

# Introduction to Software Testing

## Chapter 2.4

### Graph Coverage for Design Elements

**Paul Ammann & Jeff Offutt**

[www.introsoftwaretesting.com](http://www.introsoftwaretesting.com)

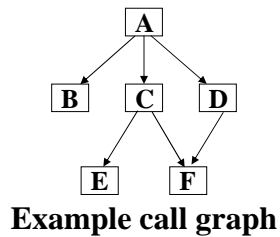
This version is abbreviated.  
Data flow coverage is omitted.

## OO Software and Designs

- **Emphasis on modularity and reuse puts complexity in the design connections**
- **Testing design relationships is more important than before**
- **Graphs are based on the connections among the software components**
  - Connections are dependency relations, also called couplings

## Call Graph

- The most common graph for structural design testing
- **Nodes** : Units (in Java – methods)
- **Edges** : Calls to units



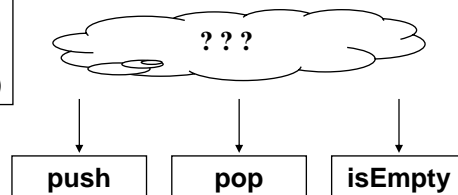
**Node coverage** : call every unit at least once (method coverage)

**Edge coverage** : execute every call at least once (call coverage)

## Call Graphs on Classes

- Node and edge coverage of class call graphs often do not work very well
- Individual methods might not call each other at all!

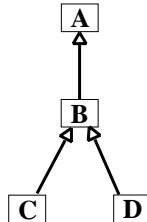
```
Class stack
public void push (Object o)
public Object pop ( )
public boolean isEmpty (Object o)
```



Other types of testing are needed – do not use graph criteria

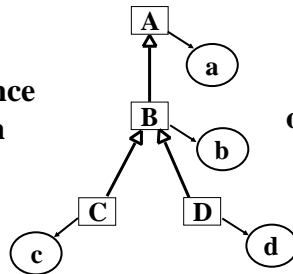
# Inheritance & Polymorphism

**Caution** : Ideas are preliminary and not widely used



Example inheritance hierarchy graph

Classes are not executable, so this graph is not directly testable  
We need objects



objects

What is coverage on this graph ?

# Coverage on Inheritance Graph

- Create an object for each class ?
  - This seems weak because there is no execution
- Create an object for each class and apply call coverage?

**OO Call Coverage** : TR contains each reachable node in the call graph of an object instantiated for each class in the class hierarchy.

**OO Object Call Coverage** : TR contains each reachable node in the call graph of every object instantiated for each class in the class hierarchy.

- **Data flow is probably more appropriate ...**  
Data flow not covered in undergraduate course

## Summary

- **Applying structural coverage techniques (non-data flow) to these graphs is easy**
- **The hard part is the data flow :**
  - Variables are defined in one method, passed as a parameter, then used in a called method ...
  - Class variables are defined in one method and used in another ...
  - Values are transferred to another class ...
  - Values are sent by message to another web component ...