

Test Frameworks and Agile Software Development : Like Peanut Butter and Chocolate

Rachel Silber
rachel@butinpractice.com



What is a Test Framework?

- Distinguished from test code
- The code you're writing to assist automated testing that is separable from the specific tests that exercise the system under test.
- Distinguished from special purpose tools and single use scripts
- Code that's intended to be reused to support multiple tests.

What is Agile Development?

A process that is marked by:

- Iterative development
- Acceptance of change as an organic part of software development
- A clear separation of roles between the “customer” and those engineering the system
- Collaborative ownership of the code

**Test framework development
needs to be managed.**

Test tool development problems

- The “it’s easy” belief.
- Unclear requirements
- Estimation difficulties
- Reuse is imperative

**A test development framework often
has an on-site customer**

**Delivering timely subsets
of the desired requirements
can provide business value.**

‘Spike solutions’

- Example: implement the code so that the system under test can be programmatically installed.
- Save for a later iteration the flexibility to select which build gets installed.

Estimating the test development effort is hard

- Unknown requirements
- New technology
- Changing resources
- Competing priorities

Take advantage of task-based estimation and the concept of velocity.

- Start with an estimate of required effort for each task
- Measure progress based on achieved goals, not time elapsed
- Use historical data to predict future progress

**Collective ownership makes the test harness
less prone to failure and abandonment.**

Agile development practices

- Refactoring
- Unit testing

Refactoring : Improving design without changing system behavior

- Set a baseline for behavior with tests
- Change implementation
- Retest to confirm nothing was unintentionally changed

Refactoring Example

Pseudocode (starting state):

```
Function GeneralCase (param) {  
    [setup code]  
    for (items in collection) do  
        if (param == 1) then:  
            [case 1]  
        else if (param == 2) then:  
            [case 2]  
        else :  
            [case 3]  
        end do  
    }  
}
```

1. Replace Parameters with Explicit Methods

```
Function LogicOfCase1()
```

```
  { [case 1] }
```

```
Function LogicOfCase2()
```

```
  { [case 2] }
```

```
Function LogicOfCase3()
```

```
  { [case 3] }
```

```
Function GeneralCase(param)
```

```
  { [setup code]
```

```
  for (items in collection) do
```

```
    if (param == 1)
```

```
      then { LogicOfCase1() }
```

```
    else if (param == 2)
```

```
      then { LogicOfCase2() }
```

```
    else { LogicOfCase3() }
```

```
  }
```

2. Transforming Procedural Code to Objects

```
Class TestObject
  Create() {
    [setup code]
  }

  Logic_For_Param1() { [case 1] }
  Logic_For_Param2() { [case 2] }
  Logic_For_Param3() { [case 3] }

  GeneralCase(param) {
    if (param = 1)
      Logic_For_Param1()
    else if (param = 2)
      Logic_For_Param2()
    else
      Logic_For_Param3()
  }
}
```

Unit testing the test framework

- Encourages collaborative ownership of the code
- Encourages developer confidence in test results
- Can give us the same productivity gains for internal code that we get for product code

Aren't tests on test code redundant?

- No.

Using tests on the test framework

- A suite of regression tests on the test framework
- Can be used when the test framework is deployed to a new test environment
- Reveals any issues in the test environment without starting a (potentially expensive) system test cycle.

Some further reading

- *Refactoring* Martin Fowler (Addison-Wesley, 1999)
- *Working Effectively with Legacy Code* Michael Feathers (Prentice Hall 2004)

Blogs:

A list of blogs and websites that deal with agile development is available on my weblog.

My blog: *But In Practice*: www.butinpractice.com/blog

Any Questions?

