Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

Clase 7
Cobertura de grafos aplicada

IIC3745 – Testing

Rodrigo Saffie

rasaffie@uc.cl

7 de agosto de 2020

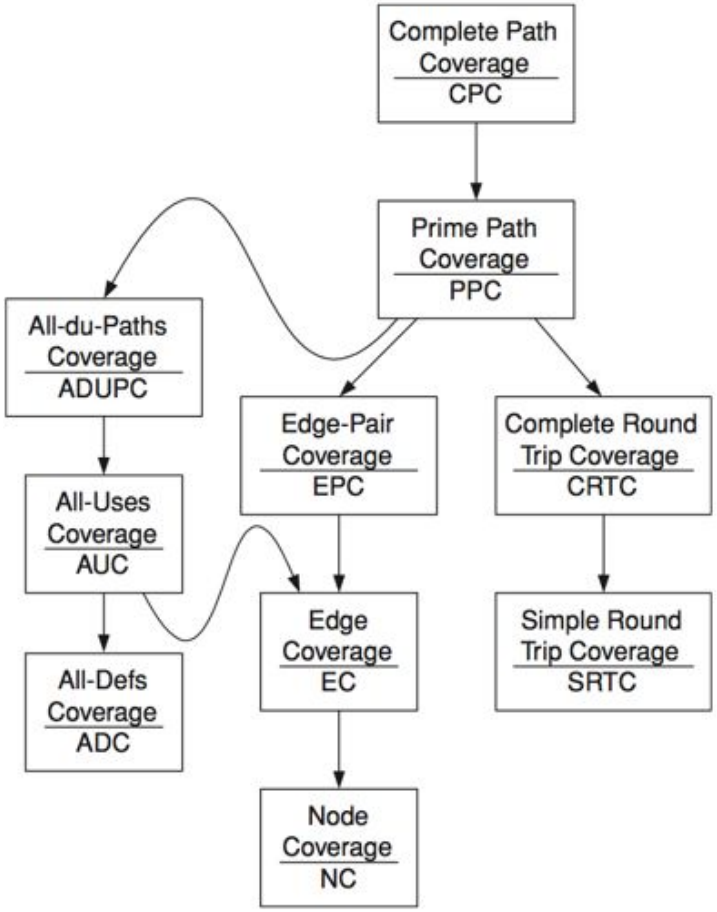
1. Criterios de cobertura
    - Cobertura basada en grafos
        - Estructurales
        - Flujo de información
    - Cobertura de grafos aplicada

Criterios de cobertura en grafos

- Estructural:
  - **NC**: Cobertura de nodos
  - **EC**: Cobertura de aristas
  - **EPC**: Cobertura de pares de aristas
  - **PPC**: Cobertura de camino primo
  - **SRTC**: Cobertura simple con *round-trips*
  - **CRTC**: Cobertura completa con *round-trips*
  - **CPC**: Cobertura de camino completo
  - **SPC**: Cobertura de camino específico
- Flujo de información
  - **ADC**: *All-defs coverage*
  - **AUC**: *All-uses coverage*
  - **ADUPC**: *All-du-paths coverage*

Subsumir

"Incluir algo como componente en una clasificación más abarcadora"



Cobertura de grafos aplicada

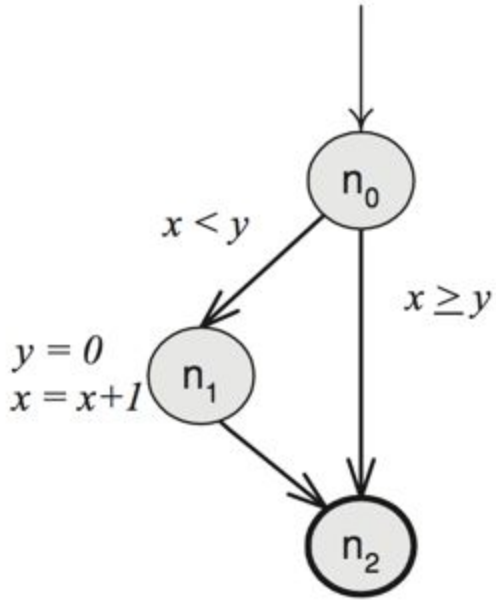
- Código fuente
- Elementos de diseño
- Especificación de diseño
- Casos de uso

Cobertura de código fuente

- Principal aplicación de los criterios de cobertura en grafos, que son representados como diagramas de control de flujo (***CFG**: control flow graph*).
  - **Nodos**: Instrucciones o secuencias de instrucciones (bloques básicos)
  - **Aristas**: Opciones de flujo (ramificaciones)

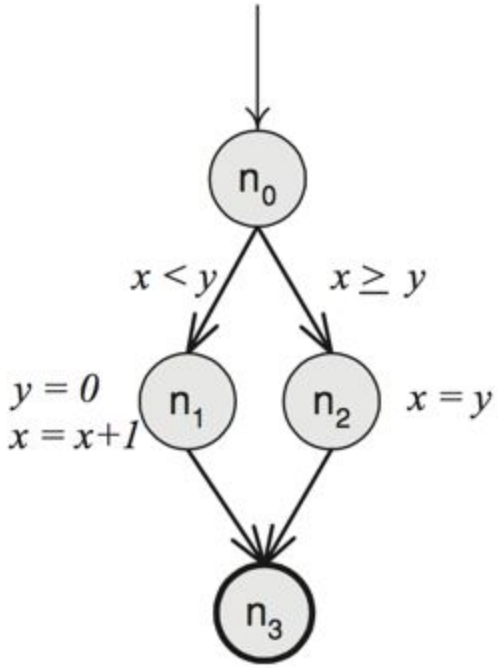
CFG: If

```
if ( x < y )
{
   y = 0;
   x = x + 1;
}
```



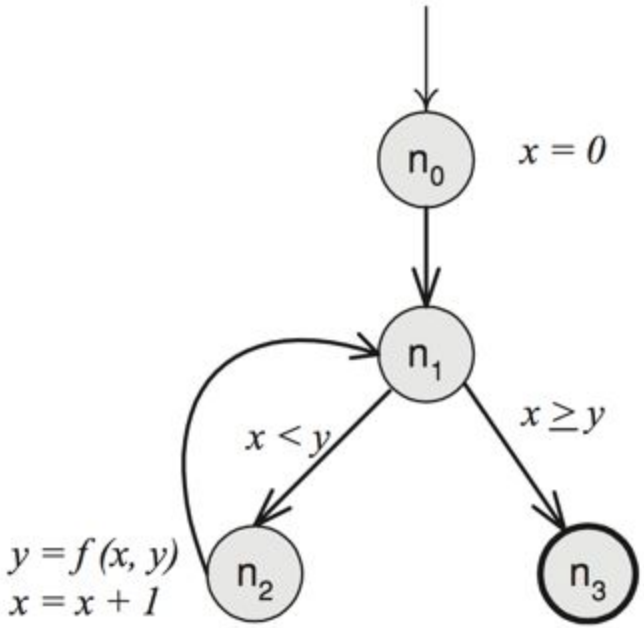
CFG: If-else

```
if (x < y)
{
   y = 0;
   x = x + 1;
}
else
{
   x = y;
}
```



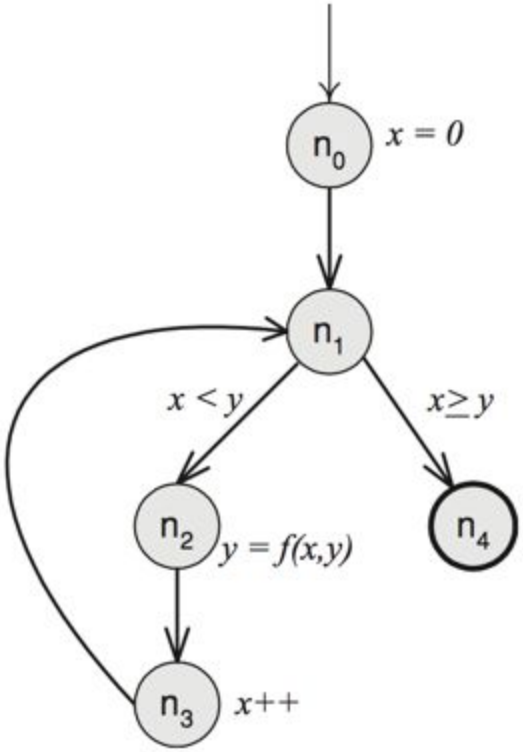
CFG: While

```
x = 0;
while (x < y)
{
   y = f (x, y);
   x = x+1;
}
```



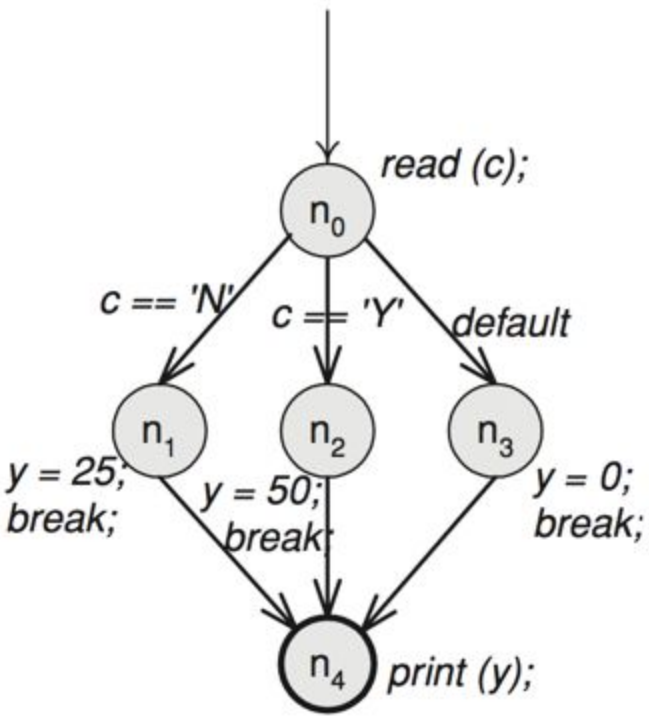
CFG: For

```
for (x = 0; x < y; x++)
{
   y = f(x,y);
}
```



CFG: Flujo de información

```
read (c);
switch (c)
{
case 'N':
     y = 25;
     break;
case 'Y':
     y = 50;
     break;
default:
     y = 0;
     break;
}
print (y);
```



Cobertura de grafos aplicada

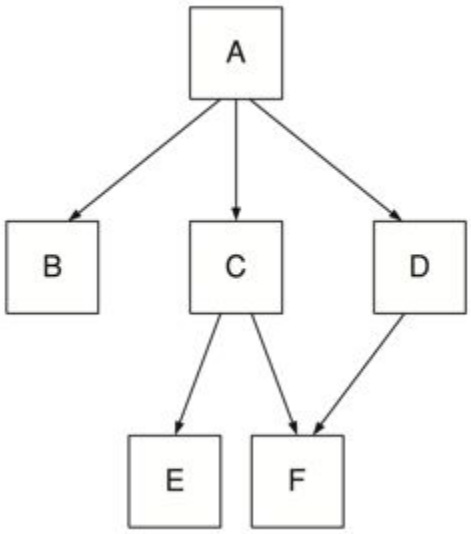
- Código fuente
- **Elementos de diseño**
- Especificación de diseño
- Casos de uso

Cobertura para elementos del diseño: **Estructural**

- Los grafos estructurales suelen representar el acoplamiento entre componentes
- No son muy útiles para encontrar defectos
- A través de *grafos de llamadas* se puede representar:
  - Dependencia entre modulos
  - Herencia y Polimorfismo

Grafo de llamadas

- **Nodos**: unidades de software (ej: métodos)
- **Aristas**: llamadas



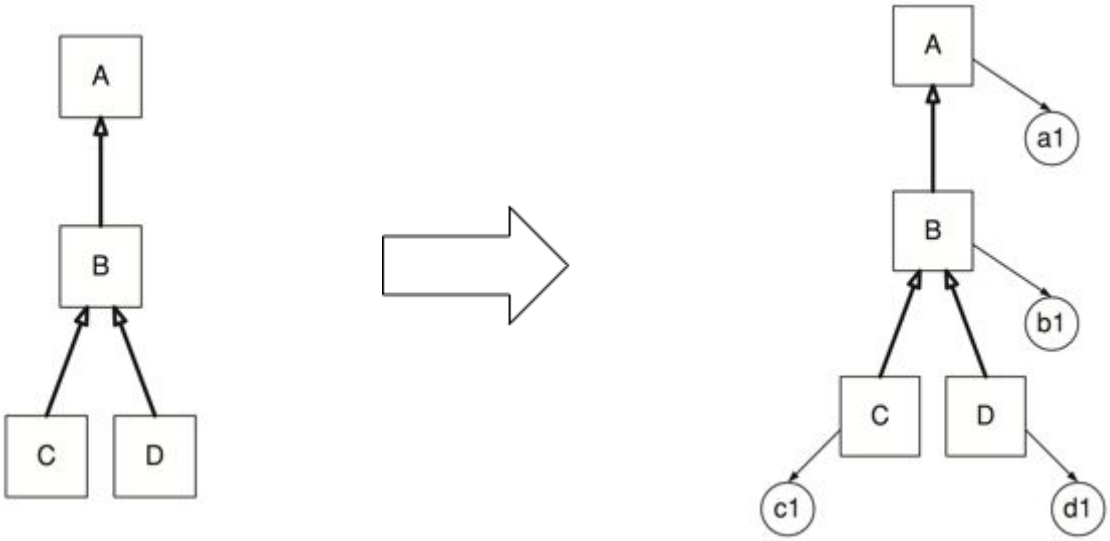
- **Cobertura de nodos**: visitar cada unidad al menos una vez
- **Cobertura de aristas**: ejecutar cada llamada al menos una vez

Grafo de llamadas: clases / módulos

- Útil para representar relaciones de métodos dentro de una clase
- Generalmente entre clases puede no ser útil si existe bajo acoplamiento
  - Se generan grafos disconexos

Herencia y Polimorfismo

- No existe consenso en la mejor forma de probarlo estructuralmente
- Las clases no son *testeables*, por lo que se agregan nodos de instancias al grafo de herencia.



Herencia y Polimorfismo: Cobertura

- Cobertura de Nodos
  - Crear un objeto de cada clase
  - Poco confiable ya que no incluye ejecución
- Cobertura de Aristas
  - Cobertura agregada: cobertura de llamadas por lo menos para una instancia de cada clase en la jerarquía
  - Cobertura total: cobertura de llamadas para cada instancia de cada clase en la jerarquía

Cobertura para elementos del diseño
Flujo de datos

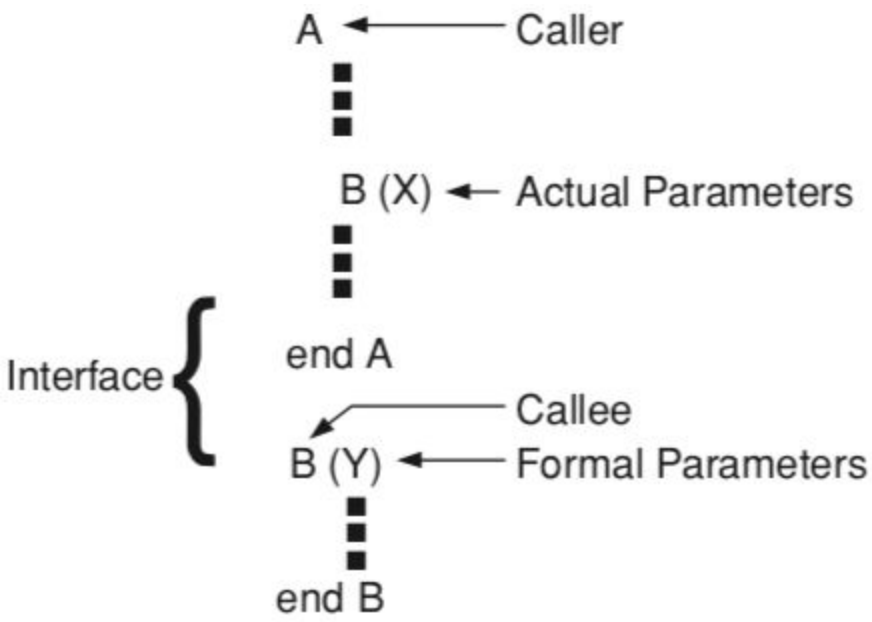
- Son grafos complejos y difíciles de analizar
  - Los parámetros pueden cambiar de nombre entre llamadas
  - Hay múltiples formas de compartir información
  - Encontrar *def* y *uses* es difícil

- Son útiles al momento de diseñar *tests* de integración

Cobertura para elementos del diseño
Flujo de datos

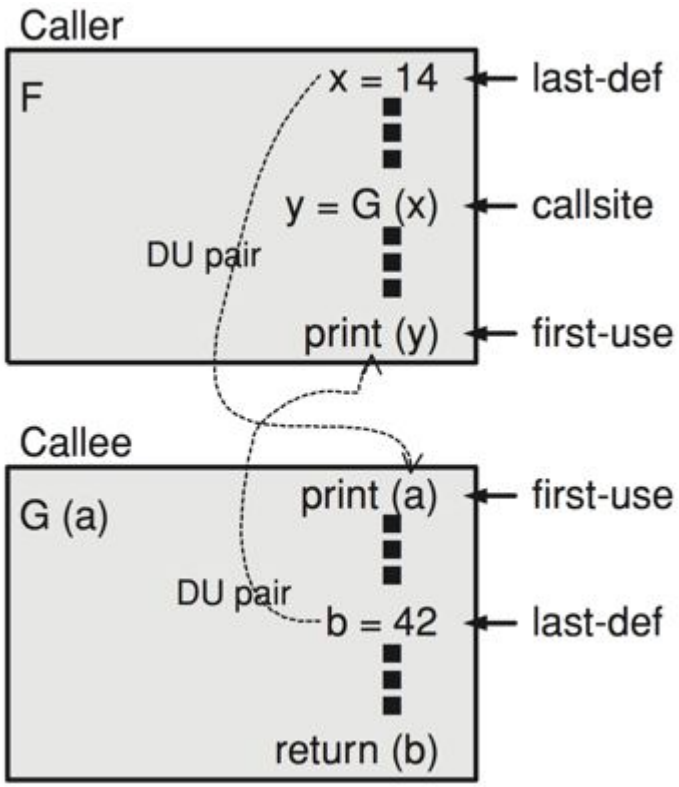
- Definiciones
  - ***Caller***: unidad que llama a otra
  - ***Callee***: unidad que es llamada
  - ***Call site***: lugar donde ocurre la llamada
  - ***Actual parameter***: valor en el *caller*
  - ***Formal parameter***: valor en el *callee*
  - ***Interface***: Mapeo de parámetro actual a formal

Cobertura para elementos del diseño
Flujo de datos



- Probar todo es muy costoso, pero probar la interfaz entrega un grado de seguridad razonable

Pares-DU entre unidades: ejemplo



Cobertura de grafos aplicada

- Código fuente
- Elementos de diseño
- **Especificación de diseño**
- Casos de uso

Cobertura para especificación de diseño

- Describe el comportamiento que debe exhibir un elemento de *software*

- La implementación puede (o no) reflejar la especificación

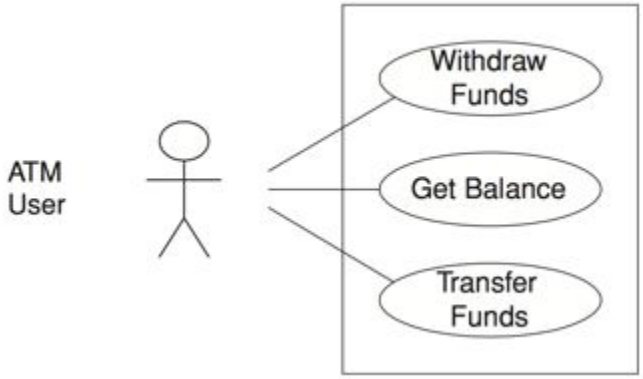
- Existen 2 formas de describir comportamiento:
  - Restricciones de secuencia
  - Según estado (máquina de estados finitos)

Cobertura de grafos aplicada

- Código fuente
- Elementos de diseño
- Especificación de diseño
- **Casos de uso**

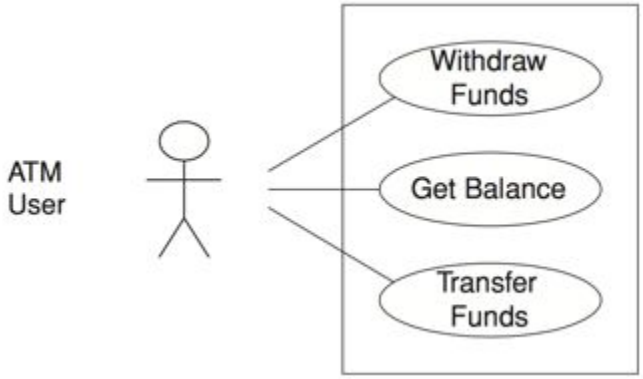
Casos de Uso

- Son utilizados generalmente para expresar requerimientos de *software*.
- Ayudan a expresar el flujo de la aplicación.



Casos de Uso

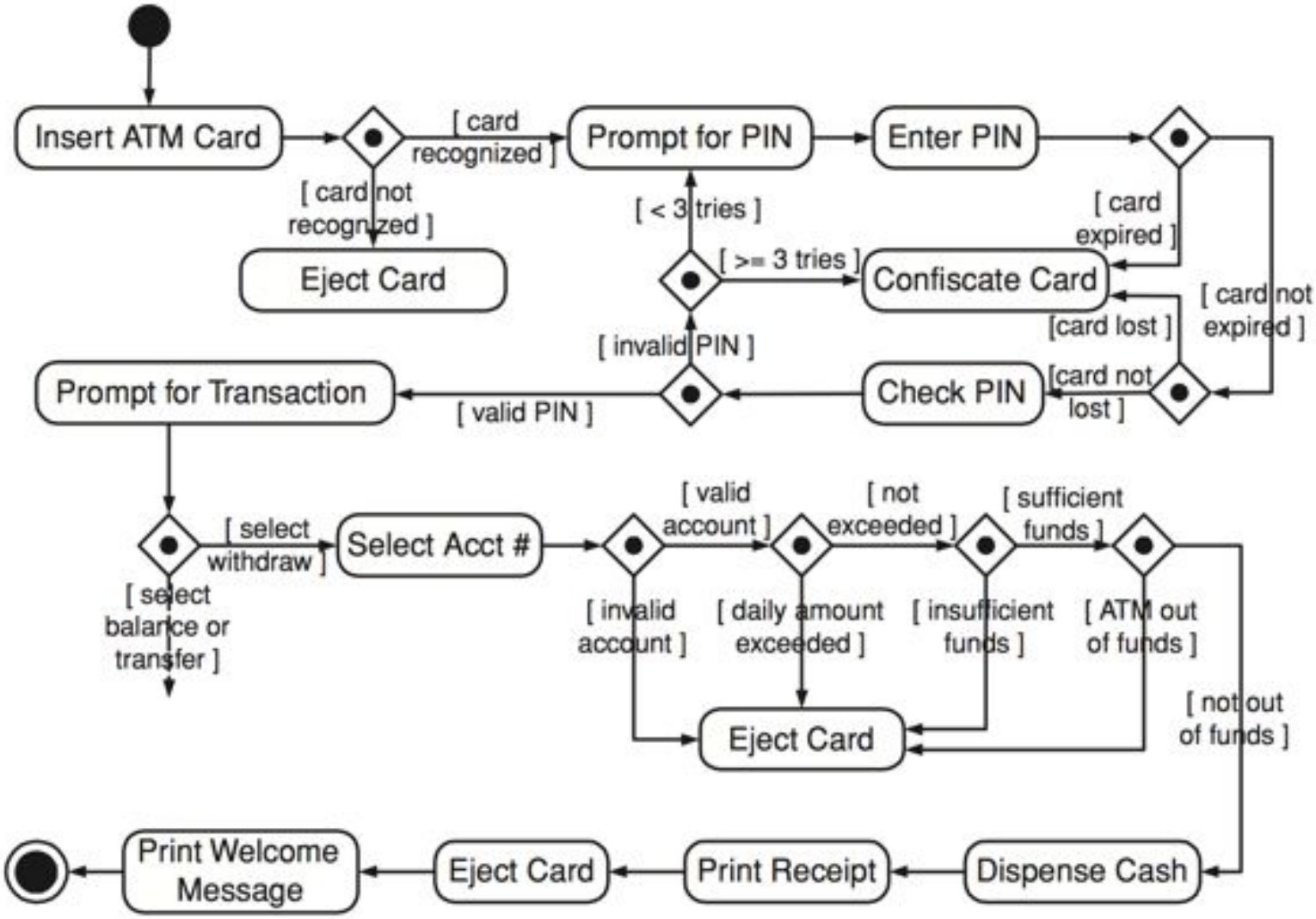
- *Node Coverage*: cubrir cada caso de uso
- No es muy útil analizar estos diagramas



Diagramas de actividad (o flujo)

- Indican el flujo entre las actividades
- Actividades deben modelar pasos a nivel del usuario
- 2 tipos de nodos:
  - Estados de acción
  - Ramificaciones
- En general estos diagramas tienen características deseables:
  - Pocos *loops*
  - Predicados simples

Giro en cajero automático



Cobertura en diagramas de actividad

- Flujos de datos no aplica

- Estructural:
  - Escenario: un camino completo a través del diagrama
    - Debe tener sentido semántico para el usuario
    - Número de caminos es finito a menudo
  - *Node Coverage*
  - *Edge Coverage*
  - Cobertura de camino específico (SPC)

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

Clase 7
Cobertura de grafos aplicada

IIC3745 – Testing

Rodrigo Saffie

rasaffie@uc.cl

7 de agosto de 2020