Wrap-up of testing topics

• Testing in the large
• Non-functional testing concerns

Testing in the large

Module testing
– testing a single module
Integration testing
– integration of modules and subsystems
System testing
– testing the entire system
Acceptance testing
– performed by the customer

Module testing

Scaffolding needed to create the environment in which the module should be tested
– stubs
  • modules used by the module under test
– driver
  • module activating the module under test

Testing a functional module

Integration testing

Big-bang approach
– first test individual modules in isolation
– then test integrated system
Incremental approach
– modules are progressively integrated and tested
  • can proceed both top-down and bottom-up according to the USES relation

Integration testing and USES relation
M1 USES M2 and M2 IS_COMPOSED_OF {M2,1, M2,2}

CASE 1
Test M1, providing a stub for M2 and a driver for M1
Then provide an implementation for M2,1 and a stub for M2,2

CASE 2
Implement M2,2 and test it by using a driver,
Implement M2,1 and test the combination of M2,1 and M2,2
(i.e., M2) by using a driver
Finally, implement M1 and test it with M2, using a driver for M1

Inheritance

Personnel
Consultant
Employee
Manager
Administrative_Staff
Technical_Staff

Testing OO programs

New issues
- inheritance
- genericity
- polymorphism
- dynamic binding

Open problems still exist

How to test classes of the hierarchy?

“Flattening” the whole hierarchy and considering every class as a totally independent component
- does not exploit incrementality
Finding an ad-hoc way to take advantage of the hierarchy

A sample strategy

A test that does not have to be repeated for any heir
A test that must be performed for heir class X and all of its further heirs
A test that must be redone by applying the same input data, but verifying that the output is not (or is) changed
A test that must be modified by adding other input parameters and verifying that the output changes accordingly

Separate concerns in testing

Testing for functionality is not enough
Overload testing
Robustness testing
Regression testing
- organize testing with the purpose of verifying possible regressions of software during its life—that is, degradations of correctness or other qualities due to later modifications
Testing concurrent and real-time systems

Nondeterminism inherent in concurrency affects repeatability
For real-time systems, a test case consists not only of input data, but also of the times when such data are supplied