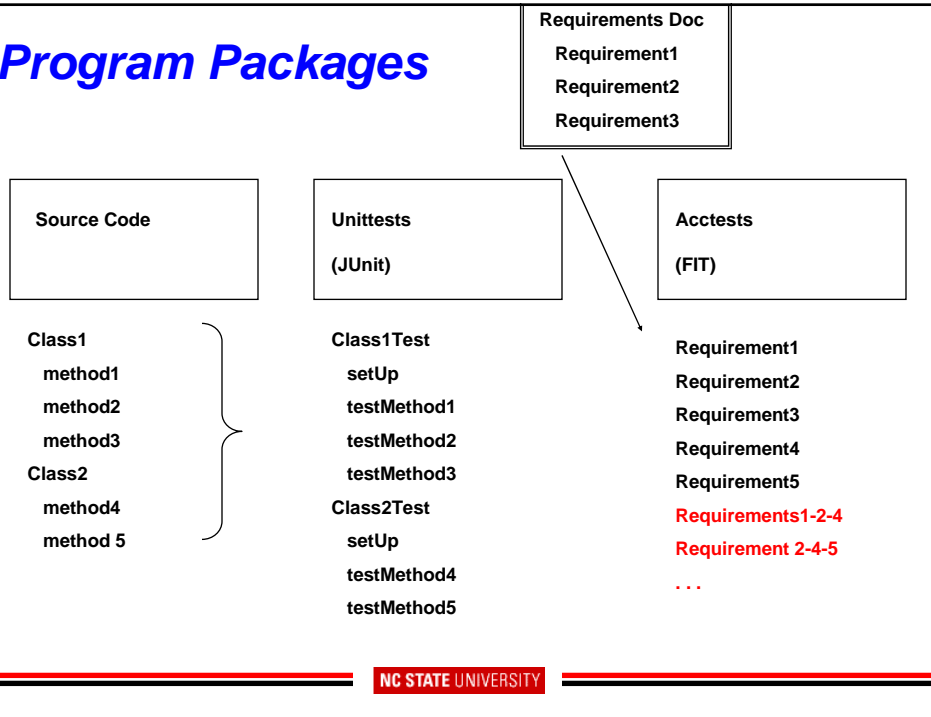


Agenda

- More advanced topics on writing black box test cases:
 - Scenario Testing
 - See http://testingeducation.org/k04/documents/bbst6_2004.pdf
 - Which uses:
 - Buwalda, Hans (2000a) "The three holy grails of test development," presented at EuroSTAR conference.
 - Transaction-flow Testing

Program Packages



Introduction

- Scenario [2] – a sequence of steps describing an interaction between user and a system
- Transaction [1] – a unit of work seen from a user's point of view. A transaction consists of a sequence of operations, some of which are performed by a system, persons, or devices that are outside the system
- . . . basically the same but Beizer's methods can help you think systematically about choosing scenarios.
- Validation or verification?
- Integration, functional, or system test?
- When in testing cycle should these be run?

[1] Beizer, B., *Software Testing Techniques*, International Thompson, 1990

[2] Fowler, M., *UML Distilled*, Addison Wesley, 2004.

NC STATE UNIVERSITY

Scenario testing

- The ideal scenario has several characteristics:
 - The test is *based on a story* about how the program is used, including information about the motivations of the people involved.
 - The story is *motivating*. A stakeholder with influence would push to fix a program that failed this test.
 - The story is *credible*. It not only *could* happen in the real world; stakeholders would believe that something like it probably *will* happen.
 - The story involves a *complex use of the program or a complex environment or a complex set of data*.
 - The test results are *easy to evaluate*. This is valuable for all tests, but is especially important for scenarios because they are complex.

NC STATE UNIVERSITY

Scenarios

- **Designing scenario tests is much like doing a requirements analysis, but is not requirements analysis. They rely on similar information but use it differently.**
 - The requirements analyst tries to foster agreement about the system to be built. The tester exploits disagreements to predict problems with the system.
 - The tester doesn't have to reach conclusions or make recommendations about how the product should work. Her task is to expose credible concerns to the stakeholders.
 - The tester doesn't have to make the product design tradeoffs. She exposes the consequences of those tradeoffs, especially unanticipated or more serious consequences than expected.
 - The tester doesn't have to respect prior agreements. (Caution: testers who belabor the wrong issues lose credibility.)
 - The scenario tester's work need not be exhaustive, just useful.

Soap operas

- [a special type of scenario test in which more careful attention is paid to creating complex, lifelike stories]
- **Build a scenario based on real-life experience. This means client / customer experience.**
- **Exaggerate each aspect of it:**
 - example, for each variable, substitute a more extreme value
 - example, if a scenario can include a repeating element, repeat it lots of times
 - make the environment less hospitable to the case (increase or decrease memory, printer resolution, video resolution, etc.)
- **Create a real-life story that combines all of the elements into a test case narrative.**

Examples of story lines when used for testing

Pension fund

William starts as a metal worker for Industrial Entropy Incorporated in 1955. During his career he becomes ill, works part time, marries, divorces, marries again, gets 3 children, one of which dies, then his wife dies and he marries again and gets 2 more children....

World wide transaction system for an international bank

A fish trade company in Japan makes a payment to a vendor on Iceland. It should have been a payment in Icelandic Kronur, but it was done in Yen instead. The error is discovered after 9 days and the payment is revised and corrected, however, the interest calculation (value dating)...

From a talk by Hans Buwalda

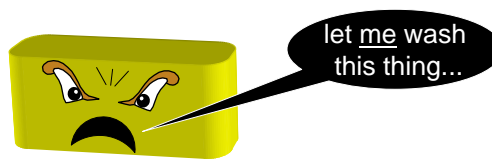
Soap operas (in testing) are not necessarily:

- “Extreme”
- Far fetched
- Long and elaborate
- Pieces of art and creativity

From a talk by Hans Buwalda

“Killer Soaps”

- More specifically aimed at finding hidden problems
- Run when everything else has passed
- One option: put a killer soap at the end of a normal cluster
- Ask the “specialists” for input



From a talk by Hans Buwalda

Risks of scenario testing

- Other approaches are better for testing early, unstable code.
 - A scenario is complex, involving many features. If the first feature is broken, the rest of the test can't be run. Once that feature is fixed, the next broken feature blocks the test.
 - Test each feature in isolation before testing scenarios, to efficiently expose problems as soon as they appear.
- Scenario tests are not designed for coverage of the program.
 - It takes exceptional care to cover all features or requirements in a set of scenario tests. Statement coverage simply isn't achieved this way.
- Reusing scenarios may lack power and be inefficient
 - Documenting and reusing scenarios seems efficient because it takes work to create a good scenario.
 - Scenarios often expose design errors but we soon learn what a test teaches about the design.
 - Scenarios expose coding errors because they combine many features and much data. To cover more combinations, we need new tests.
 - Do regression testing with single-feature tests or unit tests, not scenarios.

Scenario Testing

- Car wants to drive through an intersection.
- The car can only clear through the intersection if the traffic light is green and there are not obstructions of any kind in the intersection. Otherwise, the car needs to join a queue of cars.

Scenario Testing II

- Come up with three scenario tests using (a selected subset of) the use cases in Iterations #1-3
 - UC1: *Create and Disable Patients and Health Care Personnel*
 - UC2: *Authenticate Users*
 - UC3: *Enter/edit Demographics*
 - UC4: *Log Transaction (some)*
 - UC5: *Declare/undeclared designated licensed health care professional*
 - UC9: *Enter/edit diagnosis*
 - UC10: *Document office visit*

Transaction-Flow Testing

- Beizer “the ultimate black-box technique”
- This is a way to catch nasty behavioral bugs early in testing.
- “Let the nightmare paths be your guide.”
- Can be the cornerstone of system functional testing just as path testing is the cornerstone of unit testing.

Transaction-flow Testing Steps

- Draw the flow graph of transaction flows.
- Find and document a covering set of transaction flows. Beizer recommends “statement” plus “branch” coverage.
- Beizer recommends asking the designers for the ten most important transactions a system can process. [The act of getting the information on which to base tests can be more effective at catching and exterminating bugs than the tests that result from the information. Good to walkthrough transactions.]
- Sensitize the paths . . . Meaning create specific test cases
 - Prepare inputs
 - Predict outputs
- Instrument the paths.
- Run the tests.

