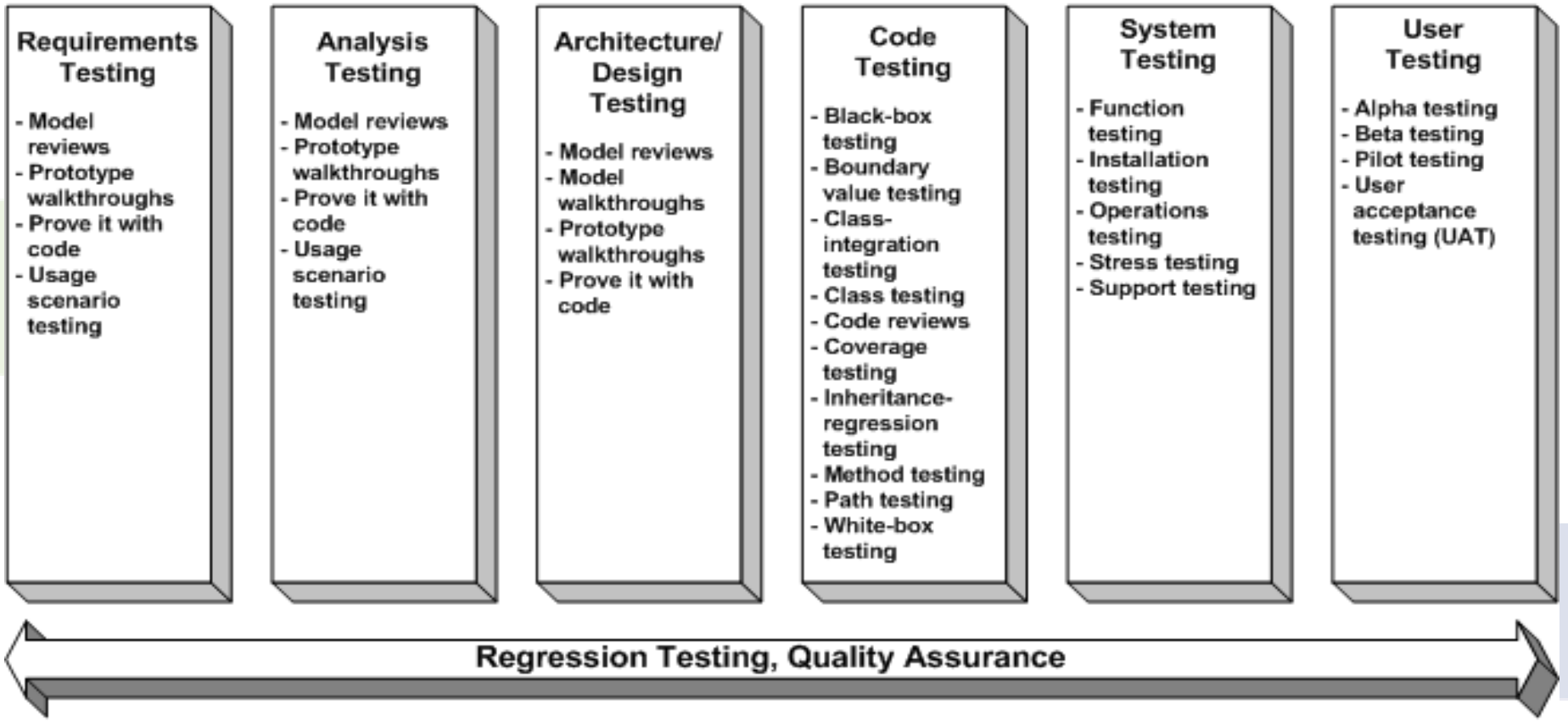
Where do We Test?

Practical Agile
Agile in your Context

Are we Confused?



Practical Agile
Agile in your Context

How About Now?

Taken from a random blog post:

"Below is an exhaustive list of "types of software testing" and a brief description about each type of software testing, like load testing, stress testing, unit testing, system testing, acceptance testing, certification testing, performance testing, user acceptance testing, penetration testing, automated testing, beta testing, compatibility testing, security testing, benchmark testing, functional testing, negative testing, destructive testing, integration testing, regression testing, alpha testing, end-to-end testing, path testing, smoke testing, black box testing, stability testing, usability testing etc., and many more, …"

Practical Agile
Agile in your Context

Lets Focus

- Lets focus on two major Testing Levels:
  - Unit Tests
  - End to End (Integration) Tests

Unit Tests

- What is Unit Testing?
- What is a “Unit”?
  - A line of code? one method? A Class? A group of classes? A subsystem? The entire system?
- Lowest Level of testing
  - A unit is the smallest testable part of a system.
  - Usually one or a small number of classes

End 2 End Testing

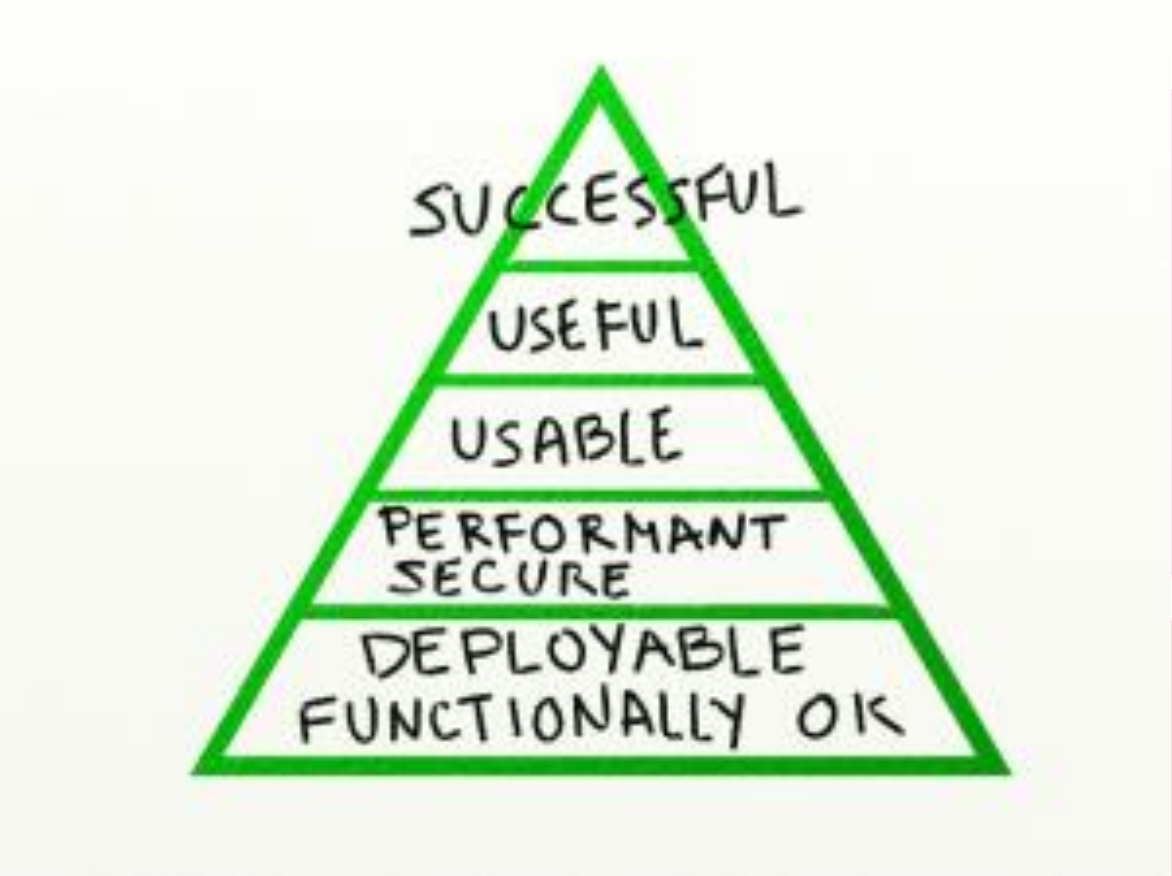
- What End to End?
- Where are the ends of our system?
  - All the system? A bunch of Servers? One server? Over the GUI? A single Subsystem?
- Highest Level of testing
  - Usually we aim for the entire system (or as much of it as we can practically include)
  - User point of view (Acceptance)

Lets Think

What are the pros and cons of testing at each level?

Practical Agile
Agile in your Context

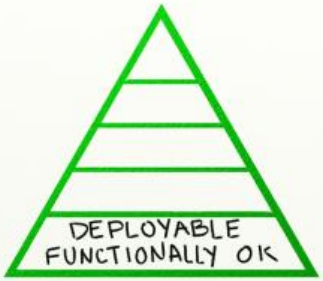
The Quality Pyramid



Practical Agile
Agile in your Context

Deployable, Functionally OK

- The system need to work
  - We need to be able to deploy it
  - Satisfy minimum functionality
- Enough is enough
  - Minimum viable product (Lean Startup)



Practical Agile
Agile in your Context

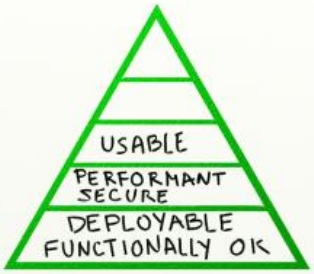
Performant, Secure

- The system needs to work **well**
  - Scalability
  - Security
  - Availability…
- Enough is enough
  - Quantify your goals.

Practical Agile
Agile in your Context

Usable

- Can the system be used?
  - User Interaction design
  - Graphical design
  - Creating a community



Practical Agile
Agile in your Context

Usefull

- The system needs to help the users
  - Are the feature in use?
  - The 80-20 rule
- Take out the feature which are not used
  - they have negative ROI

Practical Agile
Agile in your Context

Successful

- The system solves a business problem
  - Does it saves money?
  - Does it save time?
  - What are the business goals?
- Was it right for the business in the first place?

Practical Agile
Agile in your Context

Lets Think

Where does your organization invest most of its testing effort?

Practical Agile
Agile in your Context

Lets Simplify

- If I Could have 3 magic boxes, I would like to know:

1. Am I doing things right?
2. Am I doing the right things?
3. Am I Adding Business Value?

Am I Doing things Right

- This is what unit test are used for.

- Unit tests:
  - Are fast
  - Test each unit in isolation
  - Enable me to test all paths of my code
  - Will improve my technical design

Am I Doing The right thing?

- E2E tests are a good tool for:
  - Help me understand the requirements
- E2E tests:
  - Goes through all the system
  - Help me understand how the system behaves
  - Help me refine Acceptance Criteria

Am I adding Value?

Practical Agile
Agile in your Context

Where to Invest?

What	How much?
Automated Unit	60% - 70%
Automated E2E	20% - 30%
Exploratory	20% - 30%

When to Invest?

- Unit Testing
  - An integral part of the coding phase. (TDD)
  - All code should be tested before it moves to next stage
- E2E Testing
  - Most of the effort is done as part of the requirement.
  - Actual automation in parallel to coding
- Exploratory Testing
  - Final activity before Done.

Who?

- Unit Testing
  - Programmers (Each on his own code)
- E2E Testing
  - Product + Testers (Programmers) – define the test scenarios
  - Test Engineers/Programmers - Automation
- Exploratory Testing
  - Expert testers

Summary

- The only Way to go fast is with High Quality
- Quality is a joint team effort
- A single testing Level will never be enough
  - **All** code needs to be at least “Checked, Accepted & Explored”

Some Testing Smells

- A very short list that usually indicate that testing effort is not balanced between testing levels:
  - One Tool to win them all
  - Full E2E regression
  - We Don’t Trust them
  - …
  - …

One Tool to win them all

- Symptoms:
  - all automated test are written using the same tool
- Discussion:
  - Use the correct tool for the job
  - Unit and E2E are just too different
  - May lead to single level of testing
  - We neglect to answer and important question

Full E2E Regression

- Symptom:
  - We try to cover ALL cases in E2E Tests
- Discussion:
  - E2E tests are a tool to verify requirements
  - Use unit testing to do full regression
  - Full regression is slow
  - Not enough control over the system
- May indicate “We don’t trust them” smell

We Don’t Trust them

- Symptom:
  - Tests are duplicated by QA team
  - Also known as *“The person who wrote the code can’t test it”*
- Discussion:
  - Effort is duplicated
  - Since QA usually focus on E2E testing, most likely this will result in one level of testing

Practical Agile
Agile in your Context