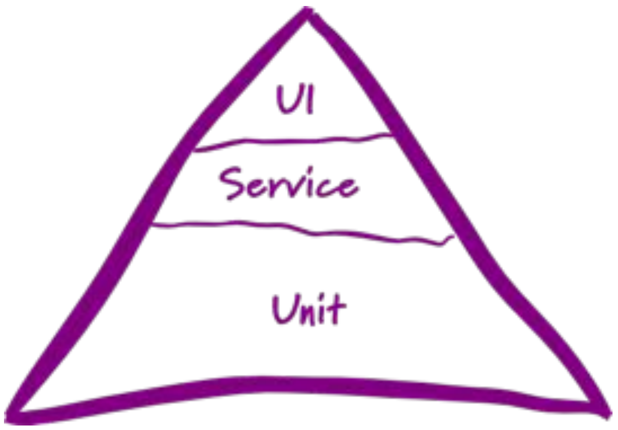
THE TESTING PYRAMID: IT'S NOT ABOUT TESTING

PYRAMIDS, PEOPLE, AND TESTS

THE (AUTOMATION) TESTING PYRAMID



Ideas from Mike Cohn's *Succeeding With Agile*, graphic from http://martinfowler.com/bliki/TestPyramid.html

THE TESTING PYRAMID: IT'S NOT ABOUT TESTING

PYRAMIDS, PEOPLE, AND TESTS

SCRUM

Agile Software Development with Scrum
red yellow green blue red blue yellow green blue
Ken Schwaber Mike Beedle

CSM ScrumAlliance CERTIFIED ScrumMaster

PST

Scrum.org
Improving the Profession of Software Development

GARY PEDRETTI

- Over 18 years in the software industry with highly cross-functional experience – DBA, Developer, BA, Application Architect
- Currently Trainer, Coach, Consultant, Owner at Sodoto Solutions
- SODOTO = See One, Do One, Teach One
- Scrum: Development Team member, Scrum Master, Coach, Professional Scrum Trainer for Scrum.org
- http://blog.GaryPedretti.com/
- http://www.linkedin.com/in/garypedretti
- Twitter: @GaryPedretti
- MCSD:ALM 2012

KEY TAKEAWAYS

New ways for architects, testers, BAs, and coders to work together in a cross-functional manner

New ways to think about Mike Cohn's popular Testing Pyramid

New patterns for developing a well-tested application

AGENDA

- People, culture, teams
- The Test Pyramid – going through the layers
- Timing – “moving left”
- Test automation

QUESTIONS – TRUE OR FALSE?

- There is a natural conflict of interest between coders and testers.
- At the very least, testers should “keep their distance” from coders.
- Testers should be measured and compensated by the number of bugs they find.
- The “cross-functional teams” that Agile processes talk about mean “anyone should feel comfortable (and become qualified for) doing any sort of task.”
- The “cross-functional teams” and “generalizing specialists” that Agile processes talk about diminish and devalue specialists (like a testing specialist with few other skills outside of testing).

TRUE?

- Testers and coders have different skillsets, concerns, motivations, and incentives
- Cross-functional teams sound nice, but we still pretty much work in our silos, and in a sequential manner, anyway

I'D LIKE TO ADDRESS THE "PEOPLE" SIDE FIRST

"Culture eats strategy for breakfast."

- Peter Drucker

I’D LIKE TO ADDRESS THE “PEOPLE” SIDE FIRST

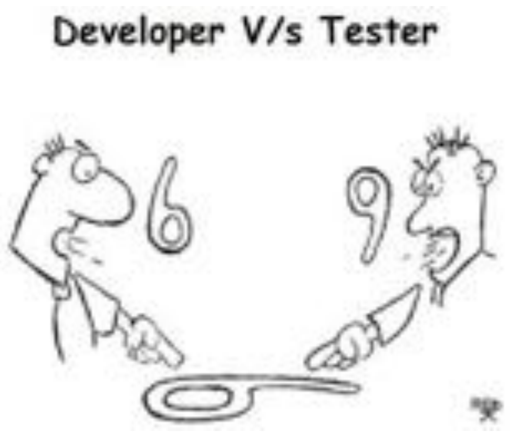
“Every tester has the heart of a coder.

On his desk.

In a jar.”

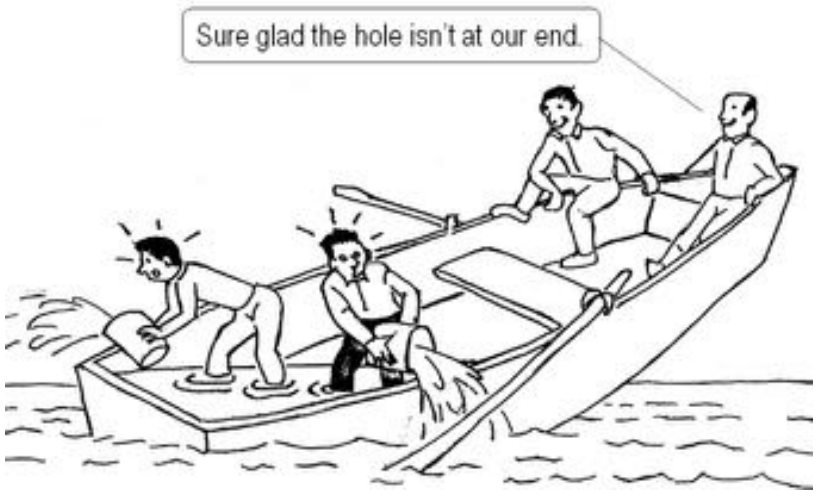
- Richard Hundhausen

AN ADVERSARIAL STANCE?



From https://www.linkedin.com/pulse/testers-vs-developers-stephen-kanagaraj-istqb-ctfl-

TEAMWORK



WHY A CROSS-FUNCTIONAL TEAM?

Wait a minute…

HAVE YOU EVER HEARD?

A coder or yourself say things like:

- “Code complete”
- “Done, but not tested”
- “Done, but not Done Done”
- “Done, but don’t ship it yet because I want to do something else to make it ready”

WHAT WAS THE NEXT QUESTION?

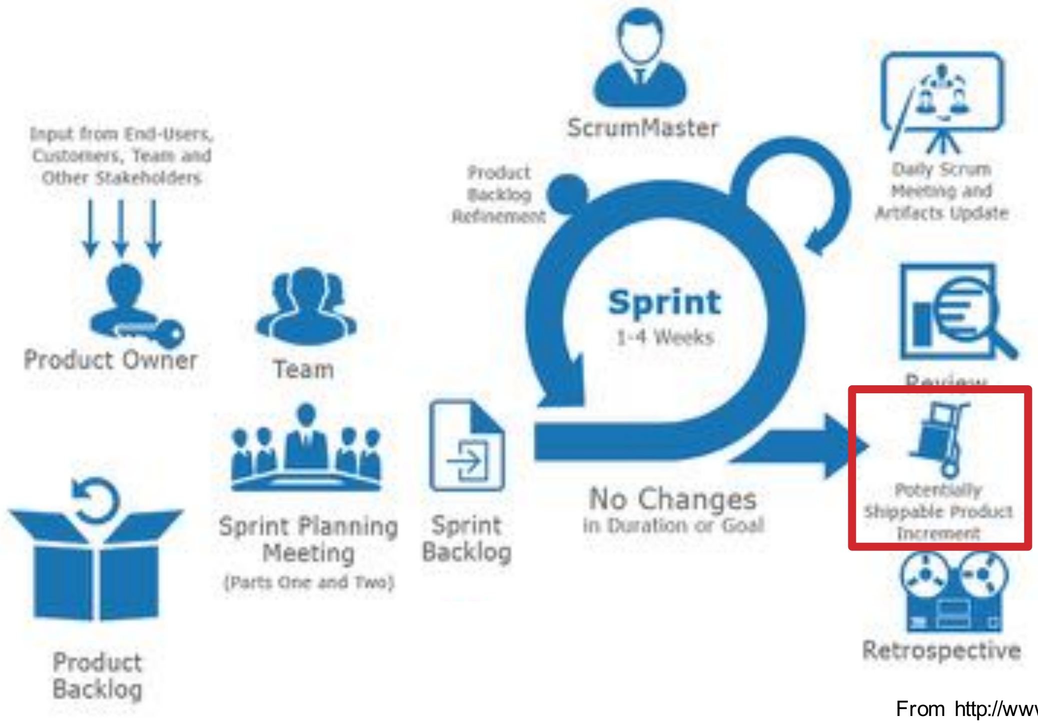
”Well, when WILL it be done????”

- Every Manager and Stakeholder Ever

WHY A CROSS-FUNCTIONAL TEAM?

How will we get "done"??

Let's true-up with the business' and stakeholders' concept of "done"!!


Input from End-Users, Customers, Team and Other Stakeholders
Product Owner
Team
ScrumMaster
Product Backlog Refinement
Sprint
1-4 Weeks
Daily Scrum Meeting and Artifacts Update
Review
Potentially Shippable Product Increment
No Changes
in Duration or Goal
Sprint Planning Meeting
(Parts One and Two)
Sprint Backlog
Product Backlog
Retrospective

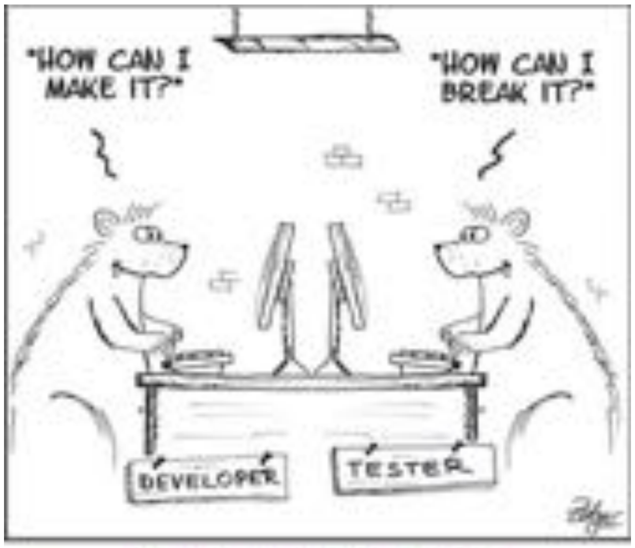
From http://www.ileadco.com

From http://www.casinodailynews.eu/quickspin-to-release-the-3-musketeers-video-slot/

HONORING PEOPLE FOR THEIR SPECIALTIES

A Testing or Quality Specialist

- Has a sixth sense for finding bugs
- Knows how to break stuff
- Thinks about the negative paths, edge cases, etc.

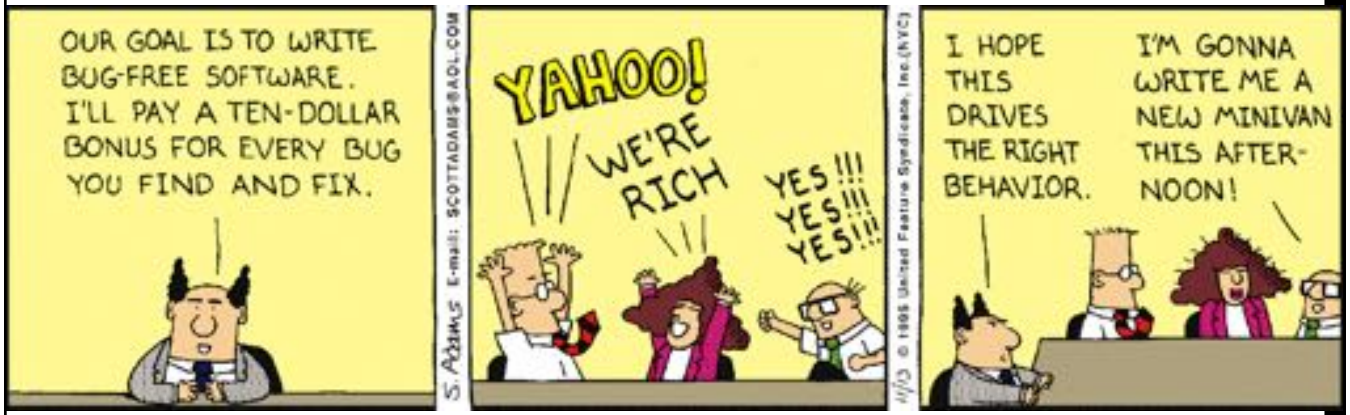


From http://qa-india.blogspot.com/2010/12/testers-vs-developers.html

WHAT'S THE COMMON GOAL?

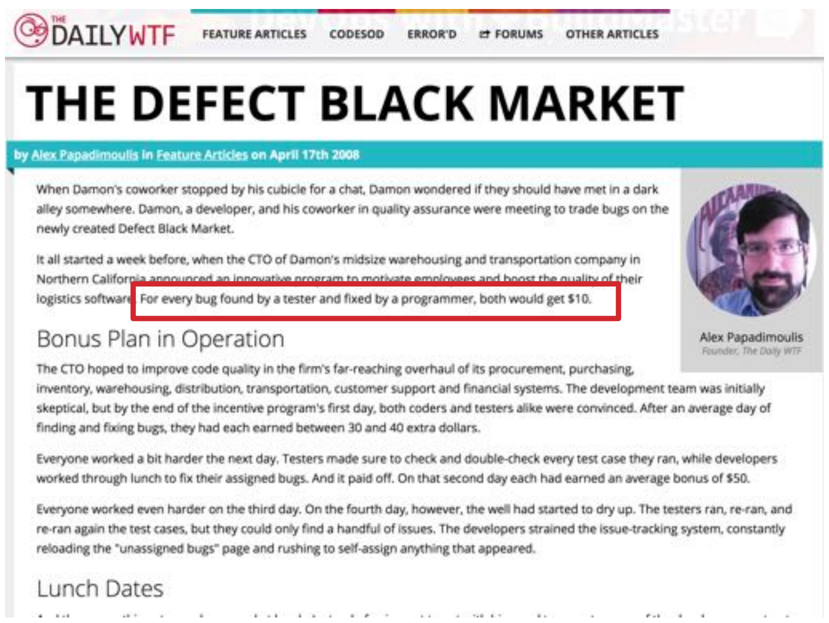
QUALITY

BEWARE INCENTIVES...


OUR GOAL IS TO WRITE BUG-FREE SOFTWARE. I'LL PAY A TEN-DOLLAR BONUS FOR EVERY BUG YOU FIND AND FIX.
YAHOO!
WE'RE RICH
YES!!! YES!!! YES!!!
I HOPE THIS DRIVES THE RIGHT BEHAVIOR.
I'M GONNA WRITE ME A NEW MINIVAN THIS AFTERNOON!
S. Adams E-mail: SCOTTADAMS@AOL.COM
11/13 © 1995 United Feature Syndicate, Inc.(NYC)

From http://dilbert.com/strip/1995-11-13

BEWARE INCENTIVES...



THE DEFECT BLACK MARKET

by Alex Papadimoulis in Feature Articles on April 17th 2008

When Damon's coworker stopped by his cubicle for a chat, Damon wondered if they should have met in a dark alley somewhere. Damon, a developer, and his coworker in quality assurance were meeting to trade bugs on the newly created Defect Black Market.

It all started a week before, when the CTO of Damon's midsize warehousing and transportation company in Northern California announced an innovative program to motivate employees and boost the quality of their logistics software. For every bug found by a tester and fixed by a programmer, both would get $10.

Alex Papadimoulis
Founder, The Daily WTF

Bonus Plan in Operation

The CTO hoped to improve code quality in the firm's far-reaching overhaul of its procurement, purchasing, inventory, warehousing, distribution, transportation, customer support and financial systems. The development team was initially skeptical, but by the end of the incentive program's first day, both coders and testers alike were convinced. After an average day of finding and fixing bugs, they had each earned between 30 and 40 extra dollars.

Everyone worked a bit harder the next day. Testers made sure to check and double-check every test case they ran, while developers worked through lunch to fix their assigned bugs. And it paid off. On that second day each had earned an average bonus of $50.

Everyone worked even harder on the third day. On the fourth day, however, the well had started to dry up. The testers ran, re-ran, and re-ran again the test cases, but they could only find a handful of issues. The developers strained the issue-tracking system, constantly reloading the "unassigned bugs" page and rushing to self-assign anything that appeared.

Lunch Dates

From http://thedailywtf.com/articles/The-Defect-Black-Market

TRUE?

- Testers and coders have different skillsets, concerns, motivations, and incentives
- Cross-functional teams sound nice, but we still pretty much work in our silos, and in a sequential manner, anyway

NO…

- Testers and coders have different specialties, which we want to honor and cultivate
- Testers and coders have the same goal – quality
- Cross-functional teams can be a reality when different specialties pair up, and when testing starts immediately…

QUESTIONS – REVIEW

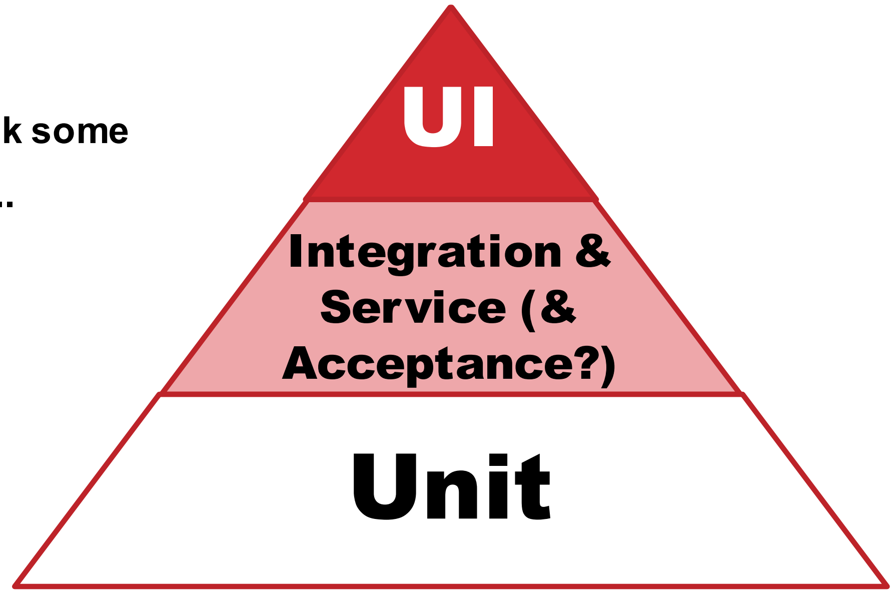
- There is a natural conflict of interest between coders and testers. FALSE
- At the very least, testers should "keep their distance" from coders. FALSE
- Testers should be measured and compensated by the number of bugs they find. FALSE
- The "cross-functional teams" that Agile processes talk about mean "anyone should feel comfortable (and become qualified for) doing any sort of task." FALSE
- The "cross-functional teams" and "generalizing specialists" that Agile processes talk about diminish and devalue specialists (like a testing specialist with few other skills outside of testing). FALSE

QUESTIONS – TRUE OR FALSE?

- Unit tests are written by coders, and only coders.
- BAs could write acceptance tests.
- Automated UI tests, by their very nature, are brittle and require a lot of maintenance.

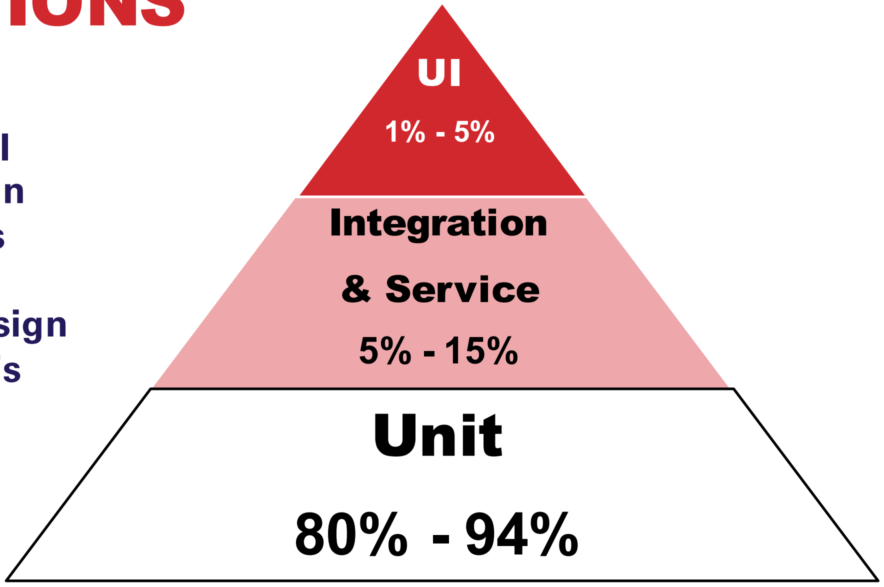
TESTING PYRAMID

Let’s talk some details...



PROPORTIONS

To achieve these proportions and still cover the application adequately requires good application architecture and design patterns! “Quality is built-in from the beginning.”



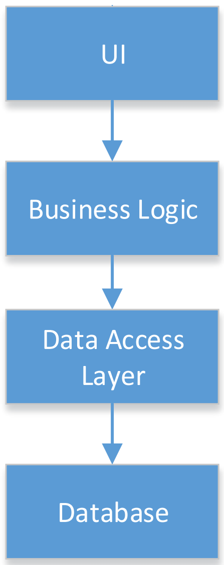
BUILD QUALITY IN FROM THE BEGINNING?

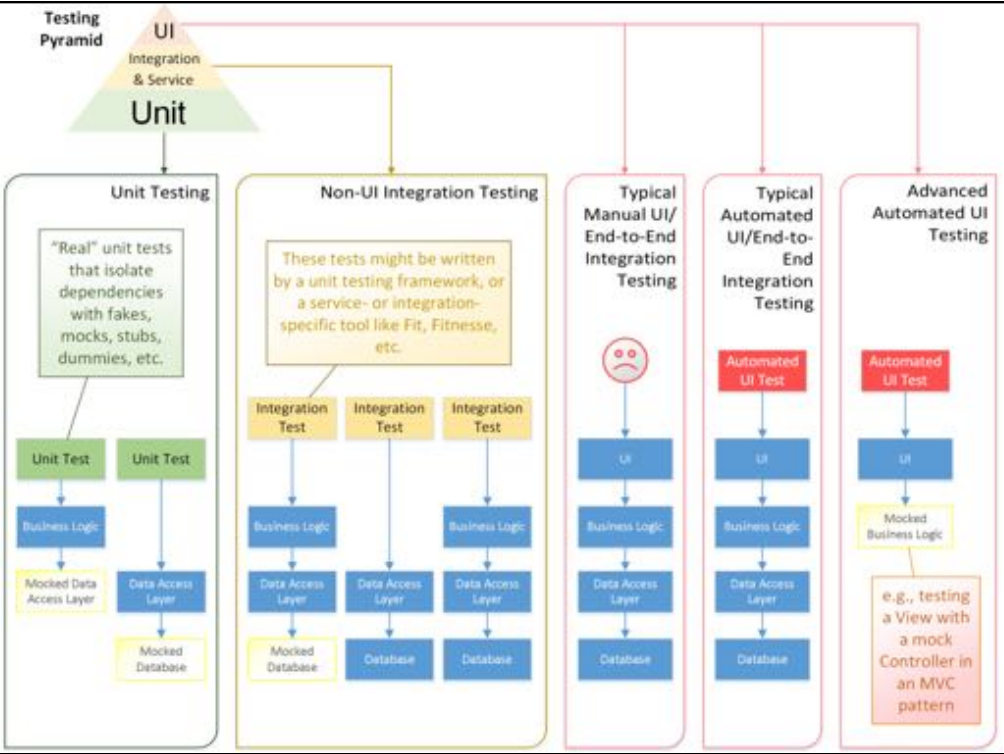
"Quality is job #1!"

Following the Test Pyramid requires good design and architecture?

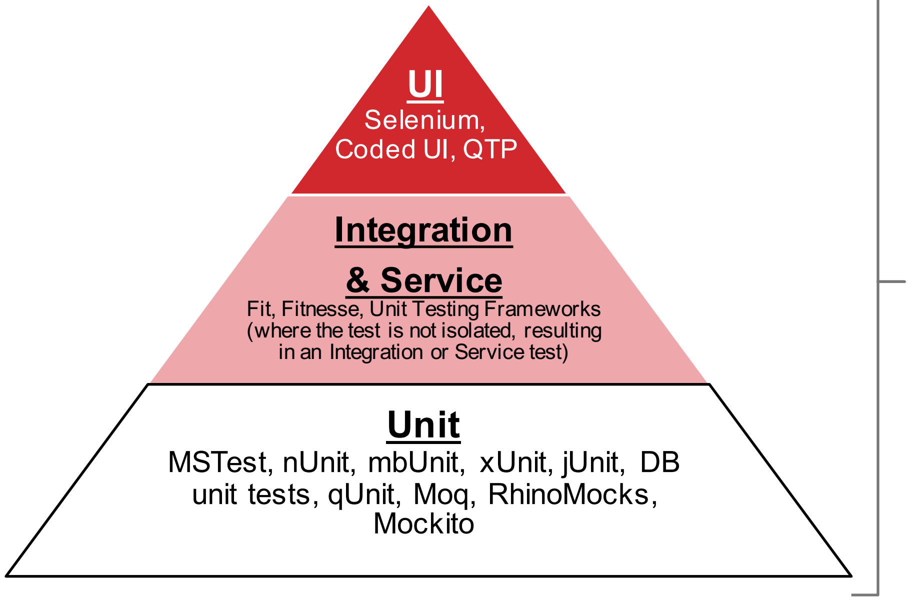
So…it's an architectural statement? It's for architects???

AN EXAMPLE APP…



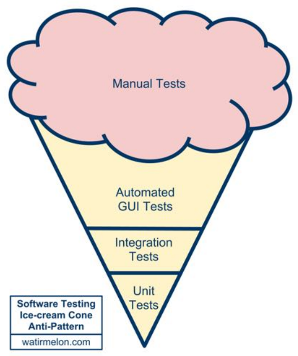


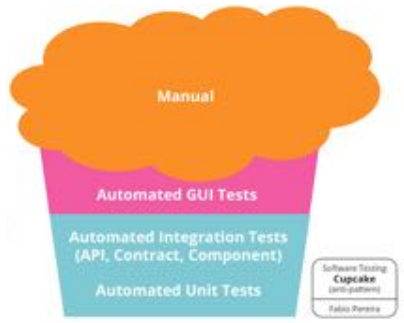
TOOLS

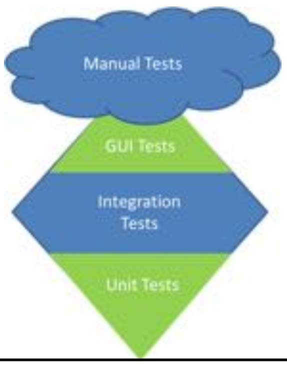

UI
Selenium, Coded UI, QTP
Integration & Service
Fit, Fitnesse, Unit Testing Frameworks (where the test is not isolated, resulting in an Integration or Service test)
Unit
MSTest, nUnit, mbUnit, xUnit, jUnit, DB unit tests, qUnit, Moq, RhinoMocks, Mockito

BDD frameworks (Cucumber, SpecFlow) could use any of these test types and tools to implement their test steps

TEST PYRAMID ANTI-PATTERNS

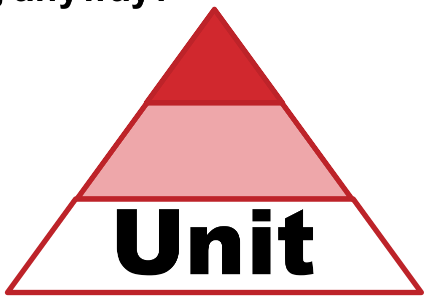



Manual
Automated GUI Tests
Automated Integration Tests (API, Contract, Component)
Automated Unit Tests
Software Testing Cupcake (anti-pattern)
Fabio Pereira



HEY – I'M NOT A CODER!

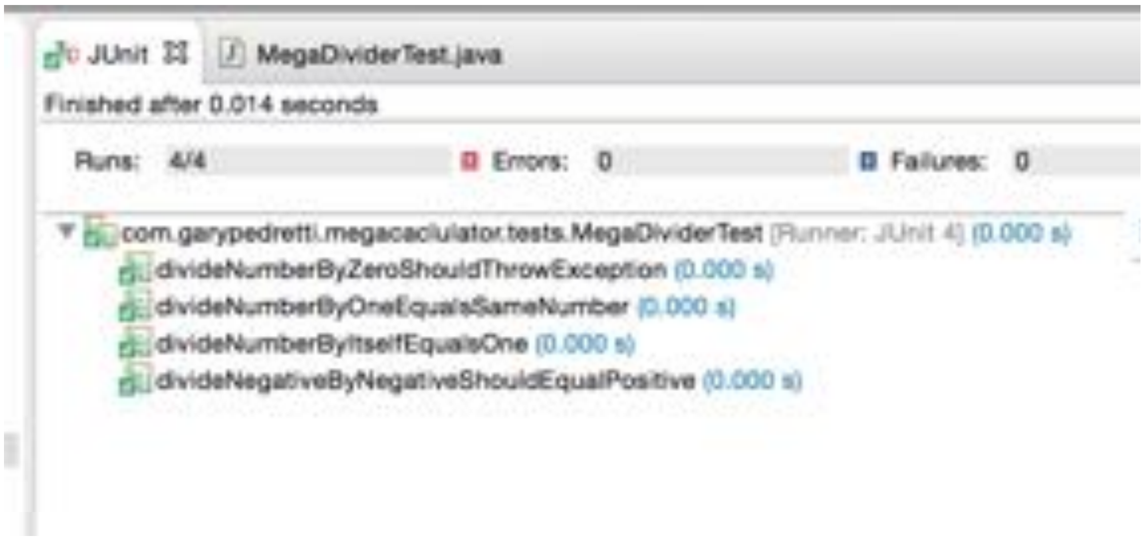
- Isn't that unit testing stuff hardcore code?
- WHAT are the coders actually unit testing, anyway?



UNIT TESTS – CODE THAT TESTS CODE

```
[TestClass]
public class DivisionMachineTests
{
  [TestMethod]  // This is the code that tests the other code...it tests the DivisionMachine “target”
  public void DivideANumberByItselfShouldReturnOne()
  {
    int theNumber = 5;
    DivisionMachine dm = new DivisionMachine();
    int answer = dm.Divide(theNumber,  theNumber);
    Assert.AreEqual(1, answer);
  }
}
public class DivisionMachine  // "Target" code being tested - an class that contains functionality
{
  public int Divide(int p1, int p2)
  {
    return p1 / p2;
  }
```

UNIT TESTS – PAIRING ACROSS SPECIALTIES

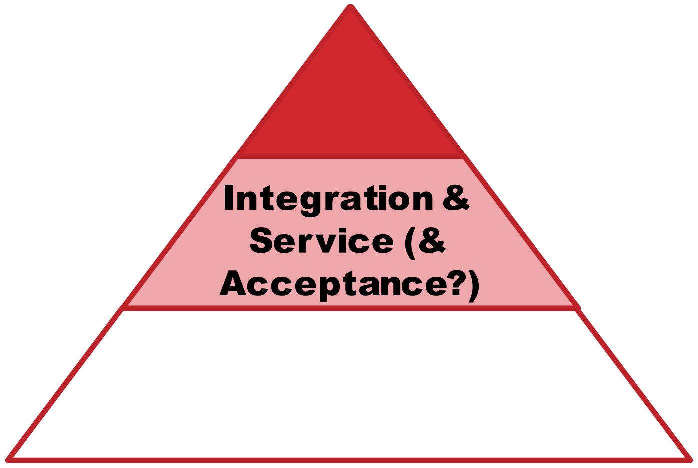


```
Finished after 0.014 seconds

Runs: 4/4        Errors: 0        Failures: 0

▼ com.garypedretti.megacaclulator.tests.MegaDividerTest [Runner: JUnit 4] (0.000 s)
    divideNumberByZeroShouldThrowException (0.000 s)
    divideNumberByOneEqualsSameNumber (0.000 s)
    divideNumberByItselfEqualsOne (0.000 s)
    divideNegativeByNegativeShouldEqualPositive (0.000 s)
```

PAIRING

- There when tests are being written, offering your unique viewpoint
- Encourage plain language test names – these are system documentation!

HOW ABOUT INTEGRATION / SERVICE / ACCEPTANCE TESTS?



HOW ABOUT INTEGRATION / SERVICE / ACCEPTANCE TESTS?

- **Integration and Service Tests are really just high-level, declarative steps of standard test cases**
  - Think about the test cases you shell in before there is any system to describe in detail
  - More on Declarative vs. Imperative later…
- **Acceptance Tests are also an area where "traditional" testers can get involved**
  - Think about your existing connection to BAs - flushing out requirements via test cases, etc.
  - Now, take that pairing to the next level!

TOOLS IN THIS AREA

Acceptance Test Driven Development

- **Fit and Fitnesse – (primarily) table-based acceptance testing tool that allows business users to specify expectations in Excel, etc.**

Behavior-Driven Development (BDD) – all about conversations, but recently often focused on toolsets

- **Cucumber and SpecFlow**
- **Scenario based, with common language Given…When…Then constructs**

BDD – BEHAVIOR-DRIVEN DEVELOPMENT

Having conversations among the team members and stakeholders about the behaviors

is more important than

capturing the behaviors

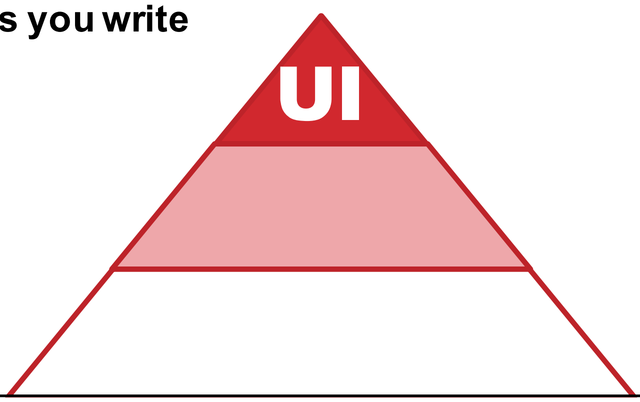
which is more important than

automating the behaviors.

– Liz Keogh, ALM Chicago 2013

UI TESTS

- The automation tools keep getting better and better…more robust
- But they will never change the fact: automated UI tests will always be the most brittle tests you write



TOOLS IN THIS AREA

- **QTP**
- **Microsoft Coded UI**
  - Works with Windows apps, Excel sheets, anything – not just browsers
- **Selenium**
  - Currently best-of-breed for browsers / web apps
  - Has server farm capabilities for lightweight load and performance testing

A WORD ON TRADITIONAL SOFTWARE QUALITY CONTROL – V&V

Verification and Validation (from Boehm, *Software Risk Management*, 1989)

- **Verification**: Are we building the product right? (lack of defects, meets requirements – does what we say it would do, internal?)

- **Validation**: Are we building the right product? (meets stakeholder needs, external?)

SOFTWARE QUALITY CONTROL – V&V – IN AN AGILE ENVIRONMENT

Verification

- Test to find bugs – Exploratory Testing
- Test we built to the best of our understanding – Acceptance Tests, Service/Integration Tests, Unit Tests

Validation

- Product Owner/Business Lead acceptance
- Use Sprint Review/Demo/Showcase and other stakeholder feedback to validate the requirements were/are still valuable

QUESTIONS – REVIEW

- **Unit tests are written by coders, and only coders. FALSE**
- **BAs could write acceptance tests. TRUE**
- **Automated UI tests, by their very nature, are brittle and require a lot of maintenance. TRUE…**

QUESTIONS – TRUE OR FALSE?

- It's impossible to start testing right at the beginning of a project or Sprint / Iteration / slice of time in an Agile process.
- When using an Agile process, it's OK to have one Sprint or Iteration for coding followed by one Sprint or Iteration of testing.
- Responsible testing can't start until coding is complete.
- Even if you can get functional testing "inside the Sprint" when using Scrum or another Agile process, Sprints or Iterations inevitably look like "mini-Waterfalls," with testing necessarily pushed to the end of the timebox.
- I am frustrated by compressed test cycles – we go last, so if there are schedule problems we always bear the brunt of them.

ACTIVITY: DRAW OUT YOUR CURRENT VALUE STREAM

What does it take at your current organization to go from idea to production?

Who is involved?

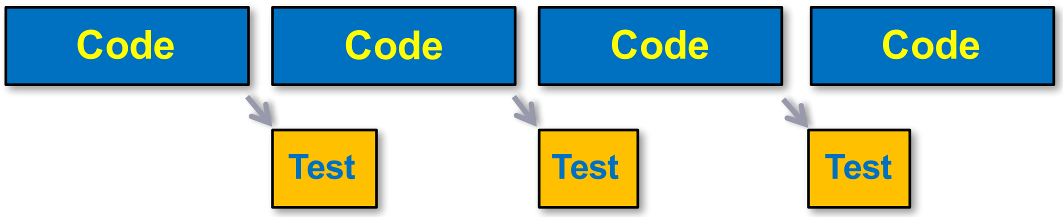
What environments are involved?

How long does it take value (software) to pass through the various stages?

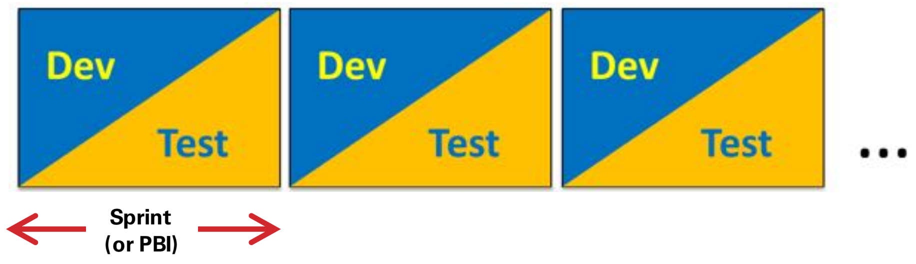
Where does testing happen? What are the prerequisites? How long does it take?

"STAGGERED CYCLES (SPRINTS, ITERATIONS, PHASES)"

A Development Team hands stuff over to QA and moves on…

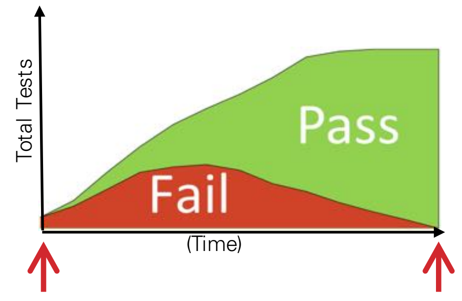


TEST IN PARALLEL WITH CODE



Remember what managers and stakeholders want???

TEST THROUGHOUT THE CYCLE



Identify tests before work starts. These will be initially **failing** tests.

All acceptance tests must pass.

HOW?

- **Quality begins with refinement (grooming) of requirements and work items**
  - Test assumptions
  - Flush out additional acceptance criteria
- **Quality continues with cycle (Sprint, phase, etc.) planning**
  - Ask – how will we test this? What test data will be necessary? How much time will we need to allot?
  - Shell out declarative acceptance tests based on acceptance criteria
  - Flush out additional acceptance criteria

HOW?

- **Inside the cycle, work closely with BAs and coders**
  - BAs for requirements, edge cases, etc.
  - Coders to help with unit tests, and understand what is being unit tested
  - Share with coders your functional test cases, from the beginning, as they get fleshed out, and at the end
  - Enlist coders to help test!!!
- **As code and features are implemented, continually revise your test cases to be up-to-date**
  - Moving from Declarative to Imperative

DECLARATIVE VS. IMPERATIVE

- **People often cast this as an Either/OR, but it is most powerful as an AND, changing through time**
- **Let's look at an example…**

Story: Animal Submission

As a Zoologist

I want to add a new animal to the site

So that I can share my animal knowledge with the community

Scenario: Successful Animal Submission

DECLARATIVE

Given I'm on the animal creation page

When I add a new animal

Then I should see the page for my newly created animal

And the notice a message thanking me for my animal submission

IMPERATIVE

Given I'm on the animal creation page

When I fill in Name with 'Alligator'

And select Phylum as 'Chordata'

And fill in Animal Class with 'Sauropsida'

And fill in Order with 'Crocodilia'

And fill in Family with 'Alligatoridae'

And fill in Genus with 'Alligator'

And check Lay Eggs

And click the Create button

Then I should see the notice 'Thank you for your animal submission!'

And the page should include the animal's name, phylum, animal class, order, family, and genus

ABOUT ENVIRONMENTS AND TEST TIMING

Fast Feedback is the goal

From worst to best in terms of fast feedback:

- Test in Production only (ouch!)
- Test in some lower environment only – e.g., the QA environment
- Test in a testing specialist's local sandbox
    - Testing the current code – the latest from the source control system
    - **Before** this code is deployed to any shared environment
- Test as code is written
    - Test-Driven Development
    - Coding/Testing specialist pairing during test writing (both unit and functional)
    - Coding/Testing specialist pairing during code writing

REDEFINING "TEST": ANY CHECK OF QUALITY

Verifying Test Cases

Proofread

Desk Check

Unit Test

Functional Test

Performance Test

- Not every change has a performance impact…think spelling error – is a fully transparent desk check OK?
- Reuse existing functional assets whenever possible

BUGS

Lubarsky's Law of Cybernetic Entomology: prov.

"There is *always* one more bug."

From The Jargon File (version 4.4.7)

http://www.catb.org/jargon/html/L/Lubarskys-Law-of-Cybernetic-Entomology.html

WHAT SHOULD YOU TEST? – RISK-BASED TESTING

If there are an infinite number of edge cases for any given scenario….

Think about

- Risk mitigation
- ROI
- High-risk
- High-value
- Quality

Don't gold plate any of your development, including tests

QUESTIONS – REVIEW

- It's impossible to start testing right at the beginning of a project or Sprint / Iteration / slice of time in an Agile process. FALSE
- When using an Agile process, it's OK to have one Sprint or Iteration for coding followed by one Sprint or Iteration of testing. FALSE
- Responsible testing can't start until coding is complete. FALSE
- Even if you can get functional testing "inside the Sprint" when using Scrum or another Agile process, Sprints or Iterations inevitably look like "mini-Waterfalls," with testing necessarily pushed to the end of the timebox. FALSE
- I am frustrated by compressed test cycles – we go last, so if there are schedule problems we always bear the brunt of them. TRUE?

ACTIVITY: REVISIT – DRAW OUT YOUR CURRENT VALUE STREAM

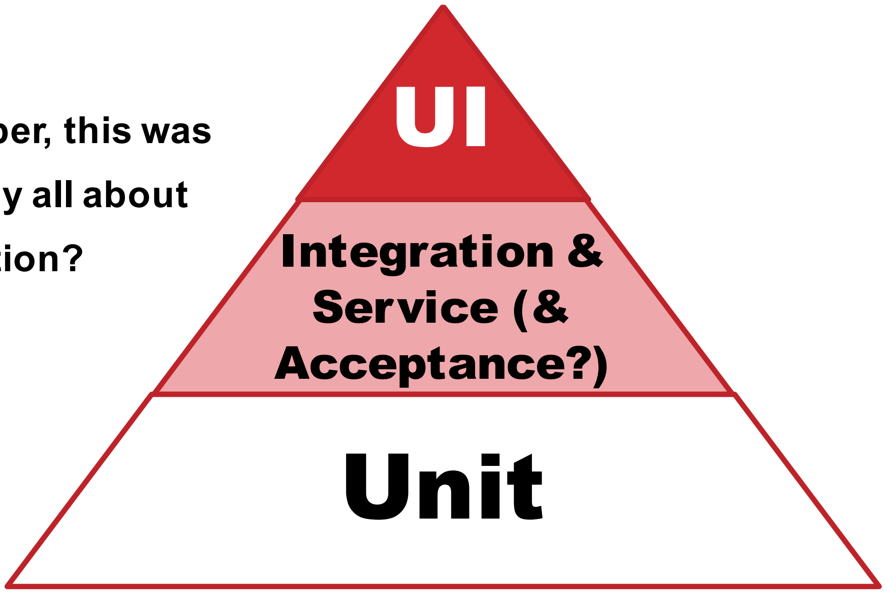
Add in what you want to change!

QUESTIONS – TRUE OR FALSE?

- Test automation is best handled by a dedicated, specialized team outside of the typical team(s) that are actively designing, building, and testing applications.
- When using an Agile process, it's highly unlikely tests can be automated inside a Sprint or Iteration – so inevitably we'll be playing "catch-up" later.

ON AUTOMATION

Remember, this was originally all about automation?



ON AUTOMATION OF UI-LEVEL TESTS…

Automation is an investment – choose wisely when to automate

- They become inventory you must maintain – they're part of your codebase forever,
- Don't allow non-updated tests and/or false negatives in these tests any more than you would for unit or other tests
- If you can't trust your test results, you'll end up with broken window syndrome and all your tests will rot…

RECORD/PLAYBACK TOOLS VS. HAND-CODED AUTOMATED UI TESTS

- Record/Playback is very low investment
- But not very maintainable or robust
- Coded tests are generally the opposite – expensive, but maintainable (but "maintained" is still up to you!)

RECORD / PLAYBACK

But Record/Playback is for amateurs, right?

- "Record-playback tools are almost always a bad idea for any kind of automation, since they resist changeability and obstruct useful abstractions. They are only worth having as a tool to generate fragments of scripts which you can then edit as a proper programming language, in the manner of Twist or Emacs."

– Martin Fowler

http://martinfowler.com/bliki/TestPyramid.html

RECORD / PLAYBACK

Go back to the flow of Declarative to Imperative as you move through a cycle

Do the same with Record/Playback -> Coded!!

RECORD/PLAYBACK VS. CODING

Not mutually exclusive options because of flow:

- Simple recordings, initially (early in the Sprint?)
- As the UI comes together and **starts** to stabilize (sooner than you think), change your investment
- Throw away the recordings (remember the investment is low)

RECORD/PLAYBACK VS. CODING

Not mutually exclusive options because of flow:

- Consciously choose to increase your investment by creating more robust coded automation
  - Before considering a PBI "Done"…otherwise where would you automate your tests, in a Test Automation Sprint?? <- THAT WAS SARCASM
- Maintain the coded automation going forward, just like any other coded test (like unit tests) – this is now part of your codebase

RECORD/PLAYBACK VS. CODING

Not mutually exclusive options because of the mix for a single test

- Part of what you put into a coded test could be code generated by record/playback tools
  - Because it's good enough
  - Because it's a reasonable base to improve on with custom code

BACK TO THE CROSS-FUNCTIONAL TEAM

- **Who could help you with custom, coded automated tests?**
- **Are they on your team?**

QUESTIONS – REVIEW

- **Test automation is best handled by a dedicated, specialized team outside of the typical team(s) that are actively designing, building, and testing applications. FALSE**
- **When using an Agile process, it's highly unlikely tests can be automated inside a Sprint or Iteration – so inevitably we'll be playing "catch-up" later. FALSE**

WRAP-UP – KEY TAKEAWAYS

New ways for architects, testers, BAs, and coders to work together in a cross-functional manner

New ways to think about Mike Cohn's popular Testing Pyramid

New patterns for developing a well-tested application

THE BAR IS HIGH

THANK YOU!

GARY@GARYPEDRETTI.COM

@GARYPEDRETTI

WWW.SODOTOSOLUTIONS.COM