TOKYO TEST FEST

From Good to Great: *Evolving Your Role as a Quality Consultant*

Udit Gattani
Engineering Manager, QA
Mercari, Inc.

Contents

The Quality Consultant

How did I become Quality Consultant?

- Worked as a developer
- Then evolved myself into testing
- Learned how to **automate tests** and **build frameworks**
- Learned about automation across **different layers, platforms**
- Gained experience in different types of projects across **multiple domains and platforms**
- Engaged in project assessment, pre-sales, workshops, delivery management
- Learning on the job and with the time

Quality Consultant Role & Skillset

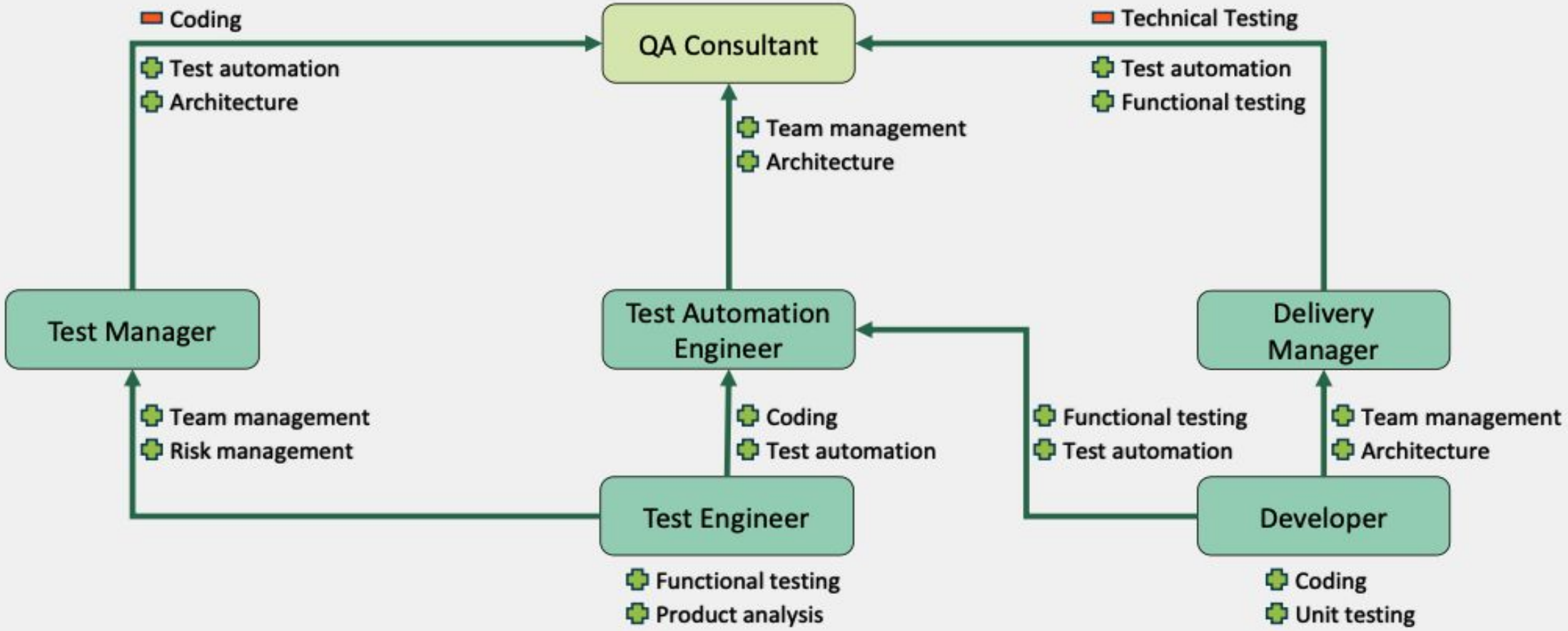
External Consultant	Internal Consultant
Pre-sales & Assessment focused	Test Management focused
Advisory & Change Mgmt role	Decision Maker role
Technical Oriented mindset	Product Oriented mindset
Architect or Consultant title	Strategist or Coach title

Skills Necessary

- Technical Testing Knowledge
- Architecture Mindset
- Stakeholder Management
- Project Management

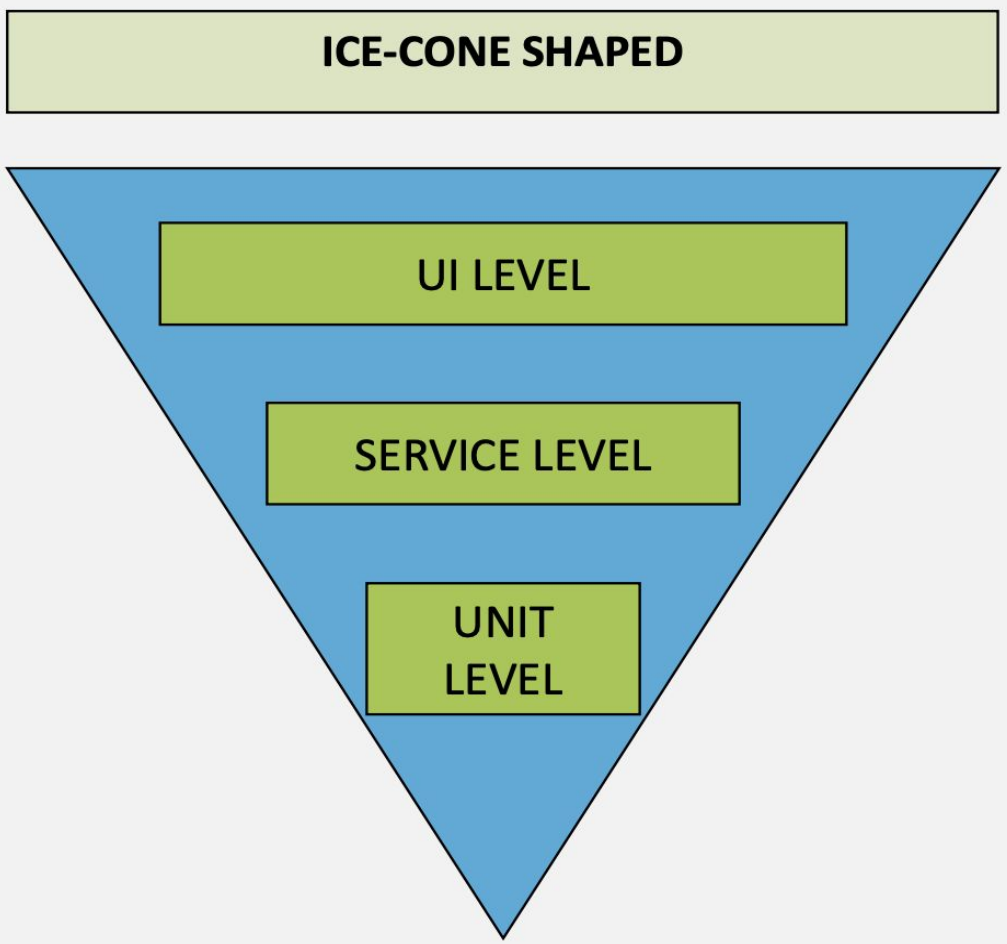
Potential Career Paths

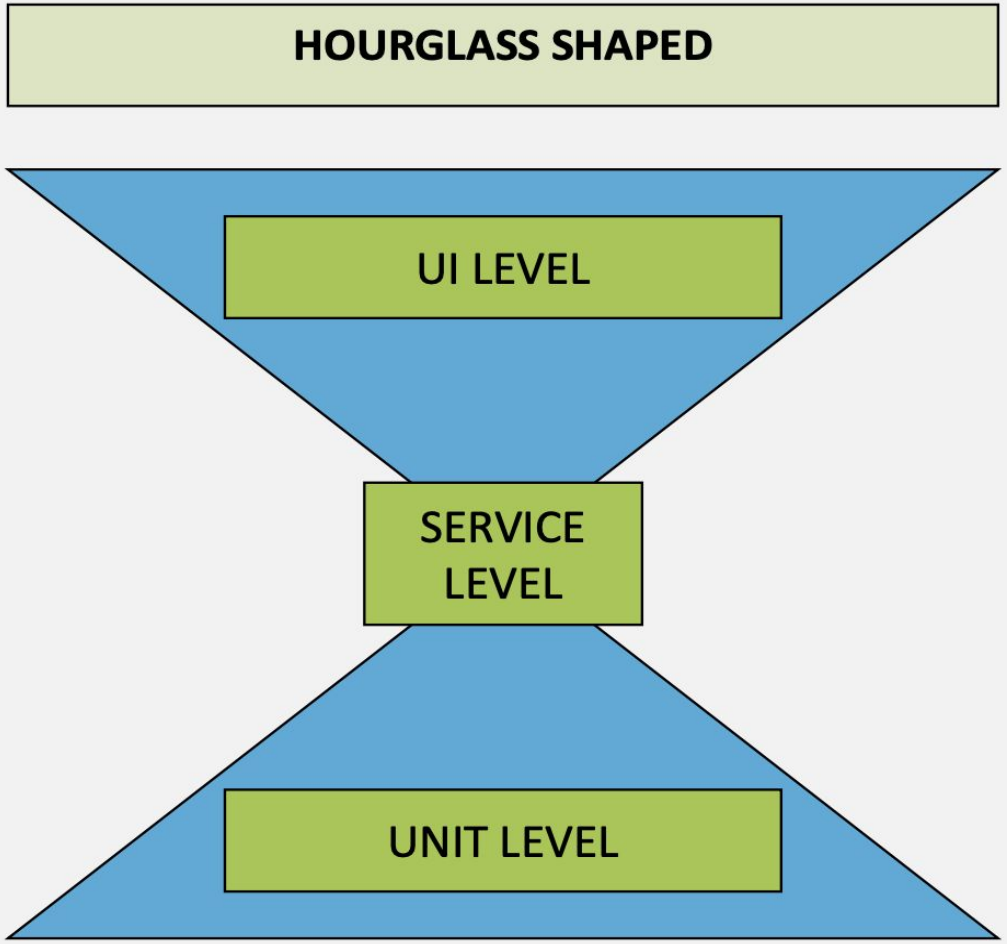
Potential Career Paths



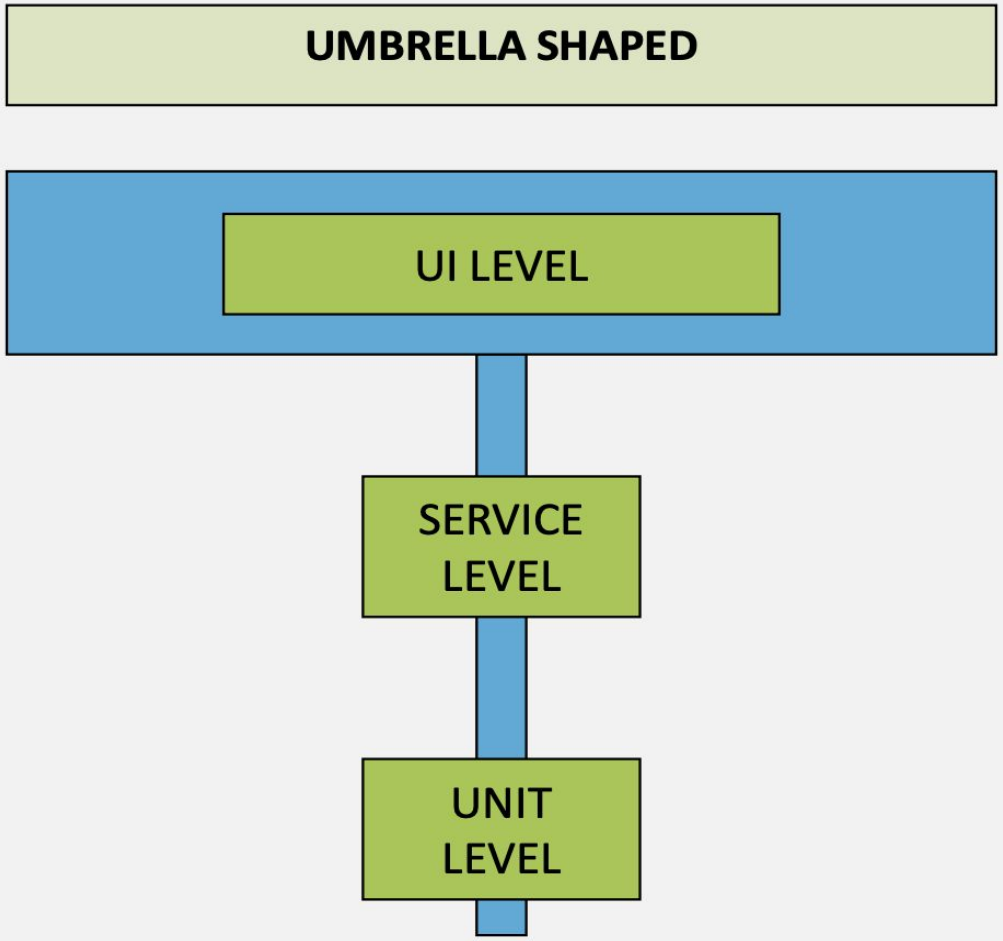
Test Pyramids

When Projects Go Wrong





When Projects Go Really Wrong



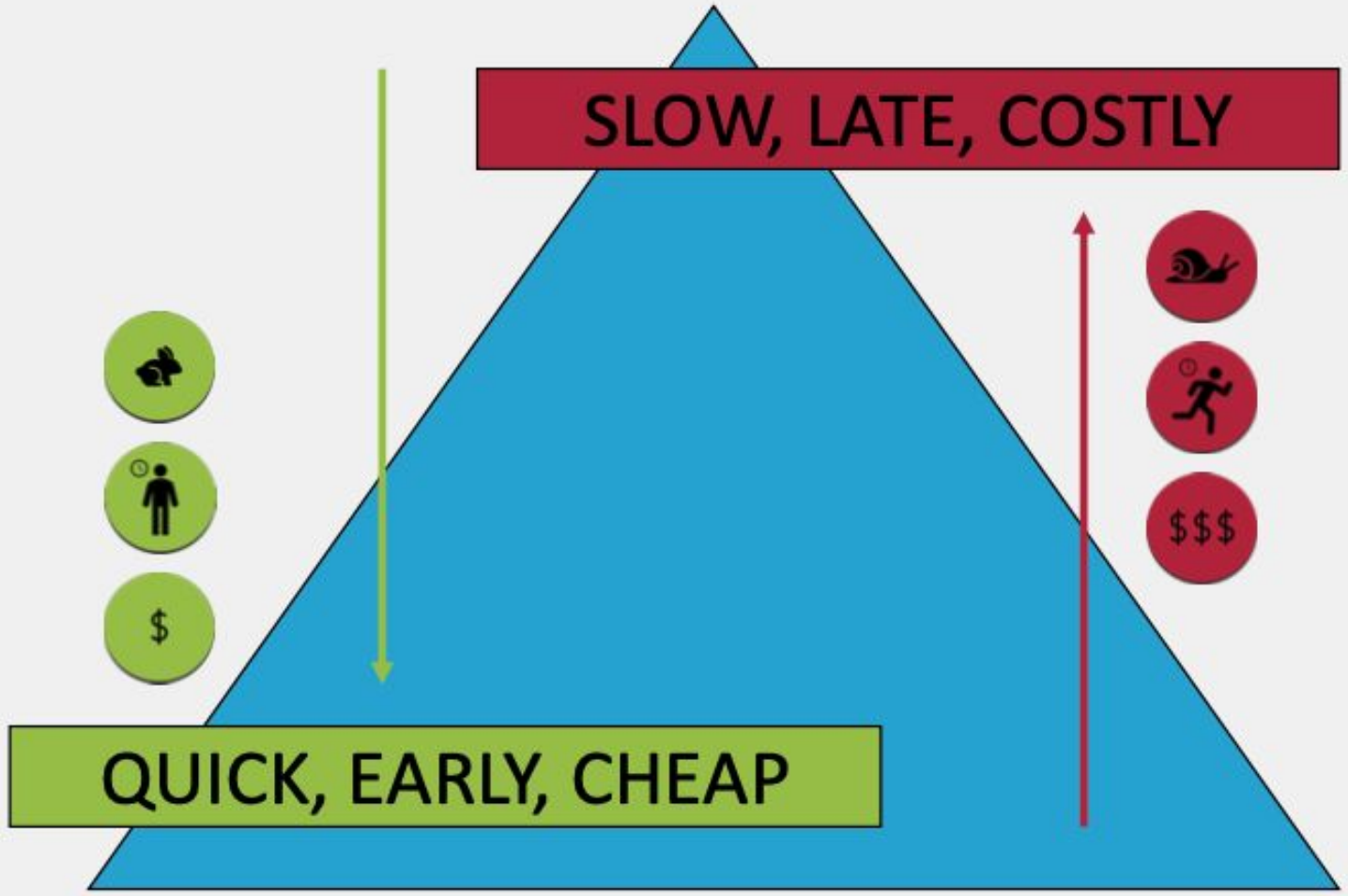
The Pyramid & Shift-Left

The pyramid indicates:

- Cost effectiveness
- Test development time
- Test execution time
- Complexity of tests
- Dependency on other services

Shift-left approach:

- Move testing as early as possible in the lifecycle
- On the pyramid it's a move down to lower levels



Agile Test Automation Pyramid

Originates from Mike Cohn (2003) and Jason Huggins (2006). Popularized by Martin Fowler (2012).

UI

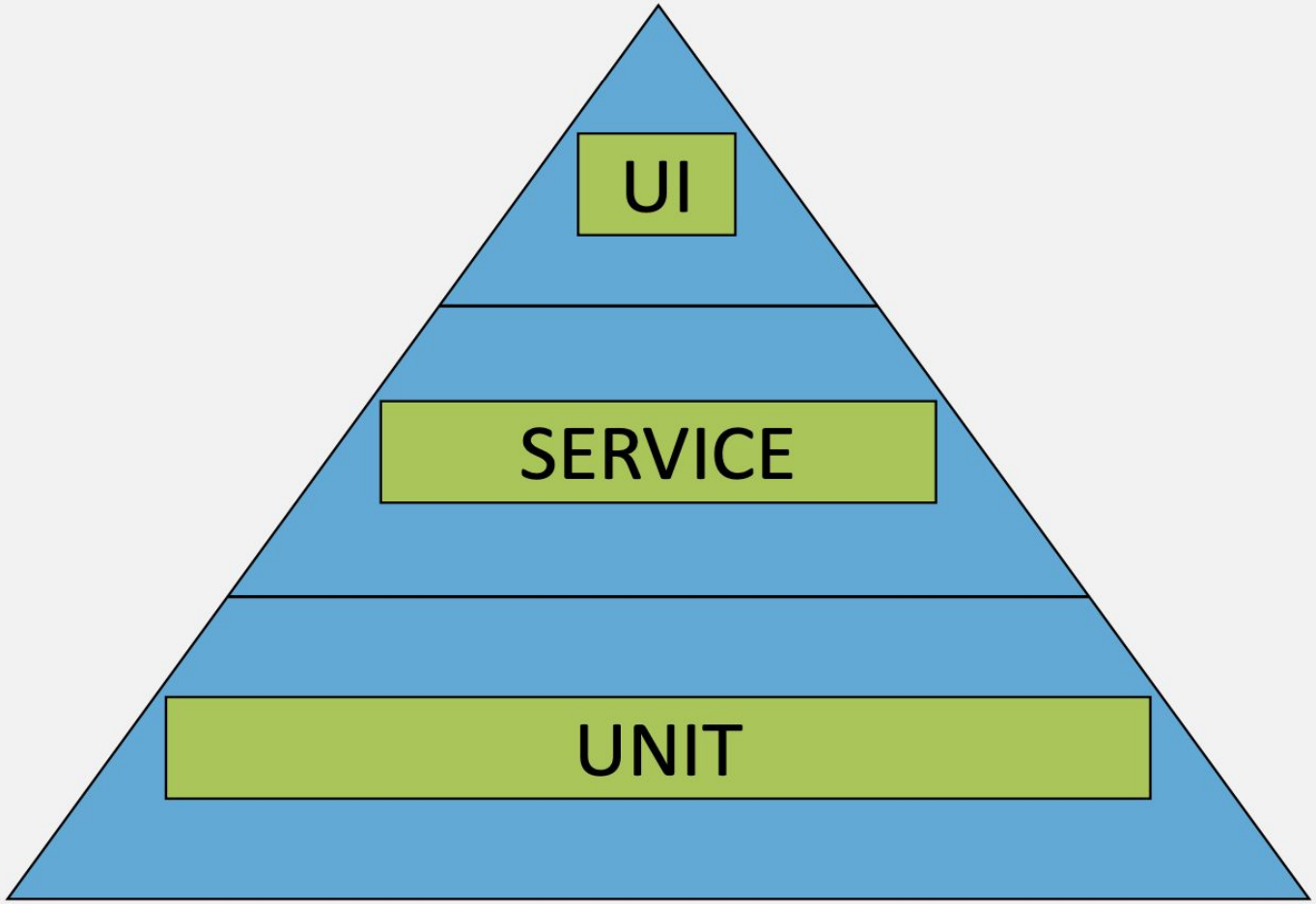
- The E2E user journeys are represented on the top of this pyramid

Service

- All kinds of service testing both with real and mocked data, real and stubbed out services

Unit

- The smallest units of a program is validated with atomic test cases



Agile Test Automation Pyramid

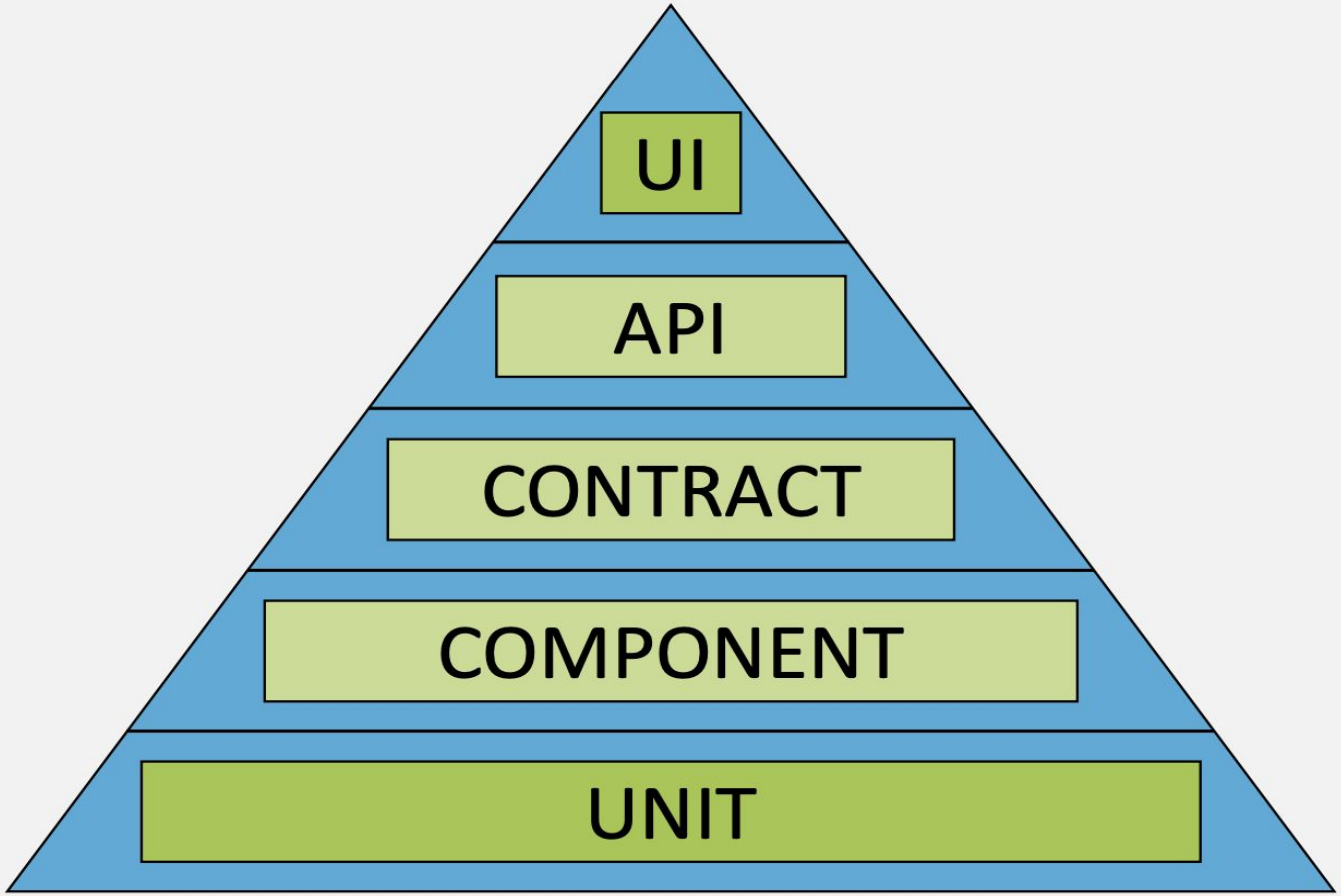
Expands the Service level of the original pyramid.
More focused on integrating to the pipeline.

UI & API
- System integration testing
- Real data and real services close to Prod

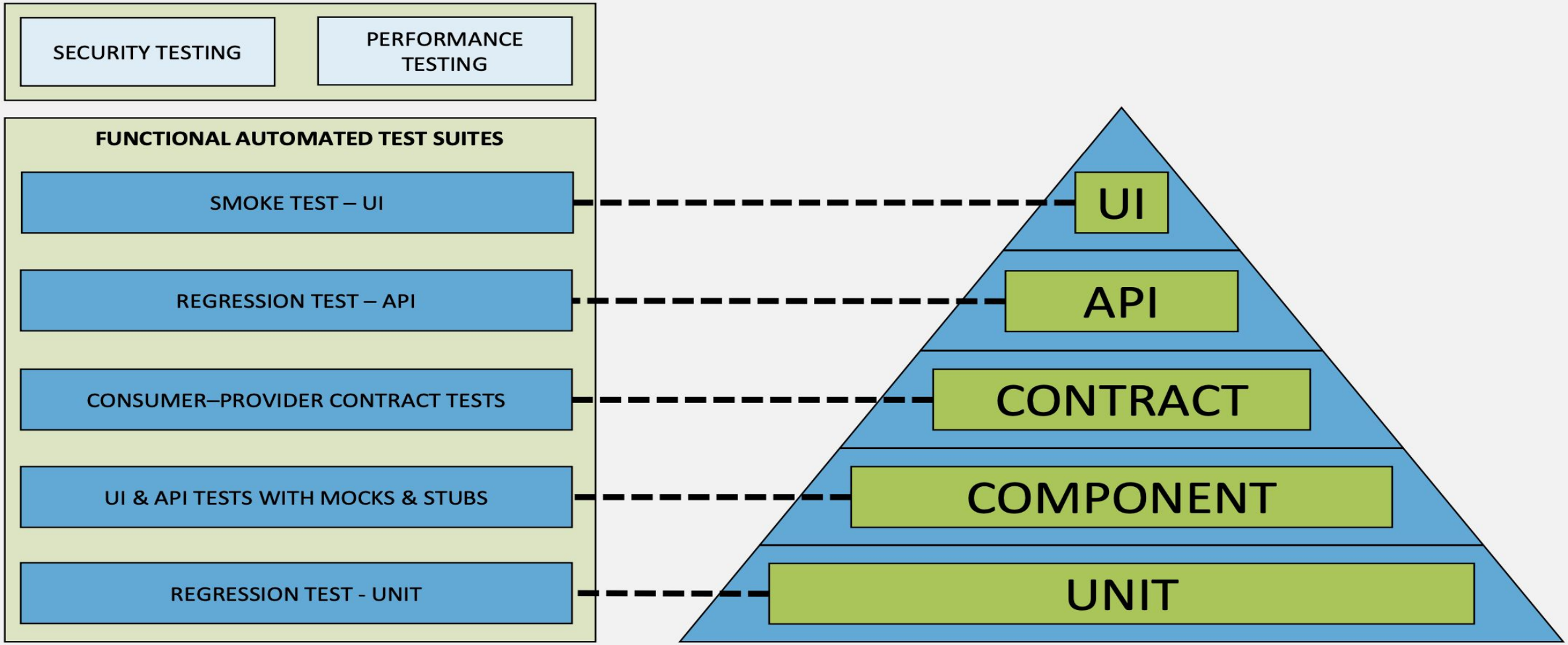
Contract
- It is a form of service integration testing
- Validates the contract between producers and consumers

Component
- Validates the components in isolation, leveraging test doubles
- Some call this simply as integration testing

Unit:
- No mocks/stubs/etc.
- Detroit/Classicist style



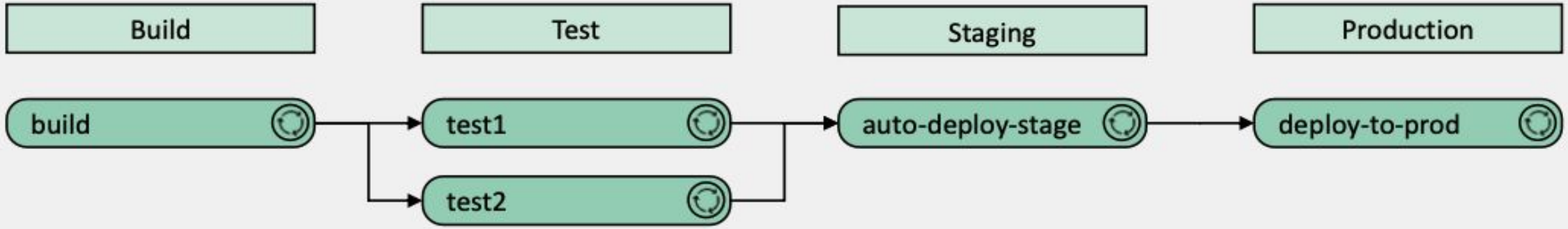
Agile Test Automation Pyramid



Pipeline Integration

What are Pipelines?

- A CI/CD pipeline is a combination of steps performed to deliver a new version of software.
- The primary goal is to get the code into production as early as possible.
- Automating the steps of a pipeline helps achieve this goal.



Quality & Integration to Pipelines

- **Slow manual testing is reduced or eliminated**
  - Dogfooding, Exploratory testing is getting more prominent
  - Focus on design, accessibility, user experience testing only

- **Robust UI automated test regression is replaced**
  - Replaced with smoke suites or domain driven tests
  - Testing is shifted at lower levels (unit, component, contract), which provides early and quick feedback on quality
  - If extensive automated end-to-end (E2E) UI testing is necessary, it is run as a separate job:
    - Either as a nightly build,
    - Or as a weekly/bi-weekly build.
    - Verification of these tests is indicated as a manual step in the pipeline.

- **Type of Tests during deployment to Staging/Pre-Prod/Release environments**
  - Component Tests
  - Contract Tests
  - API Regression Tests
  - UI Smoke Tests

Transitioning to SDET and/or Quality Consultant

Things to remember

What Works

Skill Diversification: Expand into automated testing and framework development.

Test Pyramids: Implement a layered testing approach.

Shift-Left Testing: Prioritize early lifecycle testing.

Pipeline Integration: Utilize CI/CD for process automation.

Soft Skills: Enhance communication and stakeholder management.

What Does Not Work

Siloed Approach: Isolate QA efforts from development, leading to poor integration and higher defect leakage.

Heavy Reliance on Manual Testing: Persist with manual tests in environments requiring rapid feedback and deployment.

Inconsistent Frameworks: Use multiple disparate frameworks hindering maintainability and scalability of test automation.

Thank You

udit-gattani-05900629

TOKYO TEST FEST

Q&
A

TOKYO TEST FEST