EMSYS
Plan de Verificación y Validación
Versión 5.1

Historia de revisiones

Fecha	Versión	Descripción	Autor
20/08/2016	1.1	Versión inicial	Camila Serena
27/08/2016	2.1	Versión final	Camila Serena
28/08/2016	2.2	Revisión y correcciones menores	Bruno Amaral, Camila Serena
16/09/2016	5.1	Correcciones al plan	Camila Serena

Contenido

1. Introducción

1.1. Propósito

Este Plan de Verificación para el proyecto EMSYS tiene 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.

1.2. Punto de partida

El proyecto consiste en el desarrollo de una aplicación para tablets Android con el objetivo de gestión de eventos, permitiendo su uso tanto online como offline.

Se caracteriza como una aplicación cliente-servidor, donde la parte servidor consiste en un web service que expone métodos para la comunicación con la aplicación móvil y se comunica con una base de datos para obtener y actualizar los datos utilizados en la aplicación móvil. Por otra parte el lado cliente, consiste en la aplicación móvil con funcionalidades para la creación y gestión de eventos.

Este proyecto nace por la necesidad de aumentar la dinámica de la gestión de eventos para el sistema EMSYS, ya desarrollado por SONDA. EMSYS es un sistema de gestión de eventos que integra diferentes sistemas externos, con el fin de disminuir el tiempo de respuesta en cada una de las distintas etapas por las que transcurre un evento.

1.3. Alcance

Para la verificación en este proyecto, se seguirán las siguientes fases:

- Unitaria:
  - Las pruebas unitarias se realizan sobre módulos individuales de código, para verificar el correcto funcionamiento de los mismos por separado. Se realizarán pruebas de caja negra, con criterios de clase de equivalencia y/o valores límites, para complementar luego con caja blanca, en los módulos con mayor complejidad, utilizando un criterio de recubrimiento de sentencias. Estas pruebas son diseñadas y desarrolladas por los implementadores involucrados. Para el registro de las mismas se utilizará una planilla definida por la responsable de VER, quien debe realizar una revisión semanal de lo registrado y de ser necesario exigir que se realicen más pruebas.

- Integración:
  - Las pruebas de integración refieren a las pruebas de todos los módulos unitarios que componen un proceso. Consiste en realizar pruebas tal que verifiquen que un gran conjunto de partes de software funcionan correctamente juntas. Para este proyecto se irán integrando los módulos a medida que se encuentren disponibles, trabajando en conjunto con el administrador, para realizar una planificación de la iteración tal que el orden de implementación sea con la integración de los componentes en mente (esto se incluye en *Plan de integración de la iteración*) y permita realizar las pruebas correspondiente utilizando la menor cantidad de drivers/stubs necesarios.
  -
  -

    - Estas pruebas serán diseñadas y realizadas por el equipo de desarrollo. Se entregará una planilla para el registro de las mismas.

- Sistema:
    - Esta fase contiene una variedad de pruebas que se llevarán a cabo para verificar que se cumplen todos los requisitos, funcionales y no-funcionales, que se encuentran especificados en el Documento de Requisitos. Se realizan los siguientes tipos de pruebas:

    - Pruebas de Función:
        - Verifican que el sistema integrado realiza los requerimientos funcionales especificados en Documento de Requerimientos. Se utilizará la técnica de desarrollo de casos de prueba a partir de casos de uso.
        - La responsable de VER será la encargada de diseñar y asignar dentro del equipo de verificación las pruebas a realizar.
    - Desempeño:
        - Son las pruebas que verifican que el sistema cumple con los requerimientos no funcionales especificados en Documento de Requerimientos. Se describen, con más detalle, los tipos de prueba de esta categoría en la sección 3 de este documento.
        - La responsable de VER será la encargada de diseñar y asignar dentro del equipo de verificación las pruebas a realizar. Se realizan en conjunto con el encargado de SQA.

    Antes de comenzar con las pruebas específicas, se realizan pruebas de humo al sistema para verificar que es estado del mismo es el mínimo indispensable para realizar pruebas más profundas. De no encontrarse apto se exigirá a los desarrolladores que realicen pruebas más exhaustivas y hasta dejar el sistema en un nivel aceptable.
- Aceptación:
    - Busca validar con el cliente que el software se comporta de la manera acordada. Se basan en los criterios de aceptación acordados con el cliente, que se encuentran en el documento de requisitos.
    - Participan la responsable de verificación, el responsable de SQA, cliente y administrador.

Las características destacables que se buscarán verificar y validar principalmente serán la usabilidad de la aplicación como los tiempos de respuesta en distintos entornos de uso. A su vez se buscará garantizar la integración correcta con agentes externos. Al contrario, no se buscará verificar la respuesta de nuestro sistema ante fallas en sistemas externos, por ejemplo el mal uso de la base de datos por otro actor externo al sistema.

Riesgos identificados que afectan al plan de verificación y validación:

- Atraso en el desarrollo de los módulos afecta directamente en el cronograma de pruebas.
- Aprendizaje de herramientas para pruebas, ya sea para unitarias por los implementadores, como las de cubrimiento de código para pruebas no funcionales por parte del equipo de verificación y las necesarias para la verificación de los requisitos no funcionales (uso de VM, para simulación de carga, entre otros).

Una restricción a tener en cuenta en el plan de proyecto, es la exigencia de un porcentaje mínimo de cubrimiento de código, que se encuentra aún por definir.

1.4. Identificación del proyecto

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

- Plantilla *Plan de Verificación y Validación* encontrada en la página de MUM.

- Documento Especificación *de requisitos.*
- Documento *Modelo de casos de uso.*
- Documentos de Planificación y Verificación de años anteriores extraídos de la memoria del curso.
- Presentación de V&V del curso Introducción a la Ingeniería de Software.
- Libro *Software Engineering*, Sommerville.

1.5. Estrategia de evolución del Plan

La responsable de monitorear el Plan de V&V será la responsable de verificación, realizando revisiones periódicas, al final de cada iteración. En las primeras iteraciones, la frecuencia de revisión será más asidua considerando que se está acordando el alcance del proyecto con el cliente y los documentos de requisitos y el plan de proyecto no se encuentran estables.

La evaluación y aprobación de cambios al plan serán acordados entre el responsable de verificación y el administrador, ya que los cambios en este pueden impactar directamente en el plan de proyecto. Además, en caso de ser necesario, el responsable debe consultar con los asistentes de verificación.

Para la comunicación de cambios se utilizarán las herramientas acordadas para la comunicación del proyecto.

2. Requerimientos para verificar

En la lista a continuación se presentan los elementos, casos de uso, requisitos funcionales y no funcionales, más importantes que serán verificados. Los requisitos completos a verificar se encontraran en el documento de Especificación de requisitos.

- Autenticación de usuarios.
- Creación y gestión de eventos, con archivos multimedia.
- Uso en modo offline.
- Geolocalización de eventos.
- Notificaciones en tiempo real.
- Consumo de web service externo.
- Avisos de parte de la base de datos.

3. Estrategia de Verificación

Se seguirán las fases de prueba explicitadas en el punto 1.3, debiéndose generar, en cada una, la documentación necesaria por parte del involucrado. A continuación se profundiza en los distintos tipos de pruebas a realizar.

3.1. Tipos de pruebas

3.1.1. Prueba de integridad de los datos y la base de datos

3.1.1.1. *Objetivo de la prueba*

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

3.1.1.2. *Técnica*

Se realiza la invocación de cada método o proceso de acceso a la base de datos con datos válidos y no válidos. Luego se inspecciona la base de datos verificando que los resultados obtenidos son los esperados y la misma no se encuentra en estado inconsistente.

3.1.1.3. *Criterio de aceptación*

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

3.1.1.4. *Consideraciones especiales*

- Se debe planificar con los especialistas técnicos el acceso a la base de datos del servidor, para las pruebas de caja negra en el servidor levantado.

3.1.2. Prueba de Funcionalidad

La prueba de funcionalidad se enfoca en los requisitos definidos con el cliente para verificar que se corresponden directamente a casos de usos o funciones y reglas del negocio implementadas en el producto. 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 interfaz de usuario y analizar los resultados obtenidos.

3.1.2.1. *Objetivo de la prueba*

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

3.1.2.2. *Técnica*

Ejecutar cada caso de prueba diseñado a partir de los casos de uso 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.

Los casos de prueba se encuentran estarán referenciados en el *Plan de verificación de la iteración.*

3.1.2.3. *Criterio de aceptación*

Todas las pruebas planificadas se realizaron. Todos los defectos encontrados han sido debidamente identificados y corregidos, hasta lograr pasar todas las pruebas correctamente.

3.1.2.4. *Consideraciones especiales*

No se definieron consideraciones especiales

3.1.3. Prueba de Ciclo del Negocio

Esta prueba debe simular las actividades realizadas en el proyecto en el tiempo. Se apunta a probar casos de uso seguidos donde un error en uno influye en otro. En el caso de nuestro producto, las pruebas de negocio que se van a realizar son de los casos de uso encadenados que corresponden al ciclo de vida de un evento (dependiendo de lo definido en el alcance).

3.1.3.1. *Objetivo de la prueba*

Asegurar que la aplicación funciona de acuerdo a los requerimientos del negocio. En particular, se buscará verificar el correcto pasaje entre los estados posibles de un evento con la intervención de los distintos usuarios involucrados.

3.1.3.2. *Técnica*

La prueba debe simular ciclos de negocios realizando lo siguiente:

Las pruebas de funcionalidad se deben modificar para aumentar la cantidad de veces que se ejecuta cada función, simulando varios usuarios diferentes en un período determinado.

Todas las funciones sensibles a la fecha se deben ejecutar con fechas válidas y no válidas o períodos de tiempo válidos y no válidos.

Para cada prueba realizada 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.

3.1.3.3. *Criterio de aceptación*

Todas las pruebas planificadas se realizaron. Todos los defectos encontrados han sido debidamente identificados y corregidos, hasta lograr pasar todas las pruebas correctamente.

3.1.3.4. *Consideraciones especiales*

No se definieron consideraciones especiales.

3.1.4. Prueba de Interfaz de Usuario

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

3.1.4.1. *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 menúes, tamaño, posición, estado funcionen de acuerdo a los estándares; la usabilidad y amigabilidad de la interfaz sea adecuada para su uso por personal no técnico.

3.1.4.2. *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.

3.1.4.3. *Criterio de aceptación*

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

3.1.4.4. *Consideraciones especiales*

- El diseño de la interfaz será desarrollado para uso de personal sin experiencia previa con uso de tablets necesariamente.

3.1.5. Prueba de Performance

En esta prueba se miden y evalúan los tiempos de respuesta, los tiempos de transacción y otros requisitos sensibles al tiempo. El objetivo de la prueba es verificar que se logren los requisitos de performance.

Para el cliente, es importante que la aplicación responda en tiempo real con el uso de notificaciones en el producto. Para realizar este tipo de pruebas, previamente definiremos métricas para medir lo solicitado y se acordaran con el cliente antes del término de la fase inicial.

3.1.5.1. *Objetivo de la prueba*

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

- condiciones de trabajo normales conocidas. (A definir próximamente)
- peores casos de condiciones de trabajo conocidas. (A definir próximamente)

3.1.5.2. *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).

3.1.5.3. *Criterio de aceptación*

Con una transacción o un usuario: Éxito completo de la prueba sin fallas y dentro del tiempo esperado o requerido definido con el cliente. (Ver criterios de aceptación en *Documento de requisitos*)

Con múltiples transacciones y varios usuarios: Éxito completo de la prueba sin fallas y dentro de un tiempo aceptable definido con el cliente. (Ver criterios de aceptación en *Documento de requisitos*)

3.1.5.4. *Consideraciones especiales*

La prueba de performance se debe realizar en una máquina dedicada (ver herramientas) para permitir control total y medición exacta y las bases de datos usadas para las pruebas de performance deben tener un tamaño similar a las reales.

3.1.6. 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 sensibles al tiempo.

El cliente específico que aproximadamente unos doscientos usuarios utilizarán la aplicación simultáneamente, este número se utilizará como punto de partida para los pruebas de carga. Igual que en el punto anterior, se deberán definir y aprobar métricas de tiempos aceptables.

3.1.6.1. *Objetivo de la prueba*

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

3.1.6.2. *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.

3.1.6.3. *Criterio de aceptación*

Para múltiples transacciones y múltiples usuarios: Realización exitosa de las pruebas sin fallas y dentro del tiempo aceptable definido con el cliente.

3.1.6.4. *Consideraciones especiales*

- La prueba de carga debe realizarse en una máquina dedicada (ver herramientas) para tener control total y exactitud de mediciones.
- Las bases de datos usadas para la prueba deben tener un tamaño similar a las reales.

3.1.7. Prueba de Esfuerzo (stress, competencia por recursos, bajos recursos)

La prueba de esfuerzo es 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.

3.1.7.1. *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.

3.1.7.2. *Técnica*

A definir en próximas entregas.

3.1.7.3. *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.

3.1.7.4. *Consideraciones especiales*

No se realizarán sobre el sistema pruebas de esfuerzo.

3.1.8. 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.

3.1.8.1. *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.

3.1.8.2. *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.

3.1.8.3. *Criterio de aceptación*

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

3.1.8.4. *Consideraciones especiales*

- La aplicación a desarrollar cuenta con la funcionalidad de adjuntar datos multimedia, que puede ser un gran punto de falla en condiciones de muchos usuarios concurrentes.
- Se cuenta con dos bases de datos distintas, una del lado cliente y otra en el servidor.
- Definir con el cliente el periodo de tiempo que se considera aceptable para condiciones de gran volumen de datos (a definir próximamente).

3.1.9. 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. En este caso los usuarios de la aplicación, cuentan con roles específicos que le otorgan permisos en cuanto a las funcionalidades que pueden acceder.

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. Para este punto, en nuestra realidad, debemos de verificar que los usuarios puedan ingresar teniendo las credenciales adecuadas, y también el cliente específico que agentes externos a la aplicación no puedan consumir los servicios que expone por WB el servidor. El cliente además hizo hincapié en la seguridad basada en la confidencialidad de los datos enviados entre el usuario y el servidor, por lo que también se verificará las comunicaciones seguras.

3.1.9.1. *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 y se utilicen canales de comunicación seguros.

3.1.9.2. *Técnica*

Seguridad en el ámbito de aplicación:

- Identificar y hacer una lista de cada tipo de usuario y las funciones y datos sobre las que cada tipo tiene permiso.
- Crear pruebas para cada tipo de usuario y verificar cada permiso creando operaciones específicas para cada tipo de usuario.
- Modificar el tipo de usuario y volver a ejecutar las pruebas para los mismos usuarios. En cada caso, verificar que las funciones o datos adicionales están correctamente disponibles o son denegados.

Acceso en el ámbito de sistema:

- Intentar acceder a los servicios del servidor sin utilizar la aplicación.

- Utilizar herramientas para el análisis de red (ej. Wireshark), para verificar las comunicaciones seguras de punta a punta.

3.1.9.3. *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.

3.1.9.4. *Consideraciones especiales*

No se definieron consideraciones especiales.

3.1.10. 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.

3.1.10.1. *Objetivo de la prueba*

Verificar que los procesos de recuperación (manual o automáticos) recuperen apropiadamente la base de datos, aplicaciones y sistema a un estado conocido y deseado. En la prueba se incluyen los siguientes tipos de condiciones:

- Cierre abrupto de la aplicación cliente
- Sin conexión a internet del lado cliente
- Sin GPS del lado cliente
- Sin conexión a internet del lado servidor
- Término abrupto del servidor (interrupcion de energia)
- Ciclos incompletos (envío de datos interrumpidos, procesos de sincronización de datos interrumpidos)
- Elementos de datos en la base de datos inválidos o corruptos.

3.1.10.2. *Técnica*

Se deben usar las pruebas creadas para probar Funcionalidad y Ciclos de negocio para crear una serie de operaciones. Una vez logrado el punto de comienzo deseado, se deben realizar o simular las siguientes acciones, individualmente:

- Interrumpir la aplicación cliente.
- 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.

Una vez que se lograron o simularon estas condiciones, se deben invocar los procedimientos de recuperación.

- Las pruebas de ciclos incompletos utilizan la misma técnica excepto que los procesos de bases de datos deben ser abortados a sí mismos o terminados prematuramente.
- Las últimas dos pruebas requieren que se logre un estado conocido de la base de datos. Se deben corromper manualmente campos de la base de datos, punteros y claves trabajando directamente sobre la base de datos (utilizando herramientas para la base de datos). Se deben ejecutar las pruebas de Funcionalidad y Ciclo de negocio y verificar que los ciclos se completen.

3.1.10.3. *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.

3.1.10.4. *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.

3.1.11. Prueba de Configuración

La Prueba de Configuración verifica el funcionamiento del software con diferentes configuraciones de software y hardware.

Existe un requisito del cliente de que ciertos parámetros del sistema puedan ser configurados manualmente, por ejemplo el dominio del servidor.

3.1.11.1. *Objetivo de la prueba*

Verificar que el software funcione apropiadamente en las configuraciones requeridas de hardware y software.

3.1.11.2. *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.

3.1.11.3. *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.

3.1.11.4. *Consideraciones especiales*

No se definieron consideraciones especiales.

3.1.12. 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.

3.1.12.1. *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, un nuevo dispositivo (tablet), nunca instalada previamente con EMSYS
- actualización, un dispositivo (tablet) previamente instalado con EMSYS, con la misma versión
- actualización, un dispositivo (tablet) previamente instalado con EMSYS, con una versión anterior.

3.1.12.2. *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.

3.1.12.3. *Criterio de aceptación*

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

3.1.12.4. *Consideraciones especiales*

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

No se realizaran este tipo de pruebas para el sistema. Se le entregará al cliente el producto en una máquina virtual.

3.1.13. Prueba de Documentos

La Prueba de Documentos debe asegurar que los documentos relacionados al software que se generen en el proceso sean correctos, consistentes y entendible. Se incluyen como documentos los Materiales para Soporte al Usuario, Documentación Técnica, Ayuda en Línea y todo tipo de documento que forme parte del paquete de software.

3.1.13.1. *Objetivo de la prueba*

Verificar que el documento objeto de prueba sea:

- Correcto, esto es, que cumpla con el formato y organización para el documento establecido en el proyecto.
- Consistente, esto es, que el contenido del documento sea fiel a lo que hace referencia. Si el documento es Documentación de Usuario, que la explicación de un procedimiento sea exactamente como se realiza el procedimiento en el software, si se muestran pantallas que sean las correctas.
- Entendible, esto es, que al leer el documento se entienda correctamente lo que expresa y sin ambigüedades, además que sea fácil de leer.

3.1.13.2. *Técnica*

Para verificar que el documento es correcto se debe comparar con el estándar definido si existe o con las pautas de documentación y ver que el documento cumple con ellas.

Para verificar que el documento es consistente se debe ejecutar el programa siguiendo como guía el documento *Manual para el Usuario* y comprobar que lo que se explica es exactamente lo que se ejecuta en el programa. En caso de Documentación Técnica se debe revisar el código al cual corresponde la documentación y comprobar que dicha describe el código.

Para verificar que el documento es entendible, debe comprobar que se entiende correctamente, que no tiene ambigüedades y que sea fácil de leer.

3.1.13.3. *Criterio de aceptación*

El documento expresa exactamente lo que debe expresar, no hay diferencias entre lo que está escrito y el objeto de la descripción (operación de software, código de programa, decisiones técnicas) y se entiende fácilmente.

3.1.13.4. *Consideraciones especiales*

Es importante que los documentos a verificar por los responsables de SQA y verificación se encuentren disponibles los sábados antes de las 18hs, para poder ser revisados y devueltos para corrección de ser necesario, a tiempo.

3.2. Herramientas

Las herramientas utilizadas son las siguientes

- Gestión de proyecto: Agilefant
- Gestión de material: Google Drive
- Repositorio del sistema: Github
- Gestión de DBMS: SSMS para SQLExpress
- Seguimiento de errores: Github
- Pruebas automatizadas: NUnit (Lógica de negocio) , JUnit (Interfaz gráfica)
- Cubrimiento de código: dotCover de JetBrains para backend y Android Studio IDE para el frontend.
- Monitoreo de cubrimiento de pruebas: Google Drive
- BugTracking: Trello
- Maquina dedicada a la verificación:
  Intel Core i7, RAM 8Gb, disco 100Gb SSD y SO Win10.

4. Recursos

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

4.1. Roles

En la tabla a continuación se muestra la composición de personal para el proyecto EMSYS 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 cambios.
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.

4.2. Sistema

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

Recurso	Nombre/Tipo
Servidor de base de datos	A definir. (SQLExpress y/o MySQL)
Tablet Cliente para pruebas	Android 4 o superior
Requerimientos especiales	Tablet Samsung Galaxy Tab

5. Hitos del proyecto de Verificación

La verificación del EMSYS 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
Fase inicial			
Planificar la verificación	20	21/8	11/9
Fase de elaboración			
Iteración 1			
Elaborar casos de prueba	10	12/9	17/9
Ajuste y control de verificación	5	19/9	24/9
Iteración 2			
Elaborar casos de prueba	10	26/9	1/10

Ajuste y control de verificación	5	26/9	1/10
Ejecutar la verificación	12	28/9	5/10
Ejecutar verificación relevante a la arquitectura (pruebas no funcionales)	20	1/0	5/10
Evaluar la verificación	6	3/10	8/10
Fase de construcción			
Iteración 2			
Elaborar casos de prueba	6	10/10	16/10
Ajuste y control de verificación	5	10/10	16/10
Ejecutar la verificación	12	12/10	19/10
Evaluar la verificación	6	17/10	22/10
Iteración 2			
Ajuste y control de verificación	5	24/10	29/10
Ejecutar la verificación	15	24/10	2/11
Evaluar la verificación	8	26/10	5/10
Fase de implantación			
Ajuste y control de verificación	5	7/11	12/11
Ejecutar la verificación	15	7/11	16/11
Evaluar la verificación	8	9/11	19/11

6. Entregables

6.1. Modelo de Casos de Prueba

Documento	**Modelo de Casos de Prueba**
Creado por	El responsable de verificación, Camila Serena.
Para quién	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 al final de la semana 6.

6.2. Informes de Verificación

Documento	Se genera un documento **Informe de Verificación Unitaria** por cada prueba unitaria que se realice al sistema. Este documento será una plantilla que sigue las pautas dadas por la responsable de VER.
Creado por	Las personas que ejecutan las pruebas.
Para quién	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. Las liberaciones de este informe serán en las semanas 7, 9, 11 y 12.

Documento	Se genera un documento **Informe Consolidación** por cada consolidación que se realice al sistema.
Creado por	Las personas que ejecutan las pruebas.
Para quién	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 verificación unitaria. Las liberaciones de este informe serán en las semanas 7, 9, 11 y 12.

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 quién	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. Las liberaciones de este informe serán en las semanas 6, 8, 10 y 12.

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 quién	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. Las liberaciones de este informe serán en las semanas 7, 9, 11 y 12.

6.3. 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 quién	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.

6.4. 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 quién	Indica el estado del sistema.
Fecha de liberación	Será liberado luego de la verificación final del sistema.

7. Dependencias

En esta sección se detallan las dependencias, si existen, de las actividades de verificación respecto a otros elementos del sistema.

7.1. Dependencia de personal

El equipo de verificación está integrado por el encargado de verificación y cuatro asistentes de verificación. Los asistentes de verificación tienen otros roles asignados de mayor prioridad pero de todas formas deberán contribuir a la verificación del sistema. Además cada integrante del equipo de verificación debe tener un conocimiento detallado del funcionamiento del sistema, del alcance, cronograma, modelo de casos de uso y modelo de pruebas

7.2. Dependencias de Software

Se requiere una máquina virtual con todo lo necesario para realizar las pruebas. Debe contener el mismo ambiente de desarrollo que poseen los implementadores y debe ser completamente funcional al ser entregada al equipo de verificación

7.3. Dependencia de hardware

- El cliente posee una tablet para la realización de pruebas del sistema. La misma puede ser utilizada en horario laboral con coordinación previa, hay que consultar si en algún caso excepcional se puede pedir para usarla fuera de las oficinas.
- El ambiente para las pruebas debe realizarse en una máquina dedicada que a su vez tienen requerimientos mínimos de hardware para funcionar y realizar simulaciones correctamente.

7.4. Dependencia de datos y base de datos de prueba

Se necesita definir los conjuntos de datos para cargar en forma inicial al sistema para poder realizar las pruebas en forma aceptable. Además, se necesitará acordar un horario especial en el que se puedan realizar pruebas con modificaciones en la base de datos y no exista conflicto con las pruebas que puedan realizar los implementadores.

8. Riesgos

En esta sección se detallan los riesgos detectados que puedan afectar la normal realización de las tareas de verificación.

8.1. Planificación

- Planificar instancias de verificación y que se libere tarde ese módulo para la prueba.
- La planificación del orden de implementación de los módulos influye directamente en el desarrollo de las pruebas de integración. Se deberá planificar inteligentemente para mitigar el riesgo de una integración laboriosa.
- Los asistentes no tengan disponibilidad por tener que realizar tareas de su área prioritarias.

9. Apéndice

9.1. 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.
- **Menor**: 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.
- **Mejora**: 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.

9.2. Niveles de aceptación para lo 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 se define los 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.