Quality Assurance In Microservice Architectures

Krishnan Chandran Irina Barykina

Department of Informatics,
Intelligent Adaptive Systems, UHH

2016

Outline

- What is Quality Assurance?
- QA is easy, isn't it?
- QA on Development stage.
- QA on Deployment stage.
- QA after Release.
- Conclusion.

Introduction

Definition

Quality Assurance refers to planned and systematic production processes that provide confidence in a product's suitability for its intended purposes.

- QA must prevent bugs and failures, not identify them.
- QA is wasteful on the last stages of development cycle.

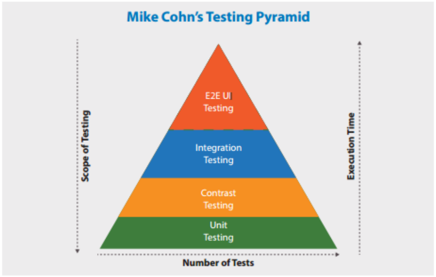
Challenges

-

Test Pyramid
A balanced test portfolio

Mike Cohen’s Test Pyramid

- Foundation Layer: Unit Tests
- Intermediate Layer: Contract Testing and Integaration Testing
- Tip of the Pyramid: E2E UI Tests

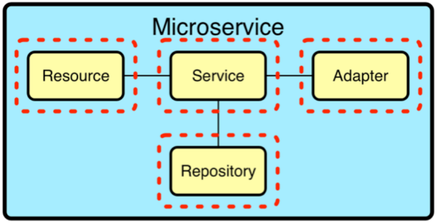


Types of Tests

Applying the layers in a microservice

Unit Tests

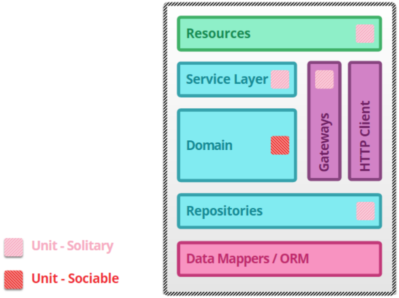
- Coverage limited to individual components
- Useful in services, resources, repositories, and adapters
- "every build should run the tests, and a failed test should fail the build"
- "Solitary Unit Test and Sociable Unit Test"
- "Also a relevant design tool when combined with TDD"



Types of Tests
Applying the layers in a microservice

Unit Tests

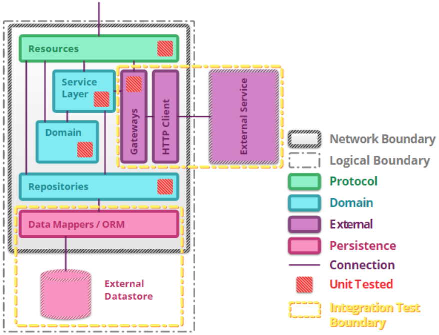
- Coverage limited to individual components
- Useful in services, resources, repositories, and adapters
- "every build should run the tests, and a failed test should fail the build"
- "Solitary Unit Test and Sociable Unit Test"
- "Also a relevant design tool when combined with TDD"



Types of Tests
Integration,Component and Contract Testing

Integration Tests

- Covers communication paths and interactions between components to detect interface defects.
- Gateway Integration and Persistence Integration



Types of Tests

Integration,Component and Contract Testing:

Component Tests

Types of Tests

Integration,Component and Contract Testing

Contract Tests

- Verifies that the contract expected by a consuming service is met.
- Integration Contract Testing and Consumer Driver Contract Testing.
- The Overall Service contract is the sum of individual contract tests.

Scenario 1

Testing between microservices internal to an application

Scenario 2

Testing between an internal microservice and an external API

Scenario 3

Microservice exposed to public domain

Deployment

RAD and Deployment Pipeline

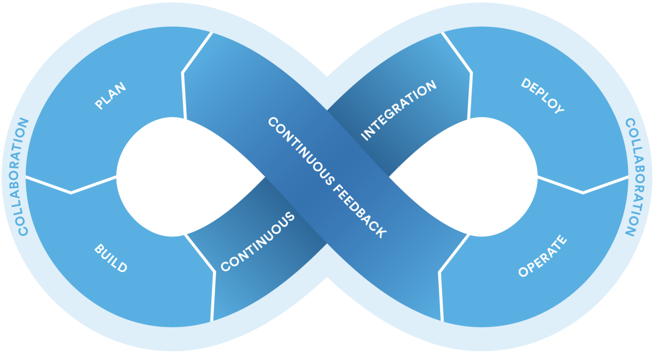
Deployment

Continuous Deployment and Delivery

Deployment
DevOps Culture

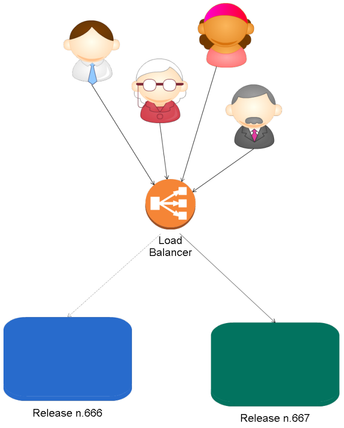
DevOps Culture:

- Aim: break silos between development and later stages
- Requirements: shared responsibility and autonomy of teams



After Deployment
Smart releasing strategies

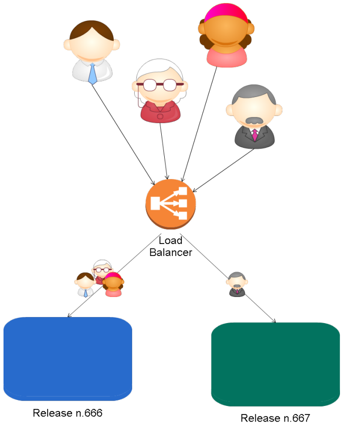
- Smoke Test Suites
- Blue/Green Deployment
- Canary releasing



After Deployment

Smart releasing strategies

- Smoke Test Suites
- Blue/Green Deployment
- Canary releasing



After Deployment

Monitoring

Tools

- xUnit framework
- stubbing and mocking (on the example of Mockito)
- smart stubbing with Mountebank
- testing of data passing between services (on the example of SOAP UI)
- consumer driven testing (on the example of Pact)
- End-to-End Testing (BDD Tools, JBehave, Cucumber)

References

Sam Newman. *Building Microservices*. O'Reilly and Associates, 2015.

Mike Cohn. *Succeeding with Agile: Software Development Using Scrum*. Addison Wesley, 2009.

Arvind Sundar. An insight into microservices testing strategies, 2016. URL `https://www.infosys.com/it-services/validation-solutions/white-papers/documents/microservices-testing-strategies.pdf`.

Toby Clemson. Testing strategies in a microservice architecture, 2014. URL `http://martinfowler.com/articles/microservice-testing`.

Martin Fowler. Continuousdelivery, 2014. URL `http://martinfowler.com/bliki/ContinuousDelivery.html`.

Vishal Naik. Architecting for continuous delivery, 2016. URL `https://www.thoughtworks.com/insights/blog/architecting-continuous-delivery`.