Diseño de Sistemas

Unidad 7: Validación del Diseño - parte 1

Martín Agüero Pablo Sabatino

2019

UTN.BA
DPTO. INGENIERÍA EN SISTEMAS DE INFORMACIÓN
CÁTEDRA DISEÑO DE SISTEMAS

Agenda

Unidad 7:

- Verificación y validación de Software
- Objetivo de la construcción de casos de prueba.
- Pruebas Unitarias y de Integración.
- Validación y Prueba del Sistema.
- Concepto de Escenario.

Validación del Diseño

Introducción - Errores informáticos de la historia

El desorbitado Mars Climate en 1998

La "Mars Climate" se estrelló en Marte porque la NASA no tradujo kilómetros a millas

Los técnicos olvidaron convertir datos de navegación del sistema métrico decimal al inglés

"La gente a veces comete errores", dijo ayer Edward Weiler, director adjunto de la agencia estadounidense, que, pese a todo, logró colocar a seres humanos en la Luna hace ya tres décadas. "El problema más grave", añadió, "no fue ese error, sino el fallo de los servicios de ingenieros de la NASA a la hora de aplicar los mecanismos para detectar y corregir el fallo. Esa es la razón por la que perdimos la nave".

Validación del Diseño

Introducción - Errores informáticos de la historia

Apagón en EUA

En agosto del 2003 varios estados del noreste de EE.UU. y la provincia canadiense de Ontario se quedaron sin luz debido a un corte de energía resultado de un accidente local.

El accidente pasó desapercibido a causa de un fallo del software de vigilancia del funcionamiento de General Electric Energy y provocó una cadena de errores.

Validación del Diseño

Introducción - Errores informáticos de la historia

Transporte: Aerolínea American Airlines

EEUU | Problemas con el sistema de reservas

Un error informático cancela y retrasa cientos de vuelos de American Airlines

Un fallo generalizado en el sistema informático de American Airlines ha provocado este martes todo un caos aéreo en la compañía aérea de EEUU. Tras horas sin que el sistema de reservas funcionara con normalidad, todas las operaciones de la compañía fueron canceladas hasta que subsanó el error. Cientos de aviones con el pasaje ya embarcado sufrieron demoras y cancelaciones.

Validación del Diseño

Importancia: Verificación y validación de Software

Importancia de la detección de bugs en nuestras aplicaciones

Validación del Diseño

Introducción - Verificación y validación de Software

La verificación y validación es el nombre que se da a los procesos de comprobación y análisis que aseguran que el software que se desarrolla *está acorde a su especificación* y *cumple las necesidades (expectativas) de los clientes*.

La V&V es un proceso de ciclo de vida completo. Inicia con las revisiones de los requerimientos y continúa con las revisiones del diseño y las inspecciones del código hasta la prueba del producto. Existen actividades de V&V en cada etapa del proceso de desarrollo del software. La verificación y la validación no son la misma cosa , aunque es muy fácil confundirlas.

Validación del Diseño

Introducción - Definición de V&V según según Boehm:

Validación: *¿Construimos el producto correcto?*

Verificación: *¿Construimos bien el producto?*

- Los procesos de validación y verificación buscan comprobar que el software cumpla con las especificaciones y brinde la funcionalidad deseada por quienes financian su desarrollo.
- Comienzan tan pronto como están disponibles los requerimientos y continúan a través de todas las etapas del proceso de desarrollo.

Boehm fue pionero de la ingeniería de software expresó en el año 1979

Validación del Diseño

Introducción - Definición de V&V según Sommerville:

Verificación:

Busca comprobar que el sistema cumple con los requerimientos especificados (funcionales y no funcionales).

- ¿El software está de acuerdo con su especificación?

Validación

- Busca comprobar que el software hace lo que el usuario espera.
- ¿El software cumple las expectativas del cliente?

Validación del Diseño

Introducción - Definición de V&V según la IEEE

Verificación

⇒ El proceso de evaluación de un sistema o componente para determinar si los productos de una determinada fase de desarrollo satisfacen condiciones impuestas al inicio de la fase.

Validación

⇒ El proceso de evaluación de un sistema o componente durante o al final del proceso de desarrollo para determinar si satisface los requisitos especificados.

IEEE: Institute of Electrical and Electronics Engineers

Validación del Diseño

Introducción

El objetivo final de los procesos de validación y verificación es establecer un ***nivel de confianza*** para que sea posible afirmar que el sistema es **adecuado**. Esto significa que debe ser lo suficientemente eficaz en función de lo esperado.

El nivel de confianza depende de:

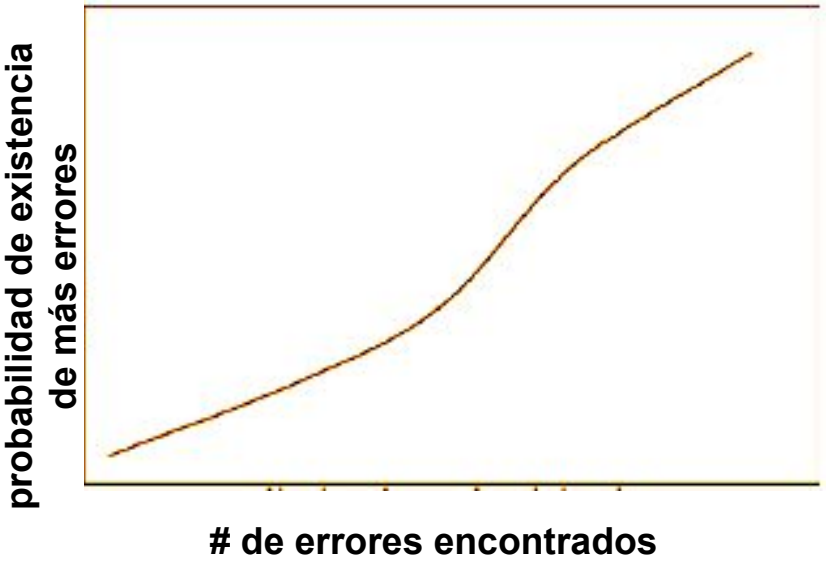
- El propósito del sistema
- Las expectativas de los usuarios
- El mercado actual para el sistema

Validación del Diseño

Introducción

Errores

Error: una equivocación de una persona al desarrollar alguna actividad.
Defecto: se produce cuando una persona comete un error. (ej: cuando un diseñador comprende mal un requerimiento y crea un diseño inadecuado).
Falla: es un desvío respecto del comportamiento esperado del sistema.



La probabilidad de existencia de errores (aún no detectados) en una sección de un programa es proporcional a la cantidad de errores detectados.

Myers, The Art of Software Testing, 2012.

Validación del Diseño

Proceso de prueba

Introducción

Las pruebas representan la instancia desde donde puede valorarse la calidad y, de manera más pragmática, descubrirse errores. Pero las pruebas no deben verse como una red de seguridad:

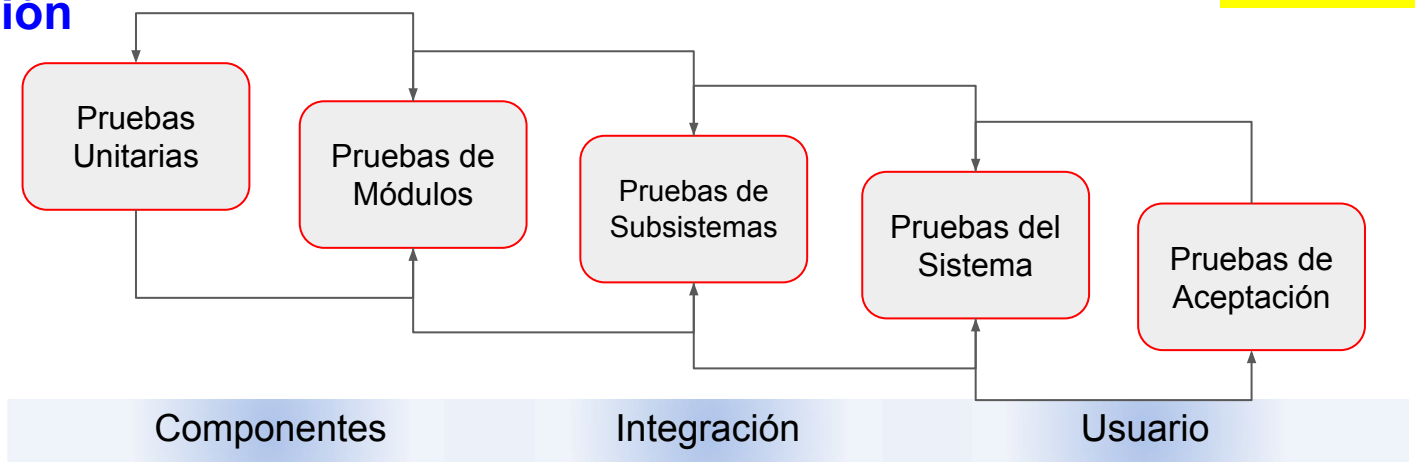
"No se puede probar la calidad. Si no está ahí antes de comenzar las pruebas, no estará cuando termine de probar".

La calidad se incorpora en el software **a lo largo de todo el proceso de ingeniería del software**. La adecuada aplicación de métodos y herramientas, revisiones técnicas efectivas, y gestión y medición sólidas conducen a la calidad que se confirma durante las pruebas.

Validación del Diseño

Introducción

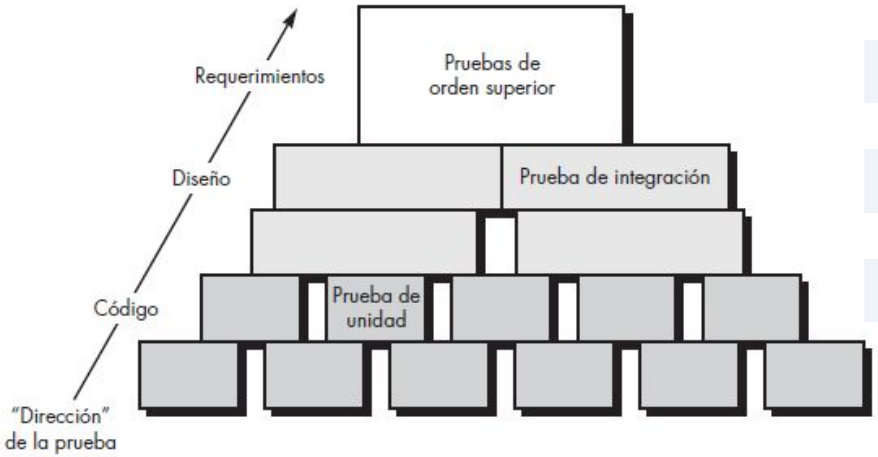
Proceso de prueba



Un sistema complejo suele probarse en varias etapas por medio de un proceso iterativo. Es decir, se debe volver a las fases anteriores cada vez que se encuentra un error en la fase en ejecución.

Validación del Diseño

Introducción



Proceso de prueba

Usuario

Integración

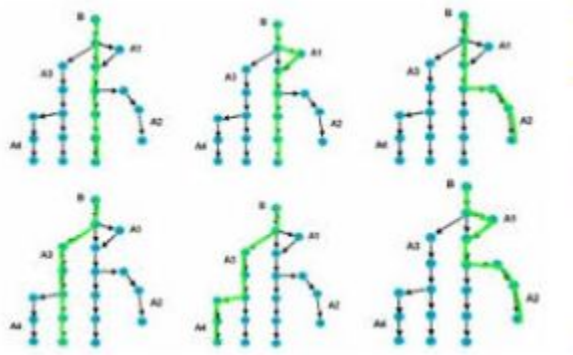
Componentes

Inicialmente, las pruebas se enfocan en cada componente de manera individual. Luego, los componentes deben ensamblarse o integrarse para formar el paquete de software completo.

La ***prueba del sistema*** verifica que todos los elementos se complementan de manera adecuada y se consigue el funcionamiento/rendimiento global esperado.

Validación del Diseño

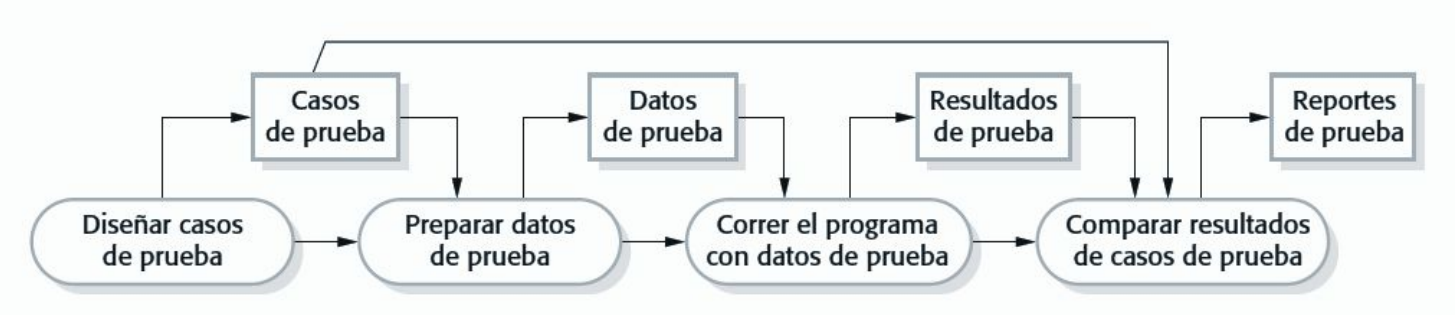
Objetivos de la construcción de Casos de Prueba



Validación del Diseño

Objetivos de la construcción de Casos de Prueba

Los ***casos de prueba o test cases*** son especificaciones de las entradas para la prueba y la salida esperada del sistema (los resultados de la prueba), además de información sobre lo que se pone a prueba. Los ***datos de prueba*** son las entradas que se diseñaron para probar un sistema.

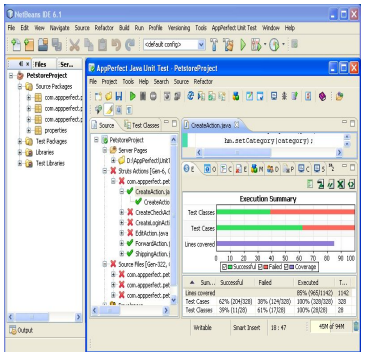


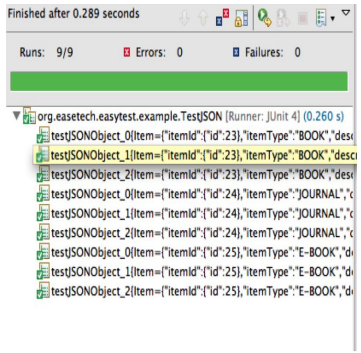
Modelo del proceso de pruebas de software

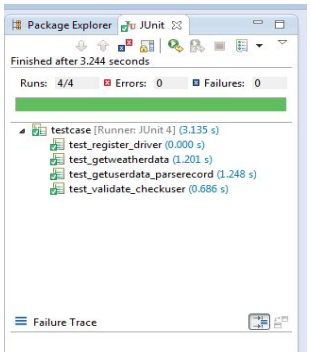
Validación del Diseño

Objetivos de la construcción de Casos de Prueba

En ocasiones pueden generarse automáticamente datos de prueba; no obstante, es imposible la generación automática de casos de prueba. Sin embargo, es posible automatizar la ejecución de pruebas. Los resultados previstos se comparan automáticamente con los resultados establecidos, de manera que no haya necesidad que un individuo busque errores y anomalías al correr las pruebas.







Validación del Diseño

Objetivos de la construcción de Casos de Prueba

Por lo general, un sistema de software debe pasar por tres etapas de pruebas:

1. Pruebas de desarrollo: el sistema se pone a prueba durante el proceso para descubrir errores (bugs) y defectos.

2. Versiones de prueba: un equipo de prueba por separado experimenta una versión completa del sistema, antes de presentarlo a los usuarios.

3. Pruebas de usuario: los usuarios reales o potenciales de un sistema prueban el sistema en su propio entorno. Las pruebas de aceptación se efectúan cuando el cliente prueba de manera formal un sistema para decidir si debe aceptarse del proveedor del sistema, o si se requiere más desarrollo.

Validación del Diseño

Pruebas unitarias y de integración

Validación del Diseño

Pruebas unitarias

Pruebas unitarias y de integración

Las pruebas unitarias son el proceso de probar componentes del programa, como métodos o clases de objetos. Las funciones o los métodos individuales son el tipo más simple de componente. Se deben diseñar las pruebas para brindar cobertura a todas las características del objeto. Esto significa que debe:

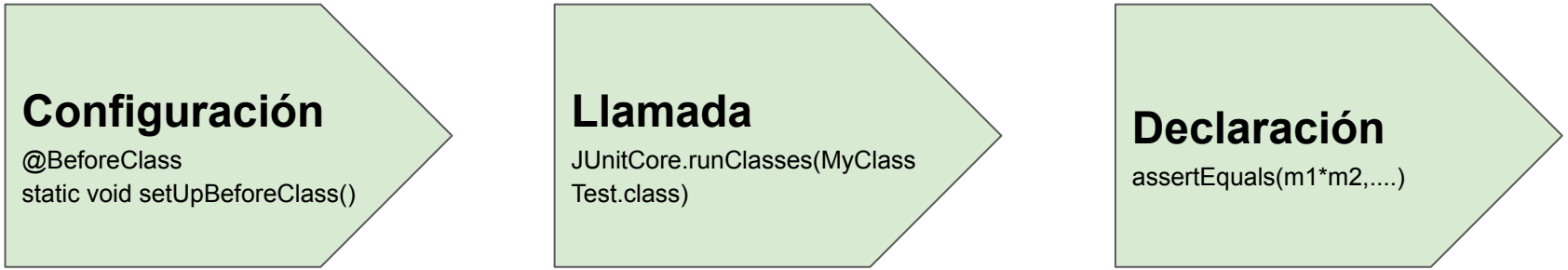
- Probar todas las ***operaciones*** asociadas con el objeto.
- Establecer y verificar valores para los ***atributos*** relacionados con el objeto.
- Poner el objeto en todos los ***estados*** posibles.

Validación del Diseño

Pruebas unitarias

Pruebas unitarias y de integración

Un conjunto automatizado de pruebas posee tres partes:



Se inicializa el sistema con el caso de prueba (entradas y salidas esperadas).

Se invoca al objeto o al método que se va a probar.

Se compara el resultado de la llamada con el resultado esperado. Si la información se evalúa como verdadera, la prueba tuvo éxito; pero si resulta falsa, entonces fracasó.

Validación del Diseño

Pruebas unitarias: objetos mock

Pruebas unitarias y de integración

Cuando el objeto que se prueba posee dependencias a otros objetos que tal vez todavía no se desarrollaron o que, si se utilizan, frenan o lentifican el proceso de pruebas. En esos casos, se puede usar ***objetos mock*** (representantes). Son objetos con la misma interfaz como los usados por objetos externos que simulan su funcionalidad.

Un objeto mock que aparenta una base de datos suele tener sólo algunos ítems de datos. De igual modo, los objetos mock pueden usarse para simular una operación anormal o eventos extraños.

Validación del Diseño

Pruebas unitarias y de integración

Pruebas unitarias: pruebas de frontera

Las pruebas de frontera son una de las tareas de prueba unitarias más importantes.

Con frecuencia, el software falla en sus fronteras.

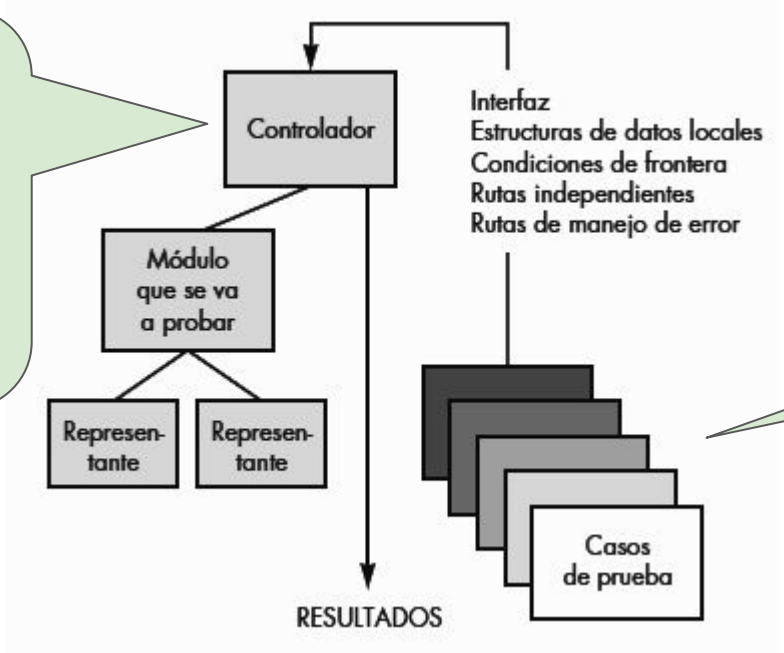
Es muy probable que los casos de prueba que trabajan con la estructura de datos, el flujo de control y los valores de datos justo abajo y arriba de máximos y mínimos descubran errores.

Validación del Diseño

Pruebas unitarias y de integración

Pruebas unitarias: Entorno

Un controlador es un "programa principal" que acepta datos de caso de prueba, pasa tales datos al componente (que va a ponerse a prueba) e imprime resultados relevantes.



Cada caso de prueba debe acoplarse con un conjunto de resultados esperados.

Validación del Diseño

Pruebas unitarias

Pruebas unitarias y de integración

Algunos de los lineamientos generales para el diseño de casos de prueba:

- Elegir entradas que fuercen al sistema a generar todos los mensajes de error.
- Diseñar entradas que produzcan que los buffers de entrada se desborden.
- Repetir varias veces la misma entrada o serie de entradas.
- Forzar la generación de salidas inválidas.
- Forzar resultados de cálculo demasiado largos o demasiado pequeños.

Validación del Diseño

Pruebas de integración

Validación del Diseño

Pruebas de integración

Pruebas unitarias y de integración

Las ***pruebas de integración*** son una técnica sistemática para construir la arquitectura del software mientras se llevan a cabo pruebas para descubrir errores asociados con la interfaz. El objetivo es tomar los componentes probados de manera individual y construir una estructura de programa que se haya dictado por diseño.

Con frecuencia existe una tendencia a intentar la integración no incremental, es decir, a construir el programa de modo tal que todos los componentes se combinan por adelantado.

Validación del Diseño

Pruebas de integración

Pruebas unitarias y de integración

En la ***integración incremental*** el programa se construye y prueba en pequeños incrementos, donde los errores son más fáciles de aislar y corregir; las interfaces tienen más posibilidades de probarse por completo; y puede aplicarse un enfoque de prueba sistemático. A continuación se exponen algunas estrategias de integración incremental.

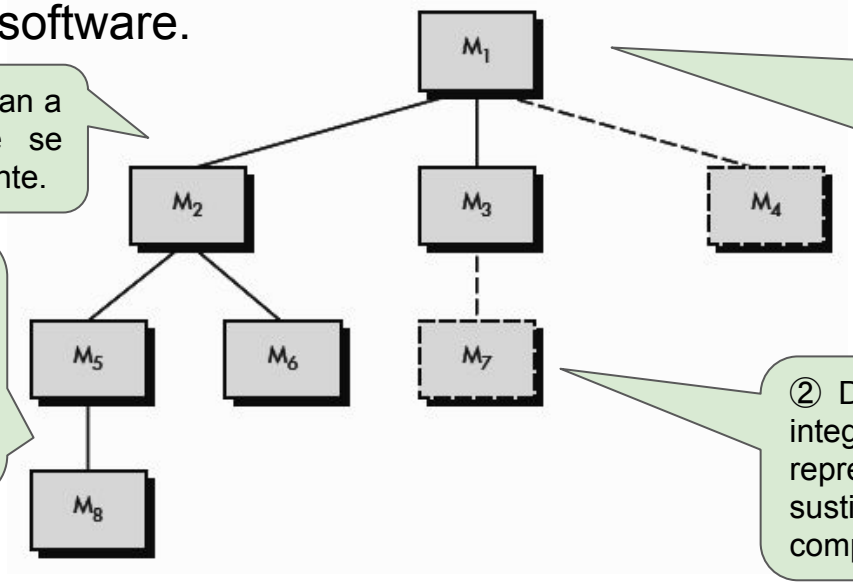
Cuando desarrolle un cronograma de proyecto, considere la forma en la que ocurrirá la integración, de modo que los componentes estén disponibles cuando se necesiten.

Validación del Diseño

Pruebas unitarias y de integración

Pruebas de integración

Integración descendente es un enfoque incremental a la construcción de la arquitectura de software.



① El módulo de control principal se usa como un controlador de prueba y los representantes (mocks) se sustituyen con todos los componentes directamente subordinados al módulo de control principal.

② Dependiendo del enfoque de integración seleccionado, los representantes subordinados se sustituyen uno a la vez con componentes reales.

③ Las pruebas se llevan a cabo a medida que se integra cada componente.

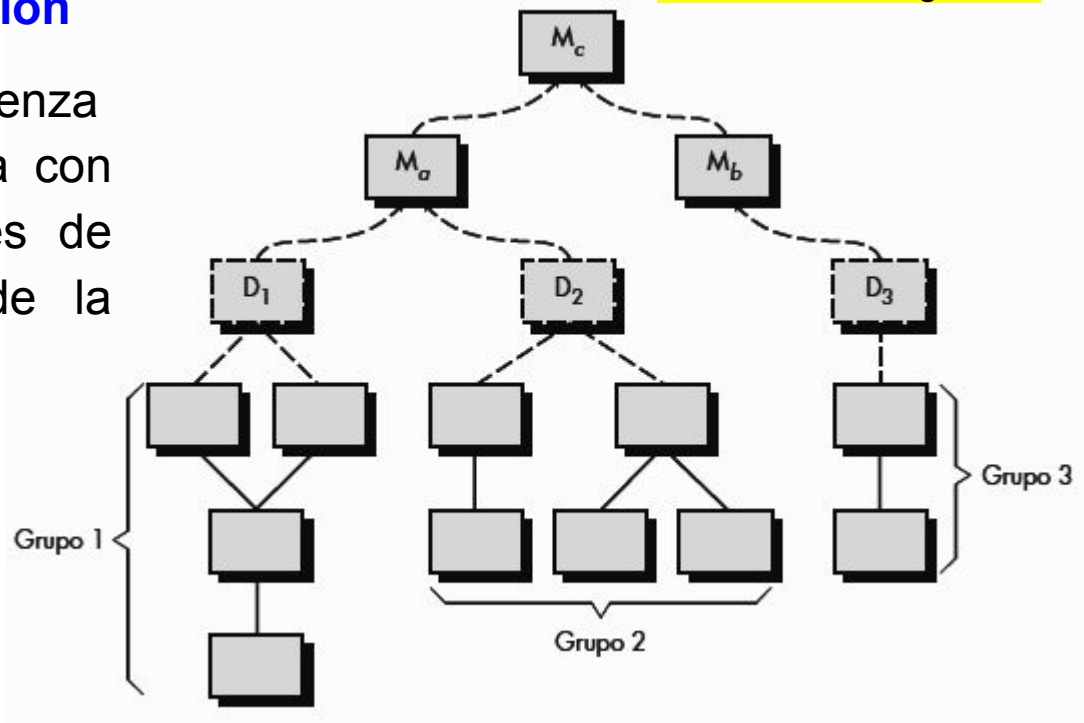
④ Al completar cada conjunto de pruebas, el representante se sustituye con el componente real.

Validación del Diseño

Pruebas unitarias y de integración

Integración ascendente comienza con la construcción y la prueba con módulos atómicos (componentes de los niveles inferiores dentro de la estructura del programa).

Pruebas de integración



Validación del Diseño

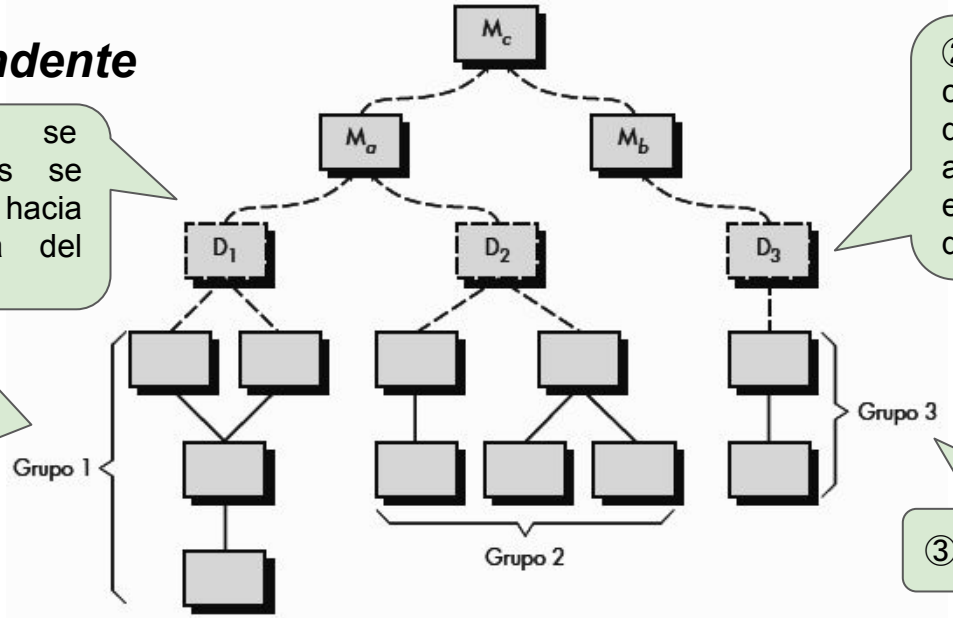
Pruebas unitarias y de integración

Pruebas de integración

Integración ascendente

④ Los controladores se remueven y los grupos se combinan moviéndolos hacia arriba en la estructura del programa.

① Los componentes en el nivel inferior se combinan en grupos (builds) que realizan una subfunción de software específica.

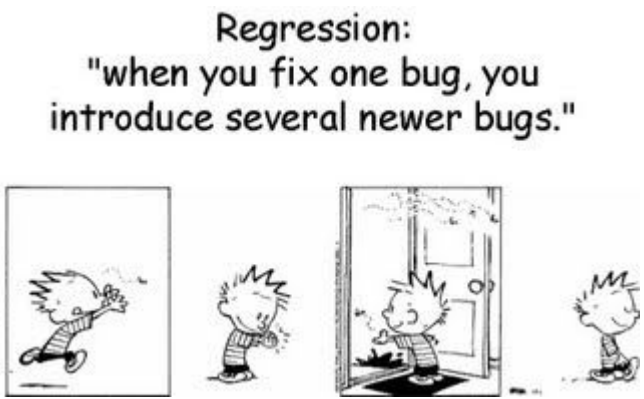


② Se desarrollan controladores (programas de control para pruebas) a fin de coordinar la entrada y salida de casos de prueba.

③ Se prueba el grupo.

Validación del Diseño

Pruebas de regresión



Validación del Diseño

Pruebas de regresión

Pruebas de regresión

Cada vez que se agrega un nuevo módulo como parte de las pruebas de integración, el software cambia. Se establecen nuevas rutas de flujo de datos, ocurren nuevas operaciones de entrada/salida y se invoca nueva lógica de control. Dichos cambios pueden causar problemas con las funciones que anteriormente trabajaban sin fallas. En el contexto de una estrategia de prueba de integración, la ***prueba de regresión*** es la nueva ejecución de algún subconjunto de pruebas que ya se realizaron a fin de asegurar que los cambios no propagaron efectos colaterales no deseados.

Validación del Diseño

Pruebas de validación

Validación del Diseño

Pruebas de validación

Las pruebas de validación comienzan cuando finalizan las pruebas de integración, es decir, cuando el software está completamente ensamblado como un paquete y los errores de interfaz se descubrieron y corrigieron.

La validación es exitosa cuando el software funciona en una forma que cumpla con las expectativas razonables del cliente.

Validación del Diseño

Pruebas de validación

Si se desarrolló una **Especificación de Requerimientos de Software** (SRS), en ella se describen todos los atributos del software visibles para el usuario y contiene una sección de ***Criterios de Validación*** que forman la base para un enfoque de pruebas de validación.

Validación del Diseño

Pruebas de validación

Criterios de validación

La validación del software se logra a través de una serie de pruebas que demuestran ***conformidad con los requerimientos***. Después de realizar cada caso de prueba de validación, existen dos posibles condiciones:

1. La característica de función o rendimiento se conforma de acuerdo con las especificaciones y se acepta.

2. Se descubre una desviación de la especificación y se crea una lista de ellos.

Con frecuencia es necesario negociar con el cliente para establecer un método para resolver los desvíos.

Validación del Diseño

Pruebas Alfa y Beta

Pruebas de validación

Antes de que un software sea liberado en el mercado o presentado al cliente final, tiene que pasar a través de muchos controles de calidad. El software debe trabajar acorde a la expectativa del usuario, y no debe tener errores. El usuario del software no está interesado en saber cómo se desarrolló el software o cómo funciona el código fuente; por esto cuando se construye software a medida para un cliente, se lleva a cabo una serie de pruebas de aceptación para permitir que el cliente valide todos los requisitos. Lo que importa para el usuario es la funcionalidad del software. Por lo tanto ***en las pruebas alfa y beta*** el código fuente no está probado, y la atención se centra en la funcionalidad del software, de acuerdo con las necesidades del usuario.

Validación del Diseño

Pruebas Alfa y Beta

Pruebas de validación - Prueba Alfa

Alfa: se llevan a cabo, por un cliente, en el lugar de desarrollo. Se usa el software de forma natural con el desarrollador como observador del usuario y registrando los errores y problemas de uso.

Las pruebas alfa se hacen en un entorno controlado. Se realizan después de que todos los procedimientos de prueba básicos, como las pruebas unitarias y pruebas de integración se han completado, y se produce después de las pruebas del sistema. En general trata de pruebas de navegación, entrada y salida de los mecanismos del software, Esta no es la versión final de software y cierta funcionalidad podría ser añadida al software incluso después de esta prueba.

Validación del Diseño

Pruebas Alfa y Beta

Pruebas de validación - Prueba Beta

Beta: se llevan a cabo por los usuarios finales del software en los lugares de trabajo de los clientes. A diferencia de la prueba alfa, el desarrollador no está presente normalmente. Así, la prueba beta es una aplicación en vivo del software en un entorno que no puede ser controlado por el desarrollador. El cliente registra todos los problemas que encuentra durante la prueba beta e informa al desarrollador. Las pruebas beta es la última fase de las fases de prueba y se hace utilizando técnicas de caja negra. A veces la versión beta también es liberada en el mercado, y en base a los comentarios de los usuarios se hacen modificaciones.

Se consideran versiones inestables.

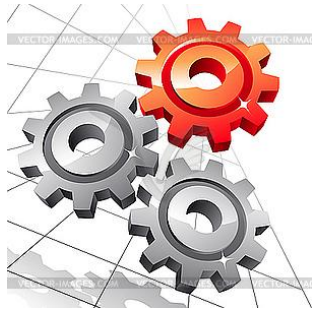
Validación del Diseño

Pruebas del sistema

Validación del Diseño

Pruebas del sistema

Las pruebas del sistema durante el desarrollo incluyen la integración de componentes para crear una versión del sistema y, luego, poner a prueba el sistema integrado. Las pruebas de sistema demuestran que los componentes son compatibles, que interactúan correctamente y que transfieren los datos correctos en el momento adecuado a través de sus interfaces.



En esta etapa el objetivo es evaluar que los elementos del sistema se hayan integrado de manera adecuada y que se realicen las funciones asignadas.

Validación del Diseño

Pruebas del sistema

Pruebas de recuperación

Es una prueba que fuerza al software a fallar en varias formas y que verifica que la recuperación se realice de manera adecuada. Si la recuperación es automática (realizada por el sistema en sí), se evalúa el reinicio, los mecanismos de puntos de verificación, la recuperación de datos y la reanudación para correcciones. Si la recuperación requiere intervención humana, se evalúa el tiempo medio de reparación (TMR) para determinar si está dentro de límites aceptables.

Validación del Diseño

Pruebas del sistema

Pruebas de seguridad

La prueba de seguridad intenta verificar que los mecanismos de protección que se construyen en un sistema en realidad lo protegerán de cualquier acceso no autorizado. Abarca un amplio rango de actividades: hackers que intentan ingresar en los sistemas como pasatiempo o desafío, empleados resentidos que intentan ingresar por venganza, individuos deshonestos que intentan acceder para obtener un lucro ilícito.

Validación del Diseño

Pruebas del sistema

Pruebas de esfuerzo (stress)

La prueba de esfuerzo ejecuta un sistema en forma que demanda recursos en cantidad, frecuencia o volumen anormales. Por ejemplo:

1) Pruebas especiales que generen diez interrupciones por segundo, cuando una o dos es la tasa promedio.
2) Aumentar las tasas de entrada de datos para determinar cómo responderán las funciones de entrada.
3) Requerimientos de memoria máxima y otros recursos.
4) Thrashing (hiperpaginación) en un sistema operativo.
5) Búsqueda excesiva por datos residentes en disco.

Validación del Diseño

Pruebas del sistema

Pruebas de rendimiento o performance

Las pruebas de rendimiento con frecuencia se suman a las pruebas de esfuerzo y por lo general requieren de medición de hardware y de software para este propósito, es decir, con frecuencia es necesario medir la utilización de los recursos (por ejemplo, ciclos del procesador, memoria consumida, conexiones a la base de datos, etc. La instrumentación externa puede monitorear intervalos de ejecución y eventos de registro (por ejemplo, interrupciones) conforme ocurren, y los muestreos del estado de la máquina de manera regular.

Validación del Diseño

Pruebas del sistema

Pruebas de despliegue

En muchos casos, el software debe ejecutarse en varias plataformas y bajo más de un entorno de sistema operativo. La prueba de despliegue, en ocasiones llamada prueba de configuración, evalúa el software en cada entorno en el que debe operar.

Además, examina todos los procedimientos de instalación y el software de instalación especializado (por ejemplo, “instaladores”) que usarán los clientes, así como toda la documentación que se usará para introducir el software a los usuarios finales.

Validación del Diseño

Pruebas basadas en escenarios

Validación del Diseño

Pruebas basadas en escenarios

La prueba basadas en escenarios se concentran en lo que hace el usuario, no en lo que hace el producto.

Esto significa capturar las tareas (por medio de casos de uso) que el usuario tiene que realizar y luego aplicar éstas y sus variantes como pruebas.

Los escenarios descubren errores de interacción. Pero, para lograr esto, los casos de prueba deben ser más complejos y más realistas que las pruebas basadas en fallo.

Validación del Diseño

Pruebas basadas en escenarios

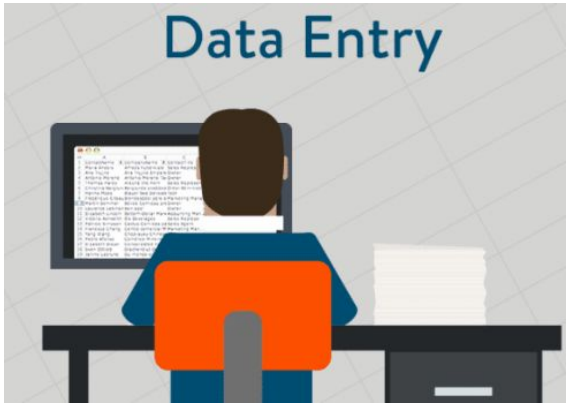
Como ejemplo, tome en cuenta el diseño de pruebas basadas en escenario para un editor de texto:

Caso de uso: Corrección del borrador final
Antecedentes: Al imprimir el borrador "final", se lo lee y descubre algunos errores no vistos en la pantalla.
1. Imprimir todo el documento.
2. Moverse en el documento, cambiar ciertas páginas.
3. Conforme cada página cambia, imprimirla.
4. En ocasiones se imprime una serie de páginas.

Validación del Diseño

Pruebas basadas en escenarios

La pruebas basadas en escenarios tienden a evaluar múltiples subsistemas en una sola prueba (los usuarios no se limitan al uso de un subsistema a la vez).



Validación del Diseño

En la industria decimos - Camino Feliz - Happy path- Test to pass

Es un escenario de uso ideal sin condiciones de excepción. Cuando escribimos software tenemos la tendencia natural de escribir pruebas que consciente o inconscientemente sabemos que van a dar pocos problemas o directamente que no va a fallar.

Probamos entonces el mejor caso y que resuelve bien nuestro código: la prueba feliz o happy path nos dice que nuestro código hace bien aquello para lo que lo hemos desarrollado.

Pero, no nos podemos limitar a escribir exclusivamente *pruebas felices*, esto solo puede ser el primer paso de nuestro caso de prueba.

Validación del Diseño

En la industria decimos - Criterios de aceptación (acceptance criteria)

Son los criterios de salida que un componente o sistema debe satisfacer para ser aceptado por un usuario, cliente u otra entidad autorizada. [IEEE 610].

Un criterio es un norma evaluadora, por lo tanto, los criterios de aceptación definen si una cierta funcionalidad cumple sus objetivos o no. Por lo tanto, deben ser redactados de manera clara, concisa, y sin subestimar su impacto.

Siempre debe tenerse en cuenta que se debe abarcar los distintos escenarios de negocio, desde el principal pasando por las distintas alternativas y excepciones que puedan existir dentro del requerimiento. Evita los diferentes puntos de vista que pueden tener los profesionales de testing.

Validación del Diseño

En la industria decimos - Métricas

Lo que no se puede medir, no se puede gestionar

Métrica: Medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo determinado.

Ejemplos:

- Cantidad de bugs críticos por Sprint.
- Cantidad de bugs en producción del último Q.
- Cantidad de bugs resueltos en la etapa de estabilización.

Validación del Diseño

En la industria decimos - Smoke Test (Prueba de humo)

⇒ Una prueba rápida de las principales funciones de una aplicación software. Se emplean para detectar lo más pronto posible si funcionan los componentes más críticos de la aplicación.

⇒ Es una prueba poco profunda y muy general en la que se prueban los principales componentes de la aplicación sin entrar en demasiado detalle.

⇒ Es una prueba superficial e insume muy poco tiempo.

⇒ Se lleva a cabo para asegurar que las funciones más importantes de un programa están operando correctamente, pero sin molestarse con los detalles más finos.

Validación del Diseño

Cuestionario

https://b.socrative.com/login/student/

Habitación/Room: **UTNDDS**

Referencias

Pressman, R., Ingeniería de Software, un enfoque práctico. 7ma Edición.
McGraw-Hill. 2010.

Sommerville, I., Ingeniería de Software. 9na Edición.
Addison-Wesley. 2011.

Myers, G., The Art of Software Testing.
Wiley. 2011.

https://elpais.com/diario/1999/10/02/sociedad/938815207_850215.html
Mayo 2018