Interpool

Plan de Verificación y Validación

Versión 1.2

Historia de revisiones

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Descripción | Autor |
| 16/08/10 | 1.0 | Creación del documento | Alejandro García |
| 20/08/10 | 1.1 | Revisión del documento | Javier Madeiro |
| 01/09/10 | 1.2 | Ajustes del Plan | Alejandro García |
| 04/09/10 | 1.2 | Revisión del documento | Javier Madeiro |

Contenido

[1. Introducción 3](#_Toc273116235)

[1.1. Propósito 3](#_Toc273116236)

[1.2. Punto de partida 3](#_Toc273116237)

[1.3. Alcance 3](#_Toc273116238)

[1.4. Identificación del proyecto 4](#_Toc273116239)

[1.5. Estrategia de evolución del Plan 4](#_Toc273116240)

[2. Requerimientos para verificar 5](#_Toc273116241)

[3. Estrategia de Verificación 6](#_Toc273116242)

[3.1. Tipos de pruebas 6](#_Toc273116243)

[3.1.1. Prueba de integridad de los datos y la base de datos 6](#_Toc273116244)

[3.1.2. Prueba de Funcionalidad 6](#_Toc273116245)

[3.1.3. Prueba de Ciclo del Negocio 7](#_Toc273116246)

[3.1.4. Prueba de Interfase de Usuario 7](#_Toc273116247)

[3.1.5. Prueba de Performance 7](#_Toc273116248)

[3.1.6. Prueba de Carga 8](#_Toc273116249)

[3.1.7. Prueba de Esfuerzo (stress, competencia por recursos, bajos recursos) 8](#_Toc273116250)

[3.1.8. Prueba de Volumen 9](#_Toc273116251)

[3.1.9. Prueba de Seguridad y Control de Acceso 10](#_Toc273116252)

[3.1.10. Prueba de Fallas y Recuperación 10](#_Toc273116253)

[3.1.11. Prueba de Configuración 11](#_Toc273116254)

[3.1.12. Prueba de Instalación 12](#_Toc273116255)

[3.1.13. Prueba de Documentos 12](#_Toc273116256)

[3.2. Herramientas 12](#_Toc273116257)

[4. Recursos 13](#_Toc273116258)

[4.1. Roles 13](#_Toc273116259)

[4.2. Sistema 13](#_Toc273116260)

[5. Hitos del proyecto de Verificación 14](#_Toc273116261)

[6. Entregables 15](#_Toc273116262)

[6.1. Modelo de Casos de Prueba 15](#_Toc273116263)

[6.2. Informes de Verificación 15](#_Toc273116264)

[6.3. Evaluación de la verificación 16](#_Toc273116265)

[6.4. Informe final de verificación 16](#_Toc273116266)

[7. Dependencias 16](#_Toc273116267)

[7.1. Dependencia de personal 16](#_Toc273116268)

[7.2. Dependencia de software 16](#_Toc273116269)

[7.3. Dependencia de hardware 16](#_Toc273116270)

[7.4. Dependencia de datos y base de datos de prueba 16](#_Toc273116271)

[8. Riesgos 17](#_Toc273116272)

[8.1. Planificación 17](#_Toc273116273)

[8.2. Técnico 17](#_Toc273116274)

[8.3. Gestión 17](#_Toc273116275)

[9. Apéndice 17](#_Toc273116276)

[9.1. Niveles de gravedad de error 17](#_Toc273116277)

[9.2. Niveles de aceptación para los elementos verificados 17](#_Toc273116278)

1. Introducción
   1. Propósito

Este Plan de Verificación para el proyecto Interpool soporta los siguientes objetivos:

* Identificar la información de proyecto existente y los componentes de software que deben ser verificados.
* Enumerar los requerimientos recomendados para verificar.
* Recomendar y describir las estrategias de verificación que serán usadas.
* Identificar los recursos necesarios y proporcionar una estimación de esfuerzo para realizar la verificación.
* Enumerar los entregables del proyecto de verificación.
  1. Punto de partida

El proyecto presentado por Microsoft tiene la finalidad de desarrollar una aplicación cliente-servidor para Windows Phone 7, utilizando dos de las últimas tecnologías que esta compañía ha generado para desarrolladores: Silverlight para Windows Phone 7 y Azure.

El destino de la verificación del proyecto es el correcto funcionamiento del producto a construir en base a lo acordado con el cliente. Se debe verificar que todos los requerimientos funcionales y no funcionales especificados por el cliente sean cumplidos de forma de lograr la satisfacción del mismo.

Tendrá como objetivo buscar, por medio de las pruebas entregar un sistema de calidad al cliente de manera de minimizar la probabilidad de que el sistema falle cuando se encuentre en producción.

Para la realización de la verificación se cuenta con un equipo integrado por:

* Alejandro García – Responsable de verificación
* Javier Madeiro – Asistente de verificación
* Marcos Sander – Asistente de verificación
* Juan Ghiringhelli – Asistente de verificación
* José Cordero – Asistente de verificación
* Leticia Vilariño – Asistente de verificación
  1. Alcance

Las distintas fases de la verificación serán las siguientes:

* **Pruebas Unitarias:** Se verificarán los diferentes módulos del sistema antes de integrarlos con la finalidad de detectar las fallas en el momento de menor impacto. Las pruebas serán realizadas por el equipo de desarrolladores y de ser posible por la persona que haya implementado cada modulo.
* **Pruebas de Integración:** Se verificará que los componentes interactúen en forma correcta entre ellos. Se tratará que sean realizadas por el equipo de desarrolladores.
* **Pruebas de Función:** Se verificará que el sistema integrado cumpla con todos los requerimientos funcionales acordados con el cliente.

Se prueba cada funcionalidad de forma individual (ej.: testeo de casos de uso) y se pruebas distintas combinaciones de funcionalidades (ej.: testeo de ciclos de casos de uso)

* **Prueba de Desempeño:** Se verificará que el sistema integrado cumpla con los requerimientos no funcionales.
* **Prueba de Diseño:** Se verificará que el diseño y la arquitectura utilicen buenas prácticas y que el código sea legible y cumpla con los estándares propuestos por Microsoft.
* **Prueba de aceptación:** Se verificará bajo la supervisión del cliente, si el sistema cumple con los requerimientos pedidos.

Cabe agregar que el cliente requiere que los casos de prueba cubran el 70% del código.

* 1. Identificación del proyecto

Los documentos usados para elaborar el Plan de Verificación son los siguientes:

* Plantilla del Plan de Verificación y Validación incluida en el Modelo de Proceso MUM.
* Documento de Especificación de Requerimientos v1.1.
* Actas de las reuniones con el cliente disponibles.
* Letra del proyecto brindada por el cliente.
* Diapositivas del curso Introducción a la Ingeniería de Software del año 2010.
* Plan de Verificación y Validación de otros años tomado de la Memoria Organizacional de la asignatura.
  1. Estrategia de evolución del Plan

El monitoreo del plan de verificación y validación está a cargo del Responsable de Verificación Alejandro García. El plan será revisado periódicamente y será modificado si se considera necesario, sobre todo en las primeras semanas donde dicho plan depende de los cambios que hayan sufrido los documentos.

Cualquier cambio que pueda afectar al plan de verificación y validación deberá ser informado al responsable del mismo, para corroborar el plan y tomar las medidas necesarias.

Los cambios en el Plan serán evaluados y aprobados directamente por el responsable de verificación junto con el equipo de asistentes. Si se tratase de un cambio mayor se consultaría a los distintos verificadores.

Cualquier cambio será informado al resto del equipo de Verificación mediante los medios de comunicación definidos (entregable COMDIG2v1\_0.pdf). La última versión del Plan estará en el repositorio del grupo.

1. Requerimientos para verificar

En la lista a continuación se presentan los elementos, casos de uso, requerimientos funcionales y requerimientos no funcionales, que serán verificados.

**Funcionales:**

* El sistema deberá manejar niveles de dificultad para cada usuario, así como un mecanismo de puntuación.
* El sistema tendrá que generar escenarios de juego a partir de información extraída de motor de búsqueda Bing.
* El sistema deberá mostrar la ciudad a la que fue en busca de pista y/o del sospechoso para atraparlo, brindando información real y una imagen o animación de la ciudad en cuestión.
* El sistema brinda la acción “Interrogar”. Se mostrarán tres personajes famosos de la ciudad actual (en el juego), extraídos del Bing, los cuales brindaran pistas acerca del sospechoso. Luego de que el usuario elija un personaje, el sistema mostrara una imagen del mismo y éste ultimo deberá dejar información “escondida” sobre el sospechoso y el próximo lugar a donde dirigirse, sacando información de los contactos de Facebook y de la web respectivamente.
* El sistema debe permitir consultar todas las pistas recogidas hasta el momento.
* El sistema brinda la acción “Mirar”. Se muestran los tres posibles personajes a interrogar por el jugador.
* El sistema brinda la acción “Viajar”. Se muestra una animación en donde el usuario podrá elegir una ciudad hacia dónde dirigirse.
* El sistema brinda la acción “Filtrar”. Se muestra un formulario de datos a llenar del sospechoso. El usuario podrá llenar estos datos y luego buscar en los contactos del Facebook quien tiene las características ingresadas en el formulario.
* El sistema brindará la posibilidad de arrestar al sospechoso siempre y cuando el sospechoso sea un único contacto.
* El sistema le brindará un código de verificación enviando una solicitud de amistad en Facebook al “Gran Sospechoso”.

**No Funcionales:**

* Cuenta de Facebook.
* Desarrollar sobre plataforma Microsoft.
* Lenguajes y herramientas para el desarrollo: Silverlight y Azure.
* Código en C#.
* Cumplir con el estándar de código de Microsoft.
* El sistema debe soportar los lenguajes Inglés y Español.
* El sistema deberá cumplir con la aprobación de MarketPlace.
* El sistema debe contar con una interfaz atractiva.
* El sistema deberá ser entretenido.
* El sistema deberá contar con una aplicación cliente y otro servidor.
* El servidor deberá almacenar información de todos los participantes del juego.
* El sistema debe contar con un mínimo de 30 ciudades disponibles para jugar.

1. Estrategia de Verificación

Esta sección describe cada una de las pruebas que se implementarán. Se indicarán las técnicas usadas y el criterio aceptación.

* 1. Tipos de pruebas
     1. Prueba de integridad de los datos y la base de datos
        1. Objetivo de la prueba

Asegurar que los métodos y procesos de acceso a la base de datos funcionan correctamente y sin corromper datos.

* + - 1. Técnica

Invoque cada método o proceso de acceso a la base de datos con datos válidos y no válidos.

Inspeccione la base de datos para asegurarse de que se han guardado los datos correctos, que todos los eventos de la base de datos ocurrieron correctamente, o repase los datos devueltos para asegurar que se recuperaron datos correctos por la vía correcta.

* + - 1. Criterio de aceptación

Todos los métodos y procesos de acceso a la base de datos funcionan como fueron diseñados y sin datos corruptos.

* + - 1. Consideraciones especiales

La prueba requiere un entorno de administración de bases de datos SQL 2008 Express para ingresar o modificar información directamente en la base de datos.

Los procesos deben ser invocados manualmente.

Se deben usar bases de datos pequeñas para aumentar la facilidad de inspección de los datos para verificar que no sucedan eventos no aceptables.

* + 1. Prueba de Funcionalidad

La prueba de funcionalidad se enfoca en requerimientos para verificar que se corresponden directamente a casos de usos o funciones y reglas del negocio. Los objetivos de estas pruebas son verificar la aceptación de los datos, el proceso, la recuperación y la implementación correcta de las reglas del negocio. Este tipo de prueba se basa en técnicas de caja negra, que consisten en verificar la aplicación y sus procesos interactuando con la aplicación por medio de la interfase de usuario y analizar los resultados obtenidos.

* + - 1. Objetivo de la prueba

Asegurar la funcionalidad apropiada del objeto de prueba, incluyendo la navegación, entrada de datos, proceso y recuperación.

* + - 1. Técnica

Ejecute cada caso de uso, flujo de caso de uso, o función usando datos válidos y no válidos, para verificar lo siguiente:

* Se obtienen los resultados esperados cuando se usan datos válidos.
* Cuando se usan datos no válidos se despliegan los mensajes de error o advertencia apropiados.
* Se aplica apropiadamente cada regla del negocio.
  + - 1. Criterio de aceptación

Todas las pruebas planificadas se realizaron. Todos los defectos encontrados han sido debidamente identificados.

* + - 1. Consideraciones especiales

Las pruebas tienen que pasar el 100%, mientras que el cubrimiento del código de las pruebas debe ser de un 70% como mínimo de acuerdo a los estándares de Microsoft.

* + 1. Prueba de Ciclo del Negocio

Esta prueba debe simular las actividades realizadas en el proyecto en el tiempo. Se debe identificar un período, que puede ser un año, y se deben ejecutar las transacciones y actividades que ocurrirían en el período de un año. Esto incluye todos los ciclos diarios, semanales y mensuales y eventos que son sensibles a la fecha.

No se realizarán pruebas de ciclo de negocio para nuestro sistema, ya que no se cuentan con operaciones sensibles a la fecha.

* + 1. Prueba de Interfase de Usuario

Esta prueba verifica que la interfase de usuario proporcione al usuario el acceso y navegación a través de las funciones apropiada. Además asegura que los objetos presentes en la interfase de usuario se muestren como se espera y conforme a los estándares establecidos por la empresa o de la industria.

* + - 1. Objetivo de la prueba

Verificar que: la navegación a través de los elementos que se están probando reflejen las funciones del negocio y los requerimientos, incluyendo manejo de ventanas, campos y métodos de acceso; los objetos de las ventanas y características, como menús, tamaño, posición, estado funcionen de acuerdo a los estándares.

* + - 1. Técnica

Crear o modificar pruebas para cada ventana verificando la navegación y los estados de los objetos para cada ventana de la aplicación y cada objeto dentro de la ventana.

* + - 1. Criterio de aceptación

Cada ventana ha sido verificada exitosamente siendo consistente con una versión de referencia o estándar establecido.

* + - 1. Consideraciones especiales

Se le dará una gran importancia a estas pruebas, ya que la interfaz de usuario es sumamente importante en este producto.

* + 1. Prueba de Performance

En esta prueba se miden y evalúan los tiempos de respuesta, los tiempos de transacción y otros requerimientos sensitivos al tiempo. El objetivo de la prueba es verificar que se logren los requerimientos de performance. La prueba de performance es implementada y ejecutada para poner a punto los destinos de pruebas de performance como función de condiciones de trabajo o configuraciones de hardware.

* + - 1. Objetivo de la prueba

Verificar la performance de determinadas transacciones o funciones de negocio bajo ciertas condiciones:

* condiciones de trabajo normales conocidas.
* peores casos de condiciones de trabajo conocidas.
  + - 1. Técnica
* Usar procedimientos de prueba desarrollados para verificar funciones o ciclos de negocio.
* Modificar archivos de datos para aumentar el número de transacciones o los procedimientos de prueba para aumentar el número de iteraciones de ocurrencia de transacciones.
* Las pruebas se deben ejecutar en una máquina (mejor caso de prueba un solo usuario, una sola transacción) y se debe repetir con múltiples usuarios (virtuales o reales).
  + - 1. Criterio de aceptación

Con una transacción o un usuario: Éxito completo de la prueba sin fallas y dentro del tiempo esperado o requerido.

Con múltiples transacciones y varios usuarios: Éxito completo de la prueba sin fallas y dentro de un tiempo aceptable.

* + - 1. Consideraciones especiales

Ninguna

* + 1. Prueba de Carga

La prueba de carga somete los objetos a verificar a diferentes cargas de trabajo para medir y evaluar los comportamientos de performance y la habilidad de los objetos de continuar funcionando apropiadamente bajo diferentes cargas de trabajo. El objetivo es determinar y asegurar que el sistema funciona apropiadamente en circunstancias de máxima carga de trabajo esperada. Además evaluar las características de performance, como tiempos de respuesta, tiempos de transacciones y otros elementos sensitivos al tiempo.

* + - 1. Objetivo de la prueba

Verificar el comportamiento de performance de determinados componentes del software bajo condiciones de trabajo diferentes.

* + - 1. Técnica

Usar pruebas desarrolladas para funciones o ciclos de negocios y modificar archivos de datos para aumentar el número de transacciones o las pruebas para aumentar la cantidad de ocurrencia de transacciones.

* + - 1. Criterio de aceptación

Para múltiples transacciones y múltiples usuarios: realización exitosa de las pruebas sin fallas y dentro del tiempo aceptable.

* + - 1. Consideraciones especiales

Ninguna

* + 1. Prueba de Esfuerzo (stress, competencia por recursos, bajos recursos)

La prueba de esfuerzo en un tipo de prueba de performance implementada y ejecutada para encontrar errores cuando hay pocos recursos o cuando hay competencia por recursos. Poca memoria o poco espacio de disco pueden revelar fallas en el software que no aparecen bajo condiciones normales de cantidad de recursos. Otras fallas pueden resultar al competir por recursos compartidos como bloqueos de bases de datos o ancho de banda de red. La prueba de esfuerzo también puede usarse para identificar el trabajo máximo que el software puede manejar.

* + - 1. Objetivo de la prueba

Verificar que el software funciona apropiadamente y sin error bajo condiciones de esfuerzo, como son:

* poca memoria o sin disponibilidad de memoria en el servidor
* cantidad máxima de clientes conectados
* múltiples usuarios realizando la misma operación sobre los mismos datos
* peor caso de volumen de operaciones.

El objetivo de la prueba de esfuerzo es también identificar y documentar las condiciones bajo las cuales el sistema falla y no continua funcionando apropiadamente.

* + - 1. Técnica

Usar las pruebas desarrolladas para Performance y Prueba de Carga.

Para probar recursos limitados, las pruebas se deben ejecutar en una sola máquina, y se debe reducir o limitar la memoria en el servidor.

Para las pruebas de esfuerzo restantes, deber usarse múltiples clientes, cualquiera que ejecute las mismas pruebas o pruebas complementarias para producir el peor caso de volumen de operaciones.

* + - 1. Criterio de aceptación

Todas las pruebas planeadas se ejecutaron y se alcanzaron o excedieron los límites del sistema sin que el software fallara o las condiciones bajo las que ocurre una falla en el software están fuera de las condiciones especificadas.

* + - 1. Consideraciones especiales

Las pruebas de esfuerzo de red pueden requerir herramientas de red para cargar la red con mensajes o paquetes.

Sincronizar el acceso simultáneo de varios clientes accediendo a los mismos datos.

* + 1. Prueba de Volumen

La Prueba de Volumen somete el software a grandes cantidades de datos para determinar si se alcanzan límites que causen la falla del software. La Prueba de Volumen identifica la carga máxima continua que puede manejar el software a prueba en un período dado.

* + - 1. Objetivo de la prueba

Verificar que el software funciona correctamente con volúmenes de datos grandes:

* Máximo (real o físicamente posible) número de clientes conectados, o simulados, todos realizando la misma operación (peor caso de operación) por un período de tiempo extenso.
* Máximo tamaño de base de datos y múltiples consultas ejecutadas simultáneamente.
  + - 1. Técnica

Usar pruebas desarrolladas para Prueba de Performance y Prueba de Carga.

* Se deben usar múltiples clientes, ejecutando las mismas pruebas o pruebas complementarias para producir el peor caso de volumen de operaciones o mezcla en un período de tiempo extenso.
* Se debe crear el tamaño máximo de base de datos (real, escalado o con datos representativos) y múltiples clientes ejecutando consultas simultáneamente por un período de tiempo extenso.
  + - 1. Criterio de aceptación

Todas las pruebas planificadas se ejecutaron y se han alcanzado o excedido los límites especificados sin que el software falle.

* + - 1. Consideraciones especiales

Ninguna

* + 1. Prueba de Seguridad y Control de Acceso

La Prueba de Seguridad y Control de Acceso se enfoca en dos áreas de seguridad:

* Seguridad en el ámbito de aplicación, incluyendo el acceso a los datos y a las funciones de negocios.
* Seguridad en el ámbito de sistema, incluyendo conexión, o acceso remoto al sistema.

La seguridad en el ámbito de aplicación asegura que, basado en la seguridad deseada los actores están restringidos a funciones o casos de uso específicos o limitados en los datos que están disponibles para ellos.

La seguridad en el ámbito de sistema asegura que, solo los usuarios con derecho a acceder al sistema son capaces de acceder a las aplicaciones y solo a través de los puntos de ingresos apropiados.

* + - 1. Objetivo de la prueba

Seguridad en el ámbito de aplicación: Verificar que un actor pueda acceder solo a las funciones o datos para los cuales su tipo de usuario tiene permiso.

Seguridad en el ámbito de sistema: Verificar que solo los actores con acceso al sistema y a las aplicaciones, puedan acceder a ellos.

* + - 1. Técnica

Se verificará que el usuario administrador tenga los permisos adecuados y que sea el encargado de acceder a los datos de la aplicación.

* + - 1. Criterio de aceptación

Para cada tipo de actor conocido las funciones y datos apropiados están disponibles, y todas las operaciones funcionan como se espera y ejecutan las pruebas de Funcionalidad de la aplicación.

* + - 1. Consideraciones especiales

Ninguna

* + 1. Prueba de Fallas y Recuperación

Las Pruebas de Fallas y Recuperación aseguran que el software puede recuperarse de fallas de hardware, software o mal funcionamiento de la red sin pérdida de datos o de integridad de los datos.

La Prueba de Recuperación es un proceso en el cual la aplicación o sistema se expone a condiciones extremas, o condiciones simuladas, para causar falla, como fallas en dispositivos de Entrada/Salida o punteros a la base de datos inválidos. Los procedimientos de recuperación se invocan y la aplicación o sistema es monitoreado e inspeccionado para verificar que se recupera apropiadamente la aplicación o sistema y se logre la recuperación de datos.

* + - 1. Objetivo de la prueba

Verificar que el software luego de una falla se puede recuperar correctamente y que existe un adecuado manejo de los mensajes de error.

* + - 1. Técnica

Se crearán casos de prueba que desplieguen mensajes de error y se verificará que el sistema mantenga un estado estable y los mensajes de error sean correctos. Por ejemplo:

* Apagar el Windows Phone 7 mientras se está jugando.
* Interrumpir la energía del servidor.
* Simular o iniciar la pérdida de comunicación con la red.
* Simular o iniciar la pérdida de comunicación con la red social Facebook.
* Simular o iniciar la pérdida de comunicación con el buscador Bing
  + - 1. Criterio de aceptación

En todos los casos, la aplicación, la base de datos y el sistema deben, en la realización procedimientos de recuperación, volver a un estado conocido y deseable. Este estado incluye corrupción de datos limitada a los campos, punteros o claves corruptos conocidos, y reportes indicando los procesos u operaciones que no se completaron debido a las interrupciones.

* + - 1. Consideraciones especiales

Los procedimientos para desconectar cables (simulando falta de energía o pérdida de comunicación) no son deseables o factibles. Se pueden requerir métodos alternativos, como software de diagnóstico. Se requieren los grupos de recursos de Sistemas, Bases de datos y Red.

Estas pruebas deben ejecutarse fuera del horario de trabajo normal o en una máquina aislada.

* + 1. Prueba de Configuración

La Prueba de Configuración verifica el funcionamiento del software con diferentes configuraciones de software y hardware. En nuestro caso, se busca probar el sistema en las 2 configuraciones de idioma posibles (inglés y español).

* + - 1. Objetivo de la prueba

Verificar que el software funcione apropiadamente en las configuraciones requeridas de hardware y software. En nuestro caso, se busca que el producto a construir funcione correctamente para el celular Windows Phone 7.

* + - 1. Técnica

Usar las pruebas de Funcionalidad.

* Abrir y cerrar varias sesiones de software que no son objeto de prueba, como parte de la prueba o antes de comenzar la prueba.
* Ejecutar operaciones seleccionadas para simular la interacción del actor con el software objeto de prueba y con el software que no es objeto de prueba.
* Repetir los procedimientos anteriores minimizando la memoria convencional disponible en la máquina cliente.
  + - 1. Criterio de aceptación

Por cada combinación de software objeto de prueba y software que no es objeto de prueba, todas las operaciones son completadas exitosamente sin fallas.

* + - 1. Consideraciones especiales

Ninguna.

* + 1. Prueba de Instalación

La Prueba de Instalación tiene dos propósitos. Uno es asegurar que el software puede ser instalado en diferentes condiciones (como una nueva instalación, una actualización, y una instalación completa o personalizada) bajo condiciones normales y anormales. Condiciones anormales pueden ser insuficiente espacio en disco, falta de privilegios para crear directorios, etc. El otro propósito es verificar que, una vez instalado, el software opera correctamente. Esto significa normalmente ejecutar un conjunto de pruebas que fueron desarrolladas para Prueba de Funcionalidad.

* + - 1. Objetivo de la prueba

Verificar que el software objeto de prueba se instala correctamente en cada configuración de hardware requerida bajo las siguientes condiciones:

* instalación nueva, una nueva máquina, nunca instalada previamente con Interpool
* actualización, máquina previamente instalada con Interpool, con la misma versión
* actualización, máquina previamente instalada con Interpool, con una versión anterior.
  + - 1. Técnica

Manualmente o desarrollando programas, para validar la condición de la máquina destino (nueva, nunca instalado, misma versión, versión anterior ya instalada).

Realizar la instalación.

Ejecutar un conjunto de pruebas funcionales ya implementadas para la Prueba de Funcionalidad.

* + - 1. Criterio de aceptación

Las pruebas de funcionalidad de Interpool se ejecutan exitosamente sin fallas.

* + - 1. Consideraciones especiales

¿Qué operaciones se deben ser seleccionar para realizar una prueba confiable de que la aplicación Interpool ha sido exitosamente instalada sin dejar fuera ningún componente importante?

* + 1. Prueba de Documentos

Estas pruebas no son necesarias ya que no existen requerimientos en nuestro sistema que impliquen la realización de documentos.

* 1. Herramientas
* Lenguaje: Se deberá utilizar la plataforma .NET y se desarrollara en lenguaje C#.
* Herramienta de implementación: Visual Studio 2010.
* Manejador de bases de datos SQL 2008 Express.
* Emulador de Windows Phone 7.
* Repositorio de documentos: TortoiseSVN.

1. Recursos

En esta sección se presentan los recursos recomendados para el proyecto Interpool, sus principales responsabilidades y su conocimiento o habilidades.

* 1. Roles

En la tabla a continuación se muestra la composición de personal para el proyecto Interpool en el área Verificación del Software.

|  |  |  |
| --- | --- | --- |
| **Rol** | **Cantidad mínima de recursos recomendada** | **Responsabilidades** |
| Responsable de verificación | 1 | * Identifica, prioriza e implementa los casos de prueba. * Genera el Plan de Verificación. * Genera el Modelo de Prueba. * Evalúa el esfuerzo necesario para verificar. * Proporciona la dirección técnica. * Adquiere los recursos apropiados. * Proporciona informes sobre la verificación. |
| Asistente de verificación | 4 | * Ejecuta las pruebas * Registra los resultados de las pruebas. * Recuperar el software de errores. * Documenta los pedidos de cambio. |
| Administrador de Base de Datos | 1 | * Realiza la gestión y mantenimiento del entorno de los datos (base de datos) de prueba y los recursos. * Administra la base de datos de prueba. |

* 1. Sistema

En la siguiente tabla se establecen los recursos de sistema necesarios para realizar la verificación.

|  |  |
| --- | --- |
| **Recurso** | **Nombre/Tipo** |
| Servidor de base de datos | SQL 2008 server |
| Red o subred | Azure |
| Nombre del servidor | No definido |
| Nombre de la base de datos | No definido |
| PC Cliente para pruebas | No definido |
| Requerimientos especiales | No definido |
| Repositorio de pruebas | CodePlex |

1. Hitos del proyecto de Verificación

La verificación del Interpool debe incorporar actividades de prueba para cada verificación identificada en las secciones anteriores. Se deben identificar los hitos del proyecto de verificación separados para comunicar los logros de estado de proyecto.

|  |  |  |  |
| --- | --- | --- | --- |
| **Actividad que determina el hito** | **Esfuerzo** | **Fecha de comienzo** | **Fecha de finalización** |
| Planificar la verificación | 15 hs | 16/08/2010 | 22/08/2010 |
| Elaborar casos de prueba | 8 hs | 01/09/2010 | 05/09/2010 |
| **Fase 1 - Iteración II - 3 y 4** |  |  |  |
| Ajuste y control de verificación | 10 hs | 30/08/2010 | 05/09/2010 |
| Ejecutar la verificación | 5-10 hs | 03/09/2010 | 05/09/2010 |
| Evaluar la verificación |  |  |  |
| **Fase 2 - Iteración I - 5 y 6** |  |  |  |
| Elaborar casos de prueba | 15 hs | 13/09/2009 | 19/09/2010 |
| Ajuste y control de verificación | 15 hs | 13/09/2010 | 19/09/2010 |
| Ejecutar la verificación | 10 hs | 06/09/2010 | 12/09/2010 |
| Evaluar la verificación | 10 hs | 06/09/2010 | 12/09/2010 |
| **Fase 2 - Iteración II – 7 y 8** |  |  |  |
| Elaborar casos de prueba | 15 hs | 27/09/2010 | 03/10/2010 |
| Ajuste y control de verificación | 10 hs | 27/09/2010 | 03/10/2010 |
| Ejecutar la verificación | 10 hs | 20/09/2010 | 26/09/2010 |
| Evaluar la verificación | 10 hs | 20/09/2010 | 26/09/2010 |
| **Fase 3 –Iteración I – 9 y 10** |  |  |  |
| Elaborar casos de prueba | 10 hs | 11/10/2010 | 17/10/2010 |
| Ajuste y control de verificación | 15 hs | 04/10/2010 | 17/10/2010 |
| Ejecutar la verificación | 10-15 hs | 04/10/2010 | 17/10/2010 |
| Evaluar la verificación | 15-20 hs | 04/10/2010 | 10/10/2010 |
| **Fase 3 - Iteración II – 11 y 12** |  |  |  |
| Elaborar casos de prueba | 10 hs | 25/10/2010 | 31/10/2010 |
| Ajuste y control de verificación | 5-10 hs | 18/10/2010 | 31/10/2010 |
| Ejecutar la verificación | 15-20 hs | 18/10/2010 | 25/10/2010 |
| Evaluar la verificación | 15 hs | 18/10/2010 | 31/10/2010 |
| **Fase 4 - Iteración I – 13** |  |  |  |
| Ajuste y control de verificación | 2 hs | 01/11/2010 | 07/11/2010 |
| Ejecutar la verificación | 20 hs | 01/11/2010 | 07/11/2010 |
| Evaluar la verificación | 15-20 hs | 01/11/2010 | 07/11/2010 |
| **Fase 4 - Iteración II – 14** |  |  |  |
| Ajuste y control de verificación |  |  |  |
| Ejecutar la verificación |  |  |  |
| Evaluar la verificación | 15-20 hs | 08/11/2010 | 14/11/2010 |

1. Entregables
   1. Modelo de Casos de Prueba

|  |  |
| --- | --- |
| Documento | **Modelo de Casos de Prueba** |
| Creado por | El responsable de verificación, Alejandro García. |
| Para quien | Es la guía para realizar las pruebas del sistema y lo usarán los asistentes de verificación y el responsable de verificación cuando se ejecuten las pruebas del sistema. |
| Fecha de liberación | Será liberado el 06/09/2010. |

* 1. Informes de Verificación

|  |  |
| --- | --- |
| Documento | Se genera un documento **Informe de Verificación** **Unitaria** por cada prueba unitaria que se realice al sistema. |
| Creado por | Las personas que ejecutan las pruebas. |
| Para quien | Es el retorno para los implementadores de la tarea de verificación, que detalla los errores encontrados para que puedan ser corregidos. |
| Fecha de liberación | Será liberado luego de cada verificación unitaria.  Aún no se han definido las fechas de la liberación. |

|  |  |
| --- | --- |
| Documento | Se genera un documento **Informe Consolidación** por cada consolidación que se realice al sistema. |
| Creado por | Las personas que ejecutan las pruebas. |
| Para quien | Es el retorno para los implementadores de la tarea de consolidación, que detalla los errores encontrados para que puedan ser corregidos. |
| Fecha de liberación | Será liberado luego de cada consolidación.  Aún no se han definido las fechas de la liberación. |

|  |  |
| --- | --- |
| Documento | Se genera un documento **Informe de Verificación** **de Integración** por cada prueba de integración que se realice al sistema. |
| Creado por | Las personas que ejecutan las pruebas. |
| Para quien | Es el retorno para los implementadores de la tarea de verificación, que detalla los errores encontrados para que puedan ser corregidos. |
| Fecha de liberación | Será liberado luego de cada verificación de integración. Aún no se han definido las fechas de la liberación. |

|  |  |
| --- | --- |
| Documento | Se genera un documento **Informe de Verificación** **de Sistema** por cada prueba de sistema que se realice. |
| Creado por | Las personas que ejecutan las pruebas. |
| Para quien | Es el retorno para los implementadores de la tarea de verificación, que detalla los errores encontrados para que puedan ser corregidos. |
| Fecha de liberación | Será liberado luego de cada verificación de sistema. Aún no se han definido las fechas de la liberación. |

* 1. Evaluación de la verificación

|  |  |
| --- | --- |
| Documento | Se genera un documento **Evaluación de la verificación** por cada prueba que se realice al sistema. Este documento contiene las fallas encontradas en el sistema, la cobertura de la verificación realizada y el estado del sistema. |
| Creado por | El Responsable de verificación, que toma como fuente de su trabajo los Informes de verificación. |
| Para quien | Es el resumen de la tarea de verificación y es el retorno para todo el equipo de trabajo del estado del sistema. |
| Fecha de liberación | Será liberado luego de cada verificación, unitaria, de integración y de sistema.  Aún no se han definido las fechas de la liberación. |

* 1. Informe final de verificación

|  |  |
| --- | --- |
| Documento | El documento **Informe final de verificación** es el resumen de la verificación final del sistema antes de que sea liberado al entorno del usuario. |
| Creado por | El Responsable de verificación, que toma como fuente de su trabajo los Informes de verificación. |
| Para quien | Indica el estado del sistema. |
| Fecha de liberación | Será liberado luego de la verificación final del sistema. |

1. Dependencias
   1. Dependencia de personal

Para la construcción y ejecución de pruebas del prototipo se estima que se necesitarán entre 2 o 3 personas (el responsable de verificación y 1 o 2 asistentes). Una vez comenzado la construcción del proyecto se necesitará de la colaboración de todo el equipo de Verificación.

* 1. Dependencia de software

El software a verificar debe haber sido probado por el equipo de desarrolladores mediante pruebas unitarias y de integración. El software debe ser entregado en fecha disponible para verificar.

* 1. Dependencia de hardware

Se necesitarán equipos que puedan soportar las tecnologías que se utilizarán.

* 1. Dependencia de datos y base de datos de prueba

Se necesitará una base de datos con una cantidad de datos similar a la que el cliente manejará habitualmente.

1. Riesgos
   1. Planificación

El mayor riesgo es cometer errores en la planificación y que esto impacte en la calidad y cantidad de las pruebas a realizar. Por otro lado ante demoras en la liberación de las distintas versiones a verificar también se podría alterar el cumplimiento de la planificación, lo que podría impactar de forma muy grave.

* 1. Técnico

El mayor riesgo técnico hasta el momento es no contar con el hardware sobre el que correrá la aplicación.

* 1. Gestión

Para mitigar los riesgos de las desviaciones en el calendario de verificación, se hizo una sobreestimación del tiempo dedicado a cada tarea, para contar con un tiempo extra en caso de ser necesario.

1. Apéndice
   1. Niveles de gravedad de error

En muchas actividades del proceso de verificación se deben clasificar los errores según su nivel de gravedad. Se asigna un nivel de gravedad a los errores para poder capturar de alguna manera su impacto en el sistema. Además para poder evaluar la verificación y el sistema.

A continuación se da una sugerencia de cuatro niveles diferentes de gravedad de error:

* **Catastrófico**: un error cuya presencia impide el uso del sistema.
* **Crítico**: un error cuya presencia causa la pérdida de una funcionalidad crítica del sistema. Si no se corrige el sistema no satisfará las necesidades del cliente.
* **Marginal**: un error que causa un daño menor, produciendo pérdida de efectividad, pérdida de disponibilidad o degradación de una funcionalidad que no se realiza fácilmente de otra manera.
* **Menor**: un error que no causa perjuicio al sistema, pero que requiere mantenimiento o reparación. No causa pérdida de funcionalidades que no se puedan realizar de otra manera.
  1. Niveles de aceptación para los elementos verificados

Se debe establecer un nivel de aceptación para los elementos verificados para poder establecer el estado en el que se encuentra el proyecto.

En esta sección defina niveles de aceptación y los criterios de pertenencia a cada nivel.

Como ejemplo de niveles de aceptación:

* **No aprobado**: el elemento verificado tiene errores catastróficos (uno o varios) que impiden su uso o tiene errores críticos (uno o varios) que hacen que el elemento verificado no sea confiable. El usuario no puede depender de él para realizar el trabajo.
* **Aprobado con Observaciones**: el elemento verificado no tiene errores catastróficos, ni errores críticos, pero tiene errores marginales (uno o varios) que hacen que el elemento de software se degrade en algunas situaciones.
* **Aprobado**: el elemento verificado no tiene errores o tiene errores menores que no afectan el normal funcionamiento del elemento.