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.

Introduction
Challenges

- ▶ unpredictable timely availability for testing
- ▶ hard to perform exhaustive integration testing
- ▶ separated logs and data storages
- ▶ hard to maintain proper configuration of testing environments
- ▶ *but (!)* easy to organize low-level testing and catch most of the bugs early
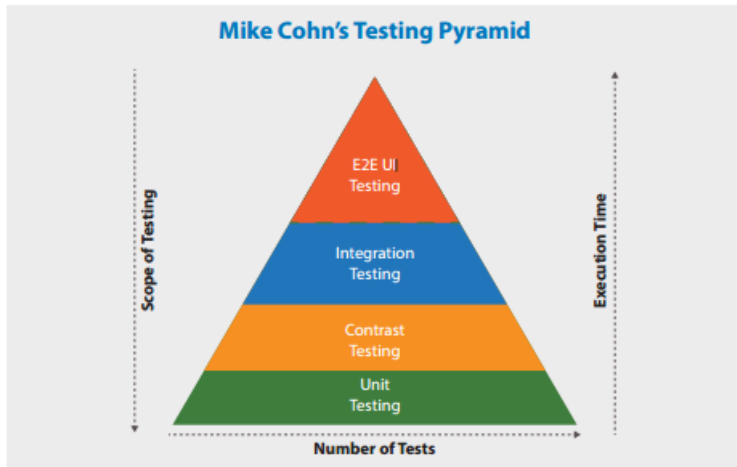
Introduction
Case Study
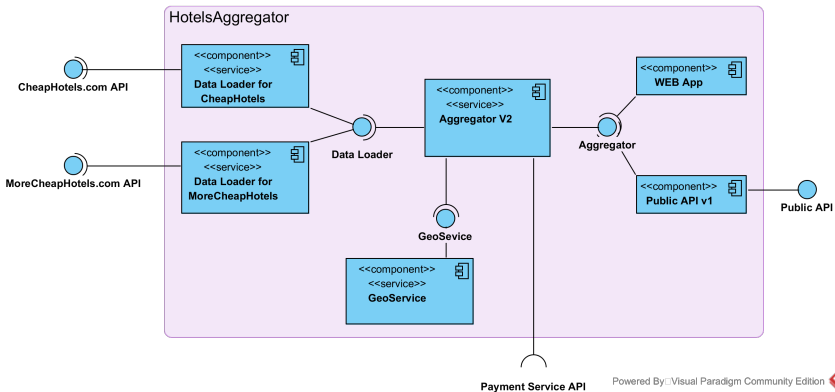
Test Pyramid
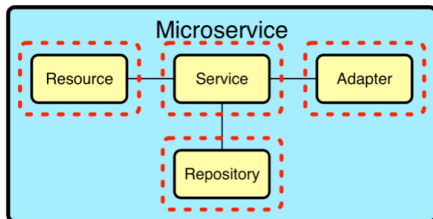**A balanced test portfolio**

Mike Cohen's Test Pyramid

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"
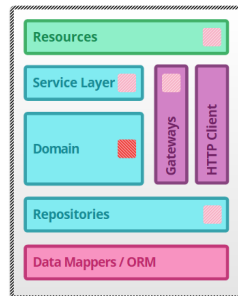
Types of Tests
Applying the layers in a microservice

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"
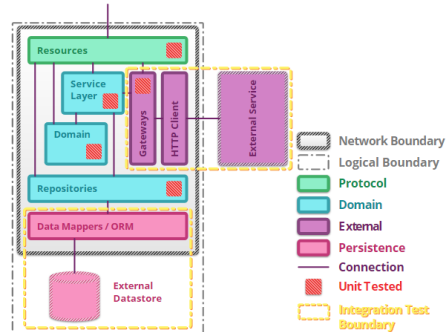

Unit - Solitary

Unit - Sociable

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

▨	Network Boundary
⋯	Logical Boundary
▨	**Protocol**
▨	**Domain**
▨	**External**
▨	**Persistence**
—	**Connection**
▨	**Unit Tested**
⌐	**Integration Test Boundary**

Types of Tests
Integration,Component and Contract Testing

Integration Tests

- ▶ A component is any
  well-encapsulated, coherent
  and independently
  replaceable part of a larger
  system.
- ▶ Isolation of the service is
  achieved by replacing
  external collaborators with
  test doubles
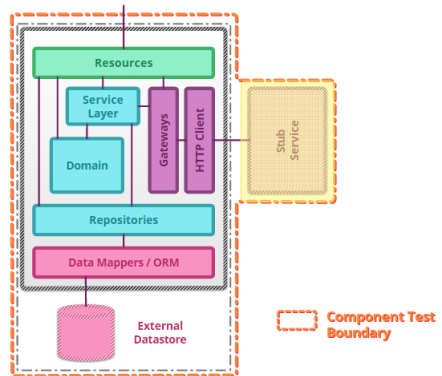
Types of Tests
Integration,Component and Contract Testing

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.
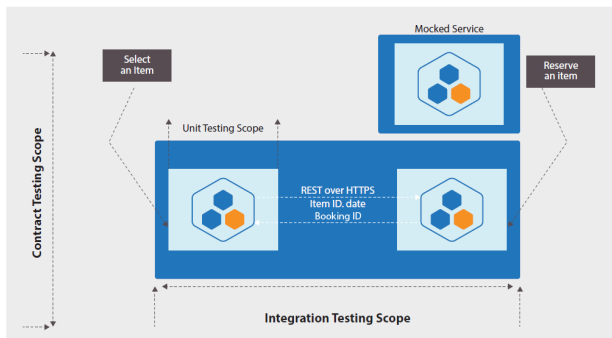
Types of Tests
Non Functional Tests

Non Functional Tests validate the quality characteristics of the component.

- ▶ Performance Tests.
- ▶ Tests for Scalability.
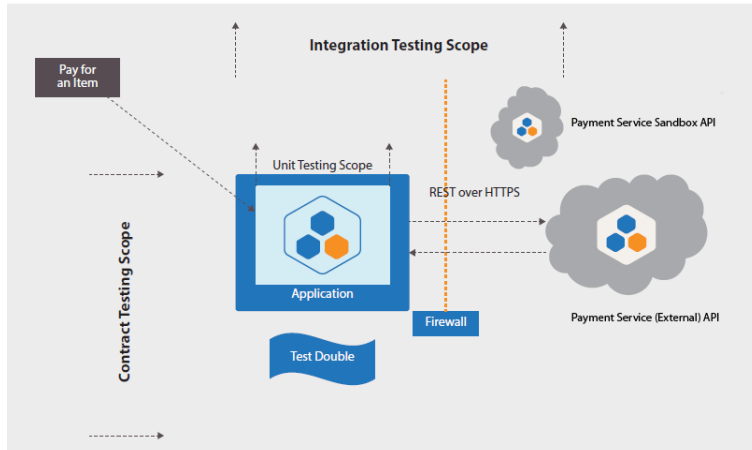- ▶ Resiliency Tests.
- ▶ Security Tests.

Testing between Microservices internal to an application or residing within the same application
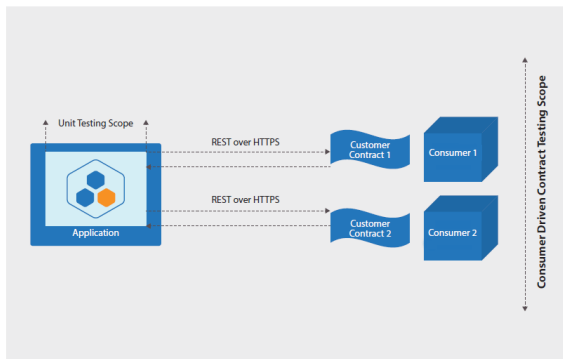
Tasks like Selecting a Hotel and Booking a Hotel.

Testing between an internal microservice and an external API
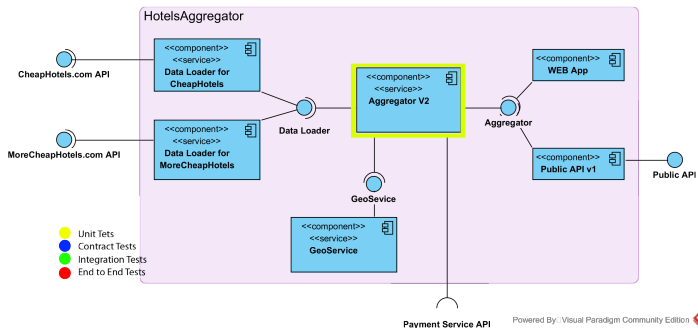**Interaction with a Payment API**

Microservice exposed to public domain
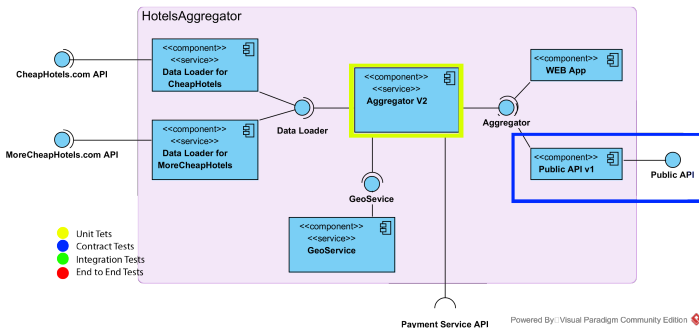## A publicly exposed application which is accessed by a Web API

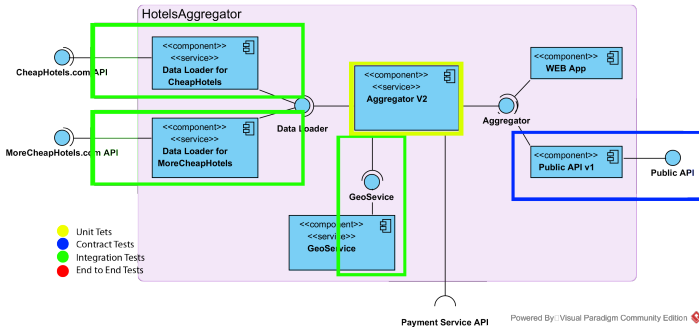# Test Boundaries

Test boundaries for the example
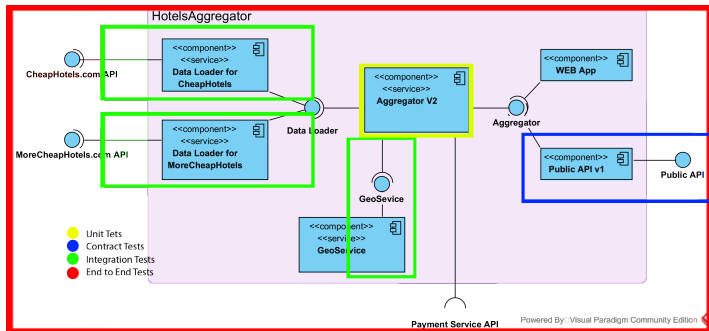
Test Boundaries

Test boundaries for the example

Powered By: Visual Paradigm Community Edition

Test Boundaries

Test boundaries for the example

Test Boundaries
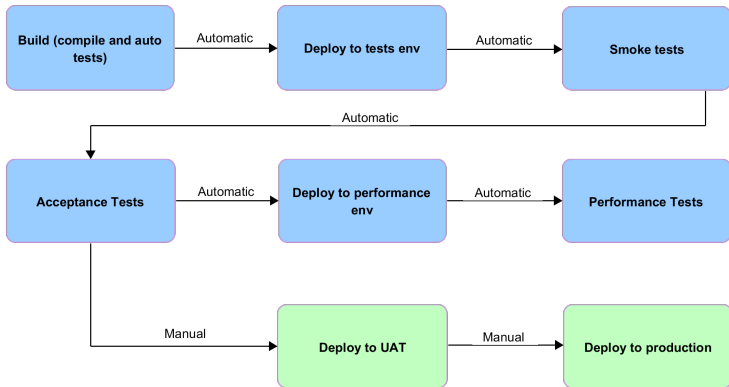
Test boundaries for the example

Deployment
Rapid Application Delivery

- ► RAD is a prerequisite for microservices []
- ► Exhaustive tests could be slow.
- ► Remedy: Deployment Pipeline.
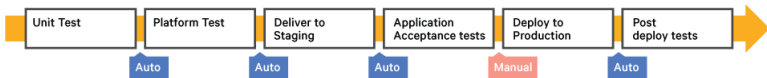
Deployment
Deployment Pipeline

Deployment
Continuous Deployment and Delivery

Continuous Delivery

| Unit Test | Platform Test | Deliver to Staging | Application Acceptance tests | Deploy to Production | Post deploy tests |

Auto — Auto — Auto — Manual — Auto

Continuous Deployment

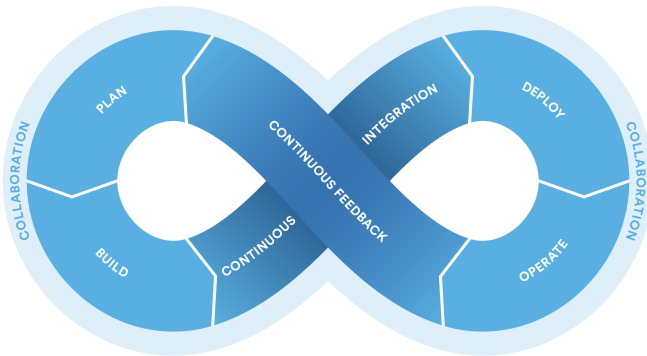| Unit Test | Platform Test | Deliver to Staging | Application Acceptance tests | Deploy to Production | Post deploy tests |

Auto — Auto — Auto — Auto — Auto
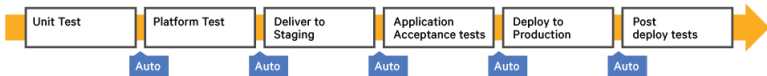
Deployment
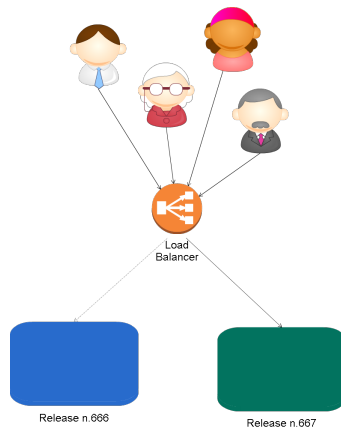**DevOps Culture**

DevOps Culture:

- ▶ Aim: break silos between development and later stages
- ▶ Requirements: shared responsibility and autonomy of teams
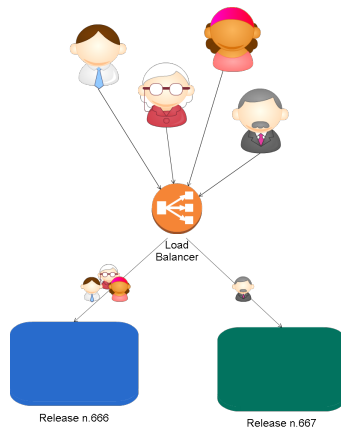
After Deployment
Smart releasing strategies

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

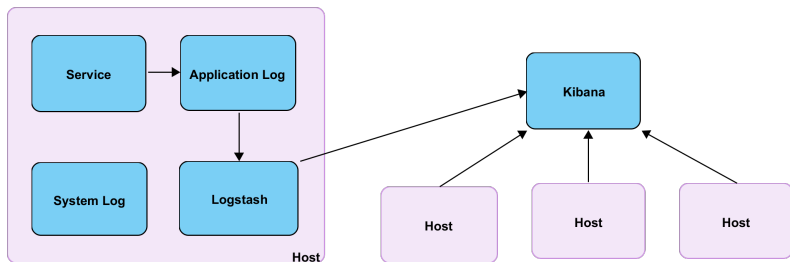**After Deployment**
Smart releasing strategies

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

Load Balancer

Release n.666

Release n.667

After Deployment
Logging

Powered By□Visual Paradigm Community Edition

After Deployment
Monitoring

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.