Testowanie Oprogramowania

autorzy~
Marianna Kubsik i Szymon Bobrowski

Ogólna teoria testowania czyli co to znaczy *testować oprogramowanie*?

Testowanie oprogramowania- proces zapewnienia jakości oprogramowania. *Jakość* to termin określający stopień zgodności implementacji kodu z oczekiwaniami, potrzebami i założonymi wymaganiami postawionymi przez zamawiającego.

One Word, Different Reactions

"BUG"

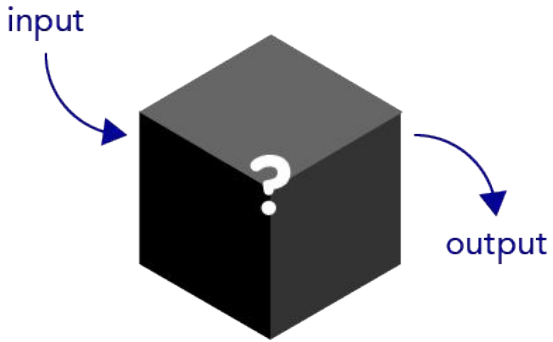
Tester Developer Manager

Techniki testowania

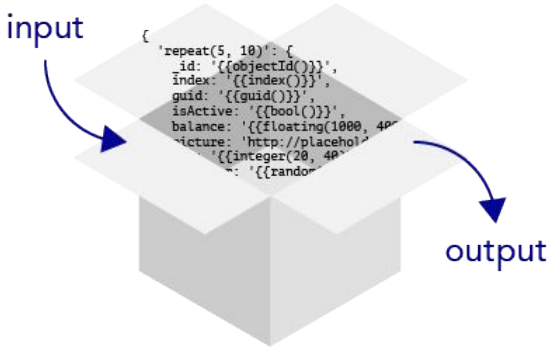
Istnieją dwie podstawowe techniki testowania:

- metoda czarnej skrzynki - tester nie zna struktury programu, musi testować system z zewnątrz,
- metoda białej skrzynki - opiera się na analizie struktury kodu, tester zna budowę systemu, ma wiedzę jak odbywają się poszczególne procesy.

Black-box Testing



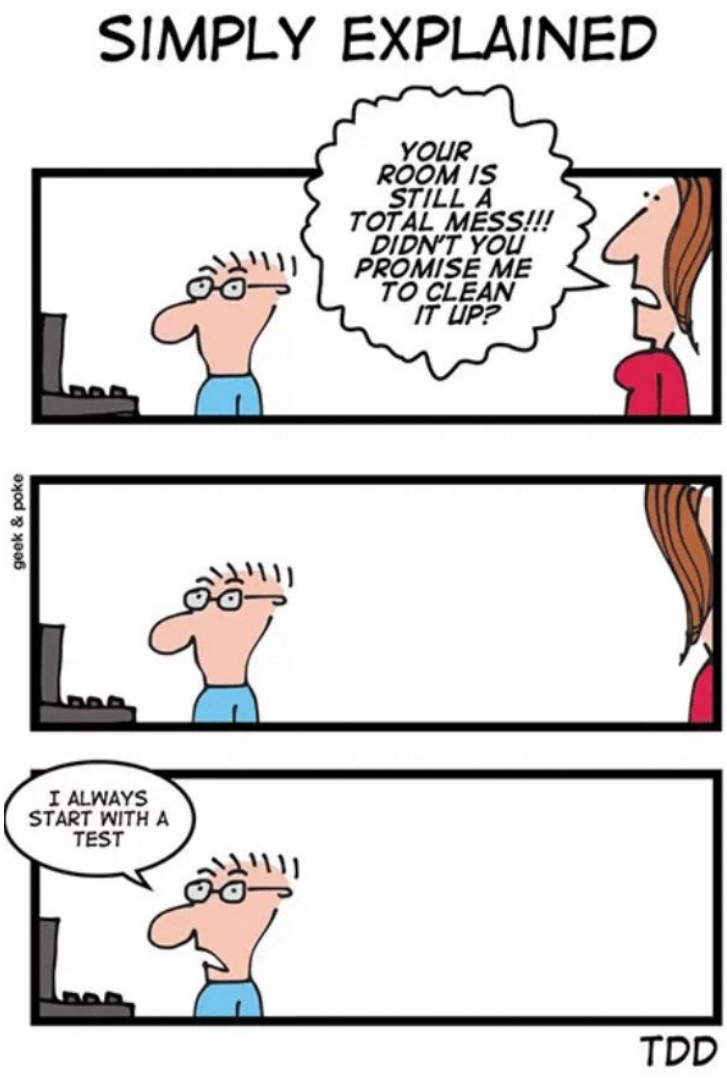
White-box Testing



hidden or unknown	INTERNAL STRUCTURE	known
not needed	KNOWLEDGE OF IMPLEMENTATION	needed
not required	KNOWLEDGE OF PROGRAMMING	required
functional, behavioural test	TYPE OF TEST	structural, logic test
on the basis of req. spec document	TESTING INITIATED...	after detail design document
higher levels	LEVEL OF SOFTWARE TESTING	lower levels

Test-Driven Development

1. Add a test for the new functionality or behavior.
2. See it fail.
3. Write enough code to make the test pass.
4. Make sure all the previous tests pass as well.
5. Refactor the code.
6. Repeat until done.

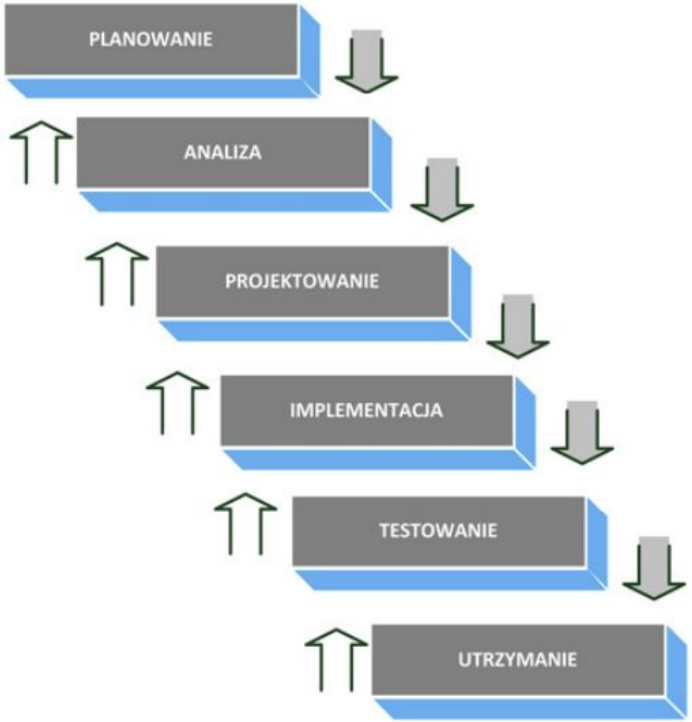

SIMPLY EXPLAINED
YOUR ROOM IS STILL A TOTAL MESS!!! DIDN'T YOU PROMISE ME TO CLEAN IT UP?
geek & poke
I ALWAYS START WITH A TEST
TDD

Testowanie w cyklu życia oprogramowania

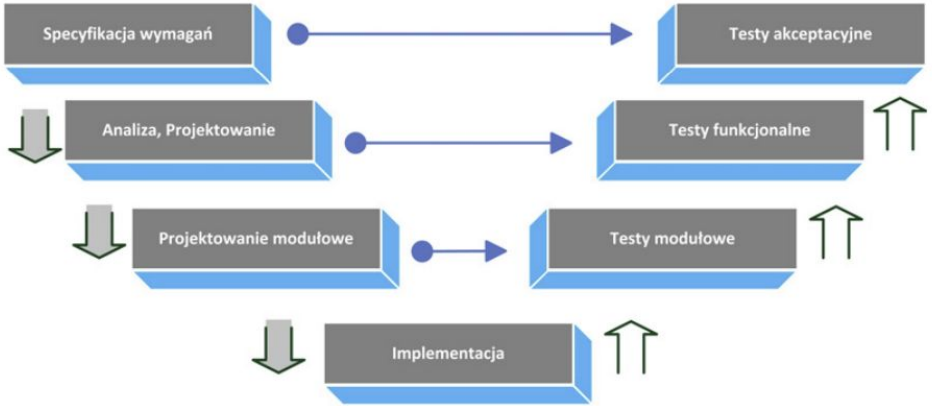
Cykl życia oprogramowania to okres od zmaterializowania się koncepcji jego powstania do momentu, kiedy zostanie ono wycofane z użytku.

W najprostszym ujęciu oprogramowanie podlega następującym fazom:

- określenie wymagań,
- projektowanie,
- implementacja,
- testy,
- utrzymanie (pielęgnacja produkcyjna), ewolucja.



Rys.1 Model Kaskadowy



Rys.2 Model V

Profil profesjonalnego testera

- Znajomość zastosowanych technologii przynajmniej w podstawowym zakresie np. umiejętność czytania kodu Java
- Ugruntowane umiejętności twarde ściśle związane z wykonywaniem testów np. specjalistyczne narzędzia testerskie (SoapUI, JMeter, OpenScript)
- wiedza z obszaru, jakiego dotyczy oprogramowanie, np. telekomunikacji, gastronomii itp.

Replikacja błędów

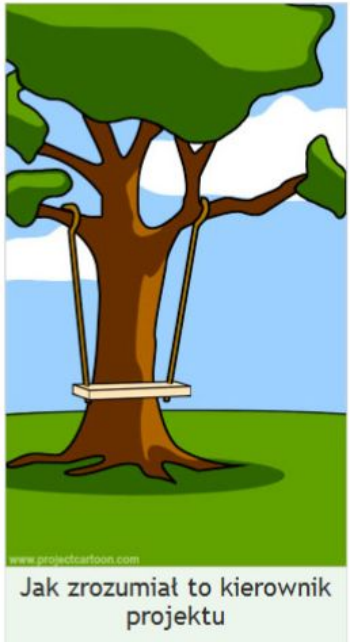
Stosuje się w celu:

- weryfikacji zasadności zgłoszenia, które spłynęło ze środowiska produkcyjnego (klient zewnętrzny),
- przygotowania środowiska testowego, weryfikacji skuteczności poprawki,
- wsparcia prac programistycznych.

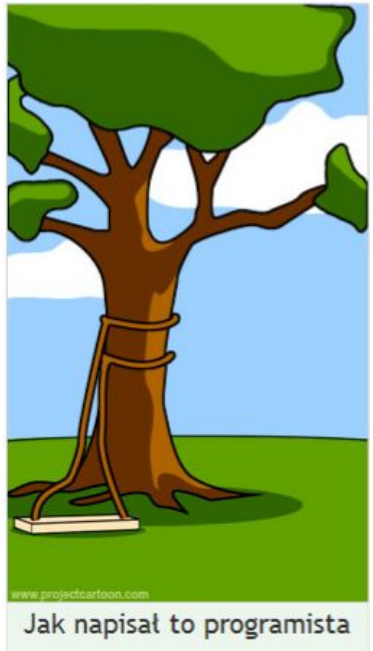
Realizacja projektu w praktyce

Najwięcej trudności w trakcie realizacji projektu IT nastręcza różnica w interpretacji i zrozumieniu oczekiwań różnych stron biorących udział w realizacji projektu.

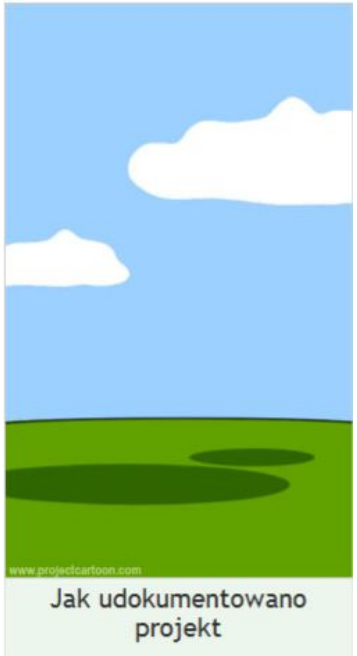
Jak opisał to klient



Jak zrozumiał to kierownik projektu



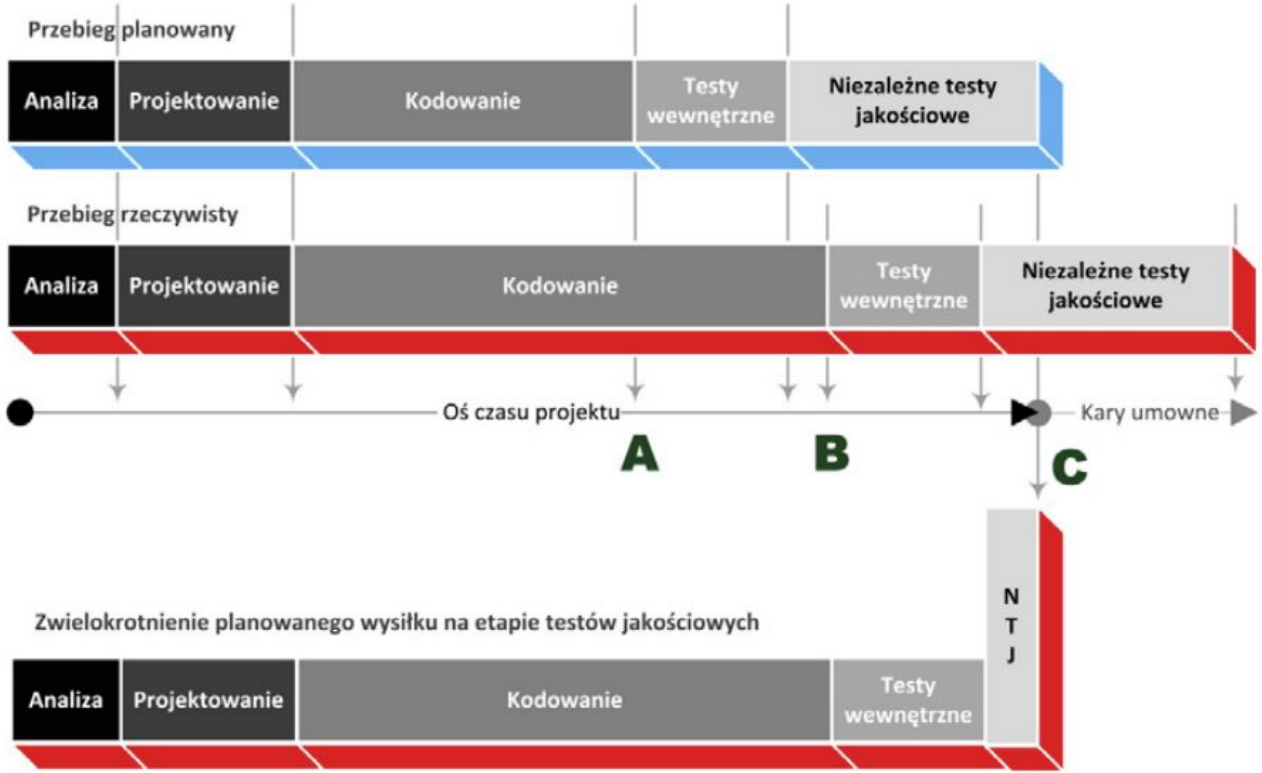
Jak napisał to programista



Jak udokumentowano projekt

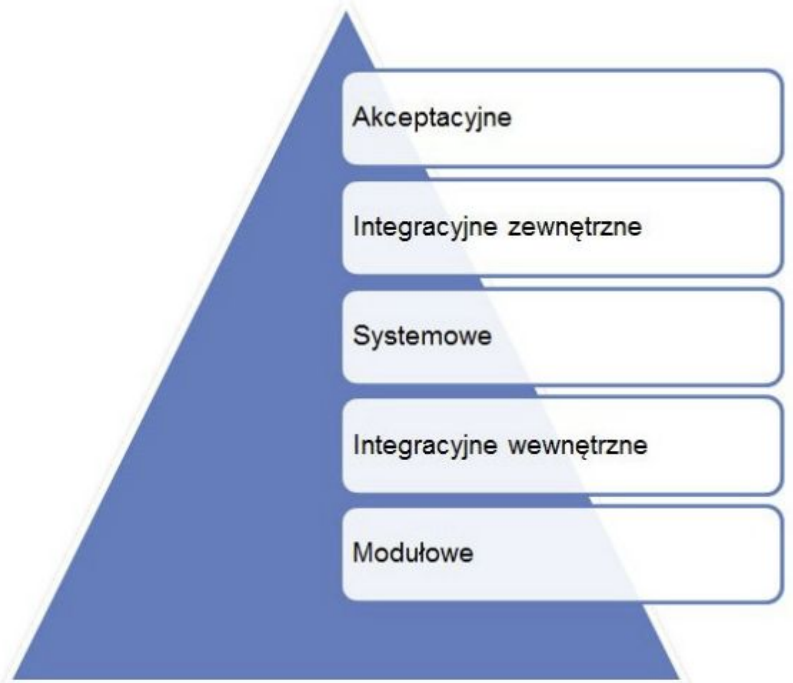
Czego klient naprawdę potrzebował

Presja czasu w życiu testera



Poziomy wykonywania testów

- testy modułowe (jednostkowe)
- testy integracyjne wewnętrzne
- testy systemowe
- testy integracyjne zewnętrzne
- testy akceptacyjne (odbiorcze)



Typy testów

Testy możemy podzielić według przyczyny ich wykonywania. Rozróżniamy odpowiednio:

- testy funkcjonalne- odpowiadają na pytanie ***co robi system?***
- testy niefunkcjonalne- odpowiadają na pytanie ***jak działa system?***
- testy regresywne- zapewniają, że modyfikacja programu nie wpłynęła na uprzednio działające funkcjonalności

JUnit 5

JUnit 5 to narzędzie służące do tworzenia testów jednostkowych.

Podstawowe adnotacje używane w testach pochodzą z pakietu `org.junit.jupiter.api` i są to:

- `@BeforeAll`- metoda oznaczona tą adnotacją będzie wykonana przed wszystkimi innymi metodami w klasie
- `@BeforeEach` - metoda oznaczona tą adnotacją będzie wykonana przed każdym kolejnym testem
- `@Test` - właściwa metoda testowa
- `@AfterEach` - metoda oznaczona tą adnotacją będzie wykonana po każdym kolejnym teście
- `@AfterAll` - metoda oznaczona tą adnotacją będzie wykonana po wszystkich innych metodach w klasie
- `@Nested` - wewnętrzna klasa oznaczona z tą adnotacją pozwala to na lepszą organizację testów

Dokumentacja: http://junit.org/junit5/docs/current/user-guide/

Pisanie własnych testów

```
class IP_CalculatorTest {

    @Test
    public void testLowestPossibleMask() { assertTrue(IP_Calculator.isMaskCorrect( maskText: "1")); }

    @Test
    public void testNegativeMask() { assertFalse(IP_Calculator.isMaskCorrect( maskText: "-5")); }

    @Test
    public void testNotNumericalMask() { assertFalse(IP_Calculator.isMaskCorrect( maskText: "pizza")); }

    @Test
    public void testMaximumAllowedAddress() { assertTrue(IP_Calculator.isAddressCorrect("255.255.255.255")); }
```

```java
public class NetworkTest {

    static private Network network;

    @BeforeAll
    public static void init() { network = new Network( networkAddressDecimal: "192.168.128.0",  networkMaskNumeral: 18); }

    @AfterAll
    public static void tearDown() { network = null; }

    @Test
    public void testConvertMaskToDecimal() { assertEquals( expected: "255.255.192.0", network.getNetworkMaskDecimal()); }

    @Test
    public void testConvertMaskToDecimalIncorrect() {
        assertNotEquals( unexpected: "119.0.29.0", network.getNetworkMaskDecimal());
    }
```

```java
@Nested
@TestInstance(TestInstance.Lifecycle.PER_CLASS)
class NumberOfHostsCorrectness {

    ArrayList<Integer> list;

    @BeforeAll
    public void init() {
        list = new ArrayList<>();
        list.add(38);
        list.add(1);
        list.add(191);
    }

    @AfterAll
    public void tearDown() { list = null; }

    @Test
    public void testIfNumberOfHostsIsCorrect() { assertTrue(network.isNumberOfHostsCorrect(list)); }
}
```

Literatura

[1] Java Unit Testing with JUnit 5 ~ Shekhar Gulati, Rahul Sharma

[2] Testowanie Oprogramowania- Podręcznik dla początkujących ~ Rafał Pawlak

[3] https://www.janio.careers/post/grey-box-testing-with-a-janio-qa-engineer

[4] https://it-academy.pl/testowanie-metoda-bialej-i-czarnej-skrzynki/

[5] https://blog.qalabs.pl/junit/junit5-pierwsze-kroki/

[6] https://www.softwaretestinghelp.com/junit-5-nested-class/

[7] https://www.youtube.com/playlist?list=PLxbRjI3sr4mzq4rbTivcBDdcDCLDSmrUD