Types and terms of testing

• Unit
• Integration
• System
• Exploratory
• Non-functional
Unit testing

- The unit: single class or a method
- The functionality of methods, construction and destruction of objects
- At the unit testing level methods are procedural, so testing is the same
- Easy to generate test cases
- Readily automatable, suitable for regression testing
Integration testing

- Fewer obvious structural relationships to guide testing
- Design for reuse implies many possible compositions
- How we define a unit has an impact on integration testing
  - (unit == method) => intraclass integration
  - (unit == class) => interclass integration
Integration testing (cont.)

• Some ideas for integration levels
  – Objects in the same package
  – Objects involved in a design pattern
  – Objects with relationships visible in class diagrams
• More difficult to generate test cases
• Generally automatable with some work
• Jorgensen/Erickson proposed in ‘94: MM-Path
MM-Path

- Method Message-Path: sequence of method executions linked by messages
- MM-Paths may branch off from other MM-Paths
- End point is a method that issues no messages itself
- ATM example
ATM PIN entry use case
System testing

- Limited to events visible at the boundaries
- What a user can do with a system
- Correspond to user stories, or use cases
- Useful for acceptance tests
- Many possible (too many) test cases
- Very tough to automate (GUI problem)
- Jorgensen/Erickson proposed in ‘94: ASF
Atomic System Function (ASF)

- Starts at an input port event
- Set of MM-Paths until an output port event is reached
- Often corresponds to something a user would do with the application
ATM PIN entry use case
Exploratory

• From context-driven school
• What a smart tester thinks about trying
• Testing as you learn about the app
• Not automatable (not the point), though may lead to automated system tests
• Requires different skills than automated U/I/S tests
Non-functional

• Performance
• Usability
• Backward compatibility
• Installation
Regression

• Running all the tests, all the time
• Not a kind of test
  – though you hear the term used that way
• Bug discovery per test is lower in this mode
  – so you hear traditional testers denigrate it
• Absolutely requires automation
• Key to test-driven development