ECE459: Programming for Performance Winter 2015

Lecture 35 — April 1, 2015

Patrick Lam *version 1*

Badly Designed Tests

Today's lecture is about test design. We'll start by talking about test smells.

A *smell* is a symptom of a problem. "Something is wrong here."

We are going to talk about three kinds of smells: project smells, behaviour smells, and code smells. Given the presence of a smell, you need to identify the root cause (ask "why?" a bunch of times; maybe 5 is a good number) and do a cost/benefit analysis for fixing the underlying problem.

Project Smells

These smells are the highest-level ones. They are usually detected by project managers, who are monitoring functionality, quality, resource usage, and cost.

Production Bugs. (aka escapes) If your project has too many production bugs, something is wrong. The team needs to figure out why this is happening. Are there enough tests? Is something wrong with the development process? Are there enough development resources?

Excessive Continuous Integration Failures. If the CI system keeps on reporting failures, something's probably wrong. It might be buggy tests, or tests that are too expensive to maintain. Or, there might not be enough tests being run at commit time.

Behaviour smells

These typically manifest as compile errors or test failures.

Fragile Tests. These are the most common behaviour smells. One cause is the use of tests created through record/playback, which induces UI dependency. Possible root causes include:

- interface sensitivity;
- behaviour sensitivity (only a few tests—relevant ones—should break per change);
- data sensitivity: tests might be relying on data stored in the system under test, or in a database;
- context sensitivity: e.g. the environment, including the time and date, or external devices.

Assertion Roulette. Another common behaviour smell. This manifests as difficult-to-diagnose continuous integration failures, where the error messages don't tell you enough to fix the problem.

Erratic and Flaky Tests. Sometimes you change nothing, but the test randomly fails. Causes of this smell include interacting tests which share a fixture; test run wars; unrepeatable tests; dependencies on external systems which randomly get hosed; etc.

Frequent Debugging. Might be caused by insufficient test coverage, or insufficiently fine-grained tests.

Tests require Manual Intervention. That's bad. Automate your tests!

Slow Tests. Tests should be fast. 30-second tests, for instance, are way, way too slow.

Code Smells

These smells affect maintenance cost and are also early warning signs of behaviour smells.

- Obscure Tests. (What's the meaning of that test?) Caused by poorly-named or poorly-implemented tests. Leads to buggy tests.
- Conditional Test Logic. Avoid whenever possible. Use `fail()` to guard.
- Hard-Coded Test Data.
- Hard-to-test Code.
- Test Code Duplication.
- Test Logic in Production. Try to avoid putting test code in your production system.