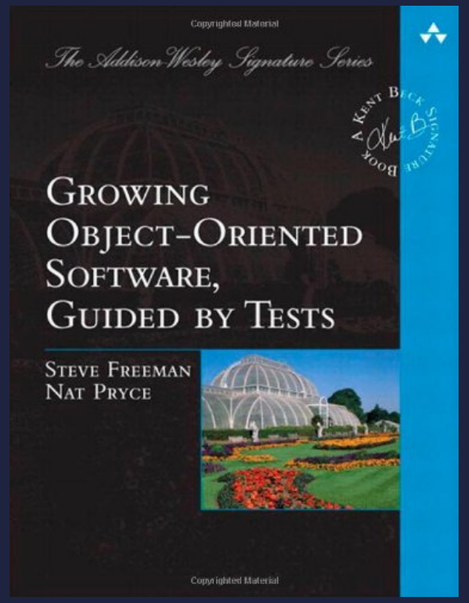
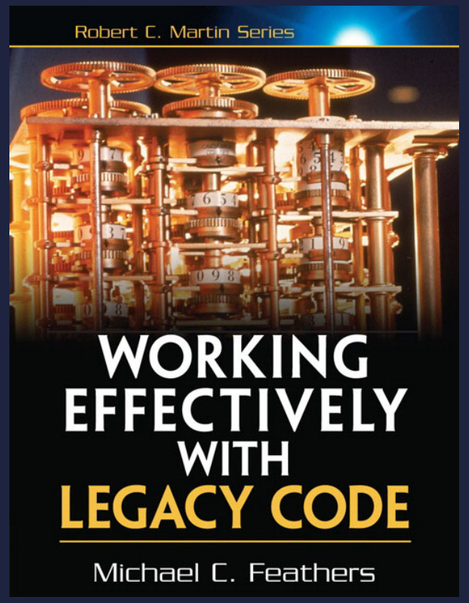
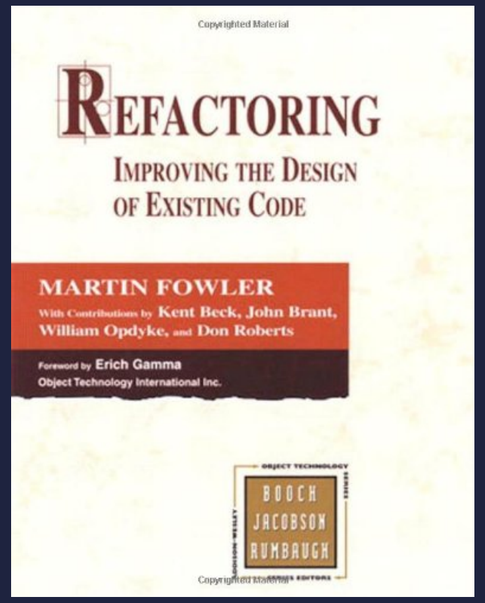
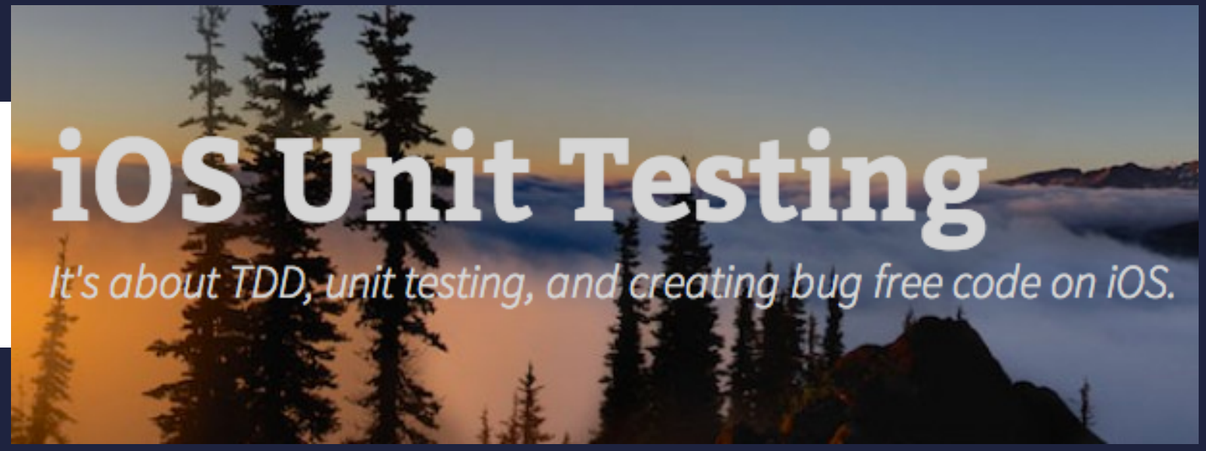
TESTS MAKE IT EASIER TO CHANGE CODE







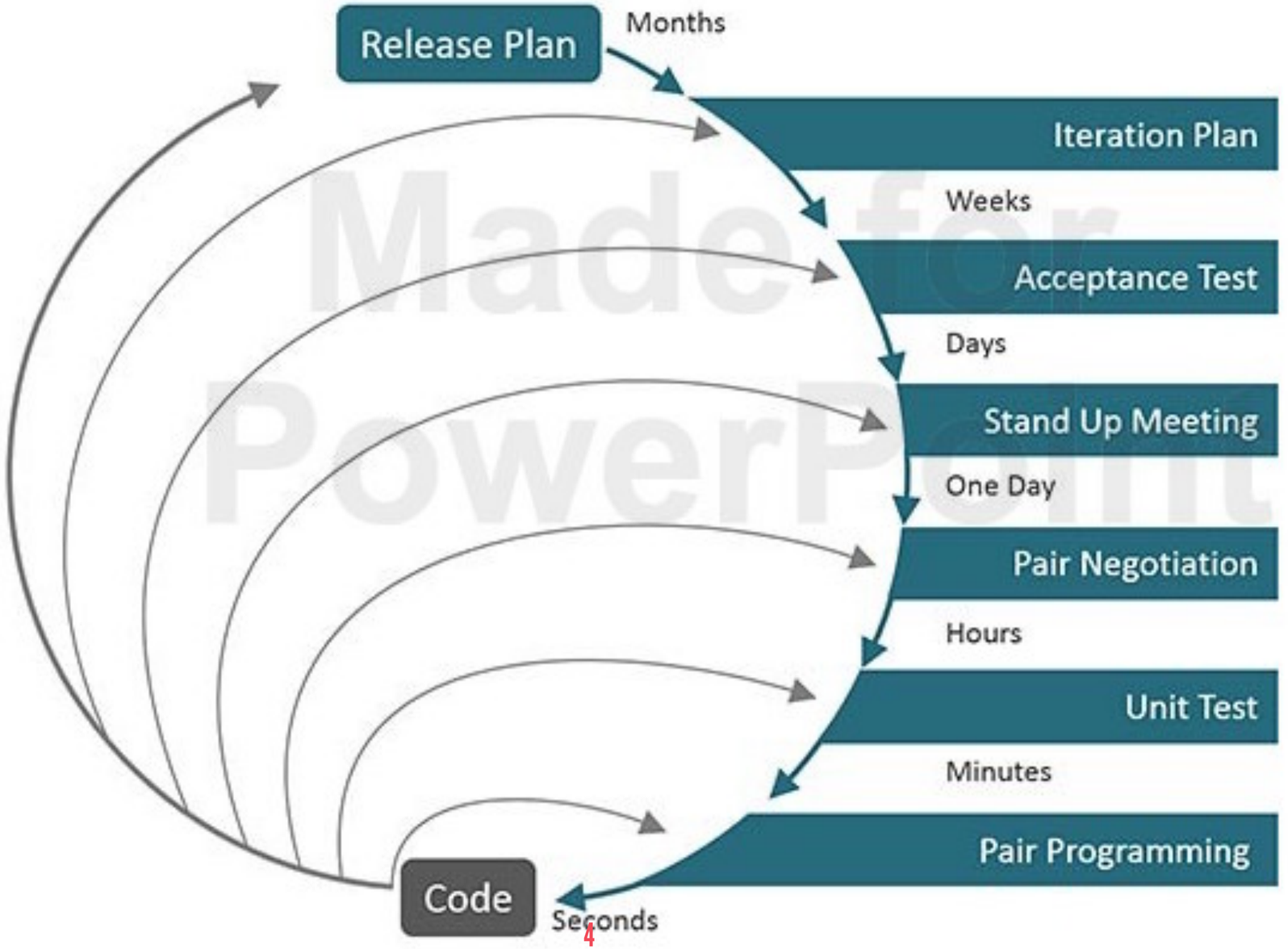
Google™ testing blog



TIGHT FEEDBACK LOOP

TESTS HELP US CATCH BUGS EARLY

Planning/Feedback Loops



CONFIDENCE

TESTS PROVIDE A SAFETY NET

RAPID FEEDBACK LETS US MAKE CHANGES WITHOUT PRAYING

IMPROVED CODE QUALITY

WELL DESIGNED CODE IS EASIER TO TEST

WELL DESIGNED CODE IS EASIER TO UNDERSTAND

HOW TO WRITE GOOD TESTS

TEST INTERFACE NOT IMPLEMENTATION

Implementation changes more often than interface

Refactoring, bugfixes, new features, optimizations

TDD

WRITE TESTS FIRST!

TDD ENCOURAGES BETTER DESIGN

THINK ABOUT HOW NEW OBJECT CAN BE TESTED

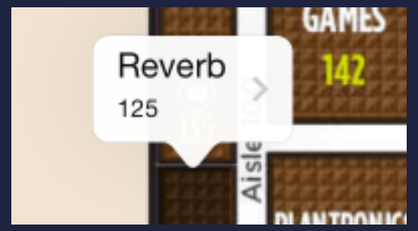
WRITE LOOSELY COUPLED COMPONENTS THAT CAN BE TESTED IN ISOLATION

LEADS TO BETTER INTERFACES

TDD SPEEDS UP IMPLEMENTATION

DETECT ERRORS WHILE CONTEXT IS FRESH IN OUR MIND

BDD IS EASIER TO READ



BDD IS EASIER TO READ

```
it(@"Scroll view should present callout", ^{
    // Given a scroll view
    GBMapScrollView *scrollView = [AutomatedTestingViewController applicationTestingInstance].scrollView;

    // And the scroll view has a small image
    UIImageView *imageView = [[UIImageView alloc] initWithImage:[UIImage imageNamed:@"100x200.jpg"]];
    scrollView.contentView = imageView;

    // When I present a callout
    GBMapScrollViewCalloutConfiguration *calloutConfiguration = [[GBMapScrollViewCalloutConfiguration alloc] init];
    calloutConfiguration.title = @"Callout";
    calloutConfiguration.rect = CGRectMake(0, 100, 100, 100);
    [scrollView presentCalloutWithConfiguration:calloutConfiguration];

    // Then the scroll view should show a callout
    [[scrollView should] haveMatchingSnapshotNamed:@"ScrollViewCalloutVisible" inReferenceImagesDirectory:GB_KIWI_MATCHERS_REFERENCE_IMAGE_DIR];
});
```

BDD IS EASIER TO READ

```
it(@"Scroll view should present callout", ^{
    // Given a scroll view
    GBMapScrollView *scrollView = [AutomatedTestingViewController applicationTestingInstance].scrollView;

    // And the scroll view has a small image
    UIImageView *imageView = [[UIImageView alloc] initWithImage:[UIImage imageNamed:@"100x200.jpg"]];
    scrollView.contentView = imageView;

    // When I present a callout
    GBMapScrollViewCalloutConfiguration *calloutConfiguration = [[GBMapScrollViewCalloutConfiguration alloc] init];
    calloutConfiguration.title = @"Callout";
    calloutConfiguration.rect = CGRectMake(0, 100, 100, 100);
    [scrollView presentCalloutWithConfiguration:calloutConfiguration];

    // Then the scroll view should show a callout
    [[scrollView should] haveMatchingSnapshotNamed:@"ScrollViewCalloutVisible" inReferenceImagesDirectory:GB_KIWI_MATCHERS_REFERENCE_IMAGE_DIR];
});
```

WHEN TO WRITE TESTS

100% CODE COVERAGE NOT ALWAYS WORTH IT

COST TO BENEFIT RATIO

New feature => high benefit

Fixing bug => high benefit

Every edge case => high cost

COST TO BENEFIT RATIO

"Risk Driven Testing" (Google Testing Blog)

Mission-critical components more important to test

Listen to gut feeling

TYPES OF TESTS

UNIT, INTEGRATION, END TO END, MANUAL

UNIT TESTS

Create object

Provide dependencies

Interact with it

Verify behavior

WEB OF DEPENDENT OBJECTS

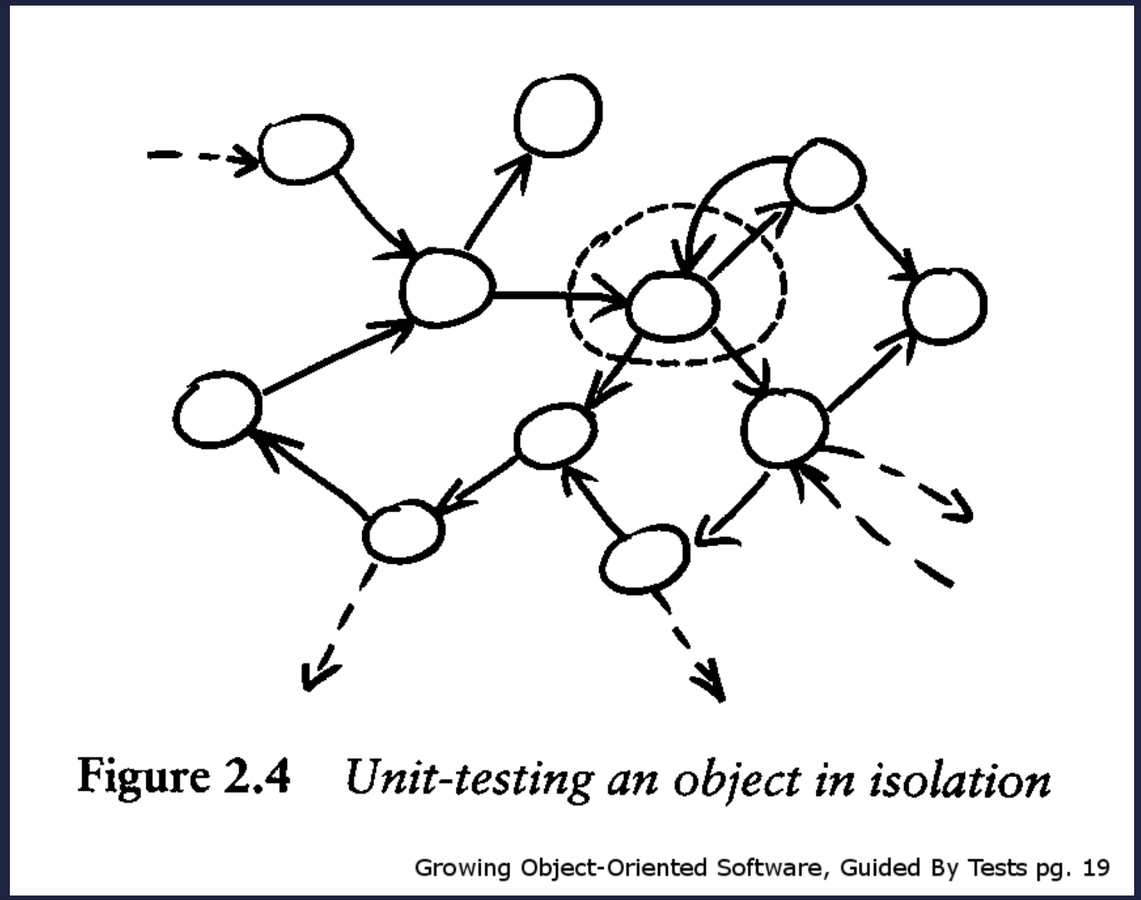


Figure 2.4 *Unit-testing an object in isolation*

Growing Object-Oriented Software, Guided By Tests pg. 19

FAKES AND MOCKS

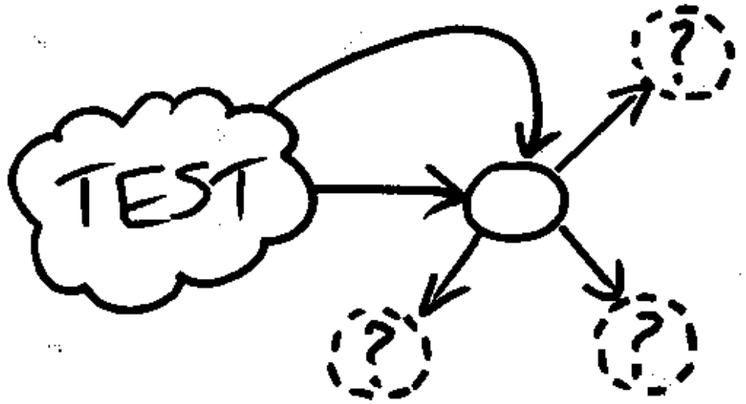


Figure 2.5 *Testing an object with mock objects*

Growing Object-Oriented Software, Guided By Tests pg. 19

INTEGRATION TESTS

Verify that multiple objects work together

Does our code work against code we can't change?

END TO END TESTS

Exercise system from the outside

Mobile: Automate UI, automatically verify screenshots against baseline

Web API: Invoke services, parse responses

MANUAL TESTS

FOR THINGS THAT CANNOT BE AUTOMATED

HELPS TO WRITE DOWN TEST CASES

EXTERNAL AND INTERNAL QUALITY

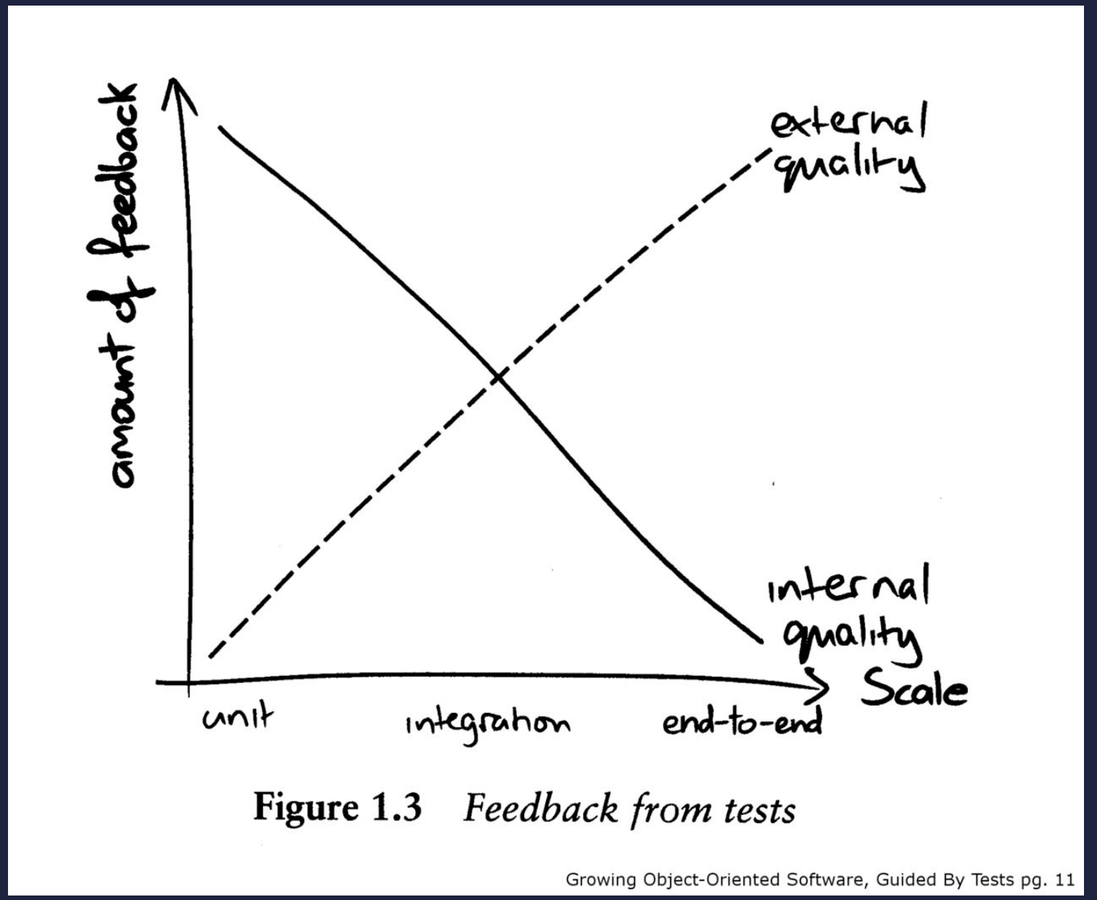
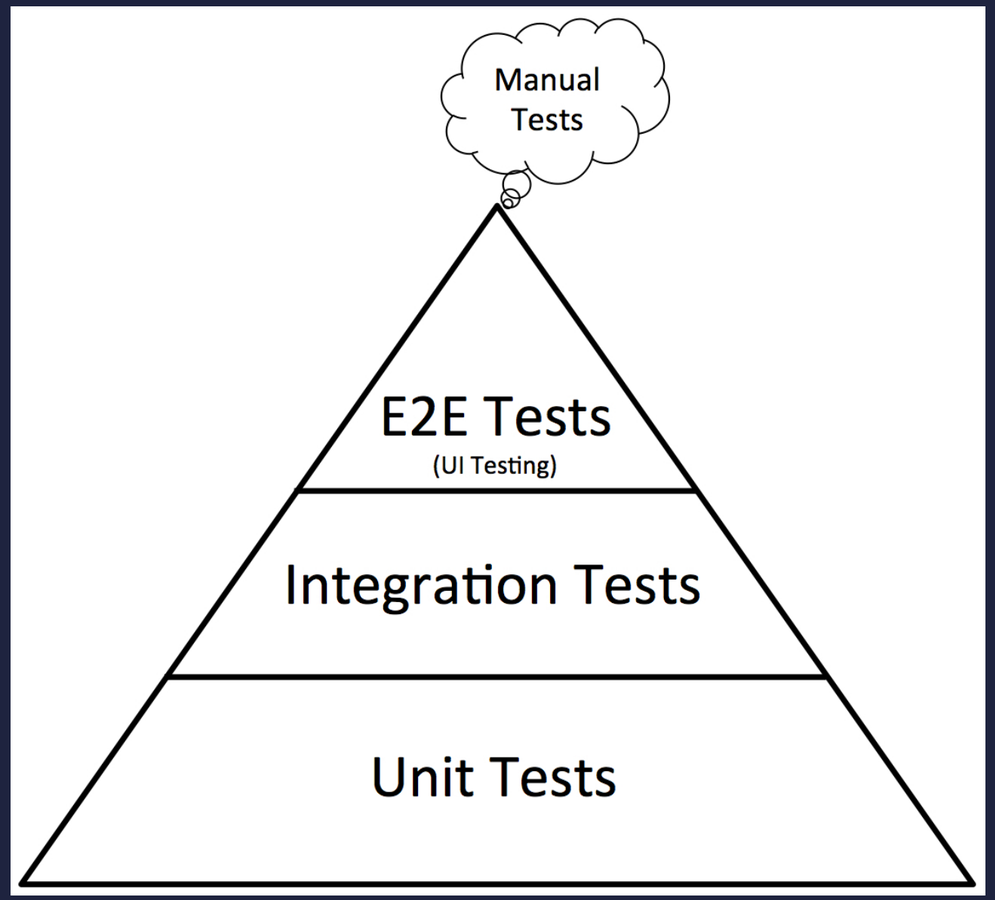


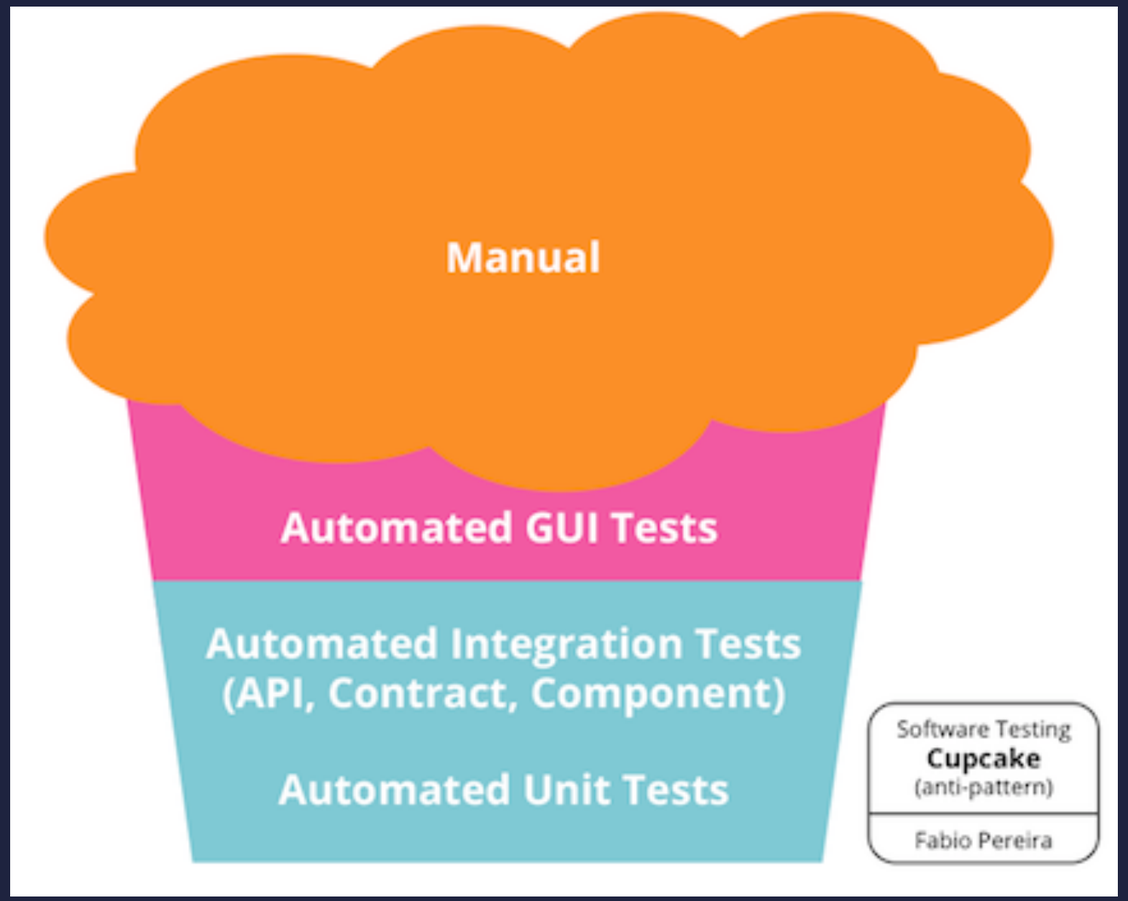
Figure 1.3 *Feedback from tests*

Growing Object-Oriented Software, Guided By Tests pg. 11

GOOD


Manual
Tests
E2E Tests
(UI Testing)
Integration Tests
Unit Tests

BAD


Manual
Automated GUI Tests
Automated Integration Tests
(API, Contract, Component)
Automated Unit Tests
Software Testing
Cupcake
(anti-pattern)
Fabio Pereira

SNAPSHOT TESTS

EFFECTIVE FOR TESTING VIEWS

EFFECTIVE FOR END TO END TESTING

PERFORMANCE TESTS

EFFECTIVE FOR TESTING OPTIMIZATIONS

WRAPPING UP

Tight feedback loop

Safety net gives us confidence

Tests encourage good design