOGÍA DE SEGUIMIENTO Y CONTROL DE INCIDENCIAS.