UTN.BA FACULTAD REGIONAL BUENOS AIRES

SECRETARÍA DE EXTENSIÓN UNIVERSITARIA FRBA UTN

Centro de e-Learning

Centro de Formación, Investigación y Desarrollo de Soluciones de e-Learning.
UTN - FRBA. Secretaría de Cultura y Extensión Universitaria
Medrano 951 2do piso (1179) // Tel. +54 11 4867 7589 / Fax +54 11 4032 0148 // e-learning@sceu.frba.utn.edu.ar

www.sceu.frba.utn.edu.ar/e-learning

Professional Testing Master

Presentación:

En esta Segunda Unidad del curso, nos adentraremos en los detalles de la estrategia de testing, incluyendo los métodos de diseño de casos de prueba, y el testing en entornos especializados.

Unidad 2:

Técnicas de testing y diseño de casos de prueba (Test Cases)

Objetivo:

Al terminar la Unidad los participantes:

- ❑ Se habrán familiarizado con los diferentes tipos de cobertura de test.
- ❑ Estarán en condiciones de diseñar casos de prueba efectivos usando pruebas de caja negra y caja blanca.
- ❑ Conocerán los aspectos fundamentales a tener en cuenta al encarar el testing en entornos especializados.

Contenido

1. Visión interna y externa del testing
2. Prueba de caja negra
3. Prueba de caja blanca
4. Testing especializado

Visión interna y externa del testing

Como se mencionó en la Unidad 1, el testing se puede realizar de dos maneras:

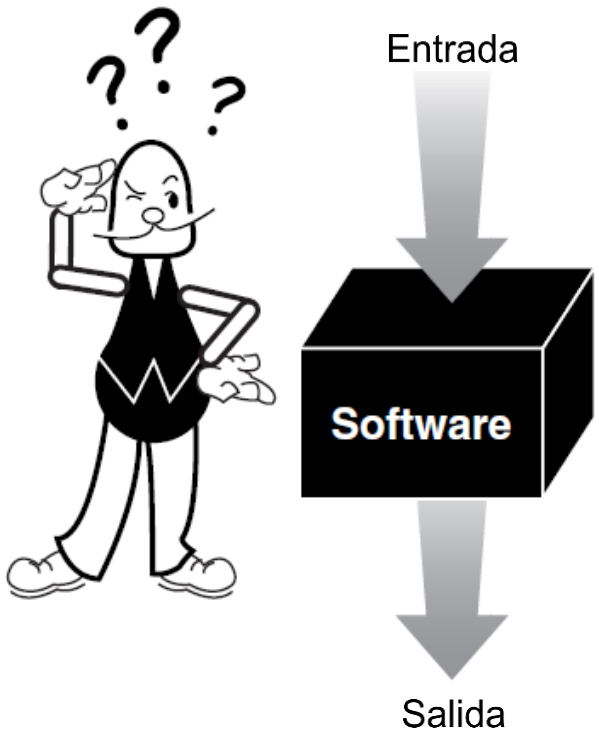
Conociendo la estructura interna del programa

→ Testing de caja blanca (white-box)

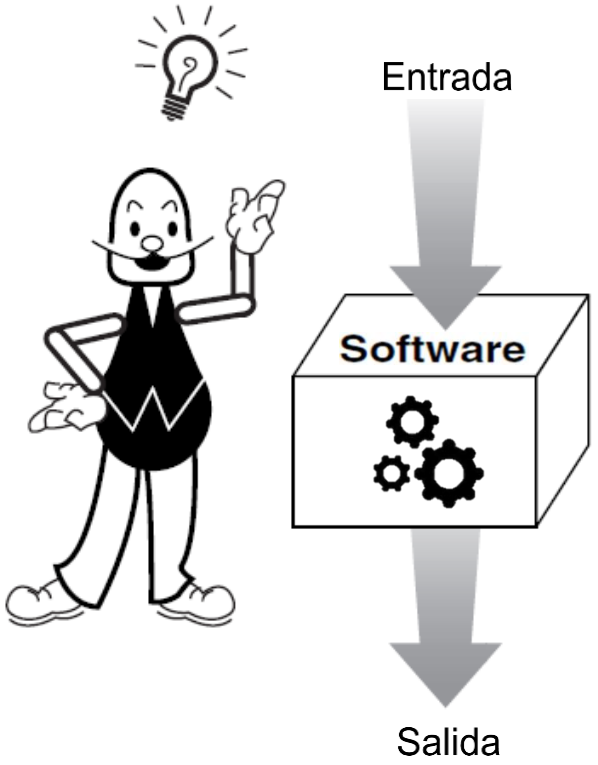
Guiándose sólo por los resultados esperados de acuerdo a la especificación

→ Testing de caja negra (black-box)

Visión interna y externa del testing (cont.)



Testing de caja negra



Testing de caja blanca

Pruebas de caja negra

En esta sección:

- ❑ Partición de equivalencia
- ❑ Análisis de valor límite

Partición de equivalencia

Es la clasificación de los posibles casos de prueba en distintos grupos (clases de equivalencia), cada uno de los cuales contiene los casos que son similares entre sí, es decir, prueban lo mismo o revelan el mismo tipo de error (bug).

Las clases de equivalencia pueden definirse de acuerdo con los siguientes lineamientos:

Partición de equivalencia (cont.)

1. Si una condición de entrada especifica un rango, se define una clase de equivalencia válida y dos inválidas (inferior y superior al rango).
2. Si una condición de entrada requiere un valor específico, se define una clase de equivalencia válida y dos inválidas (inferior y superior al valor).
3. Si una condición de entrada especifica un miembro de un conjunto, se define una clase de equivalencia válida y una inválida.
4. Si una condición de entrada es booleana, se define una clase válida y una inválida.

Análisis de valor límite

Un mayor número de errores ocurre en las fronteras del dominio de entrada y no en el “centro”.

En lugar de seleccionar algún elemento de una clase de equivalencia, este método selecciona casos de prueba en los “bordes” de la clase.

Se intenta seleccionar valores:

- Justo en los límites
- Apenas por encima
- Apenas por debajo

Análisis de valor límite (cont.)

El análisis de valor límite centra la exploración en los límites

Prueba de caja blanca

En esta sección:

- Prueba de ruta básica
  - Notación de grafo de flujo
  - Rutas de programa independientes
  - Derivación de casos de prueba
- Prueba de la estructura de control
  - Prueba de condición múltiple
  - Prueba de ciclo

Prueba de caja blanca (cont.)

Prueba de ruta básica:

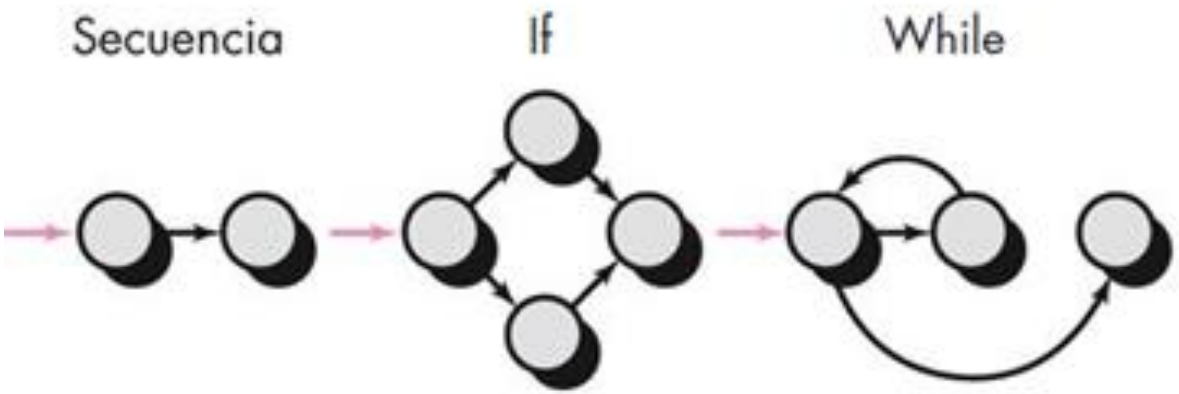
Garantiza ejecutar toda instrucción en el programa, al menos una vez durante la prueba.

Notación de grafo de flujo:

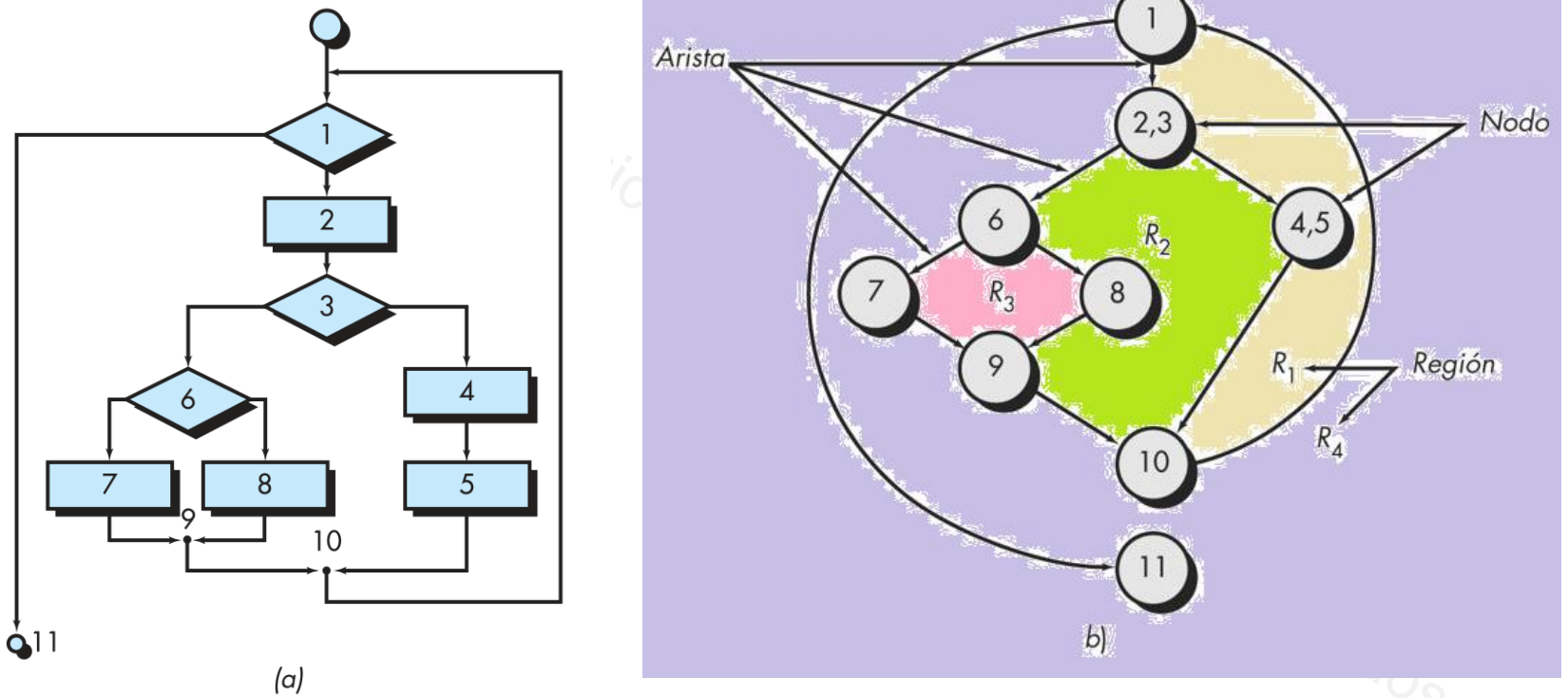
Se genera a partir del diseño (pseudocódigo, diagramas de flujo) representando todas las construcciones estructuradas del programa.

Notación de grafo de flujo

Los constructos estructurados en grafo de flujo forman:



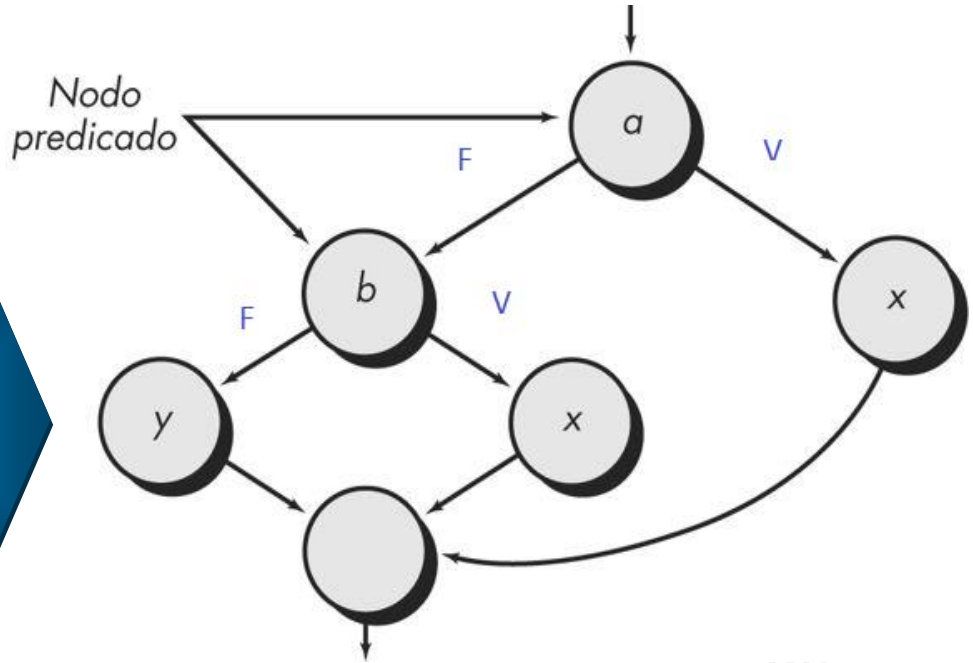
Notación de grafo de flujo (cont.)



a) Diagrama de flujo vs. b) grafo de flujo

Notación de grafo de flujo (cont.)

```
Si a o b Entonces
    procedimiento x
Sino
    procedimiento y
Fin Si
```



Representación de condiciones compuestas

Rutas de programa independientes

Una ruta independiente es cualquiera que introduce al menos un nuevo conjunto de instrucciones de procesamiento o una nueva condición en el programa.

En el ejemplo del grafo anterior:

1-11
1-2-3-4-5-10-1-11
1-2-3-6-8-9-10-1-11
1-2-3-6-7-9-10-1-11

} Son un conjunto de rutas básicas

1-2-3-4-5-10-1-2-3-6-8-9-10-1-11 No es una ruta básica

Rutas de programa independientes (cont.)

La complejidad ciclomática (cyclomatic complexity) es una métrica que cuantifica la complejidad lógica de un programa.

Define el número de rutas independientes del conjunto básico de un programa y le brinda una cota superior para el número de pruebas que debe realizar a fin de asegurar que todas las instrucciones se ejecutaron al menos una vez.

Rutas de programa independientes (cont.)

A partir del grafo de flujo, la complejidad ciclomática se puede calcular de tres formas:

1. El número de regiones del grafo de flujo
2. E − N + 2

   donde E es el número de aristas del grafo de flujo y N el número de nodos del grafo de flujo
3. P + 1

   donde P es el número de nodos predicado contenidos en el grafo de flujo

Derivación de los casos de prueba

Los pasos para obtener los casos de prueba son:

1. Usando el diseño o el código como base, dibujar el grafo de flujo G correspondiente.
2. Determinar la complejidad ciclomática V(G).
3. Determinar un conjunto básico de rutas independientes. V(G) es una cota superior sobre el número de rutas independientes.
4. Preparar casos de prueba que fuercen la ejecución de cada ruta en el conjunto básico.

Prueba de condición múltiple

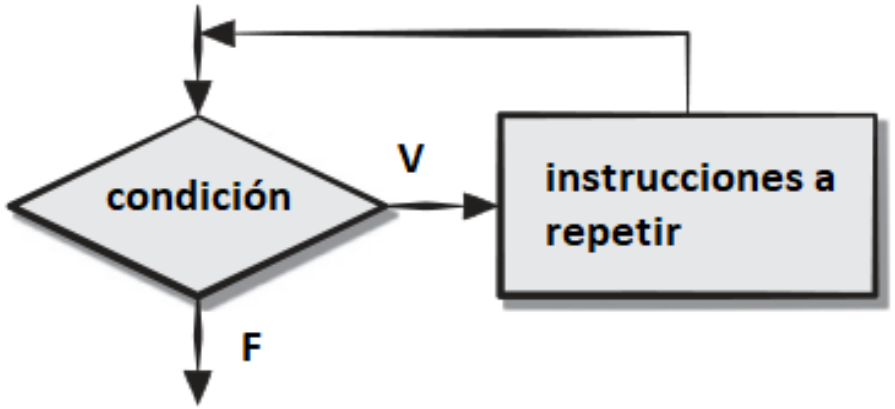
Este método se enfoca en la prueba de cada condición del programa para asegurar que no contiene errores.

Para esto se deben generar casos en forma de tablas de verdad que cubran todas las posibilidades de las condiciones compuestas del programa.

Es más efectivo que la prueba de ruta básica

Prueba de ciclo

Esta técnica de prueba se enfoca exclusivamente en la validez de las construcciones de tipo ciclo o bucle.



Prueba de ciclos

Se deben ejecutar las siguientes pruebas, donde n es el número máximo de pasadas del ciclo:

1. Saltar por completo el ciclo.
2. Sólo una pasada a través del ciclo.
3. Dos pasadas.
4. m pasadas, donde m < n es el valor típico de pasadas.
5. n − 1, n, n + 1 pasadas. Por supuesto n + 1 no debe ser posible pero es lo que se debe intentar.

Testing especializado

En esta sección:

- ❑ Prueba de interfaces gráficas de usuario
- ❑ Prueba de arquitecturas cliente-servidor
- ❑ Prueba de documentación y ayuda
- ❑ Prueba para sistemas de tiempo real

Prueba de interfaces gráficas de usuario

La complejidad de las GUI ha crecido

Mayor dificultad en el diseño y ejecución de los casos de prueba.

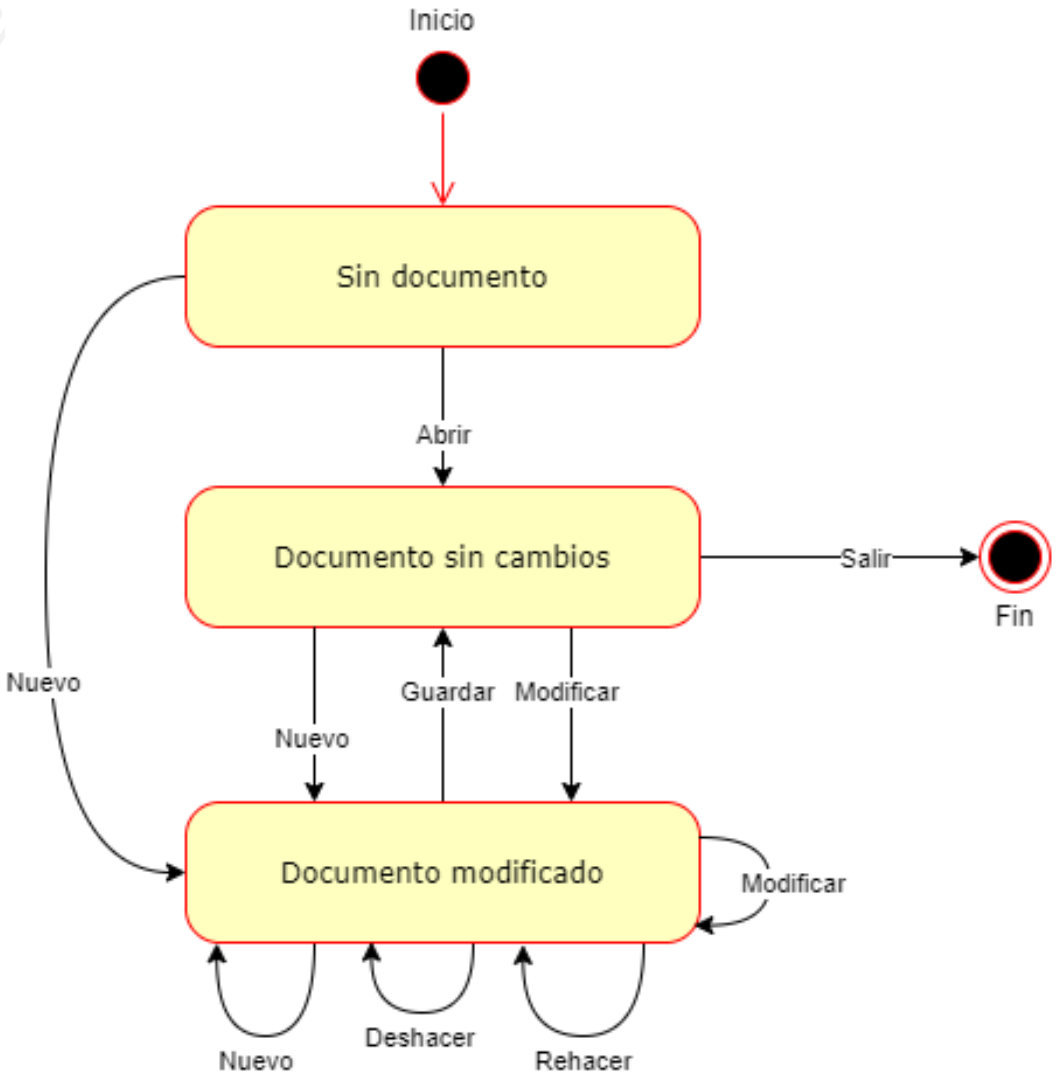
Diagramas de estados

Herramientas automatizadas

Prueba de GUI basada en modelo

Utiliza la información contenida en el diagrama de estado UML como la base para el diseño de los casos de prueba.

Ejemplo de diagrama de estados GUI



Prueba de arquitecturas cliente-servidor

La naturaleza distribuida de estas arquitecturas dificulta el testing.

Las pruebas se efectúan en tres etapas:

1. Modo de desconectado
2. Cliente conectado con el servidor
3. Monitoreo de red y rendimiento

Prueba de arquitecturas cliente-servidor (cont.)

Las pruebas incluyen:

- Pruebas funcionales de aplicación
- Pruebas de servidor
- Pruebas de base de datos
- Pruebas de transacción
- Pruebas de comunicación de red

Prueba de documentación y ayuda

Se deben verificar estos aspectos entre otros:

- ¿La documentación describe con precisión cómo lograr cada modo de uso?
- ¿Los ejemplos son precisos?
- ¿La terminología, descripciones de menú y respuestas del sistema son consistentes con el programa real?
- ¿La solución de problemas puede lograrse con facilidad usando la documentación?

Prueba para sistemas de tiempo real

Se agrega el tiempo como variable, lo que aumenta la complejidad.

Una estrategia consiste en seguir estos pasos:

- Prueba de tareas (cada tarea en forma aislada)
- Prueba de comportamiento (modelos simulados)
- Prueba intertarea (comunicación en tiempo real)
- Prueba de sistema (software y hardware)

Esperamos hayan disfrutado y aprovechado del estudio y las actividades propuestas en esta unidad!!!!!!!!!!

UTN.BA FACULTAD REGIONAL BUENOS AIRES

SECRETARÍA DE EXTENSIÓN UNIVERSITARIA FRBA UTN

Centro de e-Learning

Centro de Formación, Investigación y Desarrollo de Soluciones de e-Learning.
UTN - FRBA. Secretaría de Cultura y Extensión Universitaria
Medrano 951 2do piso (1179) // Tel. +54 11 4867 7589 / Fax +54 11 4032 0148 // e-learning@sceu.frba.utn.edu.ar

www.sceu.frba.utn.edu.ar/e-learning