Kelvin Glaß
Software Engineering Working Group

Freie Universität Berlin

Why is Integration Testing Complex?

Integration Testing in Practice

Literature Research

Multiple definitions of „integration testing“

Nearly no qualitative research concerning integration testing

Mostly research about model based integration test generation

Non-scientific literature contains only a little information about integration testing

Initial Method: Unstructured Interviews

Topic: Testing

Number of Interviews: 7

Duration: 0.5h – 1.5h

Development Experience: 4 years – 30 years

Companies: Small (30 employees) – Big (10.000+ developers)

Common information:

- Whether tests exist
- Which test types are used
- Their used definition of integration test
- Who writes the integration tests
- Information concerning their integration test infrastructure

Rare information:

- How write/debug/maintain integration tests

Observations and New Questions

I. Only a small group of developers know how to write integration tests

→ Why is writing an integration test more complex than writing an unit test?

→ Complexity results from additional dependencies (compared to unit tests)?

- Complexity should be handled by modularization

→ Are the same quality standards applied for test code as for productive code?

- Is there a double standard?

II. The execution time of the tests is to long

→ Test to much with integration tests?

→ Is there a consideration which test type is used?

→ What should be tested via integration tests?

→ How many integration test are required to be confident?

→ Which tests are executed when?

III. Integration tests require a custom infrastructure

→ Do the developers know their tools?

→ Are existing frameworks used?

IV. Integration test are required to detect API breaking changes

- Common reason why integration tests fail

- increased by microservice architecture

- Complex if newest nightly releases of other services are consumed

- It is not clear which service should change something

V. There is no employee that is responsible for integration (testing)

Separate topic: **API change management is a challenge**

→ When should an API breaking change be published?

→ What are reasons to break an API?

→ Is it possible to avoid API breaking changes?

→ How to communicate API breaking changes? (Deprecation, Deprecation Alerts, …)