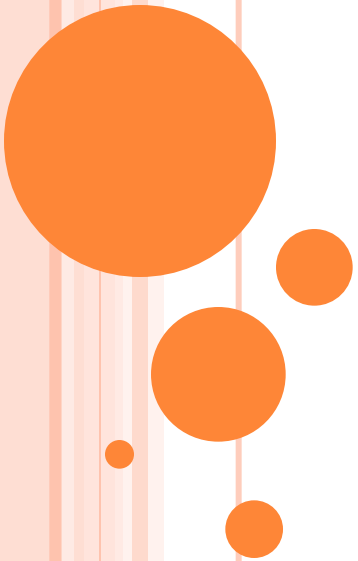
Software Engineering



LECTURE-33

Software Testing

Topics Covered

- Testing Strategy
- Testing Equivalence Class
- Unit Testing
- Equivalence Testing
- System Testing

System/Software Testing

- Error detection and removal
- determine level of reliability
- well-planned procedure - Test Cases
- done by independent quality assurance group (except for unit testing)

Test Strategy

- UNIT TESTING (Module testing)
  - debuggers, tracers
  - programmers
- INTEGRATION TESTING
  - communication between modules
  - start with one module, then add incrementally
- SYSTEM TESTING
  - manual procedures, restart and recovery, user interface
  - real data is used
  - users involved
- ACCEPTANCE TESTING
  - user-prepared test data
  - Verification Testing, Validation testing, Audit Testing

- White Box Testing
  - knowing internal working ,and exercising different parts
  - test various paths through software; if exhaustive testing is impossible, test high-risk paths
- Black Box Testing
  - knowing functions to be performed and testing whether these are performed properly
  - correct outputs from inputs
    DBs accessed/updated properly
  - test cases designed from user requirements
  - appropriate at Integration, Systems and Acceptance testing levels

Testing Equivalence Cases

- Two inputs are in the same Equivalence Class if they are handled similarly by system
  - eg. data field valid value in 1-50
  - So, 20, 38, 1, 47 belong to the same Equivalence Class
  - no need to test multiple values from same Equivalent Class
  - Bounds testing
    - eg. test 38, then end points 1 and 50
  - test valid and invalid equivalence classes
  - reduces the number of test cases required

Example: 3 inputs

I_1 has 10 equivalence classes

I_2 has 10 equivalence classes

I_3 has 10 equivalence classes

Total tests cases required:
10 x 10 x 10 = 1000 test cases.

Dependency Islands

- Each output is usually not dependent on all inputs

Example: suppose we have 6 inputs I_1,.. I_6 and 3 outputs O_1, .., O_3

Suppose O_1 depends on I_1, I_2, I_3

O_2 depends on I_4, I_5

O_3 depends on I_6

If each input has 5 equivalence classes:

To test I_1 we need 5 test cases

To test I_2 we need 5 test cases

To test I_1 and I_2 together, we need 5 x 5 test cases

Thus for all 67 inputs, we need 5^6 = 15,625 test cases

Using dependency islands:

For O_1: test only I_1, I_2, I_3 : 5^3 test cases

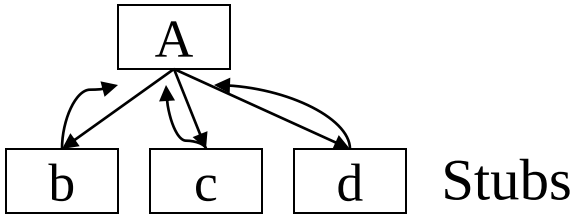
For O_2: test only I_4, I_5 : 5^2 test cases Total test cases = 155

For O_3: test only I_6 : 5 test cases

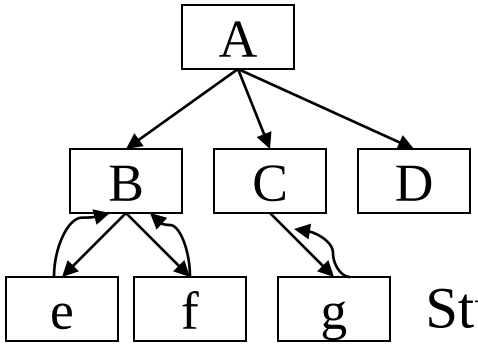
Stub Testing (Stubs and Drivers)

Unit and Integration testing

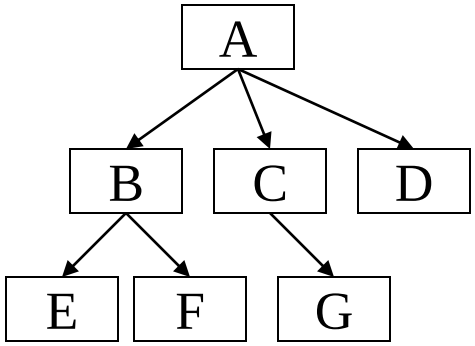
Top-Down Integration Testing



Stubs

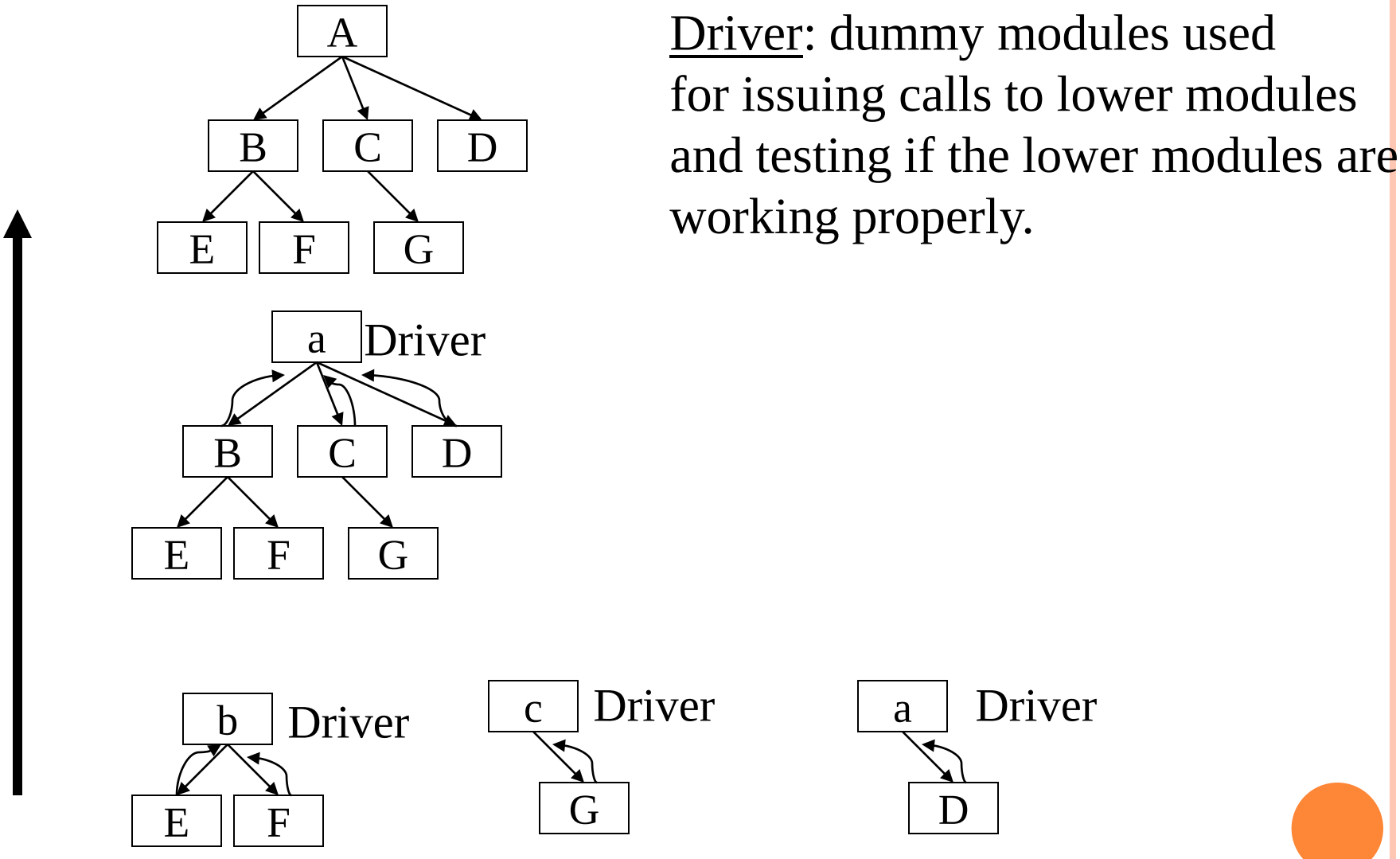


Stubs



Stubs: dummy modules used for testing if higher level modules are working properly.

Bottom-Up Integration testing



Driver: dummy modules used for issuing calls to lower modules and testing if the lower modules are working properly.

- Cost of developing stubs and drivers
  - generally, Driver modules are easier to develop -- so, bottom-up integration testing is less costly.
- With Top-Down Integration Testing, major modules are coded and tested first - strong psychological boost when major modules are done.
- With Bottom-Up Integration testing, no working program can be demonstrated until the last module is tested -- major design errors may not be detected till the end, when entire programs may need revision!
- Meet-in-the-middle approach may be best.

System Testing

- Recovery Testing
  - forces software failure and verifies complete recovery
- Security Testing
  - verify proper controls have been designed
- Stress Testing
  - resource demands (frequency, volume, etc.)

Acceptance Testing

- Alpha Testing (Verification testing)
  - real world operating environment
  - simulated data, in a lab setting
  - systems professionals present
    - observers, record errors, usage problems, etc.
- Beta Testing (Validation Testing)
  - live environment, using real data
  - no systems professional present
  - performance (throughput, response-time)
  - peak workload performance, human factors test, methods and procedures, backup and recovery - audit test