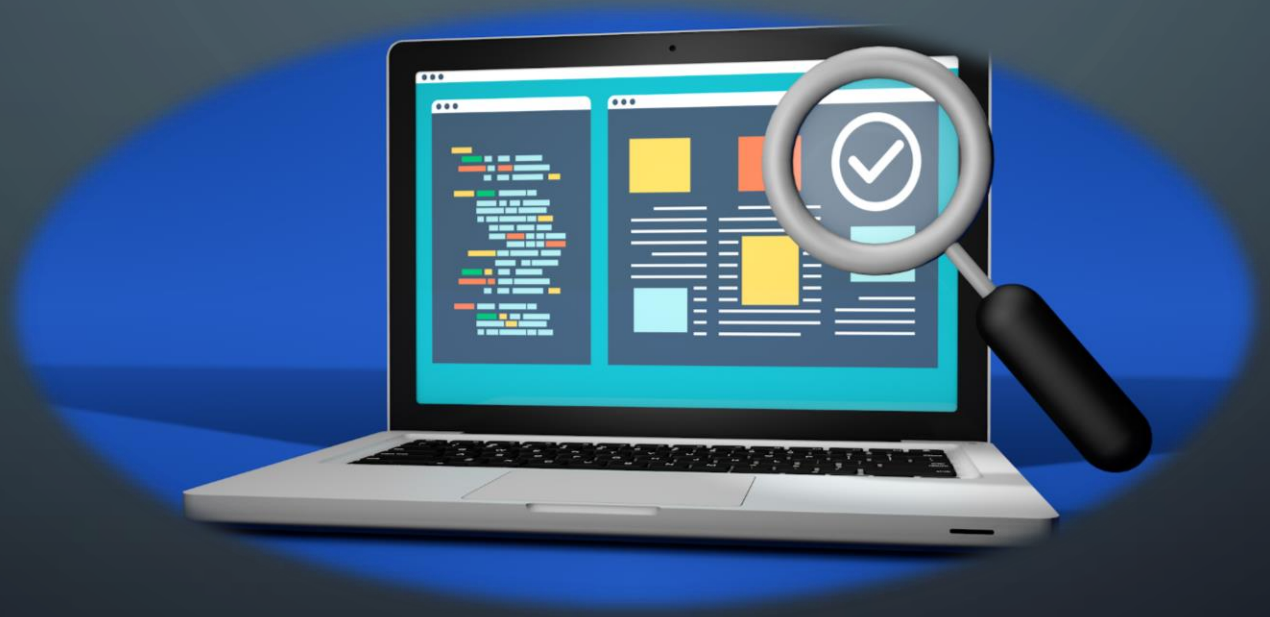
IFTS 18

METODOLOGÍA DE PRUEBAS DE SOFTWARE



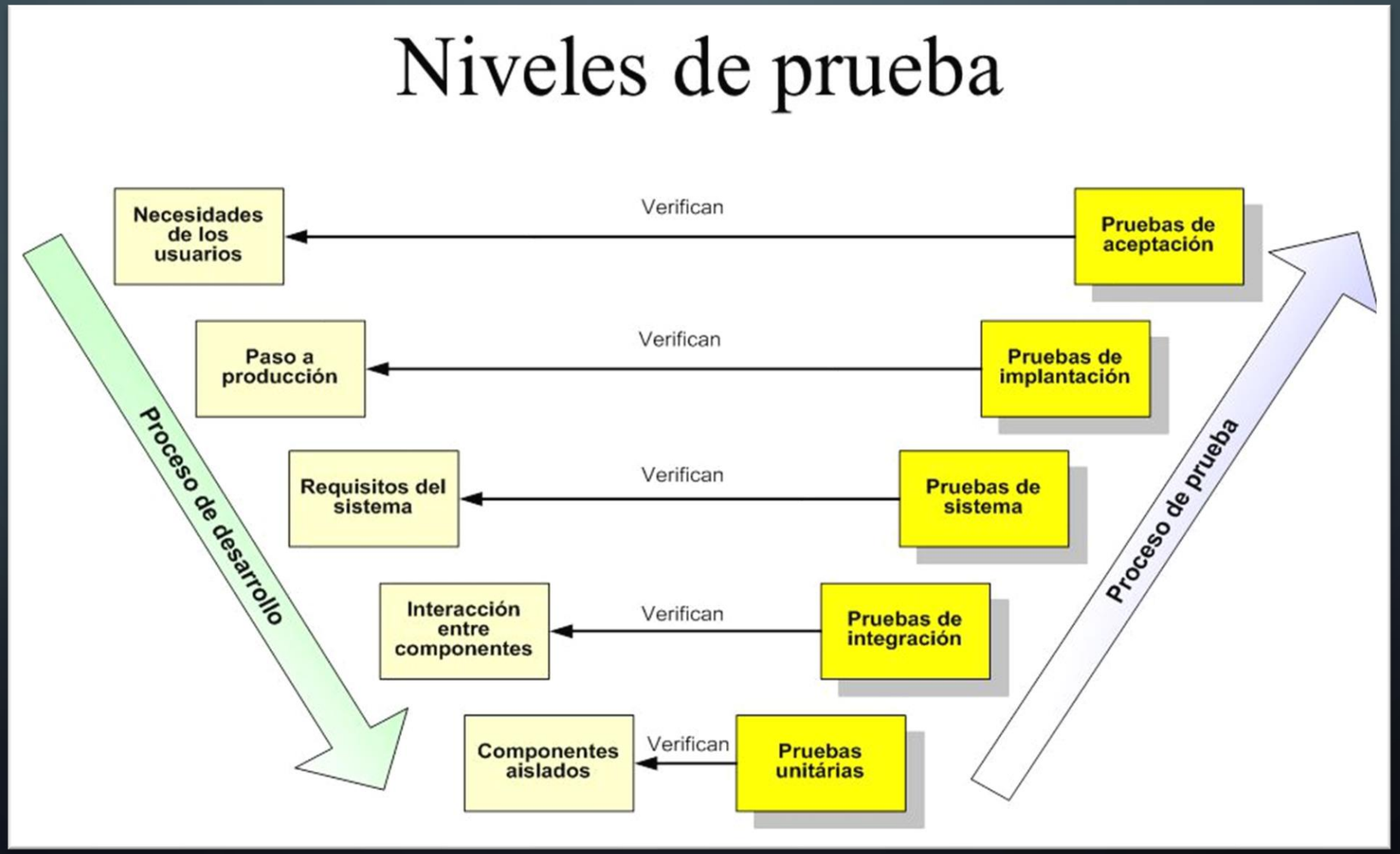
Materia: Metodología de Pruebas de Software

Profesor: Ing. Pablo Andrés Pérez

RESUMEN DE LO VISTO

CLASIFICACIÓN DE LAS PRUEBAS DINÁMICAS

PRUEBAS DINÁMICAS



PRUEBAS DE INTEGRACIÓN

Objetivo:

- *Verificar que el comportamiento del software entre las distintas interfaces y componentes es el esperado*
- *Asegurar la calidad de las interfaces*
- *Encontrar defectos en las comunicaciones*
- *Prevenir la aparición de defectos en fases posteriores de pruebas*
- *Verificar el correcto ensamblaje de los componentes una vez que han sido probados unitariamente con el fin de comprobar que interactúan correctamente a través de sus interfaces tanto interna como externas*

PRUEBAS DE INTEGRACIÓN

¿Qué son las pruebas de integración? - **Definición**

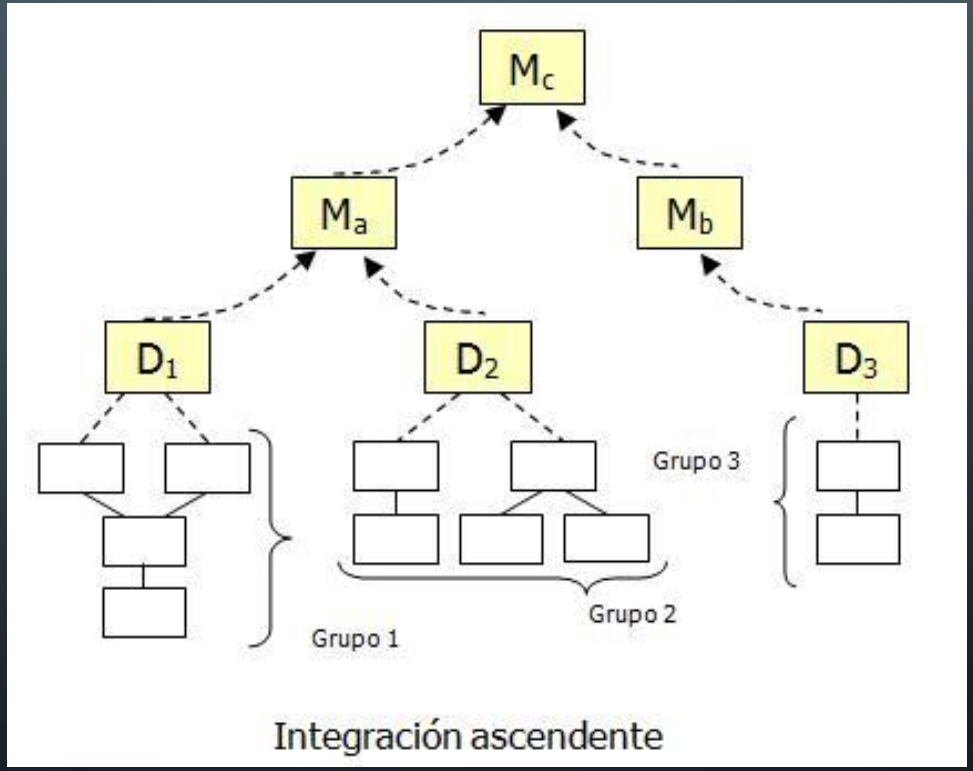
- Las pruebas de integración son aquellas que se realizan para **comprobar las interacciones entre distintos componentes o sistemas tras su integración.**

PRUEBAS DE INTEGRACIÓN

- ***Definición***
- *Las pruebas de integración **aseguran** que todas las unidades desarrolladas **funcionan en conjunto** y **ayudan a prevenir** contratiempos posteriores.*

PRUEBAS DE INTEGRACIÓN

- *El **propósito** de las pruebas de integración es **validar la integración de diferentes módulos** juntos e identificar los **errores y problemas** relacionados con ellos.*



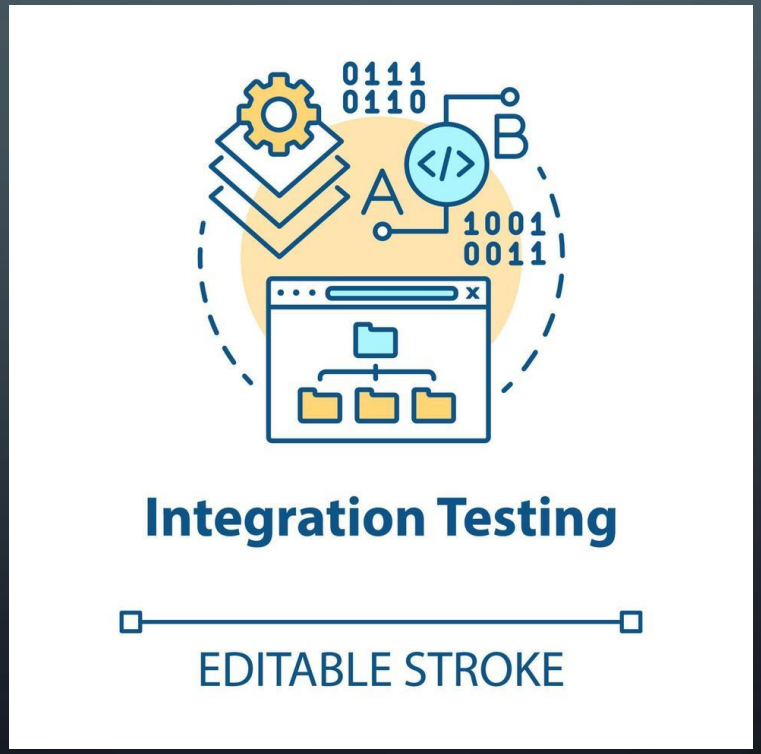
Integración ascendente

PRUEBAS DE INTEGRACIÓN

- *Las pruebas de integración **verifican** que los diferentes módulos y/o servicios usados por nuestra aplicación **funcionen en armonía cuando trabajan en conjunto.** Por ejemplo, pueden probar la interacción con una o múltiples bases de datos, o asegurar que los microservicios operen como se espera.*

PRUEBAS DE INTEGRACIÓN

- *Las pruebas de integración son típicamente el **paso siguiente a las pruebas unitarias** y son generalmente **más costosas de ejecutar**, ya que **requieren que más partes de nuestra aplicación se configuren y se encuentren en funcionamiento.***



PRUEBAS DE INTEGRACIÓN

NIVELES DE PRUEBAS DE INTEGRACIÓN

Pruebas de integración de componentes

Principalmente enfocado en la interacción de interfaces entre los componentes integrados.

Este tipo de pruebas debe realizarse de forma ***posterior a las pruebas unitarias.***

Generalmente se automatizan, lo que supone obviamente una ventaja en cuanto a tiempo y dinero se refiere.

En desarrollos ágiles, la integración de los distintos componentes, son habitualmente parte de un proceso de ***integración continua.***

PRUEBAS DE INTEGRACIÓN

NIVELES DE PRUEBAS DE INTEGRACIÓN

Pruebas de integración de sistemas

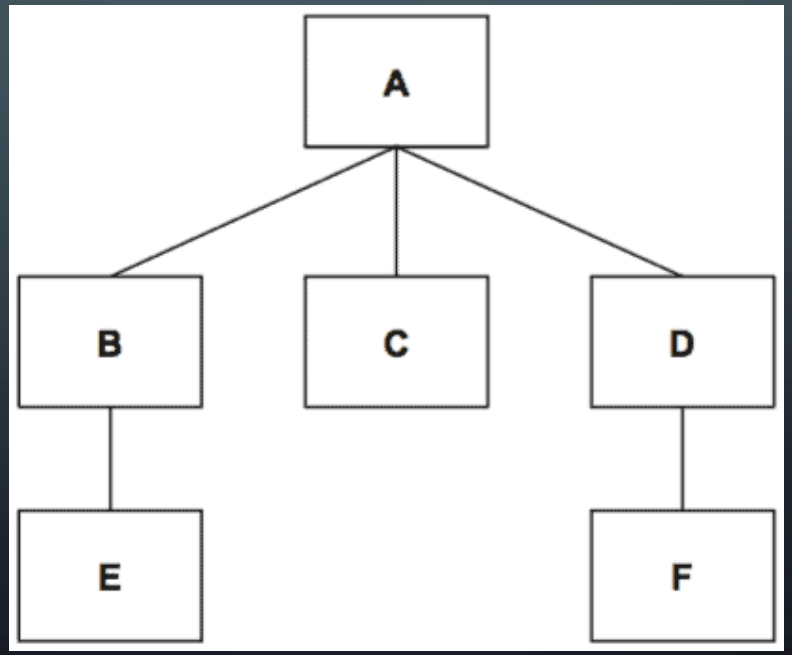
En este caso, se centran en la interacción e interfaces entre distintos sistemas, paquetes o microservicios. Aquí también entrarían las ***pruebas de interacciones con elementos externos a la organización*** como, por ejemplo, servicios web (mail, almacenamiento cloud, redes sociales, etc…) Esto supone un reto mayor al no tener todo el control de estos elementos externos.

Los test de ***integración de sistemas*** se pueden realizar tanto después de las ***pruebas de sistema***, como de forma paralela a ellas

PRUEBAS DE INTEGRACIÓN

ESTRATEGIAS DE INTEGRACIÓN:

TOP-DOWN (De arriba hacia abajo). ***El primer componente que se desarrolla y prueba es el primero de la jerarquía (A). Los componentes de nivel más bajo se sustituyen por componentes auxiliares para simular a los componentes invocados*** *(se necesita hacer uso de* ***stubs*** *que recibe llamadas de un método).* ***En este caso no son necesarios componentes conductores. Una de las ventajas de aplicar esta estrategia es que las interfaces entre los distintos componentes se prueban en una fase temprana y con frecuencia.***



PRUEBAS DE INTEGRACIÓN

*Un **STUB** es, en el contexto del testeo del software, un trozo de código usado como sustituto de alguna otra funcionalidad. Un stub puede simular el comportamiento de código existente (tal como un procedimiento en una máquina remota) o ser el sustituto temporal para un código aún no desarrollado. Los stubs son muy útiles para porting, computación distribuida así como en el desarrollo y pruebas de software en general.*

```
INICIO
    Temperatura = LeerTermometro(Afuera)
    SI Temperatura > 40 ENTONCES
         ESCRIBIR "Hace calor!"
    FIN SI
FIN
```

```
INICIO LeerTermometro(Fuente adentroOafuera)
     RETORNAR 28
FIN LeerTermometro
```

PRUEBAS DE INTEGRACIÓN

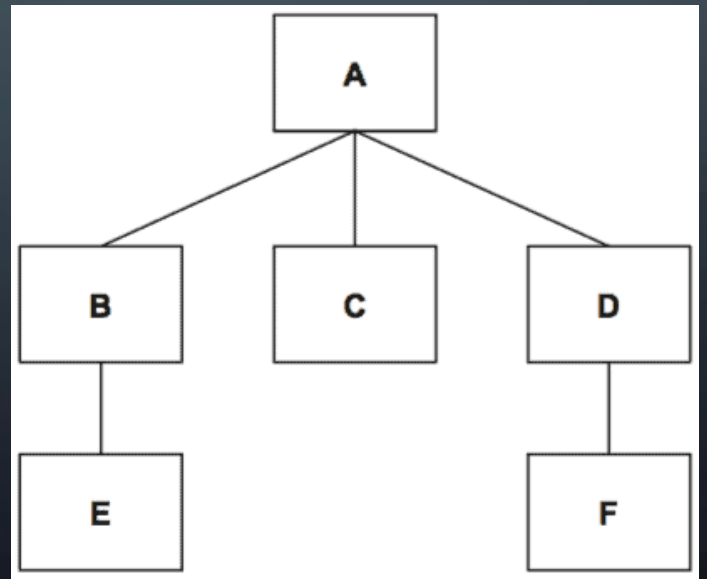
*Un **STUB** es una rutina que realmente no hace otra cosa que declararse a sí misma y a los parámetros que acepta y que devuelve un valor habitual dentro de los 'escenarios felices' del que llama al stub. Los stubs se usan habitualmente como sustitutos de la implementación aún no finalizada de una interfaz ya definida. El stub contendría sólo el código necesario para que compile y enlace con el resto del programa.*

PRUEBAS DE INTEGRACIÓN

ESTRATEGIAS DE INTEGRACIÓN:

DOWN-TOP/BUTTON-UP: *(De abajo hacia arriba) En este caso se crean primero los componentes de más bajo nivel (E, F) y se crean componentes conductores para simular a los componentes que los llaman. A continuación se desarrollan los componentes de más alto nivel (B, C, D) y se prueban. Por último dichos componentes se combinan con el que los llama (A). Los componentes auxiliares son necesarios en raras ocasiones.*

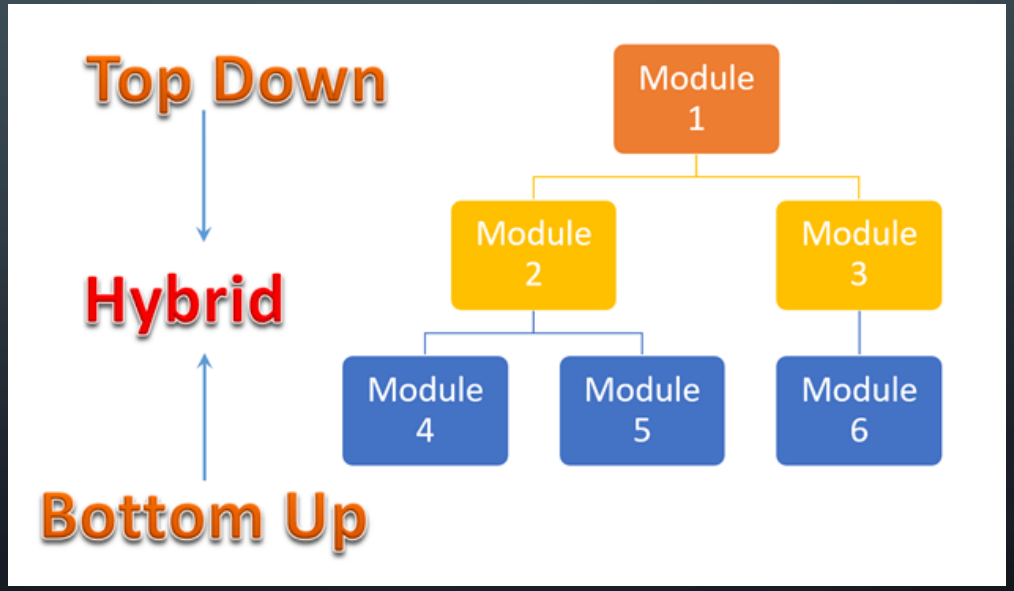
Este tipo de enfoque permite un desarrollo más en paralelo que el enfoque de arriba hacia abajo, pero presenta mayores dificultades a la hora de planificar y de gestionar.



PRUEBAS DE INTEGRACIÓN

ESTRATEGIAS DE INTEGRACIÓN:

HYBRID: *(Estrategias combinadas). A menudo es útil aplicar las estrategias anteriores conjuntamente. De este modo, se desarrollan partes del sistema con un enfoque «top-down«, mientras que los componentes más críticos en el nivel más bajo se desarrollan siguiendo un enfoque «down-top«. En este caso es necesaria una planificación cuidadosa y coordinada de modo que los componentes individuales se «encuentren» en el centro.*



RECORDEMOS…

Stubs and Drivers son los programas ficticios en las pruebas de integración utilizados para facilitar la actividad de prueba de software. *Estos programas actúan como sustitutos de los modelos que faltan en las pruebas. No implementan toda la lógica de programación del módulo de software, pero simulan la comunicación de datos con el módulo de llamada durante la prueba.*

Stub: Es llamado por el Módulo bajo Prueba.

Driver: Llama al Módulo a probar.

PRUEBAS DE INTEGRACIÓN

ESTRATEGIAS DE INTEGRACIÓN:

BIG BANG

Esta integración es no incremental, ya que se realiza cuando todos los componentes están completamente integrados. De ahí su nombre. Una vez está todo integrado, se inician estos test juntos.

El inconveniente de esta estrategia es que, como se ha consumido ya bastante tiempo esperando a que estuviera todo integrado, en esta fase los retrasos estarían más penalizados.

La ventaja es que cuentas con el resto de componentes y, por ejemplo, puedes aportar más información en muchos casos sobre el resultado.

PRUEBAS DE INTEGRACIÓN

ESTRATEGIAS DE INTEGRACIÓN:

AD HOC

Es un término utilizado para la realización de pruebas de software ***sin planificación ni documentación,*** *pero puede aplicarse a las aplicaciones de una manera rápida pero no exhaustiva.*

Las pruebas están destinadas a ser ejecutadas sólo una vez, a menos que aparezca un fallo. Las pruebas ad hoc son el método de prueba menos formal. Como tal, ha sido criticado por no estar estructurado y, por lo tanto, los errores encontrados utilizando este método pueden ser más difíciles de reproducir (ya que no hay casos de prueba escritos).

Sin embargo, la ventaja de las pruebas ad hoc es que los errores importantes pueden ser encontrados rápidamente.

Las pruebas ad hoc son improvisadas: el tester busca encontrar errores de software por cualquier medio posible. Las pruebas ad hoc pueden ser vista como una pequeña versión de la predicción de errores, que a su vez es un versión menor del exploratory testing (https://www.youtube.com/watch?v=W0-PFvqmxPY)

PRUEBAS DE INTEGRACIÓN

DEFECTOS ENCONTRADOS HABITUALMENTE EN LAS PRUEBAS DE INTEGRACIÓN

- Gracias a este tipo de pruebas, lo defectos que puedes encontrar habitualmente son:

Cálculos incorrectos

Comportamiento incorrecto en aspectos funcionales y no funcionales

Flujo incorrecto de la información entre los componentes

Ḋefectos únicamente reproducibles en entornos de producción

Funcionamientos incorrectos respecto a la documentación del producto

http://oscarmoreno.com/pruebas-de-integracion/

PLAN DE INTEGRACIÓN

PLAN DE INTEGRACIÓN

El procedimiento de prueba de Integración independientemente de las estrategias mencionadas anteriormente son:

- *Preparar el Plan de Pruebas de Integración*
- *Diseñar los escenarios de prueba, los casos y los guiones.*
- *Ejecutar los casos de prueba seguidos de informar los defectos.*
- *Seguimiento y nueva prueba de los defectos.*
- *Los pasos 3 y 4 se repiten hasta que la integración se completa con éxito.*

PLAN DE INTEGRACIÓN

ID de caso de prueba	Caso de prueba Objetivo	Descripción del caso de prueba	Resultado Esperado
1	Verifique el enlace de la interfaz entre el módulo de inicio de sesión y buzón	Ingrese las credenciales de inicio de sesión y haga clic en el botón Iniciar sesión	Para ser dirigido al Buzón de Correo
2	Verifique el enlace de la interfaz entre el buzón y el módulo de eliminación de correos	Desde el buzón, seleccione el correo electrónico y haga clic en el botón Eliminar	El correo electrónico seleccionado debe aparecer en la carpeta Eliminado/Papelera

EJERCITEMOS LA MENTE….

- Realizar un caso de prueba de integración de acuerdo a la siguiente especificación:
  - Se ha desarrollado un software de atención de pacientes para un hospital y ya se encuentran terminados 4 módulos, admisión, empadronamiento, atención de consultorio e Internación.
  - Para poder internarse un paciente debe empadronarse y ser admitido al nosocomio.
  - Para poder atenderse por consultorio un paciente debe estar empadronado.

Id TC	Objetivo	descripción	Resultado
1	Validar un paciente no empadronado		

Id TC	Modulo	Objetivo	descripción	Resultado	estrategia
1	empadronamiento	Validar un paciente no empadronado	Ingresar el DNI del paciente y presionar el botón validar	Devuelve mensaje de paciente no existente	Top - down
2	Atención consultorio	Que un paciente se atienda en un servicio (consultorio)	Cargar al paciente con su DNI y generar un turno dando clic en turno	El paciente aparece en la lista de atención de un profesional medico.	
3	Atención consultorio	Buscar un consultorio externo	Cargar el dni en la interfaz de turnos y solicitar haciendo clic en SOLICITAR	Se abre la interfaz de selección de consultorio y profesional	
4	internación	Asignar habitación y piso a un paciente	Buscar una habitación disponible y asociarla al DNI y dar ACEPTAR	Que el paciente quede internado en esa habitación	

STUBS

validarPadron (dni)

retornar true

Especificación de requerimientos:

que un paciente pueda atenderse en un servicio/especialidad siempre y cuando este empadronado en el nosocomio

* Redactar un stubs y un driver para alguno de los casos de prueba/objetivos

BREVE DESCRIPCIÓN DE LOS PLANES DE PRUEBA DE INTEGRACIÓN:

Incluye los siguientes atributos:

1. Métodos/Enfoques para las pruebas (como se discutió anteriormente).
2. Ítems Alcances y Fuera de Alcances de las Pruebas de Integración.
3. Funciones y responsabilidades.
4. Requisitos previos para las pruebas de integración.
5. Entorno de prueba.
6. Planes de Riesgo y Mitigación.

MEJORES PRÁCTICAS/DIRECTRICES PARA LAS PRUEBAS DE INTEGRACIÓN

- Primero, determine la estrategia de prueba de integración que podría adoptarse y luego prepare los casos de prueba y los datos de prueba en consecuencia.

- Estudiar el diseño de la Arquitectura de la Aplicación e identificar los Módulos Críticos. Estos necesitan ser probados en prioridad.

MEJORES PRÁCTICAS/DIRECTRICES PARA LAS PRUEBAS DE INTEGRACIÓN

- Obtenga los diseños de interfaz del equipo de arquitectura y cree casos de prueba para verificar todas las interfaces en detalle. La interfaz con la base de datos/hardware externo/aplicación de software debe probarse en detalle.
- Después de los casos de prueba, son los datos de prueba los que juegan un papel fundamental.
- Tenga siempre preparados los datos simulados antes de ejecutarlos. No seleccione datos de prueba mientras ejecuta los casos de prueba.