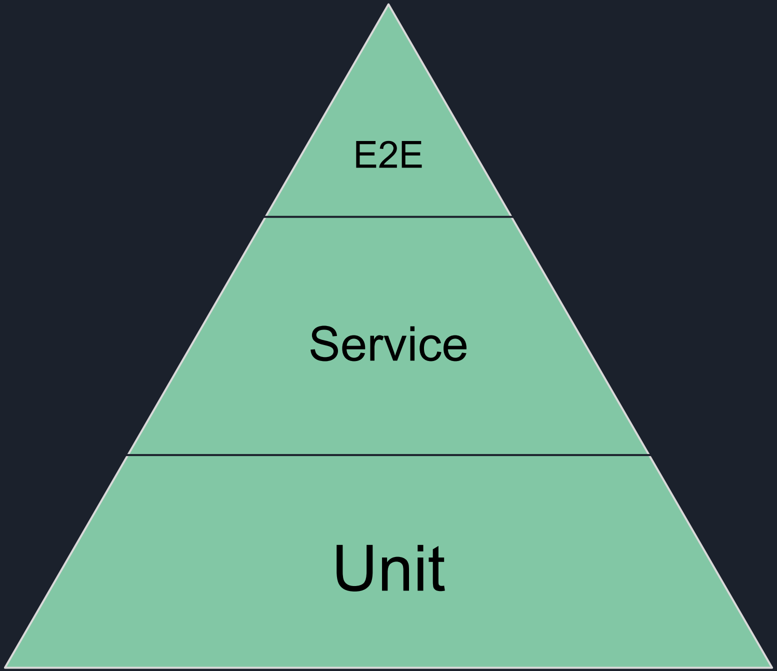
Mutation testing

Aleksandr Elmekeev

- Overview
- Terminology
- Problems
- Real Life Examples
- Summary

Overview

Test Pyramid



Test coverage

Criteria	JaCoCo	Istanbul
Function coverage	+	+
Statement coverage	+ (Instruction coverage)	+
Branch coverage	+	+
Modified condition/decision coverage	-	-
Linear Code Sequence and Jump (LCSAJ) coverage	-	-
Parameter value coverage	-	-

Quis custodiet ipsos custodes?

Who watches the watchmen?

Goals

- identify weakly tested pieces of code
- identify weak tests
- get rid of useless tests

Terminology

Mutation Operator (Mutator)

Type	Example: before	Example: after
Arithmetic	a + b	a - b
Array declaration	[1, 2, 3]	[]
Boolean	true	false
Conditional	for (var i = 0; i < 10; i++) { }	for (var i = 0; false; i++) { }
Equality	a < b	a <= b
Logical	a && b	a || b
Void	voidMethod();	// no voidMethod call

Mutant

By number of mutators:

- Simple (first order)
- Complex (high order)

By state:

- Killed
- Timeout
- Error
- Survived / Escaped
- Equivalent

Mutant: RIP

- A test must reach the mutated statement.
- Test input data should infect the program state by causing different program states for the mutant and the original program.
- The incorrect program state must propagate to the program's output and be checked by the test.

Mutation score = killed / total

Problems

High Computational Cost

- reduce number of mutants
  - **Mutant Sampling** — random subset of all mutants
  - **Selective Mutation** — certain types of mutators to generate mutants
  - **Mutant Clustering** — includes analysis of tests to identify subset
  - **Higher Order Mutation** — combines mutators (FOM × N = HOM) to make a single one with the same possibility to fail as a set of others
- optimize execution process
  - break the program by modules
  - Bytecode Translation technique
  - parallel runs
  - etc

Problems Related To Human Effort

- equivalent mutant problem
  - suggest (SEM)
  - detect (DEM)
  - avoid (AEMG)
- human oracle problem

Real Life Examples

Java

Typescript / Javascript (Angular)

Summary

Usage

When to use:

1. new tests to make sure the quality of them is good enough;
2. critical parts of software.

Be careful with:

1. file operations.

Useful Links

Read:

- Mutation testing on Wikipedia
- Mutation Testing Repository by Yue Jia and Mark Harman
- Analysis of Mutation Testing frameworks by scoban

Watch:

-