Cezary Sanecki

Testowanie aplikacji - jak robić to lepiej?

email: saneckicezary@gmail.com
instagram: @cezary.sanecki
blog: https://cezarysanecki.pl

Cezary Sanecki

```
****** ***************************** Top of Data ******************************
000001         IDENTIFICATION DIVISION.
000002         PROGRAM-ID. MARBJPMN.
000003         ENVIRONMENT DIVISION.
000004         CONFIGURATION SECTION.
000005         INPUT-OUTPUT SECTION.
000006         FILE-CONTROL.
000007             SELECT REPOUT
000008                    ASSIGN TO UT-S-SYSPRINT.
000009
000010         DATA DIVISION.
000011         FILE SECTION.
000012         FD  REPOUT
000013                 RECORD CONTAINS 127 CHARACTERS
000014                 LABEL RECORDS ARE OMITTED
000015                 DATA RECORD IS REPREC.
000016         01  REPREC                          PIC X(127).
 F1=Help       F2=Split      F3=Exit       F4=Expand     F5=Rfind      F6=Rchange
 F7=Up         F8=Down       F9=Swap      F10=Left      F11=Right     F12=Cancel
```

DECERTO

CEZARY SANECKI
PONAD KODEM

Warszawa
Java User Group

Testowanie aplikacji - jak robić to lepiej?

- **Therac-25 (naświetlenia)**

  "...W toku postępowania, rzecznik AECL przyznał, że wykonano "małą liczbę" testów urządzenia nim trafiło na rynek."

- **Ariane-5 (rakieta)**

  "... Nie było też systemu testów, ani nie przeprowadzono symulacji programowej."

- **Operacja Pustynna Burza**

  "...No właśnie nie – bo nikt nie testował systemu przez ponad 100 godzin. Nie było ku temu przesłanek."

Agenda

1

Wprowadzenie

2

Dlaczego nie piszemy testów?

3

Jak możemy to poprawić?

4

Podsumowanie i Q&A

Czym są testy jednostkowe?

“

kod wywołujący inny kod
w celu weryfikacji
czy działa on zgodnie
z wymaganiami biznesowymi

“

kod wykonujący inny kod
w kontrolowanych warunkach
w ramach jednego procesu w pamięci,
w celu weryfikacji
(bez ingerencji programisty),
że testowana logika działa w ściśle określony sposób

Maciej Aniserowicz

“

kod wykonujący inny kod
w kontrolowanych warunkach
w ramach jednego procesu w pamięci,
w celu weryfikacji
(bez ingerencji programisty),
że testowana logika działa w ściśle określony sposób

Maciej Aniserowicz

“

kod wykonujący inny kod
w kontrolowanych warunkach
w ramach jednego procesu w pamięci,
w celu weryfikacji
(bez ingerencji programisty),
że testowana logika działa w ściśle określony sposób

Maciej Aniserowicz

“

kod wykonujący inny kod
w kontrolowanych warunkach
w ramach jednego procesu w pamięci,
w celu weryfikacji
(bez ingerencji programisty),
że testowana logika działa w ściśle określony sposób

Maciej Aniserowicz

“

kod wykonujący inny kod
w kontrolowanych warunkach
w ramach jednego procesu w pamięci,
w celu weryfikacji
(bez ingerencji programisty),
że testowana logika działa w ściśle określony sposób

Maciej Aniserowicz

“

kod wykonujący inny kod
w kontrolowanych warunkach
w ramach jednego procesu w pamięci,
w celu weryfikacji
(bez ingerencji programisty),
że testowana logika działa w ściśle określony sposób

Maciej Aniserowicz

“

kod wykonujący inny kod
w kontrolowanych warunkach
w ramach jednego procesu w pamięci,
w celu weryfikacji
(bez ingerencji programisty),
że testowana logika działa w ściśle określony sposób

Maciej Aniserowicz

{i}DD

bezpieczeństwo

generowanie raportów dla biznesu

POPRAWA DESIGNU APLIKACJI

szybsze wprowadzenie nowych osób do projektu

mniej błędów = mniejsze koszty

natychmiastowy feedback

pewność

debug małych fragmentów kodu

... i pomimo tych zalet
spotyka się projekty,
gdzie się testów
nie pisze albo ignoruje

Bolączki testowe

Brak czytelności

"Zabetonowany" kod

Wolne działanie

Brak pewności co do działania aplikacji

Brak czytelności

"Zabetonowany" kod

Wolne działanie

Brak pewności co do działania aplikacji

```
@Test
void test() {
    VariantId variantId = VariantId.of(0L)
    authAsAgent(TestAgents.NEGOTIATOR.COKK_LEVEL_1.login)
    FleetId id = createFleet().id
    Regon regon = RandomUtils.randomRegon()
    createInsurer(id, CreateInsurersCommand.of(regon))
    updateInsurer(id, InsurerId.of(0), updateInsurerCommand())
    UpdateIntermediaryCommand updateIntermediaryCommand = UpdateIntermediaryCommand.builder()
            .type(IntermediaryType.BROKER)
            .identity("012345678901")
            .build()
    updateIntermediary(id, IntermediaryId.of(0L), updateIntermediaryCommand)
    editTariffGroup(id)
    TariffGroupId tariffGroupId = createTariffGroup(id).getId()
    updateTariffGroup(id, tariffGroupId, updateTariffGroupsCommand(tariffGroupId))
    acceptTariffGroup(id)
    createManuallyVehiclesWithSampleVin(id)
    selectInstallmentMethod(id, variantId, InstallmentMethodId.of(0L))
    updateVariant(id, UpdateVariantCommand.builder()
            .variantId(variantId)
            .title("WAR1")
            .build())
    updateNnw(id, variantId, VariantRangeId.of(0L), VariantRangeNNWDto.builder().insuranceSums(Set.of(BigDecimal.valueOf(10000L))).selected(true).build())
    updateCommission(id, variantId, IntermediaryCommissionsId.of(0L), UpdateIntermediaryCommissionsCommand.builder().defaultCommission(Commission.of('1')).build())
    recalculateVariant(id, variantId) // order and get quote
    AggregatedFleetResource fleet = getFleet(id, VARIANT)
    fleet.getDamage() == null
    fleet = getFleet(id, DAMAGE)
    def variantAndVariantVersionResource = getVariant(id, variantId)
    fleet.getDamage().ufgItems == [
    FleetDamageItemResource.builder()
            .year("2019")
            .oc(FleetDamageItemValuesResource.builder()
                    .vehiclesAmount(bigDecimalWithScaleOfFour("723.32"))
                    .damagesAmount(bigDecimalWithScaleOfFour("723.33"))
                    .paymentsAndProvisions(bigDecimalWithScaleOfFour("723.34"))
                    .build())
            .ac(FleetDamageItemValuesResource.builder()
                    .vehiclesAmount(bigDecimalWithScaleOfFour("823.32"))
                    .damagesAmount(bigDecimalWithScaleOfFour("823.33"))
                    .paymentsAndProvisions(bigDecimalWithScaleOfFour("823.34"))
                    .build())
            .build(),
            FleetDamageItemResource.builder()
                    .year("2020")
                    .oc(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("523.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("523.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("523.34"))
                            .build())
                    .ac(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("623.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("623.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("623.34"))
                            .build())
                    .build(),
            FleetDamageItemResource.builder()
                    .year("2021")
                    .oc(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("323.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("323.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("323.34"))
                            .build())
                    .ac(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("423.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("423.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("423.34"))
                            .build())
                    .build(),
            FleetDamageItemResource.builder()
                    .year("2022")
                    .oc(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("123.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("123.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("123.34"))
                            .build())
                    .ac(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("223.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("223.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("223.34"))
                            .build())
                    .build()
    ]
    fleet.getDamage().ufgDamageIndicator == null
    fleet.getDamage().zsiDamageIndicator == null
    fleet.getDamage().importedDamageIndicator == null
    fleet.getDamage().ocLossRatios.isEmpty()
    fleet.getDamage().acLossRatios.isEmpty()
    variantAndVariantVersionResource.damageLossRatio == DamageRatioResource.builder()
            .type("NTOP")
            .lossRatio(DamageLossRatioResource.builder()
                    .oc(DamageLossRatioItemResource.builder()
                            .premiumRisk(bigDecimalWithScaleOfFour("958.1233"))
                            .obligationRisk(bigDecimalWithScaleOfFour("961.1233"))
                            .lr(bigDecimalWithScaleOfFour("964.1233"))
                            .build())
                    .acKr(DamageLossRatioItemResource.builder()
                            .premiumRisk(bigDecimalWithScaleOfFour("957.1233"))
                            .obligationRisk(bigDecimalWithScaleOfFour("960.1233"))
                            .regressions(bigDecimalWithScaleOfFour("966.1233"))
                            .lr(bigDecimalWithScaleOfFour("963.1233"))
                            .build())
                    .ocAcKr(DamageLossRatioItemResource.builder()
                            .premiumRisk(bigDecimalWithScaleOfFour("959.1233"))
                            .obligationRisk(bigDecimalWithScaleOfFour("962.1233"))
                            .lr(bigDecimalWithScaleOfFour("965.1233"))
                            .build())
                    .build())
            .combinedRatio(DamageCombinedRatioResource.builder()
                    .oc(DamageCombinedRatioItemResource.builder()
                            .provision(bigDecimalWithScaleOfFour("968.1233"))
                            .combinedRatio(bigDecimalWithScaleOfFour("971.1233"))
                            .build())
                    .acKr(DamageCombinedRatioItemResource.builder()
                            .provision(bigDecimalWithScaleOfFour("967.1233"))
                            .combinedRatio(bigDecimalWithScaleOfFour("970.1233"))
                            .build())
                    .ocAcKr(DamageCombinedRatioItemResource.builder()
                            .provision(bigDecimalWithScaleOfFour("969.1233"))
                            .combinedRatio(bigDecimalWithScaleOfFour("972.1233"))
                            .build())
                    .build())
            .build()
}
```

```
  // ...

  @Test
  void test1() {
    // ...
  }

  @Test
  void test2() {
    // ...
  }
}
```

```java
void test() {
  Employee firstEmployee = Employee.of(
      "Jan", "Kowalski", "Programmer", Money.ofPLN(98_000.00));
  Employee secondEmployee = Employee.of(
      "Kacper", "Nowak", "Tester", Money.ofPLN(123_978.00));
  Employee thirdEmployee = Employee.of(
      "Krystian", "Kot", "Account Manager", Money.ofPLN(1_500_900.37));
  Department itDepartment = Department.builder()
      .name("IT")
      .director(Person.of("Pawel", "Mistrz"))
      .employees(List.of(firstEmployee, secondEmployee))
      .build();
  Department accountDepartment = Department.builder()
      .name("Account")
      .director(Person.of("Justyna", "Niewiadomska"))
      .employees(List.of(thirdEmployee))
      .build();
  Company company = Company.builder()
      .name("JakoTako IT Sp. z o.o.")
      .nip("8386210452")
      .regon("292875580")
      .owner(Person.of("Grażyna", "Bułka"))
      .departments(List.of(itDepartment, accountDepartment))
      .build();

  Money result = expenseCalculator.calculateCostsFor(company);

  assertThat(result).isEqualTo(Money.ofPLN(1_722_878.37));
}
```

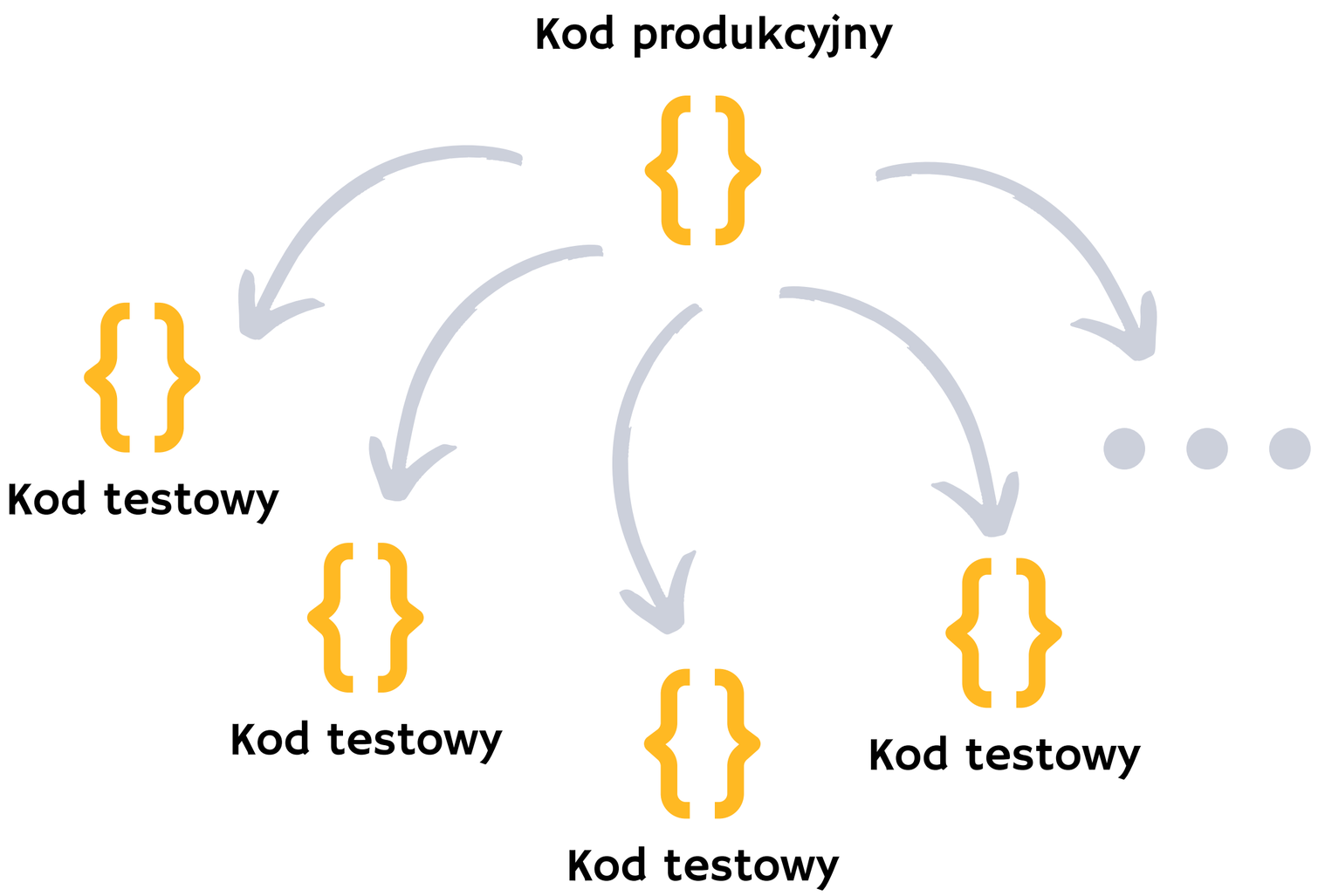
```java
@Test
void test() {
  // ... given section

  Report result = companyReportMaker.makeReportFor(company);

  assertThat(result.getAllDepartments().size()).isEqualTo(2);
  assertThat(result.getAllEmployees().size()).isEqualTo(3);
  BigDecimal salariesSum = result.getAllEmployees()
      .stream()
      .map(Employee::getSalary)
      .map(Money::getValue)
      .reduce(BigDecimal.ZERO, BigDecimal::add);
  assertThat(salariesSum.divide(
      BigDecimal.valueOf(result.getAllDepartments().size()),
      RoundingMode.HALF_UP)).isEqualTo(BigDecimal.valueOf(2));
  assertThat(result.getAllDepartments()
      .stream()
      .anyMatch(department -> department.getName().equals("IT"))).isTrue();
}
```

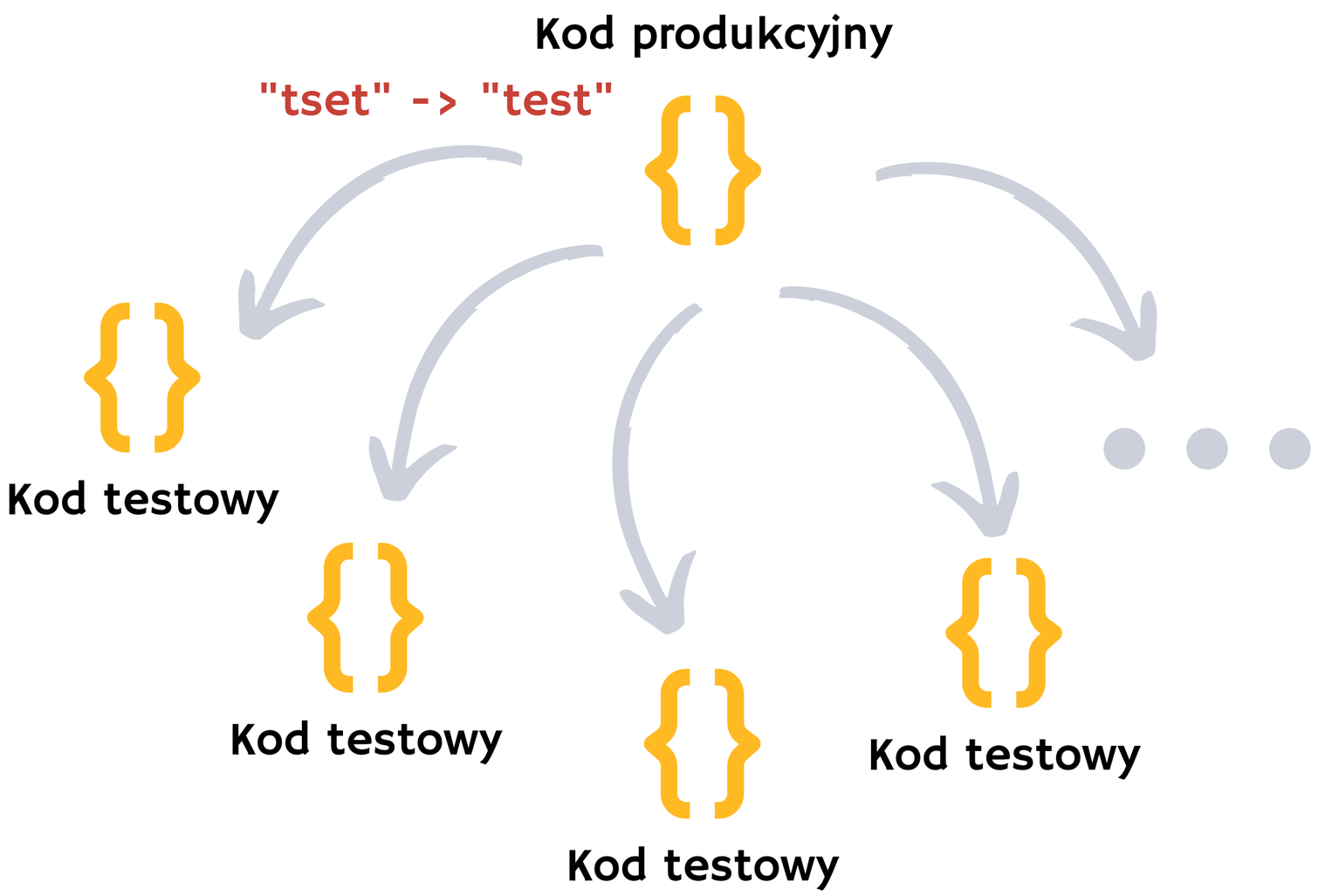
Brak czytelności

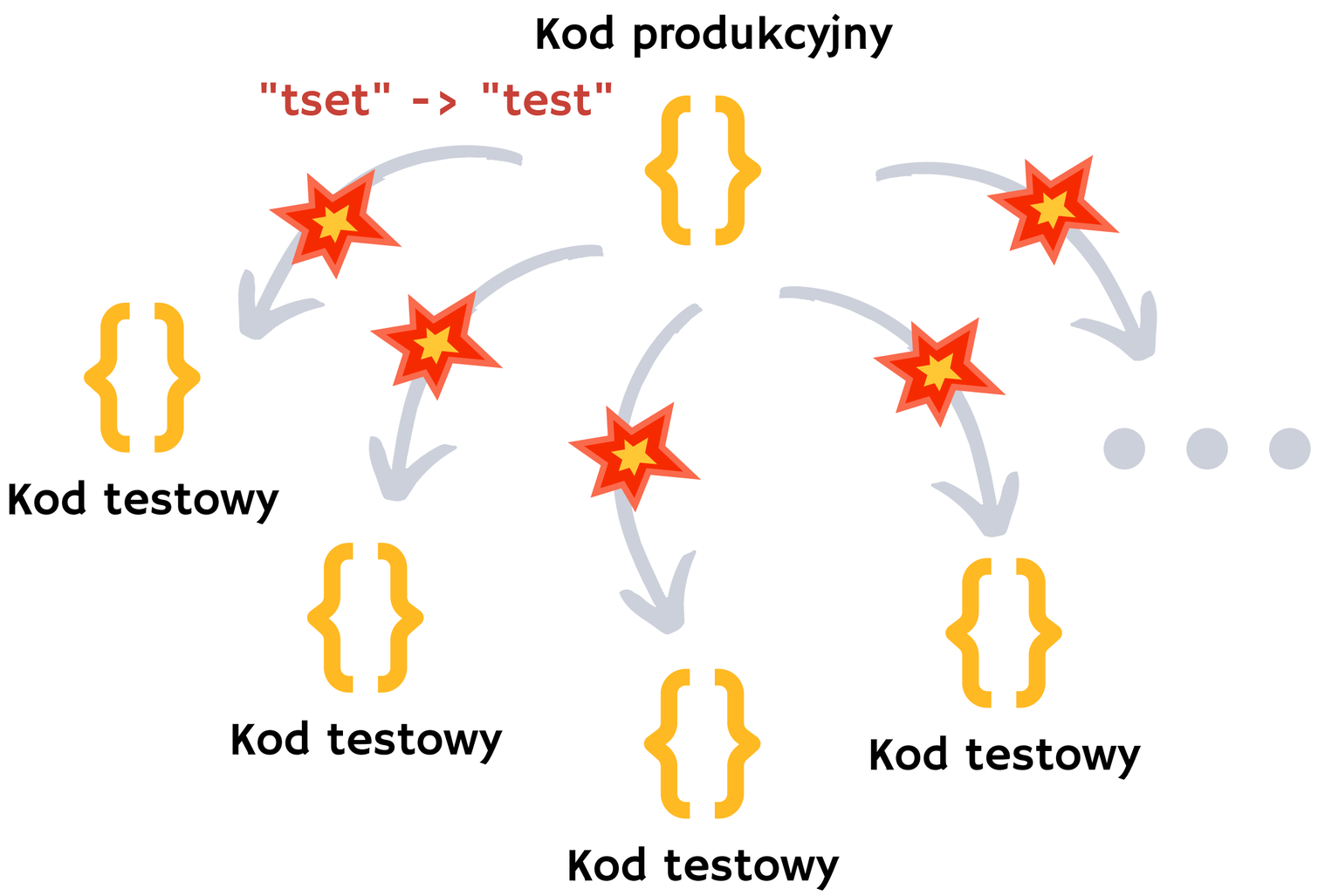
"Zabetonowany" kod

Wolne działanie

Brak pewności co do działania aplikacji


Kod produkcyjny
Kod testowy
Kod testowy
Kod testowy
Kod testowy


Kod produkcyjny
"tset" -> "test"
Kod testowy
Kod testowy
Kod testowy
Kod testowy


Kod produkcyjny
"tset" -> "test"
Kod testowy
Kod testowy
Kod testowy
Kod testowy

Brak czytelności

"Zabetonowany" kod

Wolne działanie

Brak pewności co do działania aplikacji

1.0s	tracimy flow
10.0s	tracimy uwagę
30.0s	testy tylko na push

Usability Engineering, Jakob Nielse, 1993
Modularity and hexagonal architecture in real life: Jakub Nabrdalik (https://www.youtube.com/watch?v=ILBX9fa9aJo)

Brak czytelności

"Zabetonowany" kod

Wolne działanie

Brak pewności co do działania aplikacji

“

jeszcze tylko przeklikam
u siebie...
tak dla pewności...

anonimowy deweloper

Jak robić to lepiej?

Czym są dobre testy jednostkowe?

F I R S T

F I R S T

Fast

F I R S T

Isolated

F I R S T

Repeatable

F I R S T

Self-veryfing

F I R S T

Timely

```
@Test
void test() {
    VariantId variantId = VariantId.of(0L)
    authAsAgent(TestAgents.NEGOTIATOR.COKK_LEVEL_1.login)
    FleetId id = createFleet().id
    Regon regon = RandomUtils.randomRegon()
    createInsurer(id, CreateInsurersCommand.of(regon))
    updateInsurer(id, InsurerId.of(0), updateInsurerCommand())
    UpdateIntermediaryCommand updateIntermediaryCommand = UpdateIntermediaryCommand.builder()
            .type(IntermediaryType.BROKER)
            .identity("012345678901")
            .build()
    updateIntermediary(id, IntermediaryId.of(0L), updateIntermediaryCommand)
    editTariffGroup(id)
    TariffGroupId tariffGroupId = createTariffGroup(id).getId()
    updateTariffGroup(id, tariffGroupId, updateTariffGroupsCommand(tariffGroupId))
    acceptTariffGroup(id)
    createManuallyVehiclesWithSampleVin(id)
    selectInstallmentMethod(id, variantId, InstallmentMethodId.of(0L))
    updateVariant(id, UpdateVariantCommand.builder()
            .variantId(variantId)
            .title("WAR1")
            .build())
    updateNnw(id, variantId, VariantRangeId.of(0L), VariantRangeNNWDto.builder().insuranceSums(Set.of(BigDecimal.valueOf(10000L))).selected(true).build())
    updateCommission(id, variantId, IntermediaryCommissionsId.of(0L), UpdateIntermediaryCommissionsCommand.builder().defaultCommission(Commission.of('1')).build())
    recalculateVariant(id, variantId) // order and get quote
    AggregatedFleetResource fleet = getFleet(id, VARIANT)
    fleet.getDamage() == null
    fleet = getFleet(id, DAMAGE)
    def variantAndVariantVersionResource = getVariant(id, variantId)
    fleet.getDamage().ufgItems == [
    FleetDamageItemResource.builder()
            .year("2019")
            .oc(FleetDamageItemValuesResource.builder()
                    .vehiclesAmount(bigDecimalWithScaleOfFour("723.32"))
                    .damagesAmount(bigDecimalWithScaleOfFour("723.33"))
                    .paymentsAndProvisions(bigDecimalWithScaleOfFour("723.34"))
                    .build())
            .ac(FleetDamageItemValuesResource.builder()
                    .vehiclesAmount(bigDecimalWithScaleOfFour("823.32"))
                    .damagesAmount(bigDecimalWithScaleOfFour("823.33"))
                    .paymentsAndProvisions(bigDecimalWithScaleOfFour("823.34"))
                    .build())
            .build(),
            FleetDamageItemResource.builder()
                    .year("2020")
                    .oc(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("523.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("523.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("523.34"))
                            .build())
                    .ac(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("623.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("623.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("623.34"))
                            .build())
                    .build(),
            FleetDamageItemResource.builder()
                    .year("2021")
                    .oc(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("323.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("323.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("323.34"))
                            .build())
                    .ac(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("423.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("423.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("423.34"))
                            .build())
                    .build(),
            FleetDamageItemResource.builder()
                    .year("2022")
                    .oc(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("123.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("123.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("123.34"))
                            .build())
                    .ac(FleetDamageItemValuesResource.builder()
                            .vehiclesAmount(bigDecimalWithScaleOfFour("223.32"))
                            .damagesAmount(bigDecimalWithScaleOfFour("223.33"))
                            .paymentsAndProvisions(bigDecimalWithScaleOfFour("223.34"))
                            .build())
                    .build()
    ]
    fleet.getDamage().ufgDamageIndicator == null
    fleet.getDamage().zsiDamageIndicator == null
    fleet.getDamage().importedDamageIndicator == null
    fleet.getDamage().ocLossRatios.isEmpty()
    fleet.getDamage().acLossRatios.isEmpty()
    variantAndVariantVersionResource.damageLossRatio == DamageRatioResource.builder()
            .type("NTOP")
            .lossRatio(DamageLossRatioResource.builder()
                    .oc(DamageLossRatioItemResource.builder()
                            .premiumRisk(bigDecimalWithScaleOfFour("958.1233"))
                            .obligationRisk(bigDecimalWithScaleOfFour("961.1233"))
                            .lr(bigDecimalWithScaleOfFour("964.1233"))
                            .build())
                    .acKr(DamageLossRatioItemResource.builder()
                            .premiumRisk(bigDecimalWithScaleOfFour("957.1233"))
                            .obligationRisk(bigDecimalWithScaleOfFour("960.1233"))
                            .regressions(bigDecimalWithScaleOfFour("966.1233"))
                            .lr(bigDecimalWithScaleOfFour("963.1233"))
                            .build())
                    .ocAcKr(DamageLossRatioItemResource.builder()
                            .premiumRisk(bigDecimalWithScaleOfFour("959.1233"))
                            .obligationRisk(bigDecimalWithScaleOfFour("962.1233"))
                            .lr(bigDecimalWithScaleOfFour("965.1233"))
                            .build())
                    .build())
            .combinedRatio(DamageCombinedRatioResource.builder()
                    .oc(DamageCombinedRatioItemResource.builder()
                            .provision(bigDecimalWithScaleOfFour("968.1233"))
                            .combinedRatio(bigDecimalWithScaleOfFour("971.1233"))
                            .build())
                    .acKr(DamageCombinedRatioItemResource.builder()
                            .provision(bigDecimalWithScaleOfFour("967.1233"))
                            .combinedRatio(bigDecimalWithScaleOfFour("970.1233"))
                            .build())
                    .ocAcKr(DamageCombinedRatioItemResource.builder()
                            .provision(bigDecimalWithScaleOfFour("969.1233"))
                            .combinedRatio(bigDecimalWithScaleOfFour("972.1233"))
                            .build())
                    .build())
            .build()
}
```

“

It was hard to write,
must be hard to read

anonimowy deweloper

“

Programs must be written for people to read, and only incidentally for machines to execute

Harold Abelson

rzadziej piszemy kod
niż go czytamy

- konwencje nazewnicze
- chowanie złożoności sekcji given
- rozbicie testu na kilka mniejszych
- skorzystanie z Assert Object Pattern
- ...

Pisz czytelne testy

Pisanie testów tylko po to, aby je mieć to nienajlepszy pomysł

Code coverage

0% < x% < 100%

ŹLE ŹLE

- błędy
- nowe funkcje
- refactoring

Nie "betonuj" kodu testami

Szkoła Klasyczna vs Szkoła Londyńska

- Ograniczone użycie mocków
- Jednostka to klasa albo zbiór klas
- Rozbicie ze względu na czytelność kodu

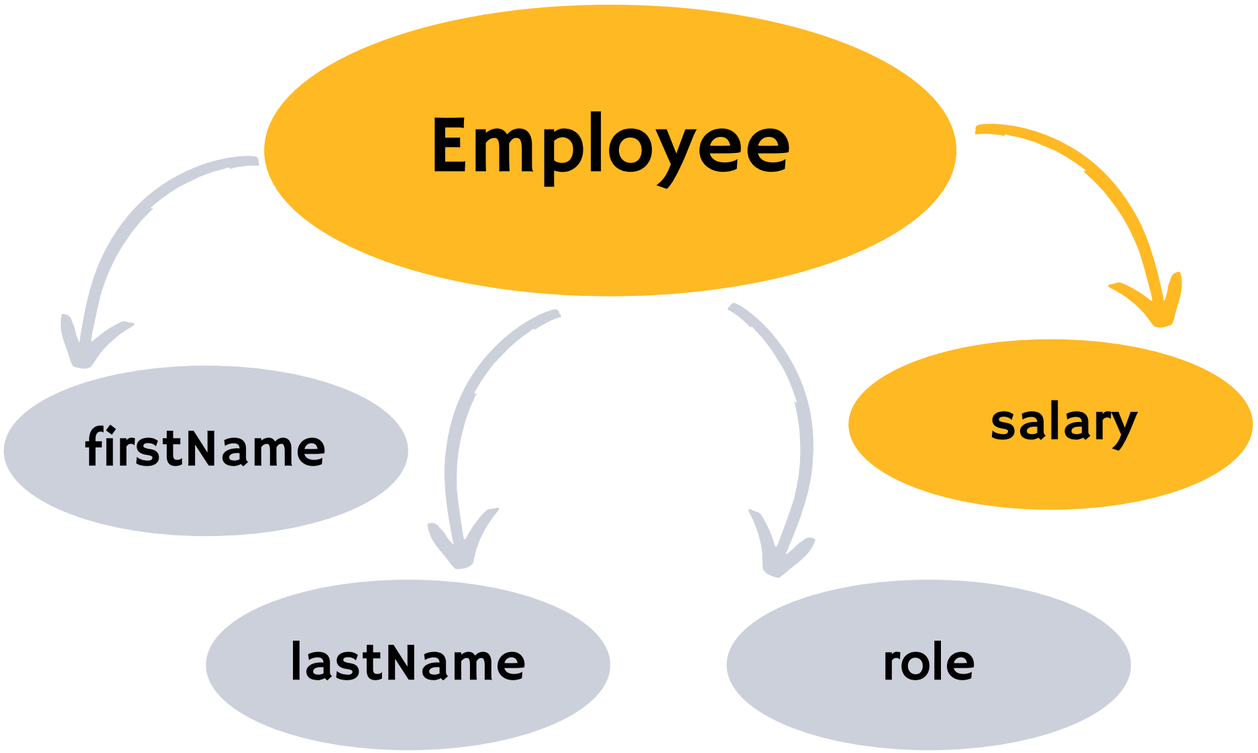
- Mockowanie wszystkich zależności
- Klasa jest jednostką
- Rozbicie ze względu na różne odpowiedzialności

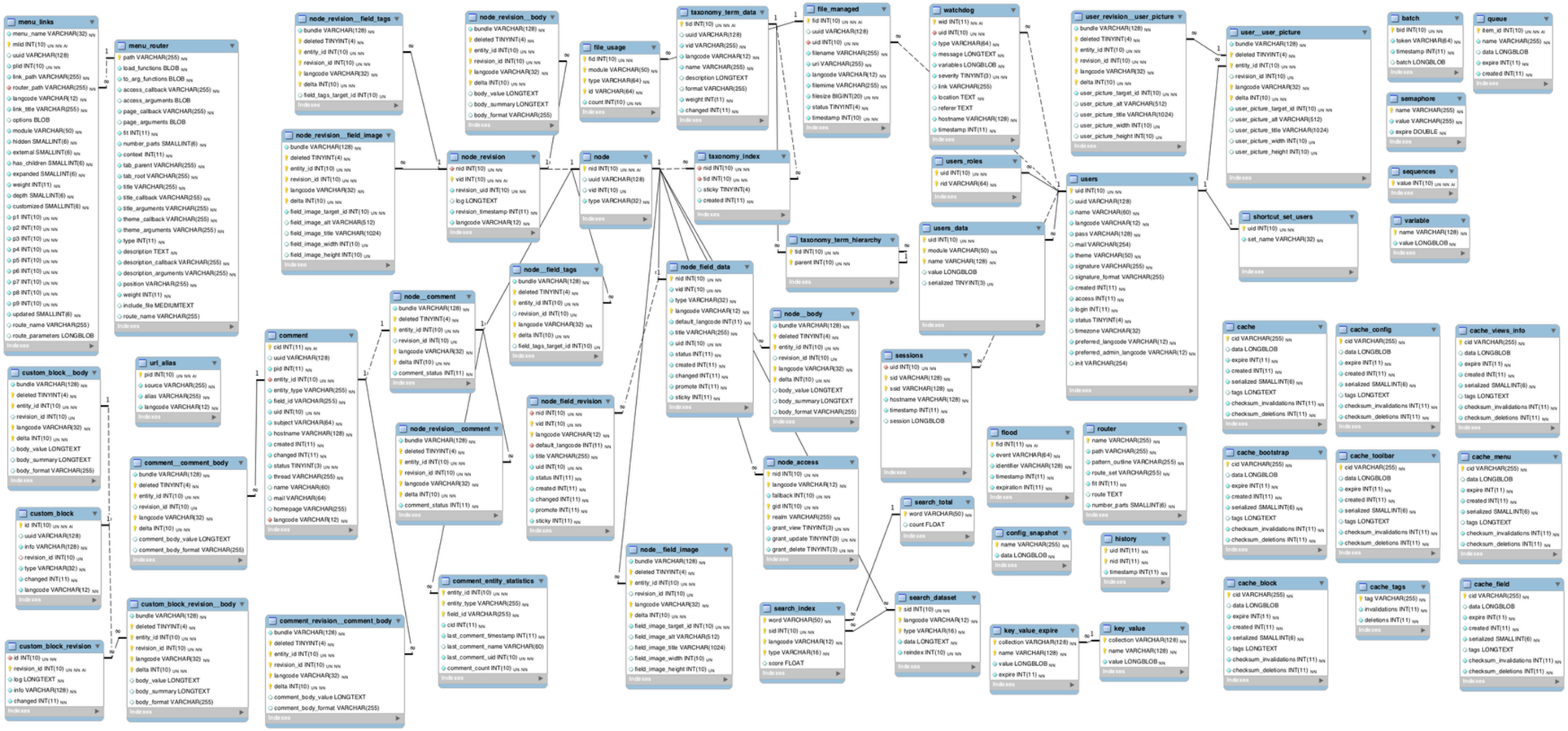
“

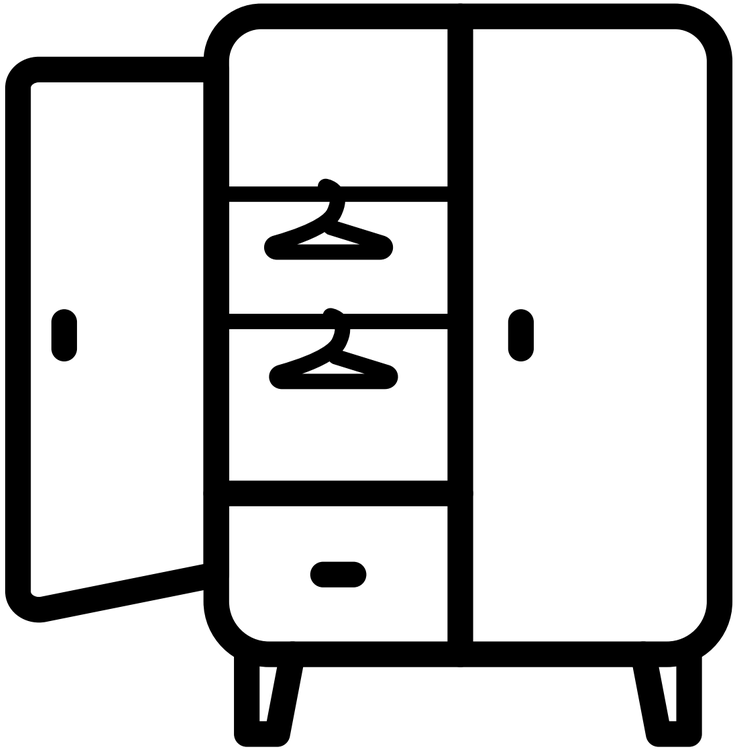
czy ja w ogóle testuję
kod produkcyjny
czy tylko
bibliotekę do mocków?

Dobierz odpowiednie podejście do pisania testów

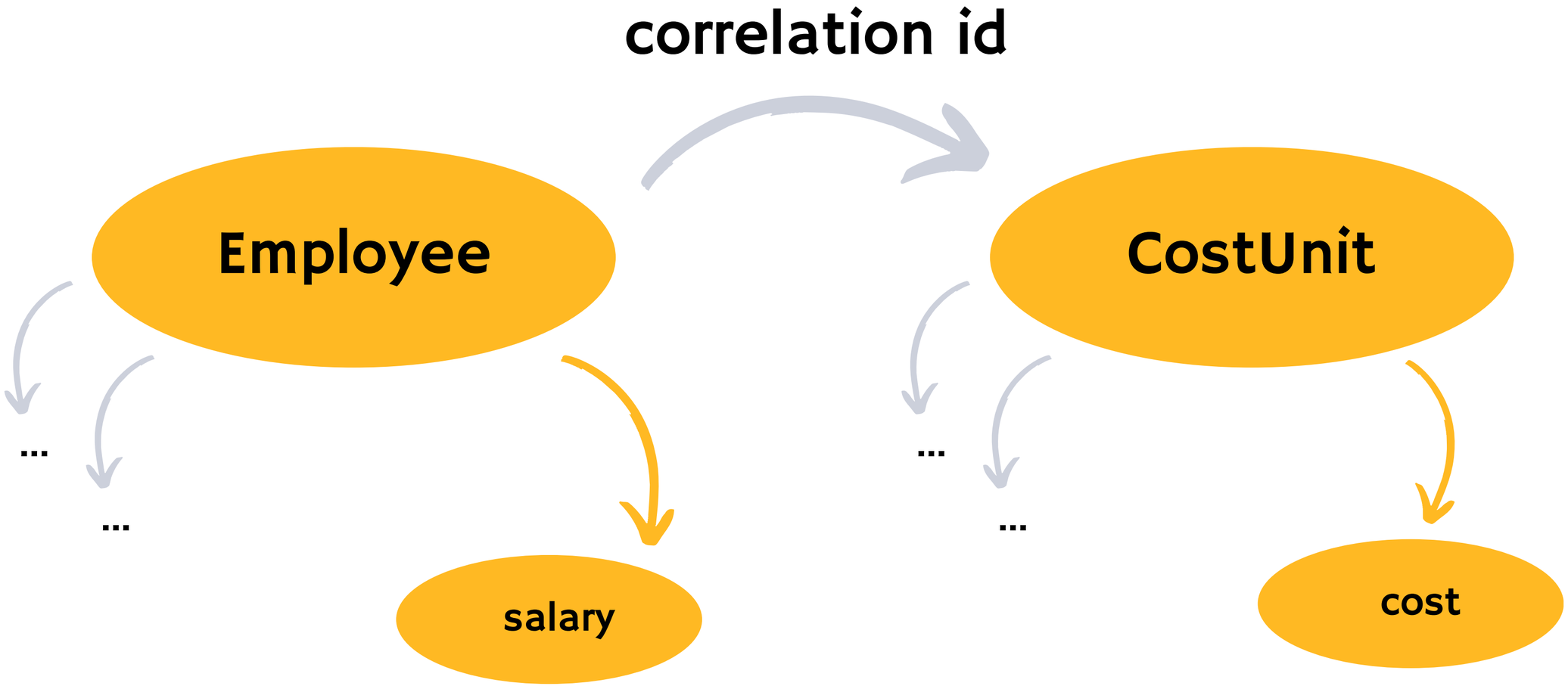
```java
void test() {
  Employee firstEmployee = Employee.of(
      "Jan", "Kowalski", "Programmer", Money.ofPLN(98_000.00));
  Employee secondEmployee = Employee.of(
      "Kacper", "Nowak", "Tester", Money.ofPLN(123_978.00));
  Employee thirdEmployee = Employee.of(
      "Krystian", "Kot", "Account Manager", Money.ofPLN(1_500_900.37));
  Department itDepartment = Department.builder()
      .name("IT")
      .director(Person.of("Pawel", "Mistrz"))
      .employees(List.of(firstEmployee, secondEmployee))
      .build();
  Department accountDepartment = Department.builder()
      .name("Account")
      .director(Person.of("Justyna", "Niewiadomska"))
      .employees(List.of(thirdEmployee))
      .build();
  Company company = Company.builder()
      .name("JakoTako IT Sp. z o.o.")
      .nip("8386210452")
      .regon("292875580")
      .owner(Person.of("Grażyna", "Bułka"))
      .departments(List.of(itDepartment, accountDepartment))
      .build();

  Money result = expenseCalculator.calculateCostsFor(company);

  assertThat(result).isEqualTo(Money.ofPLN(1_722_878.37));
}
```

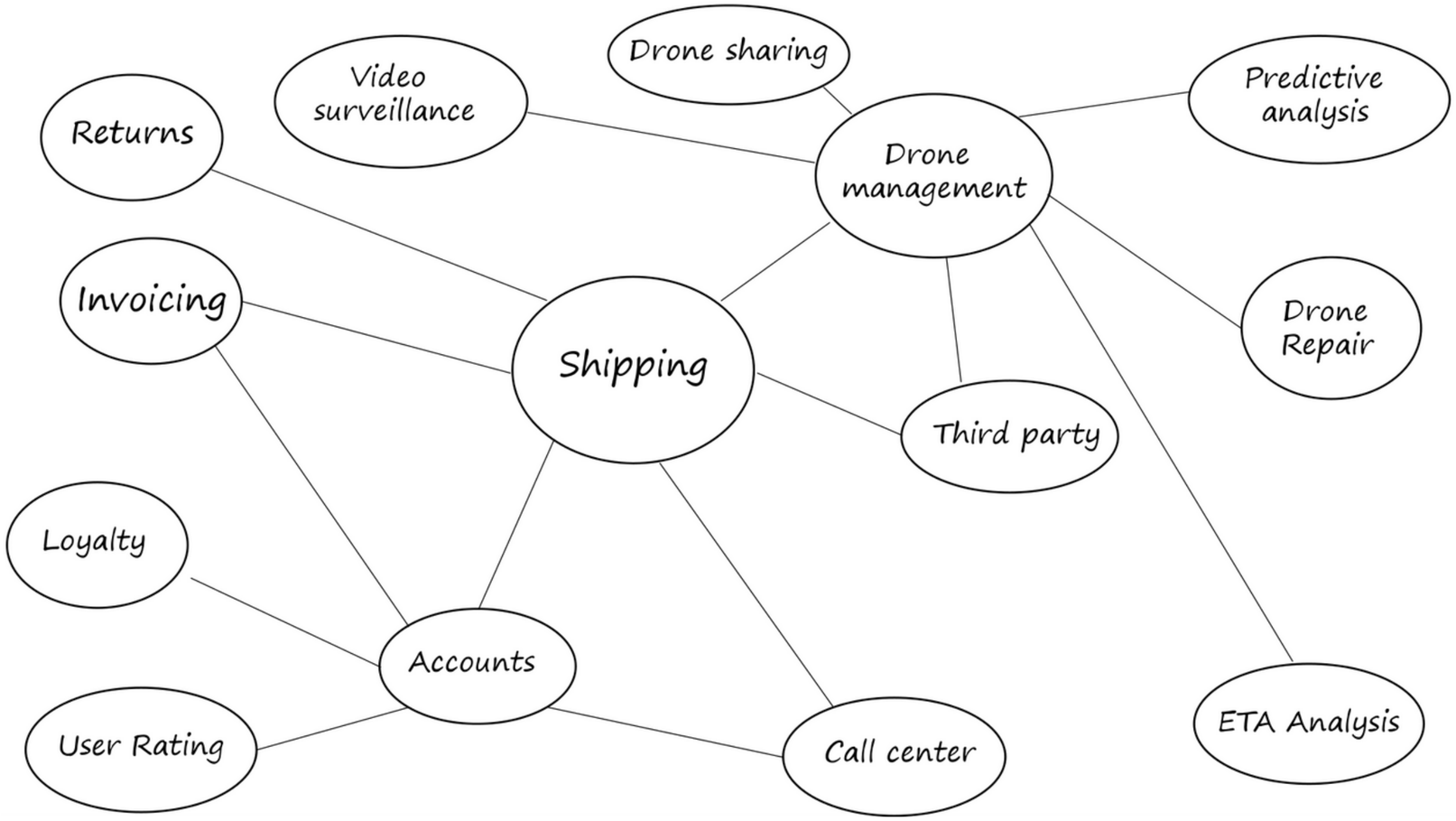



menu_links
menu_name VARCHAR(32)
mlid INT(10)
uuid VARCHAR(128)
plid INT(10)
link_path VARCHAR(255)
router_path VARCHAR(255)
langcode VARCHAR(12)
link_title VARCHAR(255)
options BLOB
module VARCHAR(50)
hidden SMALLINT(6)
external SMALLINT(6)
has_children SMALLINT(6)
expanded SMALLINT(6)
weight INT(11)
depth SMALLINT(6)
customized SMALLINT(6)
p1 INT(10)
p2 INT(10)
p3 INT(10)
p4 INT(10)
p5 INT(10)
p6 INT(10)
p7 INT(10)
p8 INT(10)
p9 INT(10)
updated SMALLINT(6)
route_name VARCHAR(255)
route_parameters LONGBLOB
menu_router
path VARCHAR(255)
load_functions BLOB
to_arg_functions BLOB
access_callback VARCHAR(255)
access_arguments BLOB
page_callback VARCHAR(255)
page_arguments BLOB
fit INT(11)
number_parts SMALLINT(6)
context INT(11)
tab_parent VARCHAR(255)
tab_root VARCHAR(255)
title VARCHAR(255)
title_callback VARCHAR(255)
title_arguments VARCHAR(255)
theme_callback VARCHAR(255)
theme_arguments VARCHAR(255)
type INT(11)
description TEXT
description_callback VARCHAR(255)
description_arguments VARCHAR(255)
position VARCHAR(255)
weight INT(11)
include_file MEDIUMTEXT
route_name VARCHAR(255)
node_revision__field_tags
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
field_tags_target_id INT(10)
node_revision__body
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
body_value LONGTEXT
body_summary LONGTEXT
body_format VARCHAR(255)
file_usage
fid INT(10)
module VARCHAR(50)
type VARCHAR(64)
id VARCHAR(64)
count INT(10)
taxonomy_term_data
tid INT(10)
uuid VARCHAR(128)
vid VARCHAR(255)
langcode VARCHAR(12)
name VARCHAR(255)
description LONGTEXT
format VARCHAR(255)
weight INT(11)
changed INT(11)
file_managed
fid INT(10)
uuid VARCHAR(128)
uid INT(10)
filename VARCHAR(255)
uri VARCHAR(255)
langcode VARCHAR(12)
filemime VARCHAR(255)
filesize BIGINT(20)
status TINYINT(4)
timestamp INT(10)
watchdog
wid INT(11)
uid INT(10)
type VARCHAR(64)
message LONGTEXT
variables LONGBLOB
severity TINYINT(3)
link VARCHAR(255)
location TEXT
referer TEXT
hostname VARCHAR(128)
timestamp INT(11)
users_roles
uid INT(10)
rid VARCHAR(64)
user_revision__user_picture
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
user_picture_target_id INT(10)
user_picture_alt VARCHAR(512)
user_picture_title VARCHAR(1024)
user_picture_width INT(10)
user_picture_height INT(10)
user__user_picture
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
user_picture_target_id INT(10)
user_picture_alt VARCHAR(512)
user_picture_title VARCHAR(1024)
user_picture_width INT(10)
user_picture_height INT(10)
batch
bid INT(10)
token VARCHAR(64)
timestamp INT(11)
batch LONGBLOB
queue
item_id INT(10)
name VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
semaphore
name VARCHAR(255)
value VARCHAR(255)
expire DOUBLE
sequences
value INT(10)
variable
name VARCHAR(128)
value LONGBLOB
node_revision__field_image
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
field_image_target_id INT(10)
field_image_alt VARCHAR(512)
field_image_title VARCHAR(1024)
field_image_width INT(10)
field_image_height INT(10)
node_revision
nid INT(10)
vid INT(10)
revision_uid INT(10)
log LONGTEXT
revision_timestamp INT(11)
langcode VARCHAR(12)
node
nid INT(10)
uuid VARCHAR(128)
vid INT(10)
type VARCHAR(32)
taxonomy_index
nid INT(10)
tid INT(10)
sticky TINYINT(4)
created INT(11)
taxonomy_term_hierarchy
tid INT(10)
parent INT(10)
users_data
uid INT(10)
module VARCHAR(50)
name VARCHAR(128)
value LONGBLOB
serialized TINYINT(3)
users
uid INT(10)
uuid VARCHAR(128)
name VARCHAR(60)
langcode VARCHAR(12)
pass VARCHAR(128)
mail VARCHAR(254)
theme VARCHAR(50)
signature VARCHAR(255)
signature_format VARCHAR(255)
created INT(11)
access INT(11)
login INT(11)
status TINYINT(4)
timezone VARCHAR(32)
preferred_langcode VARCHAR(12)
preferred_admin_langcode VARCHAR(12)
init VARCHAR(254)
shortcut_set_users
uid INT(10)
set_name VARCHAR(32)
node__field_tags
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
field_tags_target_id INT(10)
node__comment
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
comment_status INT(11)
node_field_data
nid INT(10)
vid INT(10)
type VARCHAR(32)
langcode VARCHAR(12)
default_langcode INT(11)
title VARCHAR(255)
uid INT(10)
status INT(11)
created INT(11)
changed INT(11)
promote INT(11)
sticky INT(11)
node__body
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
body_value LONGTEXT
body_summary LONGTEXT
body_format VARCHAR(255)
sessions
uid INT(10)
sid VARCHAR(128)
ssid VARCHAR(128)
hostname VARCHAR(128)
timestamp INT(11)
session LONGBLOB
cache
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
serialized SMALLINT(6)
tags LONGTEXT
checksum_invalidations INT(11)
checksum_deletions INT(11)
cache_config
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
serialized SMALLINT(6)
tags LONGTEXT
checksum_invalidations INT(11)
checksum_deletions INT(11)
cache_views_info
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
serialized SMALLINT(6)
tags LONGTEXT
checksum_invalidations INT(11)
checksum_deletions INT(11)
custom_block__body
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
body_value LONGTEXT
body_summary LONGTEXT
body_format VARCHAR(255)
url_alias
pid INT(10)
source VARCHAR(255)
alias VARCHAR(255)
langcode VARCHAR(12)
comment
cid INT(11)
uuid VARCHAR(128)
pid INT(11)
entity_id INT(10)
entity_type VARCHAR(255)
field_id VARCHAR(255)
uid INT(10)
subject VARCHAR(64)
hostname VARCHAR(128)
created INT(11)
changed INT(11)
status TINYINT(3)
thread VARCHAR(255)
name VARCHAR(60)
mail VARCHAR(64)
homepage VARCHAR(255)
langcode VARCHAR(12)
node_revision__comment
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
comment_status INT(11)
node_field_revision
nid INT(10)
vid INT(10)
langcode VARCHAR(12)
default_langcode INT(11)
title VARCHAR(255)
uid INT(10)
status INT(11)
created INT(11)
changed INT(11)
promote INT(11)
sticky INT(11)
comment__comment_body
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
comment_body_value LONGTEXT
comment_body_format VARCHAR(255)
custom_block
id INT(10)
uuid VARCHAR(128)
info VARCHAR(128)
revision_id INT(10)
type VARCHAR(32)
changed INT(11)
langcode VARCHAR(12)
custom_block_revision__body
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
body_value LONGTEXT
body_summary LONGTEXT
body_format VARCHAR(255)
comment_revision__comment_body
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
comment_body_value LONGTEXT
comment_body_format VARCHAR(255)
comment_entity_statistics
entity_id INT(10)
entity_type VARCHAR(255)
field_id VARCHAR(255)
cid INT(11)
last_comment_timestamp INT(11)
last_comment_name VARCHAR(60)
last_comment_uid INT(10)
comment_count INT(10)
custom_block_revision
id INT(10)
revision_id INT(10)
log LONGTEXT
info VARCHAR(128)
changed INT(11)
node_field_image
bundle VARCHAR(128)
deleted TINYINT(4)
entity_id INT(10)
revision_id INT(10)
langcode VARCHAR(32)
delta INT(10)
field_image_target_id INT(10)
field_image_alt VARCHAR(512)
field_image_title VARCHAR(1024)
field_image_width INT(10)
field_image_height INT(10)
node_access
nid INT(10)
langcode VARCHAR(12)
fallback INT(10)
gid INT(10)
realm VARCHAR(255)
grant_view TINYINT(3)
grant_update TINYINT(3)
grant_delete TINYINT(3)
search_index
word VARCHAR(50)
sid INT(10)
langcode VARCHAR(12)
type VARCHAR(16)
score FLOAT
search_total
word VARCHAR(50)
count FLOAT
search_dataset
sid INT(10)
langcode VARCHAR(12)
type VARCHAR(16)
data LONGTEXT
reindex INT(10)
flood
fid INT(11)
event VARCHAR(64)
identifier VARCHAR(128)
timestamp INT(11)
expiration INT(11)
router
name VARCHAR(255)
path VARCHAR(255)
pattern_outline VARCHAR(255)
route_set VARCHAR(253)
fit INT(11)
route TEXT
number_parts SMALLINT(6)
config_snapshot
name VARCHAR(255)
data LONGBLOB
history
uid INT(11)
nid INT(11)
timestamp INT(11)
key_value_expire
collection VARCHAR(128)
name VARCHAR(128)
value LONGBLOB
expire INT(11)
key_value
collection VARCHAR(128)
name VARCHAR(128)
value LONGBLOB
cache_bootstrap
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
serialized SMALLINT(6)
tags LONGTEXT
checksum_invalidations INT(11)
checksum_deletions INT(11)
cache_toolbar
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
serialized SMALLINT(6)
tags LONGTEXT
checksum_invalidations INT(11)
checksum_deletions INT(11)
cache_menu
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
serialized SMALLINT(6)
tags LONGTEXT
checksum_invalidations INT(11)
checksum_deletions INT(11)
cache_block
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
serialized SMALLINT(6)
tags LONGTEXT
checksum_invalidations INT(11)
checksum_deletions INT(11)
cache_tags
tag VARCHAR(255)
invalidations INT(11)
deletions INT(11)
cache_field
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
serialized SMALLINT(6)
tags LONGTEXT
checksum_invalidations INT(11)
checksum_deletions INT(11)



```
{
  "producer": "IKEA",
  "warrantyMonths": 24,
  "dimensionsCm: "180x100x60",
  "weightKg": 200,
  ...
}
```


correlation id
Employee
...
...
salary
CostUnit
...
...
cost


Drone sharing
Video surveillance
Returns
Predictive analysis
Drone management
Invoicing
Shipping
Drone Repair
Third party
Loyalty
Accounts
User Rating
Call center
ETA Analysis

Poświęć więcej czasu na analizę

PROGRAMISTYCZNY DEADLOCK

trzeba to zrefaktoryzować

nie mamy testów

dopiszmy testy

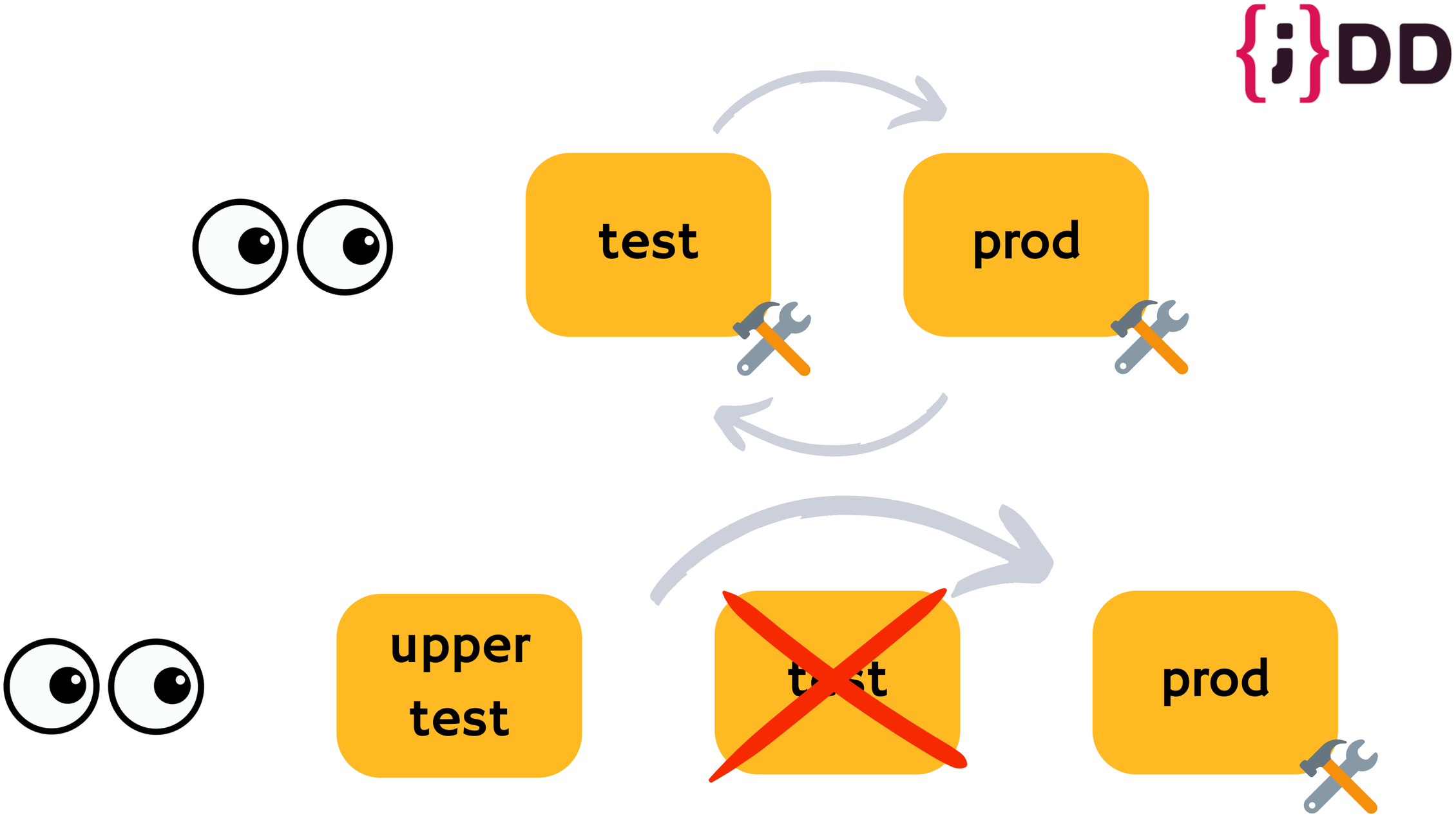
kod jest całkowicie nietestowalny

https://www.p-programowanie.pl/paradygmaty-programowania/po-co-testy-jednostkowe

{i}DD

może wtedy warto napisać test wyższego rzędu np. integracyjny

... ale tymczasowo


{i}DD
test
prod
upper test
test
prod

Dobieraj rodzaj testu do danej sytuacji

Podsumowanie

Warto korzystać z dobrodziejstwa testów

- Pisz czytelne testy
- Nie "betonuj" kodu testami
- Dobierz odpowiednie podejście do pisania testów
- Poświęć więcej czasu na analizę
- Dobieraj rodzaj testu do danej sytuacji

Eventory

Dziękuję!

email: saneckicezary@gmail.com
instagram: @cezary.sanecki
blog: https://cezarysanecki.pl

Q&A