K2BIM
Plan de Verificación y Validación
Versión 1.2

Historia de revisiones

Fecha	Versión	Descripción	Autor
22/08/2009	1.0	Creación del documento.	Alan Descoins
05/09/2009	1.1	Modificado y ampliado en base al nuevo plan de proyecto.	Alan Descoins
06/09/2009	1.2	Ajustadas las fechas.	Alan Descoins

Contenido

1. Introducción

1.1 Propósito

Este Plan de Verificación para el proyecto K2BIM 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.

1.2 Punto de partida

El proyecto a ser desarrollado es un aplicativo para la empresa ARTech, que resultará en un adicional de un producto ya existente, K2B. Esta aplicación está destinada a generar programas de planificación de recursos empresariales, *Enterprise Resource Planner (ERP)* que están orientados a la integración de la información existente dentro de una organización.
El sistema K2BIM asistirá en el proceso de implantación de K2B, permitiendo el registro de las distintas actividades de implantación y facilitando la parametrización de K2B con datos del cliente. Permitirá la comunicación entre el fabricante y el implantador y proveerá de un catálogo general de las distintas implantaciones de K2B, ofreciendo de esta manera la importación de datos de otro proyecto similar. Además, le ofrecerá al cliente la posibilidad de continuar la parametrización una vez finalizado el proceso.

K2BIM se desarrollará como prototipo de lo que ARTech visualiza como la herramienta final a construir, es decir, si bien la herramienta deberá ser completamente funcional al momento de finalizarse el proyecto, su utilización no será inmediata en entornos de producción.

El sistema K2BIM se divide en tres grandes módulos.

- **Módulo de Construcción del Árbol**
  Es el susbistema que encapsula la lógica de la construcción del árbol de proyecto, utilizado por el administrador al inicio del proyecto de implantación.

- **Módulo de Consulta y Utilización del Árbol**
  Es el subsistema con el cual interactúan implantadores, clientes y fabricantes. Tiene las funcionalidades necesarias para consultar el árbol de proyecto, navegar en él y disparar el uso de las funciones que están definidas. Para eso último puede redirigir al usuario a otra página (caso de la Wiki) o puede realizar una interacción con el módulo de integración con K2B para consumir servicios de K2B directamente.

- **Módulo de Integración con K2B**
  Es el módulo que encapsula la lógica de comunicación con los webservices que K2B ofrece para la integración. Decidimos agregar un módulo separado para este fin por motivos de modularidad.

Además, existe una interfaz web con los Javascripts que corren en el browser del usuario y

brindan la interacción con el sistema.

La verificación de estos módulos será prácticamente independiente, dado que existe cierto grado de aislamiento entre las funcionalidades de cada uno. Igualmente se realizará la prueba de integración, donde se verificará que todos funcionen en conjunto como es debido.

1.3 Alcance

El proceso de verificación del sistema K2BIM constará de las siguientes fases:

- Unitaria: se verifican los componentes del sistema en forma unitaria. Esta verificación será realizada por los implementadores, los cuales deberán verificar cada componente que construyen y elaborar los informes de verificación unitaria correspondiente. Estos últimos deben contener la cantidad de errores encontrados y los que fueron corregidos en cada iteración.
- Integración: se verifica que el sistema K2BIM funcione correctamente con los sistemas con los cuales debe integrarse. Específicamente, se verificará un correcto funcionamiento de la integración con el sistema K2B así como la integración con la Wiki.
- Funcional: se verifica que el sistema implementa todas las funcionalidades descritas en los requerimientos del cliente, siempre y cuando estos caigan dentro del alcance del producto pactado con el mismo.
- Aceptación: se verifica que el sistema cumple con lo que el cliente deseaba o pretendía. Es realizada por el cliente mismo o en su defecto el grupo de usuarios finales del producto.

No existen requisitos de de performance ni relacionados al desempeño, por lo tanto no se van a necesitar agregar pruebas de desempeño al alcance de este plan.

A continuación listamos algunos riesgos que se presentan:

- Agregar al producto requerimientos no pactados en el alcance del proyecto.
- No tener los requerimientos priorizados adecuadamente, lo que puede llevar a dedicar mayor esfuerzo en verificar requerimientos no críticos cuando se dejan otros más críticos menos probados.
- No verificar adecuadamente los requerimientos o no verificar a tiempo, esto podría afectar del diseño en adelante. Un plan de contingencia es una buena planificación de verificación.

1.4 Identificación del proyecto

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

- Acta de las reuniones con el cliente:
  - Acta de reunión del 13-08-2009 v1.2
  - Acta de reunión del 14-08-2009 v1.1
  - Acta de reunión del 17-08-2009 v1.5
  - Acta de reunión del 26-08-2009 v1.7
  - Acta de reunión del 01-09-2009 v1.2
- Documento de especificación de requerimientos v1.6

1.5 Estrategia de evolución del Plan

El responsable de monitorear el plan de verificación es la persona con el rol "Responsable de verificación" (Alan Descoins).
Los cambios al plan se podrán realizar cuando se necesiten. Sin embargo, al final de cada iteración el plan será revisado para determinar si se adecua al estado actual del proceso.
Los cambios en el plan serán comunicados a todos los integrantes del equipo a través de las vías de comunicación que se vienen utilizando. Es importante que dichos avisos queden **siempre** registrados como un mensaje en el grupo de Google que el equipo está utilizando, además de explicitados en el historial de versiones de este documento.

2. Requerimientos para verificar

Para la versión final del producto se verificarán todos los requerimientos funcionales y no funcionales presentados en el Documento de especificación de Requerimientos, que hayan caído dentro del alcance final pactado con el cliente.
En el transcurso del proceso se priorizarán requerimientos para verificar, que dependerán de la iteración y fase en la que se encuentra el producto.

3. Estrategia de Verificación

3.1 Tipos de pruebas

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

El objetivo es asegurar que los métodos y procesos de acceso a la base de datos funcionan correctamente y sin corromper datos.

Esta prueba no será ejecutada dado que todo el acceso a la base de datos se realiza a través de GeneXus, y por lo tanto el código que accede a la BD y hace chequeos pertinentes (como integridad referencial, etc.) es generado automáticamente.

3.1.2 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.

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

Técnica
Se ejecuta 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.

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

Consideraciones especiales
Identificar o describir aquellos elementos o problemas (internos o externos) que impactaron en la implementación y ejecución de las pruebas de funcionalidad.

Estas pruebas serán diseñadas y ejecutadas tomando en cuenta la importancia de las mismas para verificar la correcta implementación de los requerimientos y correcto funcionamiento del sistema.

3.1.3 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.

Objetivo de la prueba
Asegurar que la aplicación funciona de acuerdo a los requerimientos del negocio.

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.

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

Consideraciones especiales
Las fechas del sistema y eventos requieren actividades de soporte especiales. Se requieren las reglas del negocio para identificar apropiadamente los requerimientos y procedimientos a ser verificados.

Esta prueba será ejecutada, verificando que un ciclo de uso de K2BIM cumpla las expectativas.

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 muestren como se espera y conforme a los estándares establecidos por la empresa o de la industria.

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.

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.

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

Consideraciones especiales
No todas las propiedades de los objetos se pueden acceder.

Esta prueba será ejecutada pues el cliente hizo énfasis en lo importante que es la facilidad de uso del producto. Además, se debe tener en cuenta que el usuario final de K2BIM no tiene porqué tener conocimientos avanzados de computación.

3.1.5 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.

Esta prueba no se realizará pues no existen requerimientos relacionados al desempeño del sistema.

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 sensitivos al tiempo.

Esta prueba no se realizará pues no existen requerimientos relacionados al desempeño del sistema.

3.1.7 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.

Esta prueba no se realizará pues no existen requerimientos relacionados al desempeño del sistema.

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.

Esta prueba no se realizará pues no existen requerimientos relacionados al desempeño del sistema. En particular, no se tiene claro aún qué son "grandes volúmenes de datos" en el caso del sistema K2BIM.

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.
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.

Esta prueba no será ejecutada, ya que el cliente dejó claro que no necesitaba que se implemente ninguna característica concerniente a la seguridad (esto va a ser manejado con GxPortal). Nuestro sistema sí tendrá en consideración roles de usuarios para determinar qué se mostrará en la interfaz gráfica, pero no existen características de seguridad asociadas a los roles.

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.

Esta prueba no será ejecutada, ya que nuestro sistema no es un sistema crítico y no irá a producción cuando se finalice. Además, como se utiliza una base de datos para almacenar la información, tenemos otro grado de respaldo pues el manejador de base de datos MySQL provee mecanismos para asegurar integridad y consistencia de los datos.

3.1.11 Prueba de Configuración

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

Esta prueba no será ejecutada. El uso de GeneXus como herramienta de desarrollo no nos brinda libertad suficiente como para corregir errores de visualización del producto generado en distintos navegadores, entre otras limitaciones propias de la herramienta de desarrollo.

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.

Esta prueba no será ejecutada debido a que el sistema a construir es solo un prototipo que no entrará en producción. Por esto, no es de relevancia probar exhaustivamente funcionalidades relacionadas a la instalación del sistema.

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.

Esta prueba no será ejecutada. El cliente manifestó que no es de relevancia la documentación que generemos de la herramienta, dado que ellos no poseen aún documentación de K2B. Como en un futuro K2BIM se integrará como un módulo más de K2B, la documentación será realizada desde cero. Consecuentemente, no elaboraremos documentación de usuario.

3.2 Herramientas

Para seguimiento de errores utilizaremos la herramienta "Mantis Bug Tracker", disponible en el sitio http://www.deporplacita.com.uy/mantis/. Todos los errores encontrados durante las pruebas serán reportados ahi, para un correcto seguimiento de los mismos.

Para el desarrollo utilizaremos GeneXus X Evolution 1.

4. Recursos

En esta sección se presentan los recursos recomendados para el proyecto K2BIM, 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 K2BIM en el área Verificación del Software.

Rol	**Cantidad mínima de recursos recomendada**	**Responsabilidades**
Responsable de verificación (Alan Descoins)	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 (Juan Saavedra, Adrián Silveira, Guillermo Dotta, Diego Piriz)	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 (Emilio Penna)	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	MySQL 5.1
Red o subred	LAN
Sistema de seguimiento de versiones	Mantis Bug Tracker http://www.deporplacita.com.uy/mantis/

5. Hitos del proyecto de Verificación

La verificación del K2BIM 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 (horas totales por semana)	Fecha de comienzo	Fecha de finalización
Planificar la verificación	15	22/08/2009 (fines semana 2)	06/09/2009 (fin semana 4)
Elaborar casos de prueba (semana 5 + semanas pares)	20	07/09/2009 (inicio semana 5)	01/11/2009 (fin semana 12)
Ajuste y Control de Verificación	10	05/10/2009 (inicio semana 9)	11/10/2009 (fin semana 9)
Ejecutar la verificación (semanas impares)	30	21/09/2009 (inicio semana 7)	08/11/2009 (fines semana 13)
Evaluar la verificación (semanas impares)	10	21/09/2009 (inicio semana 7)	08/11/2009 (fines semana 13)

6. Entregables

6.1 Modelo de Casos de Prueba

Documento	**Modelo de Casos de Prueba**
Creado por	El Responsable de verificación, Alan Descoins.
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	La primera versión será liberada el 13/09/2009 (fin semana 5). Se perfeccionará durante 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.
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. • Elaboración, 2nda iteración: 11/10/2009 (fin semana 9). • Construcción, 1era iteración: 25/10/2009 (fin semana 11). • Construcción, 2nda iteración: 08/11/2009 (fin semana 13).

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. *A determinar fechas de 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. • Construcción, 1era iteración: 25/10/2009 (fin semana 11). • Construcción, 2nda iteración: 08/11/2009 (fin semana 13).

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. • Construcción, 2nda iteración: 08/11/2009 (fin semana 13).

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 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. • Elaboración, 1era iteración: 27/09/2009 (fin semana 7). • Elaboración, 2nda iteración: 11/10/2009 (fin semana 9). • Construcción, 1era iteración: 25/10/2009 (fin semana 11). • Construcción, 2nda iteración: 08/11/2009 (fin semana 13).

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 quien	Indica el estado del sistema.
Fecha de liberación	Será liberado luego de la verificación final del sistema. • Transición, 1era iteración: 15/11/2009 (fin semana 14).

7. Dependencias

7.1 Dependencia de personal

El equipo cuenta con cuatro asistentes de Verificación, si bien estas personas tienen asignados otros roles que van a afectar su disponibilidad, deberán trabajar con el responsable de Verificación en momentos de realizar la verificación de la integración y las pruebas de sistema.

7.2 Dependencia de software

El software a ser verificado debe tener una verificación previa por parte de los implementadores (como se describió anteriormente) en la cual se hayan realizado las pruebas unitarias y entregado los informes correspondientes.

7.3 Dependencia de hardware

No se han definido dependencias de hardware.

8. Riesgos

8.1 Planificación

Si un cronograma es muy ajustado un pequeño retraso en la liberación del software para verificar atrasa la verificación y por consiguiente las actividades que dependen de ésta, provocando un retraso en el cronograma de todo el proyecto.
Asimismo se necesita que los documentos de otras actividades del proyecto sean entregados en fecha, pues demoras en los mismos atrasarían el proceso de verificación.

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.
- **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.

9.2 Niveles de aceptación para lo elementos verificados

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.