TEMA 6

PRUEBAS DEL SOFTWARE

01 [Sep. 2006] Según Boehm, validar es:

a) ¿Estamos construyendo el producto correcto?. (pág. 420)
b) ¿Estamos construyendo correctamente el producto?.
c) ¿El producto funciona?.
d) ¿Los productos de una fase satisfacen las condiciones impuestas al principio de dicha fase?

Nota: La a) es validar, mientras que la b) es verificar. El que producto funcione no significa que lo haga correctamente.

1. DEFINICIONES

01 [Jun. 2006] [Jun. 2007] [Jun. 2008] [Sep. 2008] ¿Quién define "Probar" como proceso de ejecutar un programa con el fin de encontrar errores?

a) McCabe.
b) Firesmith.
c) Myers. (pág. 420)
d) IEEE 1990.

Nota: La definición de Myers es quizás la más sencilla de todas y explica claramente que se busca con las pruebas.

02 [Jun. 2006] En sentido estricto, ¿cuál de las siguientes NO es un error?

a) La pulsación de una tecla equivocada.
b) Un paso de procesamiento incorrecto en un programa. (pág. 420)
c) La diferencia entre el valor calculado y el verdadero.
d) Todas las anteriores son errores.

Nota: La respuesta b) es un Defecto del software.

03 [Jun. 2006] Según el IEEE 1990, "la incapacidad de un sistema para realizar las funciones requeridas dentro de los requisitos de rendimientos especificados" es:

a) Un defecto.
b) Un error.
c) Un fallo. (pág. 421)
d) Ninguna de las anteriores.

04 [Sep. 2006] ¿Cuál de las siguientes sentencias es correcta?

a) Un fallo suele ser la consecuencia de un defecto. (pág. 422)
b) Un defecto suele ser el origen de un error.
c) Un fallo suele ser el origen de un error.
d) Un error suele ser la consecuencia de un fallo.

Nota común a 03 a 04: En otras palabras un fallo se produce cuando el sistema no acaba su proceso. La secuencia es error, defecto, fallo.

3. EL PROCESO DE PRUEBA

01 [Jun. 2007] [Jun. 2008] [Sep. 2008] El proceso de pruebas consta de las siguientes actividades:

a) Planificar las pruebas, diseñar las pruebas, ejecutar las pruebas y evaluar las pruebas.
b) Planificar las pruebas, diseñar las pruebas, ejecutar las pruebas, evaluar las pruebas y depurar si es necesario.
c) Planificar las pruebas, diseñar las pruebas, ejecutar las pruebas, evaluar las pruebas, depurar si es necesario y analizar los errores. (pág. 424)
d) Ninguna de las anteriores.

Nota: El proceso de pruebas comienza con la generación de un plan de pruebas, sigue con el diseño de pruebas específicas, que se ejecutan, evaluándose a partir de los resultados obtenidos, se depuran los defectos y se analiza la estadística de errores.

4. TÉCNICAS DE DISEÑO DE CASOS DE PRUEBA

01 [Jun. 2007] [Sep. 2007] Los enfoques principales para el diseño de casos de pruebas son:

a) Estructural, funcional y aleatorio. (pág. 426)
b) Estructural, caja blanca y aleatorio.
c) Funcional, caja negra y aleatorio.
d) Ninguna de las anteriores.

Nota: Los tres enfoque principales para el diseño de casos de pruebas son el estructural o de caja blanca, funcional o de caja negra y aleatorio.

02 [Jun. 2006] ¿En qué consiste el enfoque funcional o de caja negra?

a) En probar todas las entradas y salidas del programa. (pág. 427)
b) En probar todos los posibles caminos de ejecución.
c) En probar todas las entradas al programa sin tener en cuenta las salidas.
d) En probar todas las salidas del programa y su naturaleza.

Nota: El enfoque consiste en estudiar la especificación de las funciones, la entrada y la salida para derivar los casos. La prueba ideal es la indicada en negrita.

03 [Jun. 2006] ¿En qué tipo de enfoque la prueba exhaustiva consiste en probar todas las entradas posibles?

a) Estructural.
b) Funcional.
c) Aleatorio (pág. 427)
d) De caja blanca

Nota: El enfoque aleatorio utiliza modelos (generalmente estadísticos) de las psibles entradas para crear los casos de prueba. La prueba exhaustiva incluirá todas las entradas posibles.

5. PRUEBAS ESTRUCTURALES

01 [Jun. 2007] [Sep. 2007] ¿Cuál de los siguientes criterios de cobertura lógica es el que ofrece una menor seguridad de detención de defectos?

a) Cobertura de decisiones.
b) Cobertura de condiciones.
c) Cobertura de sentencias. (pág. 429)
d) Ninguna de las anteriores.

02 [Sep.. 2005] [Sep. 2007] [Jun. 2008] ¿Cuál de los siguientes criterios de cobertura lógica ofrece una menor seguridad de detección de defectos?

a) De sentencias, (pág, 429)
b) De condiciones.
c) De condición múltiple.
d) De decisiones.

03 [Jun. 2005] ¿Cuál de los siguientes criterios de cobertura lógica es el que cuesta menos en número de ejecuciones de programa?

a) De condiciones.
b) De decisiones.
c) De condición múltiple
d) De sentencias. (pág. 429)

Nota común a 01 a 03: Con el criterio de cobertura de sentencias se generan los casos de pruebas necesarios para que cada sentencia del programa se genere al menos una vez. Por ello, es el menos seguro pero también el que menos cuesta en número de ejecuciones de programa.

04 [Jun. 2005] [Jun. 2006] [Sep. 2007] [Jun. 2008] ¿Cuál de los siguientes criterios de cobertura consiste en diseñar tantos tanto casos como sean necesarios para que cada decisión tenga por lo menos una vez un resultado verdadero y por lo menos una vez un resultado falso?

a) De sentencias.
b) De condiciones.
c) De decisiones. (pág. 429)
d) De condición múltiple.

Nota : Con el criterio de cobertura de decisiones se generan los casos de pruebas para que todas las condiciones tengan por lo menos una vez un resultado verdadero y uno falso. En general si cumple este criterio cumple también el de sentencias.

5.1 Utilización de la complejidad ciclomática de McCabe

01 [Sep. 2006] El criterio de prueba de McCabe:

a) Es equivalente al de cobertura de condiciones.
b) Es bastante similar al de cobertura de condiciones.
c) Consiste en definir un caso de prueba para cada camino posible.
d) Ninguna de las anteriores. (pág. 431)

02 [Sep. 2005] [Sep. 2007] [Jun. 2008] ¿En qué tipo de pruebas se utiliza la complejidad ciclomática de McCabe?

a) Funcionales.
b) Estructurales. (pág. 431)
c) Aleatorias.
d) De caja negra..

03 [Jun. 2005] [Sep. 2005] La complejidad ciclomática de McCabe se utiliza en:

a) Pruebas estructurales. (pág. 431)
b) Pruebas funcionales.
c) Pruebas aleatorias.
d) Pruebas de caja negra.

04 [Jun. 2005] ¿Cuál de las siguientes métricas se utiliza en el diseño de pruebas para ver el número de caminos independientes en un grafo?

a) McCabe. (pág. 431)
b) AVL.
c) Beizer.
d) Myers.

05 [Sep. 2006] La complejidad ciclomática de McCabe es una métrica que se suele utilizar para medir:

a) La legibilidad del software.
b) La reusabilidad del software.
c) La concisión del software.
d) Nada de lo anterior. (pág. 431)

Nota común a 01 a 05: La métrica de McCabe indica el número de caminos independientes existentes en un grafo, por lo que sirve para indicar el número de casos de prueba. En general, es equivalente a una cobertura de decisiones.

06 [Jun 2005] [Sep. 2005] A partir de un grafo G, siendo a el número de arcos y n el número de nodos, se calcula la complejidad V(G) como:

a) V(G) = a + n – 2.
b) V(G) = 2a + n.
c) V(G) = a – n + 2. (pág. 431)
d) V(G) = n – (a + 2).

07 [Jun. 2006] En la complejidad ciclomática de McCabe V(G) = c + 1 c es:

a) Números de nudos de condición. (pág. 431)
b) Número de arcos.
c) Número de nodos.
d) Número de regiones.

08 [Jun. 2006] La complejidad ciclomática de McCabe se mide por:

a) CC = a - n + 2, siendo “a” el número de nodos y “n” el número de arcos.
b) CC = número de regiones cerradas del grafo del programa mas 1.
c) CC = número de nodos de condición.
d) CC = Ninguna de los anteriores. (pág. 431)

Nota común a 06 a 08: Existen tres formas de medir la complejidad ciclomática; la primera en función del número de arcos y nudos (a + n – 2), la segunda en función del número de regiones cerradas (r) y la tercera en función del número de nodos de condición (c + 1).

09 [Jun. 2005] Según Beizer, V(G) indica:

a) Un límite máximo del número de casos de pruebas para un programa.
b) El número exacto de casos de pruebas para un programa.
c) Un límite mínimo del número de casos de prueba para un programa. (pág. 433)
d) Ninguna de las anteriores

Nota: La experimentación de Beizer dio como resultado que V(G) marca el límite mínimo de casos de pruebas y que cuando V(G) es mayor que 10 la probabilidad de defectos crece.

6. PRUEBA FUNCIONAL

01 [Jun. 2007] [Jun. 2008] [Sep. 2008] Entre las técnicas de diseño de caja negra se encuentra:

a) **Conjetura de errores. (pág. 434 y sig.)**
b) Complejidad ciclomática de McCabe.
c) Criterio de cobertura de sentencias.
d) Ninguna de las anteriores.

02 [Jun. 2005] ¿Cuál de las siguientes técnicas de diseño de casos NO es de caja negra?

a) Particiones o clases de equivalencia.
b) Análisis de valores límite.
c) **Complejidad ciclomática de McCabe. (pág. 434)**
d) Conjetura de errores.

03 [Jun. 2006] [Jun. 2007] [Sep. 2008] ¿Cuál de los siguientes técnicas NO es usada para el diseño de caja negra?

a) **Diagrama de flujo de control. (pág. 434)**
b) Particiones o clases de equivalencia.
c) Análisis de valores límites.
d) Conjetura de errores.

Nota común a 01 a 03: Las técnicas de diseño de pruebas funcionales o de caja negra son particiones o clases de equivalencia, análisis de valores límite y conjetura de errores.

6.1 Particiones o clases de equivalencia

01 [Jun. 3007]Una de las reglas para identificar las clases de equivalencia es que si se especifica un rango de valores para los datos de entrada se crearán...

a) Dos clases válidas y una clase no válida.
b) Dos clases válidas y dos clases no válidas.
c) **Una clase válida y dos clases no válidas. (pág. 435)**
d) Una clase válida y una clase no válida.

Nota: Evidentemente hay una clase válida (la del rango) y dos clases no válidas (la que queda por abajo y la que está por arriba).

6.2 Análisis de valores límite (AVL)

01 [Jun. 2007] [Sep. 2007] Una de las reglas para identificar las clases de equivalencia en el caso del análisis de valores límite es que si una condición de entrada especifica un rango de valores se deben generar:

a) Casos para los extremos del rango y casos no válidos para situaciones justo más allá de los extremos. (pág. 439)
b) Un caso para el extremo superior del rango.
c) Un caso para el extremo inferior del rango.
d) Un caso para el extremo superior del rango y un caso para el extremo inferior del rango.

Nota: Si una condición de entrada especifica un rango de valores se deben generar casos para los extremos del rango y casos no válidos justo más allá de los extremos.

6.3 Conjetura de errores

01 [Sep. 2006] Sobre la conjetura de errores ¿cuál de las siguientes directrices es falsa?

a) El valor 0 indica que no hay errores. (pág. 440)
b) Hay que suponer que el programador haya podido malinterpretar las especificaciones.
c) Hay que considerar las acciones de un usuario “completamente tonto”.
d) En los casos de entrada de listas, hay que considerar el caso de no introducir ningún valor y el de un sólo valor.

Nota: El valor 0 indica una situación propensa a error.

9. DOCUMENTACIÓN DEL DISEÑO DE LAS PRUEBAS

01 [Jun. 2006] [Sep. 2006] [Jun. 2007] [Sep. 2008] Los documentos de trabajo de las pruebas vienen especificados en el estándar:

a) IEEE 1980.
b) IEEE 829. (pág. 443)
c) IEEE 1980.
d) Ninguna de ellas.

02 [Jun. 2005] ¿Cuál de los siguientes documentos pertenece a la documentación del diseño de las pruebas?

a) Informe de incidente.
b) Especificación de casos de prueba. (pág. 443)
c) Informe resumen de las pruebas.
d) Histórico de pruebas.

Nota común a 01 a 02: Los documentos del diseño de pruebas incluidos en el estándar IEEE 829 son el plan de pruebas, la especificación del diseño de pruebas, la especificación de caso de prueba y la especificación de procedimiento de prueba.