Ingeniería de Software I
Pruebas de Integración

Juan Manuel Fernández Peña

Abril 2011

Idea general
Pruebas de Integración (1/3)

- Prueban grupos de unidades relacionadas; verifica su operación conjunta.
- Énfasis está en la interacción y no en funcionamiento individual.
- Si cada elemento fue adecuadamente probado, ¿no deberían funcionar bien al probarlas conjuntamente?
  - Sin embargo, el todo es más que la suma de sus partes.

Idea general
Pruebas de Integración (2/3)

- Pruebas de unidad:
  - no aseguran dominios de datos homogéneos,
  - pueden omitir algún comportamiento que sólo ocurre al interactuar con otro elemento.
- Pruebas de Integración identifican :
  - problemas de interfaces entre unidades,
  - falta de coherencia entre lo que se espera de una unidad y lo que se ofrece.

Idea general
Pruebas de Integración (3/3)

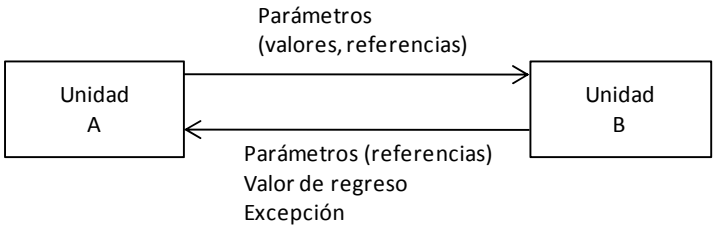
- Ejemplo: unidad A envía petición a unidad B usando un parámetro entero, posibles conflictos:
  - la primera suponía un intervalo [-100, 100] para los valores posibles, mientras la segunda suponía únicamente el intervalo [-10, 10];
  - puede ocurrir que una usa enteros de 16 bits y la otra enteros de 32 bits.

Pruebas de unidad previas

- Se hacen contra especificaciones, no siempre completas y cuidadosas
- O se hace con su implementación, dejando huecos en funciones especificadas
- Usualmente se aíslan de su contexto
  - Del propio sistema
  - Del software auxiliar
  - Del hardware

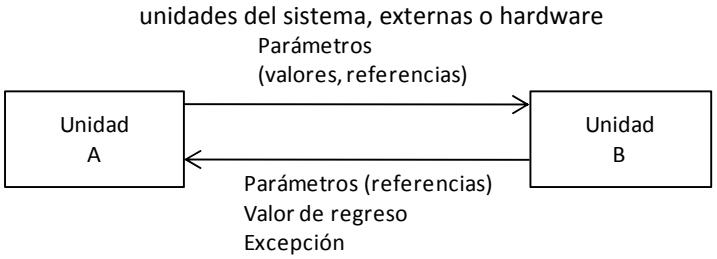
Interacciones

unidades del sistema, externas o hardware



- A pasa el control a B
- A pasa ciertos parámetros a B, como valores
- A tiene ciertas expectativas sobre la respuesta de B (poscondiciones), que pueden estar definidas más o menos formalmente o supuestas
- A recibe resultado de B
- A recibe el control de regreso de B
- A comparte con B valores de parámetros por referencia o variables globales

Interacciones



- B tiene ciertas expectativas sobre los parámetros que puede recibir (precondiciones), que pueden estar definidas más o menos formalmente o supuestas
- B tiene ciertas reglas internas que deben cumplirse (invariantes), que pueden estar definidas más o menos formalmente o supuestas
- Si expectativas de B no se cumplen o se rompen sus reglas internas, se produce una interrupción, que puede pasar como respuesta hacia A
- A y B pueden comunicarse usando un protocolo con datos viajando en una y otra dirección, por turnos.

Ejemplos de interacción

Unidad A	Unidad B	Interacción
Módulo de control de frenado de un vehículo	Módulo calcula aceleración con datos de posición en distintos tiempos	Se envían datos de posición, regresa aceleración
Módulo de nómina encargado de calcular pago de empleado	Manejador de Base de datos	Envían una consulta (query) y reciben un conjunto de datos (que puede ser vacío), un aviso de error o una excepción
Página html en cliente	Programa en servidor	clave y contraseña para ser validadas; respuesta o aviso de falla por exceso de tiempo u otra causa
Componente que almacena fotos digitales en un disco	Servicio del SO que informa espacio disponible	Solicita espacio disponible para decidir si puede almacenar la fotografía
Módulo encargado de tomar datos de temperatura	Tarjeta de adquisición de datos con sensor de temperatura conectado	Cada cierto tiempo solicita dato, lo lee y almacena

Problemas de interacción

- Problemas de interfaz entre unidades
- Problemas no funcionales (tiempo, recursos)
- Problemas de configuración (software funciona bien)
- Problemas de integridad

Problemas de interacción
interfaz

Tipo de defecto	Ejemplo o comentario	Causa posible
Parámetro seleccionado de manera errónea	Se tomó el parámetro “pre” como el que lleva el precio de un producto, pero en realidad era un valor de precedencia	Confusión por mala documentación, nombres inadecuados o descuido
Parámetros (y valor de retorno) de tipo distinto al esperado	Se envía un entero y se esperaba un número de punto flotante	Lenguaje sin refuerzo de tipos (era más frecuente en el pasado)
Carencia de aviso de excepción (en Unidad B)	Un valor no previsto genera una división por cero, causando que el programa se termine de modo abrupto	Deben considerarse posibles riesgos si los datos no cumplen condiciones preestablecidas y éstas deben informarse
Carencia de previsión sobre ocurrencia de excepciones	Excepción impide el fallo del programa, pero Unidad A la ignora dejando valores indefinidos	No se consideró que el software puede fallar de maneras no esperadas

Problemas de interacción interfaz

Tipo de defecto	Ejemplo o comentario	Causa posible
Valores inadecuados, aunque el tipo sea compatible	Se envía un entero de 32 bits pero se esperaba uno de 16 bits	Descuido entre tipos de una misma jerarquía o poco detallados en documentación
Valores que violan la definición del dominio de datos esperados	Un valor debe estar entre cero y 10000; llega valor negativo o mayor a 10000	Falta de comunicación entre autores de unidades o mala documentación
Valores con significado semántico diferente al esperado	Se envían datos en unidades del sistema decimal, pero se esperaban en sistema inglés (o datos en pesos y se esperaban en euros)	Falta documentación de semántica asociada con parámetros; unidades de medida de cualquier tipo, protocolos y cuestiones culturales.
Problema de protocolo	Una parte del diálogo entre unidades no cumple el orden del protocolo	Mala implementación del protocolo o desconocimiento del mismo

Problemas de interacción aspectos no funcionales

Tipo de defecto	Ejemplo o comentario	Causa posible
Problema de rendimiento	El tiempo conjunto es excesivo para las expectativas del cliente	Unidades ineficientes o usan recursos compartidos de manera ineficiente; por separado son rápidas pero una mala programación de hilos o control de procesos las vuelve lentas
Conflicto entre componentes	sufren abrazo mortal o pérdida de datos	No se maneja de manera adecuada la concurrencia
Fugas de memoria	Conforme operan las unidades, comienza a reducirse la memoria disponible, sin razón aparente	Problemas en el manejo de memoria, especialmente al liberar segmentos de manera descuidada. Son más frecuentes en lenguajes de bajo nivel
Recursos insuficiente para carga nominal	Al operar juntas se bloquean por falta de recursos o se degrada su funcionamiento	unidades abusan de recursos disponibles o no se calcularon necesidades conjuntas.

Problemas de interacción configuración

Tipo de defecto	Ejemplo o comentario	Causa posible
Invocación de versión obsoleta	interfaz que fue válida alguna vez, pero que ha sido desplazada por otra	Descuido al elegir los elementos a utilizar
Invocación de versión aún no instalada	interfaz que aún no está operando, aunque se haya anunciado	Descuido al elegir los elementos a utilizar
Invocación de componente no disponible	Se invoca una unidad que debiera existir, pero no se encuentra o está deshabilitada	bibliotecas no instaladas o dañadas, software de terceros en directorios equivocados o inaccesibles por cualquier causa (red desconectada, exceso de transacciones, bloqueo del firewall, falta de permisos)
Problema con unidad de hardware	Nunca recibe datos o estos no son los esperados	dispositivo requiera alimentación eléctrica adicional o alguna configuración que se omitió

Problemas de interacción integridad

Tipo de defecto	Ejemplo o comentario	Causa posible
Violación de integridad de datos	Una unidad elimina registros de una base de datos que requiere otra unidad	No se analizaron adecuadamente las responsabilidades de cada unidad y las necesidades de integridad
Estructura de datos incorrecta o inconsistente	una unidad altera archivos o registros en memoria compartida, que forman parte de la estructura de datos común	No se analizaron adecuadamente las responsabilidades de cada unidad y las necesidades de integridad

Estrategias

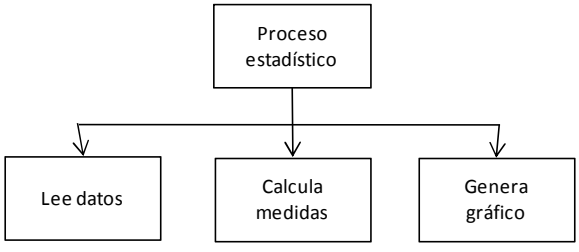
- Dos aspectos:
  - Conjunto de unidades a probar
    - Descomposición funcional
    - Comportamiento
  - Selección de valores para casos de prueba
    - Métodos ya vistos, pero asegurando que los valores obliguen a interacción

Conjuntos de unidades

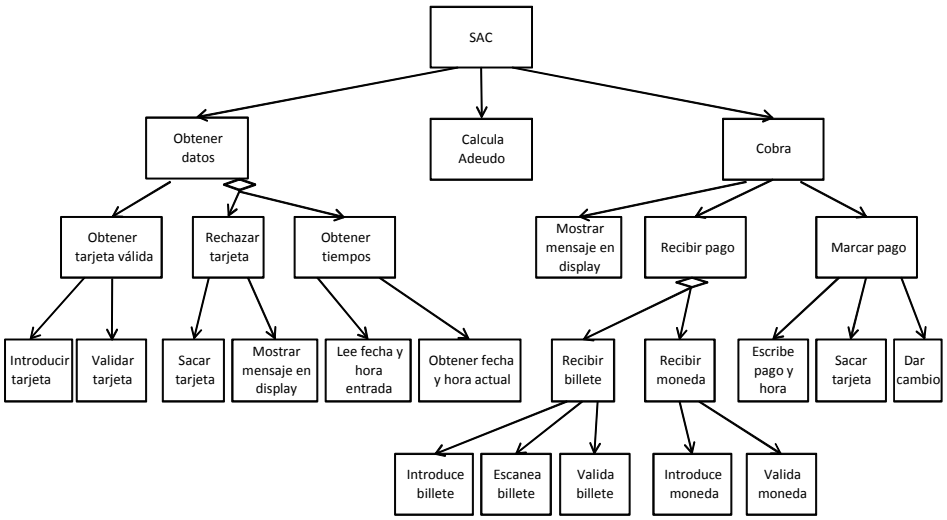
suponiendo descomposición funcional

- Big bang
- Descendente
- Ascendente
- Por parejas
- Vecindades
- Caminos de mensajes

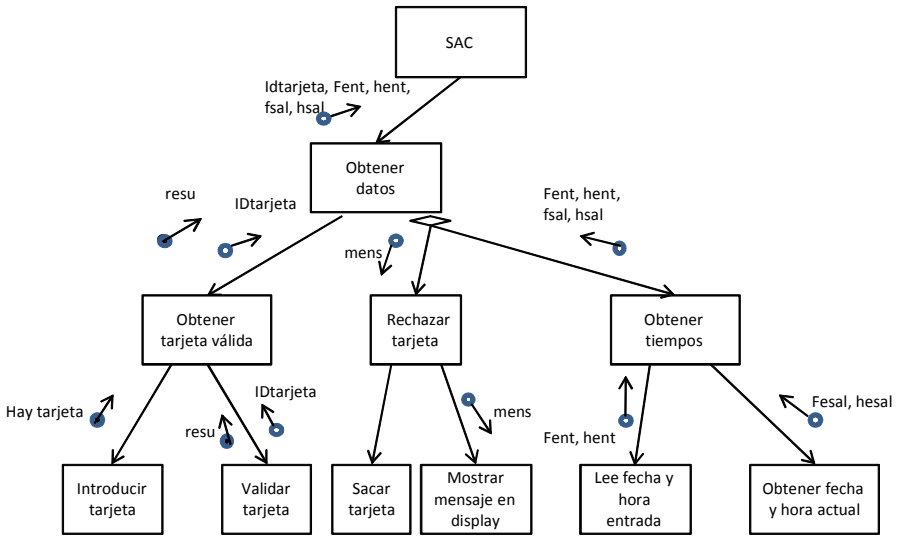
Supone árbol de módulos, como:



Ejemplo para ilustrar máquina de cobro de estacionamiento



Ejemplo: detalle



Manejadores y cabos

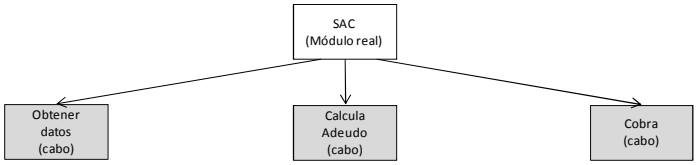
- Además de unidades a probar, puede haber otras que participan:
  - Manejadores: solicitan servicios de las unidades en prueba, les pasan parámetros
  - Cabos: ejecutan acciones auxiliares, calculan resultados, simulan elementos externos
- Razones:
  - Aún no codificadas
  - Complejas y se prefiere versión ligera
  - Se pruebas situaciones que ocurren rara vez

Big bang

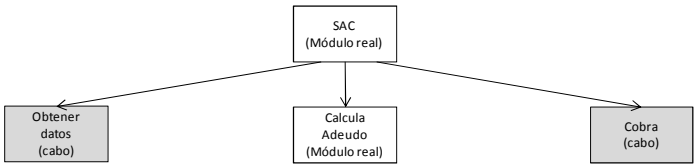
- Se prueba todo junto, produciendo una explosión; nadie sabe qué falló o por qué
- Unas fallas enmascaran otras
- Se usa en empresas sin cuidado en calidad
- No se recomienda

Descendente

- De la raíz hacia las hojas
- Raíz y cabos; raíz un módulo y cabos; …
- No especifican si uno a uno o en grupos por nivel



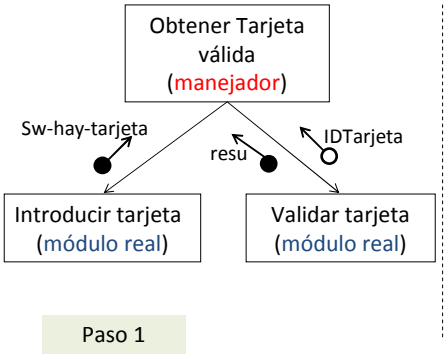
(a) Primer paso descendente

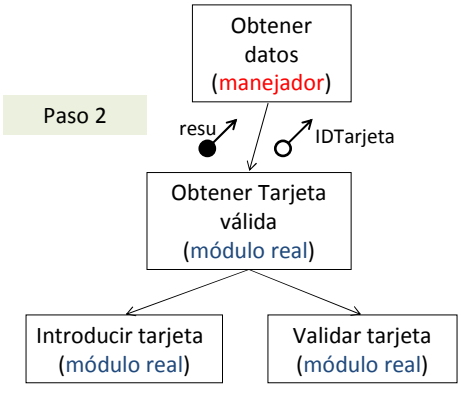


(b) Segundo paso descendente

Ascendente

- De las hojas (módulos que no llaman a otros) hacia la raiz.
- Cada módulo terminal se supone probado como unidad; se usa manejador para uno o más de ellos.
- No se especifica si primero todo un nivel o subiendo por el árbol
- Se adapta a subárboles





Parejas

- Cuando no hay un orden estricto de terminación de módulos, pueden irse probando pares conforme estén listos
- Se completa con manejadores y cabos
- Genera muchas pruebas

Vecindades

- Si se combina desarrollo ascendente y descendente, pueden formarse grupos de módulos relacionados
- Preferentemente dos niveles o tres; muchos cae en big bang
- A veces llamada estrategia sandwich

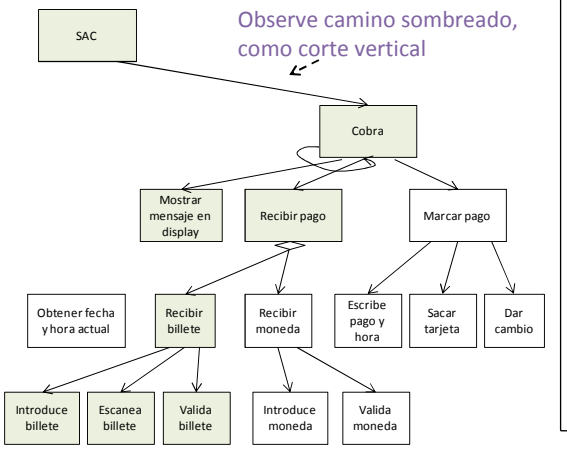
Caminos y mensajes

- En vez de estructura, se guía por el usuario, quien busca realizar tareas, más que activar módulos específicos
- Adecuado con casos de uso
- Muchas veces comienzan con una interacción del usuario con interfaz y prosiguen hasta tener una respuesta o llegar a un punto donde no hay más qué hacer

Caminos y mensajes ejemplo

- Probar cobro con un solo billete, hasta que saldo = 0

Camino de mensajes

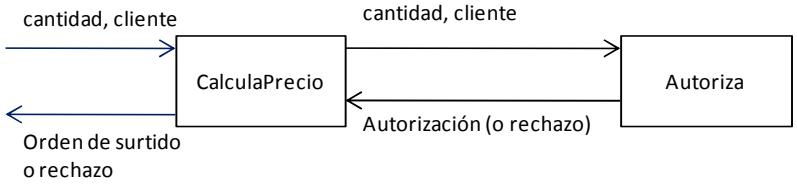


Fragmento de SAC
Llamada a Cobra
Fragmento de Cobra
Llamada a Mostrar mensaje
Código de Mostrar …
Regreso a Cobra
Fragmento de Cobra
Llamada a Recibir pago
Fragmento de Recibir pago
Llamada a Recibir billete
Fragmento de Recibir billete
Llamada a Introduce billete
Regreso a Recibir billete
Fragmento de Recibir billete
Llamada a Escanea billete
Regreso a Recibir billete
Fragmento de Recibir billete
Llamada a Valida billete
Regreso a Recibir billete
Regreso a Recibe pago
Regreso a Cobra
Fragmento de Cobra
Llamada a Mostrar mensajedisplay
Código de Mostrar …
Regreso a Cobra
Regreso a SAC

Selección de valores

- Preparar casos de prueba:
  - Todos los valores de una vez o
  - Secuencia, como en prueba de sistema
- Donde se invoca unidad, deben darse valores adecuados
  - Adecuados: si originan una interacción
- Dominio de entrada se separa en dos:
  - Datos que se consumen en la unidad
  - Datos que atraviesan la unidad (los que interactúan)

Ejemplo



- Ordinariamente, el módulo CalculaPrecio recibe un pedido y aplica reglas del negocio sobre descuentos a ciertos clientes o en virtud de la cantidad pedida.
- Si la cantidad rebasa un umbral (maxCant) requiere una autorización que corresponde a reglas del módulo Autoriza.
- Dominio de cantidad: (- ∞, ∞) en los enteros
- Datos que se consumen en CalculaPrecio: (- ∞, maxCant]
- Datos que atraviesan CalculaPrecio: (maxCant, ∞) <<= los importantes

Ejemplo (sigue)

Suponga maxCant = 1200

Valores de entrada	Resultado	Atraviesa
Cantidad <1, cliente =”CliConocido”	Rechazado por CalculaPrecio, ya que es una cantidad inaceptable	No
1 < cantidad < 1201, cliente =”CliConocido”	Calcula precio regresa orden de surtido con costo = precio * cantidad	No
Cantidad >1200, cliente =”CliConocido”	CalculaPrecio pasa los datos a Autoriza; como CliConocido está en lista, regresa autorización y CalculaPrecio emite la orden de surtido.	Sí
Cantidad >1200, cliente =”CliNuevo”	CalculaPrecio pasa los datos a Autoriza; como CliNuevo no está en lista, rechaza el pedido y CalculaPrecio emite un rechazo por volumen excesivo	Sí

Valores útiles para pruebas de integración, sombreados

Plan de pruebas de integración

- IEEE 829 general
  1. Identificación
  2. Elementos a probar
  3. Enfoque
  4. Criterio de aceptación o rechazo de un caso de prueba
  5. Criterio de suspensión
  6. Productos a entregar
  7. Tareas a realizar para satisfacer el proceso
  8. Necesidades ambientales
  9. Responsabilidades
  10. Personal necesario y si requieren entrenamiento.
  11. Calendario
  12. Riesgos y contingencias que pueden ocurrir en el proceso de prueba

Plan de pruebas de integración (1/3)

1. **Identificación**: alguna forma de reconocer planes concretos
2. **Elementos a probar**: módulos que se usarán, incluyendo cabos y manejadores.
3. **Enfoque**: estrategia a seguir y su justificación (top down, caminos de mensajes, etc.)
4. **Criterio de aceptación o rechazo de un caso de prueba**: como se sabe si pasa o no; estricto: debe coincidir con salida esperada. Otro: es aceptable para experto humano.

Plan de pruebas de integración (2/3)

5. **Criterio de suspensión**: hasta terminar todos los casos de prueba, hasta cumplir cierta cobertura, hasta agotar cierto tiempo disponible
6. **Productos a entregar**: desde el propio plan, los casos y procedimientos de prueba, los resultados.
7. **Tareas a realizar para satisfacer el proceso**: tareas de preparación (casos de prueba, manejadores y cabos), de ejecución (automática o manual), de evaluación, de reporte
8. **Necesidades ambientales**: hardware, software y espacio de trabajo necesarios.

Plan de pruebas de integración (3/3)

9. **Responsabilidades**: quién es responsable de cada cosa: módulos, cabos, manejadores, preparación, evaluación, etc.
10. **Personal necesario y si requieren entrenamiento**: en casos que lo ameriten
11. **Calendario**: tiempos para las actividades planeadas.
12. **Riesgos y contingencias:** problemas que no son seguros pero pueden afectar el proceso y qué hacer si ocurren.

Ejemplo de plan de prueba

correspondiente al ejemplo ascendente (1/2)

1. Identificación	Plan Prueba Int002
2. Elementos a probar	Obtener datos (manejador), Obtener Tarjeta válida (manejador y módulo real), Introducir Tarjeta, Validar tarjeta
3. Enfoque	Ascendente
4. Criterio aceptación	La salida coincide con la esperada
5. Criterio suspensión	Hasta lograr cobertura total del código
6. Productos a entregar	Plan de prueba, casos de prueba, lista de casos ejecutados con su resultado, lista de problemas identificados
7. Tareas	Preparar casos de prueba, preparar manejadores auxiliares, ejecutar casos de prueba, evaluar resultados, preparar informe de problemas
8. Necesidades ambientales	Computadora de escritorio con Eclipse, Java 1.6, Junit 4 y EclEmma (para cobertura)

Ejemplo de plan de prueba

correspondiente al ejemplo ascendente (2/2)

9. Responsabilidades	Probador: preparar casos de prueba, ejecutar casos de prueba, evaluar resultados, preparar informe de problemas Programador: preparar manejadores auxiliares
10. Personal	El programador será el mismo desarrollador; se requiere un probador que conozca pruebas de dominios (particiones) y de caminos
11. Calendario	Preparar casos de prueba: 2 días Preparar manejadores: 3 días Ejecutar casos de prueba: 1 día Evaluar resultados y reportar: 1 día Margen por riesgos: 2 días
12. Riesgos y contingencias	R1: dependencia imprevista del hardware de tarjeta C1: negociar más tiempo R2: programador no entrega manejadores C2: probador los realiza; descuento a programador