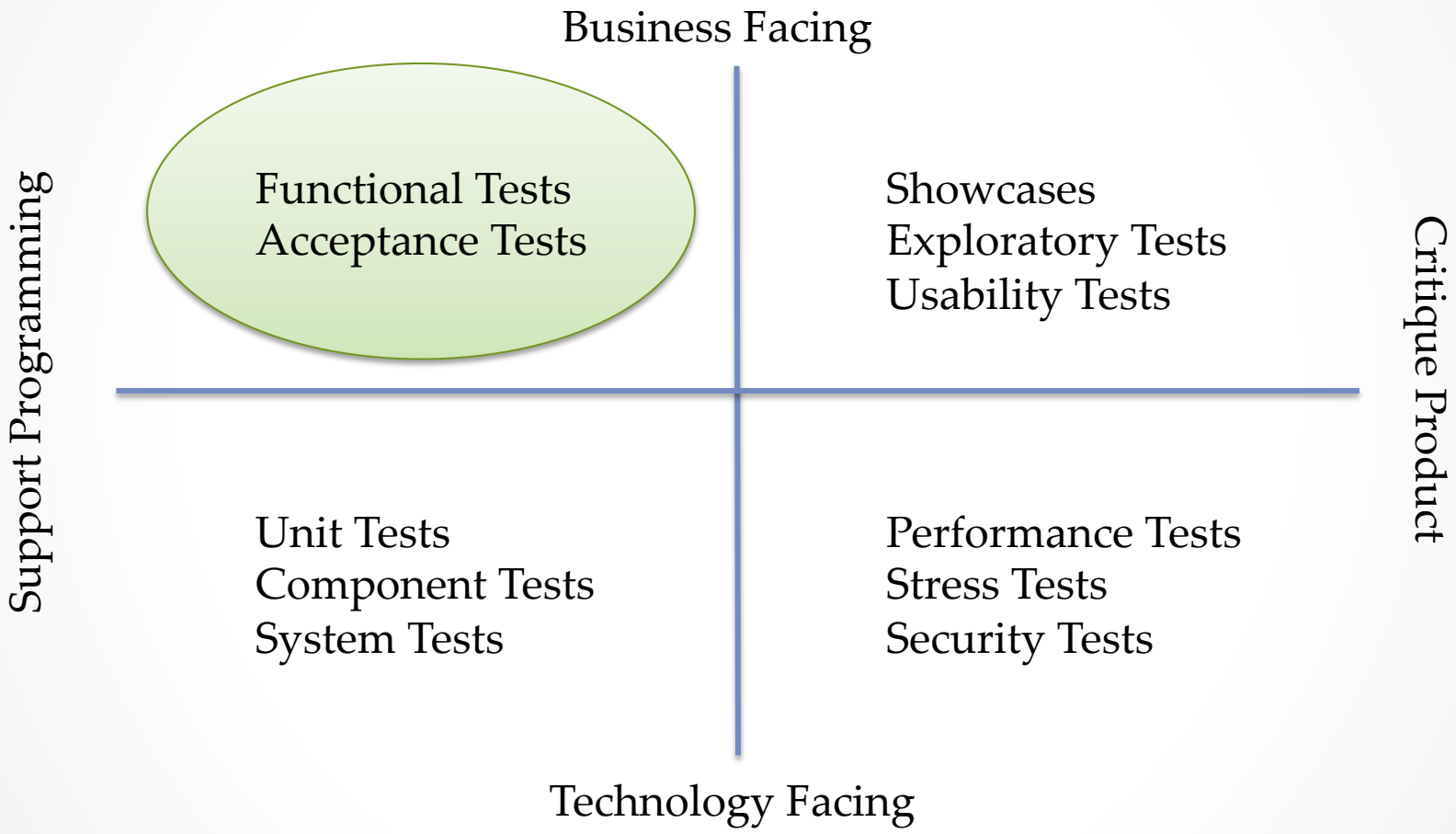
Functional Testing Patterns

Premanand Chandrasekaran
Principal Consultant
ThoughtWorks, Inc.

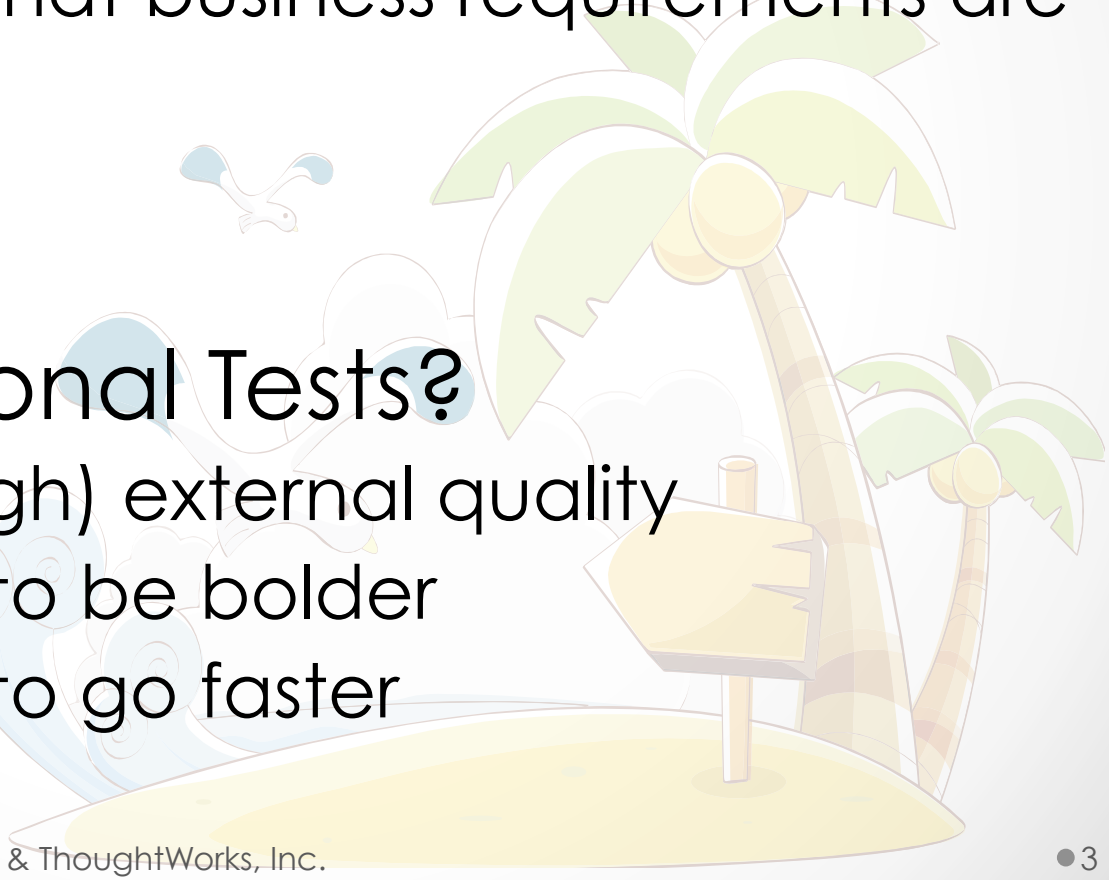
Types of Tests



http://www.exampler.com/old-blog/2003/08/21/#agile-testing-project-1

Functional Tests - The Pitch

- What Are Functional Tests?
  - Verification that business requirements are met
  - Black Box
  - Automated
- Why Functional Tests?
  - Maintain (high) external quality
  - Allow team to be bolder
  - Allow team to go faster



Functional Tests – The Reality

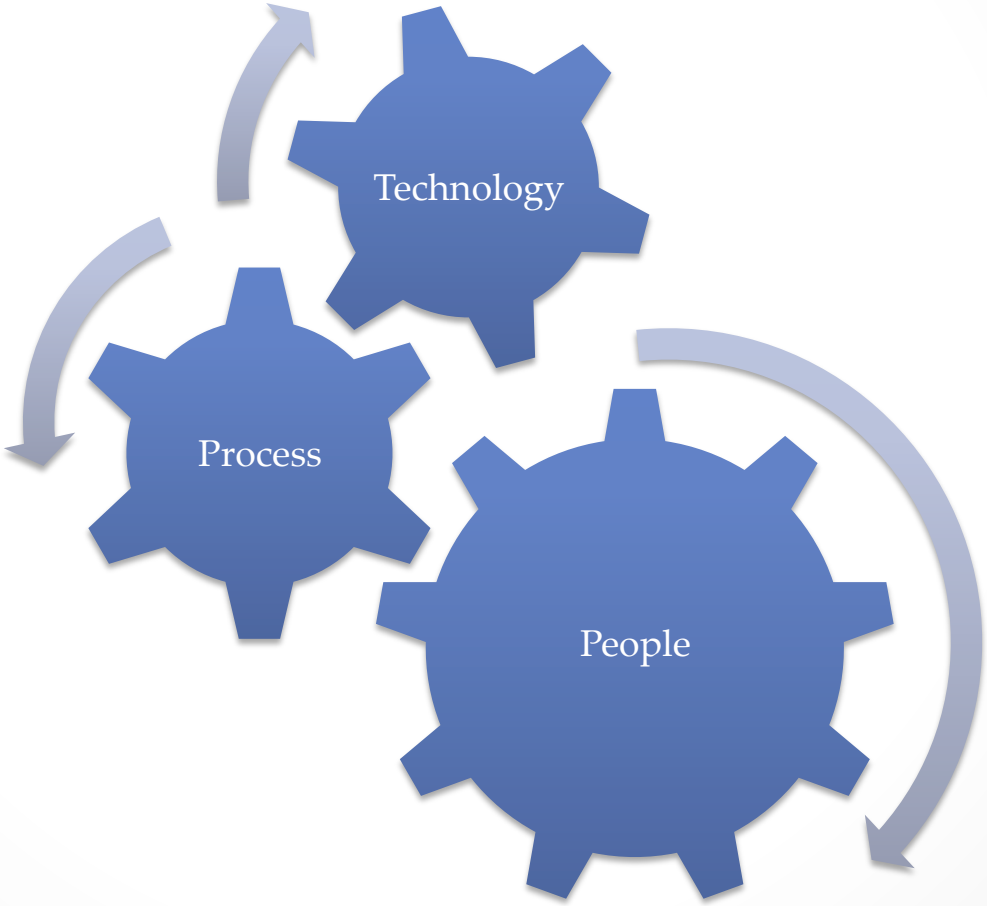
- Flaky
- Take long time to run
- Cumbersome to maintain
- Don't catch many bugs
- Some things are hard to automate
- Don't quite give you the feedback you were hoping for

Consequences

- Low confidence in automation
- Over-reliance on manual testing

- Slipped Deadlines AND/OR
- Lesser Testing AND/OR
- Testing in Production ☺

How To Get Better?



Functional Test – Example

Example - Specification

```
Feature: ShoppingCart functionality for Etsy.com

Narrative:
  In order to show the basic cart functionality
  As a user
  I want to add and remove items from the cart

Scenario: Item can be added to cart

  Given that the cart is empty
  When I search for an item
  And an item is added to the cart
  Then the cart contains that item
```

Example – Implementation

```
public class MyTest {
    private WebDriver driver;
    private String baseUrl;
    private StringBuffer verificationErrors = new StringBuffer();
    @Before
    public void setUp() throws Exception {
        driver = new FirefoxDriver();
        baseUrl = "http://www.etsy.com/";
        driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
    }

    @Test
    public void testMy() throws Exception {
        driver.get(baseUrl + "/");
        driver.findElement(By.cssSelector("span.string")).click();
        driver.findElement(By.id("search-query")).clear();
        driver.findElement(By.id("search-query")).sendKeys("hat");
        driver.findElement(By.id("search_submit")).click();
        driver.findElement(By.cssSelector("img[alt=\"White solid wood 3 peg for coats, hats and garment\"]")).click();
        driver.findElement(By.cssSelector("span.button-large.button-large-cart > span > input[type=\"submit\"]")).click();
        driver.findElement(By.cssSelector("span.string")).click();
        try {
            assertEquals("1", driver.findElement(By.cssSelector("span.quantity-value")).getText());
        } catch (Error e) {
            verificationErrors.append(e.toString());
        }
    }

    @After
    public void tearDown() throws Exception {
        driver.quit();
        String verificationErrorString = verificationErrors.toString();
        if (!"".equals(verificationErrorString)) {
            fail(verificationErrorString);
        }
    }
```

Implementation Problems?

- No correlation between specification and implementation
- Intent lost in translation
- Easy to write, hard to read, maintain
- Too monolithic, lacks abstraction

- Causes non-technical domain experts to tune out of the process

Executable Specifications

Specification Step

```
Given that the cart is empty
```

Tight Correlation

Implementation Step

```
@Given("that the cart is empty")
def cartIsEmptyAndOnStartPage() {
    home.go()
    cartIsEmpty();
}
```

Functional Test - Specification

```
Feature: ShoppingCart functionality for Etsy.com

Narrative:
  In order to show the basic cart functionality
  As a user
  I want to add and remove items from the cart

Scenario: Item can be added to cart

  Given that the cart is empty
  When I search for an item
  And an item is added to the cart
  Then the cart contains that item
```

Browse Etsy.com

etsy_browse.story

Meta:
@category browsing
@color red

Narrative:

In order to show the browsing cart functionality
As a user
I want to browse in a gallery

Scenario: Browsing around the site for items

Given I am on etsy.com
When I want to browse through a treasury gallery
And I want to buy something from etsy.com
And I want to browse the treasury
When I choose the first treasury gallery (FAILED)

More useful information in failure reports

```
org.openqa.selenium.NoSuchElementException: Unable to loc
For documentation on this error, please visit: http://sel
Build info: version: '2.13.0', revision: '14794', time: '
System info: os.name: 'Mac OS X', os.arch: 'x86_64', os.v
Driver info: driver.version: RemoteWebDriver
        (reflection-construct)
        at org.openqa.selenium.remote.ErrorHandler.create
        at org.openqa.selenium.remote.ErrorHandler.throwI
        at org.openqa.selenium.remote.RemoteWebDriver.exe
        at org.openqa.selenium.remote.RemoteWebDriver.fir
        at org.openqa.selenium.remote.RemoteWebDriver.fir
        at org.openqa.selenium.By$ByXPath.findElement(By.
        at org.openqa.selenium.remote.RemoteWebDriver.fir
        at org.jbehave.web.selenium.WebDriverPage.findEle
        at org.openqa.selenium.WebDriver$findElement.call
        (groovy-call)
        at pages.Treasury.chooseFirstGallery(Treasury.gro
        at pages.Treasury$chooseFirstGallery.call(Unknown
        (groovy-call)
        at EtsyDotComSteps.selectFirstTreasuryGallery(Ets
        (reflection-invoke)
        at org.jbehave.core.steps.StepCreator$Parameteris
        at org.jbehave.core.embedder.StoryRunner$FineSoFa
```

Acceptance Criteria DSL

- **Given** some initial context
- **When** an event occurs
- **Then** ensure outcome
- Example –

```
1  Scenario: Advanced Search for a hat
2
3  Given I am searching on Etsy.com
4  And I am not logged in
5  When I specify the Knitting sub category
6  And I search for hat
7  Then there are search results
```

Functional Test – New Implementation

```
 7  class ShoppingCartSteps {
 8
 9      Home home
10      Site site
11
12      @Given("that the cart is empty")
13      def cartIsEmptyAndOnStartPage() {
14          home.go()
15          cartIsEmpty();
16      }
17
18      @When("I search for an item")
19      def searchForItem(){
20          home.search("hat")
21      }
22
23      @When("an item is added to the cart")
24      def putThingInCart() {
25          putThingInCart("hat")
26      }
27
28      @Then("the cart will be empty")
29      def cartIsEmpty() {
30          site.cartEmpty()
31      }
```

Good Acceptance Criteria

```
Scenario: Advanced Search for a hat

Given I am searching on "http://www.etsy.com"
When I click on the select box with css class "select.handmade"
And I select "Knitting"
And I click on the text box with id "#search_query"
And type in "hat"
Then there are search results
```

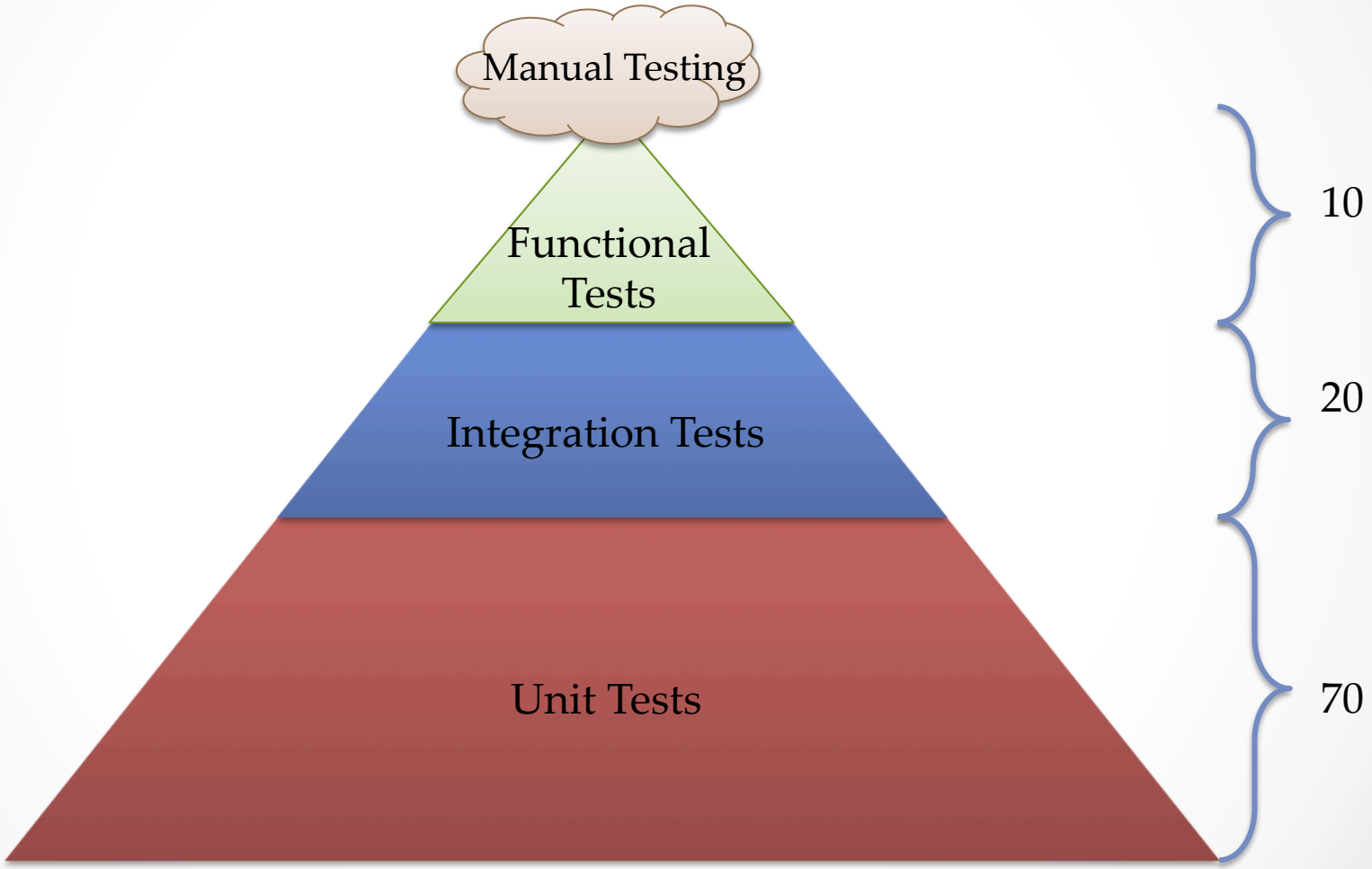
Focus on WHAT, not HOW

```
Scenario: Advanced Search for a hat

Given I am searching on Etsy.com
And I am not logged in
And I specify the Knitting sub category
And I search for hat
Then there are search results
```

Better

How Many Tests?



http://blog.mountaingoatsoftware.com/tag/agile-testing

What About Legacy Systems?

- Start with functional tests
- Cover with unit tests for new functionality
- Be on the lookout to replace functional tests more fine-grained tests at every given opportunity

What To Automate?

- The **MOST** important scenarios
  - End-To-End system level scenarios
  - Happy paths

- Avoid automating individual feature-level acceptance criteria
- Make a distinction between "ephemeral" and "long-standing" tests

When do these tests run?

- After every check-in
- As part of CI build pipeline
  - Unit → Smoke → Regression → Deploy
- Incubator Test Suites

Workflow

- Sprint [N -1]
  - Feature elaboration with entry level acceptance criteria
- Sprint [N]
  - Feature kick-off meeting
  - Developers implement feature
  - Testers (optionally) add additional acceptance criteria
- Sprint[N, N + 1]
  - Testers verify all acceptance criteria to be satisfied
  - Testers refactor functional test suite
  - Feature marked complete

Test Characteristics

- Be idempotent
  - At least re-runnable
  - Keep tests isolated from one another
- Be runnable in parallel
  - No inter-dependencies between tests
- Be deterministic
  - Quarantine/eradicate non-deterministic tests
  - Don't "sleep"
    - Prefer callbacks or at least poll for asynchronous responses
  - Consider mocking remote third-party services
    - Have separate tests to verify interaction with remote services
  - Consider using relative dates/times or mock the system clock
  - http://martinfowler.com/articles/nonDeterminism.html

Test Sources

- Don't have any dependencies on production sources
- Treat as first-class citizens
  - Keep DRY
  - Run static analysis metrics
- Leverage design patterns like
  - Page Objects
  - Compound Steps
- Consider the use of terser languages
  - Groovy, Ruby etc.

Test Data

- Use application to create test data
- Use test data creation steps
  - Frameworks like DBUnit
- Use anonymized production data subsets
  - Requires lot of time and effort
- Use anonymized production data copies
  - Huge data volumes may make it prohibitive
- Avoid the shared database anti-pattern
- Consider mocking remote third-party services

Test Infrastructure

- Invest in fast build hardware
  - CI servers support multiple remote agents
- Use same software configuration as production in all environments
- Consider running tests on the cloud
  - http://www.saucelabs.com

Who Writes/Owns The Tests?

- The specifications – acceptance criteria?
  - Domain Experts
- Step implementations?
  - Developers & Testers
- The functional suite?
  - The Team
  - Day-to-Day – Testers

Team Dynamics

- Fix broken builds and tests immediately
  - OK to occasionally break the build
  - Not OK to leave the build broken for long
- Do not allow check-ins over a broken build
  - Unless check-in is being made to fix the build
- Co-locate teams as much as possible
  - At least create fully formed remote teams

Questions?

premanand@thoughtworks.com

Thank You!