PROGRAMIŚCI DLA SZCZECINA

Zachodniopomorska Szkoła Biznesu w Szczecinie

Teoria testowania oprogramowania

Część 2 – Rodzaje testów, piramida testów

Mariusz Łazor / 26.08.2019

Fundusze Europejskie Program Regionalny

Rzeczpospolita Polska

Pomorze Zachodnie

Unia Europejska Europejski Fundusz Społeczny

Podziały testów

- Poziom, na którym są przeprowadzane (jednostkowe, modułowe, integracyjne, systemowe, akceptacyjne)
- Warunki ich przeprowadzania (testy alfa i beta)
- Stosowane techniki (białoskrzynkowe, czarnoskrzynkowe, szaroskrzynkowe)
- Typ testu (funkcjonalny, niefunkcjonalny, regresywny)

Testy jednostkowe

- Testom poddawany jest kod źródłowy (klasy, metody, funkcje, procedury)
- Są trzymane blisko kodu źródłowego (de facto same w sobie są kodem)
- Do ich rozwoju wykorzystuje się specjalistyczne narzędzia/ biblioteki (np. JUnit, TestNG)
- Za ich rozwój i utrzymanie odpowiadają programiści
- Są uruchamiane podczas budowania programu (kompilacji) ale możliwe jest również uruchomienie tych testów oddzielnie
- Każdy pojedynczy test powinien być niezależny od pozostałych
- Często są utożsamiane z testami modułowymi (np. w ISTQB)
- Zdarza się, że najpierw tworzone są testy, a dopiero kod źródłowy (Test Driven Development)
- Wykorzystywane są techniki białoskrzynkowe
- Testy jednostkowe są testami automatycznymi

Testy modułowe

- Testy pojedynczego modułu (np. ekranu, okna modalnego, ich wycinka, a nawet programu – wszystko zależy od tego, co zdefiniujemy w projekcie jako moduł)
- Podstawą testów są wymagania dotyczące konkretnego modułu (specyfikacja modułu), na podstawie których tworzone są przypadki testowe, ale wykonuje się również testy eksploracyjne
- Są wykonywane w izolacji od reszty systemu
- Reszta systemu może zostać zaślepiona (tzw. mocki)
- Na poziomie modułowym przeprowadzane są głównie testy funkcjonalne, ale czasami na tym poziomie wykonuje się też testy niefunkcjonalne (np. testy wydajnościowe mikroserwisu)
- Testy modułowe często są automatyzowane
- Za testy modułowe odpowiada tester

Testy integracyjne

- O testowaniu integracyjnym mówimy, kiedy mamy zintegrowanych kilka modułów naszej aplikacji
- Testowana jest komunikacja między modułami, systemami (testy integracji systemów), sprzętem oraz interfejsy
- Wyrocznią testów są np. projekty architektury systemu czy modele przepływu danych (np. sekwencyjny)
- Odpowiedzialność testera
- Do testów integracyjnych można używać zaślepek (np. kiedy opieramy się na kontraktach)
- Istnieje kilka podejść do testów integracyjnych (wstępująca, zstępująca, „wielki wybuch”)

Testy integracyjne bottom-up

- Podejście wstępujące
- Na początku testowana jest integracja na najniższym poziomie hierarchii przechodząc po kolei w górę
- Do przeprowadzenia testów nie jest konieczne ukończenie implementacji na najwyższym poziomie hierarchii
- Łatwa lokalizacja defektów
- Integracja modułów na najwyższym poziomie hierarchii (najbardziej krytycznych, które inicjują cały przepływ danych) jest testowana jako ostatnia

Testy integracyjne top-down

- Podejście zstępujące
- Testy integracyjne rozpoczyna się od modułów na najwyższym poziomie hierarchii schodząc po kolei w dół
- Aby móc je przeprowadzić, moduły na niższych poziomach muszą być gotowe lub musimy je mockować
- W pierwszej kolejności wykrywane są defekty dotyczące integracji na najwyższym poziomie (w miejscach najbardziej krytycznych)
- Po testach integracji na najwyższym poziomie może zostać mało czasu na testy na niższych poziomach integracji, co może doprowadzić do niewykrycia defektów

Testy integracyjne metodą „big-bang"

- Testy integracyjne są wykonywane po zintegrowaniu wszystkich modułów
- Długi czas oczekiwania na rozpoczęcie testów integracyjnych
- Trudna lokalizacja błędów
- Podejście wygodne dla małych systemów

Testy systemowe

- Całościowe testy w pełni zintegrowanego systemu lub jego części, jeśli jesteśmy w stanie ją wyodrębnić
- Testowane są zarówno cechy funkcjonalne jak i niefunkcjonalne
- Testy systemowe odzwierciedlają rzeczywiste korzystanie z systemu (np. zalogowanie się, zakup komputera w sklepie internetowym, wylogowanie się)
- Podstawą testów najczęściej jest specyfikacja funkcjonalna systemu
- Podczas testów systemowych nie powinny być używane zaślepki, a całe środowisko testowe powinno w możliwie maksymalny sposób odzwierciedlać środowisko produkcyjne włączając w to infrastrukturę fizyczną i logiczną
- Odpowiedzialność testera
- Testy systemowe są często poddawane automatyzacji

Testy akceptacyjne (tzw. UATy)

- UAT - User Acceptance Testing
- Ich celem jest akceptacja i odbiór modułu lub systemu i rekomendacja do wdrożenia produkcyjnego
- Można je wykonywać na poziomie pojedynczego modułu jak i całego systemu
- Są realizowane w celu upewnienia się, że produkt jest zgodny z wymaganiami i w pełni realizuje założone procesy biznesowe
- W trakcie testów wykorzystywane są techniki czarnoskrzynkowe
- Są podobne do funkcjonalnych testów systemowych
- Leżą w interesie odbiorcy produktu

Testy alfa i beta

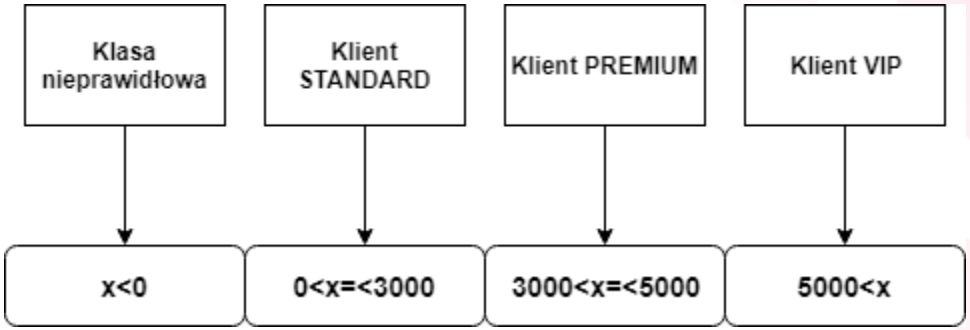
- Przeprowadzane są najczęściej w przypadku tworzenia oprogramowania „pudełkowego", np. systemy operacyjne, gry
- Ich celem jest uzyskanie opinii o systemie przez potencjalnych klientów
- Testy alfa są wykonywane wewnątrz organizacji tworzącej oprogramowanie
- Testy beta przeprowadza się poza organizacją (otwarte i zamknięte beta-testy)

Testy białoskrzynkowe

- Testy oparte na dostępie do wewnętrznej struktury systemu (głównie kodu źródłowego)
- Pokrycie instrukcji – testy projektowane są w taki sposób, żeby każda możliwa instrukcja została poddana testowi
- Stopień pokrycia instrukcji to liczba instrukcji pokrytych testami podzielona przez całkowitą liczbę instrukcji
- Pokrycie decyzji – celem testów jest pokrycie wszystkich możliwych decyzji
- Stopień pokrycia decyzji to liczba decyzji pokrytych testami podzielona przez całkowitą liczbę dostępnych decyzji, 100% pokrycia decyzji daje pewność pokrycia 100% instrukcji, ale nie działa to w drugą stronę

Testy czarnoskrzynkowe

- Nie znamy struktury wewnętrznej systemu, a jedynie specyfikację wymagań
- Podział danych na klasy równoważności – podział danych wejściowy lub wyjściowych na zbiory powodujące takie same zachowanie systemu



Testy czarnoskrzynkowe

- Analiza wartości brzegowych – rozwinięcie testów opartych na klasach równoważności o weryfikację zachowania systemu na krańcach danych klas
- Tablica decyzyjną – tabela składająca się z różnych kombinacji warunków wejściowych oraz oczekiwanego działania systemu dla tych danych

Warunki	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Zarobki > 3000 zł	1	1	1	0	0	0	0
Umowa na czas nieokreślony	1	1	0	1	1	0	0
Posiada historię kredytową	1	0	1	1	0	1	0
Akcje							
Przyznać kredyt z preferencyjną marżą	1	0	0	1	0	0	0
Przyznać kredyt z podwyższoną marżą	0	1	1	0	1	0	0
Zażądać dodatkowego ubezpieczenia	0	0	0	1	1	0	0
Nie przyznawać kredytu	0	0	0	0	0	1	1

Testy czarnoskrzynkowe

- Przypadki użycia – testy odbywają się na odstawie opisanych docelowych scenariuszy, w których system może zostać użyty uwzględniając scenariusz główny (najczęściej pozytywny) oraz alternatywne (niekoniecznie negatywne)

Testy szaroskrzynkowe

- Połączenie testów biało i czarnoskrzynkowych
- Znamy ogólną strukturę projektu bez szczegółów oraz specyfikację systemu
- Najczęściej są wynikiem konsultacji testera z programistą lub dostępu do projektu struktury oprogramowania
- Są bytem dosyć abstrakcyjnym, ale w rzeczywistości ma częste zastosowanie i pozytywnie wpływa na jakość oraz rozwija testera pod kątem technicznym

Testy funkcjonalne

- Weryfikacja funkcjonalności na podstawie specyfikacji funkcjonalnej systemu
- Sprawdzamy „co" system robi i czy robi to prawidłowo
- Stosowane są głównie testy czarnoskrzynkowe

Testy niefunkcjonalne

- Sprawdzamy „jak" system działa
- Testy bezpieczeństwa
- Testy wydajnościowe (czas odpowiedzi przy rosnącym obciążeniu lub malejących zasobach, np. pamięci operacyjnej)
- Testy obciążeniowe (zachowanie systemu przy wysokim obciążeniu, jak duże obciążenie jest w stanie znieść nasza aplikacja)
- Testy przeciążeniowe (zachowanie systemu przy przekroczeniu maksymalnego obciążenia, czy system zawiedzie w oczekiwany sposób, czy stracimy jakieś dane)
- Testy użyteczności
- Testy dostępności
- I wiele innych

Testy potwierdzające i regresywne

- Testy potwierdzające (re-testy) to retesty zgłoszonych i poprawionych defektów
- Testy regresywne to sprawdzenie, czy funkcjonalność, która wcześniej działała prawidłowo, działa nadal (np. po wprowadzeniu zmian w systemie)
- Testy regresywne wykonuje się po wprowadzeniu zmian w module lub systemie, a także po zamknięciu defektów
- W praktyce testy regresywne wykonuje się przed wdrożeniem systemu na produkcję
- Testy regresywne są często automatyzowane

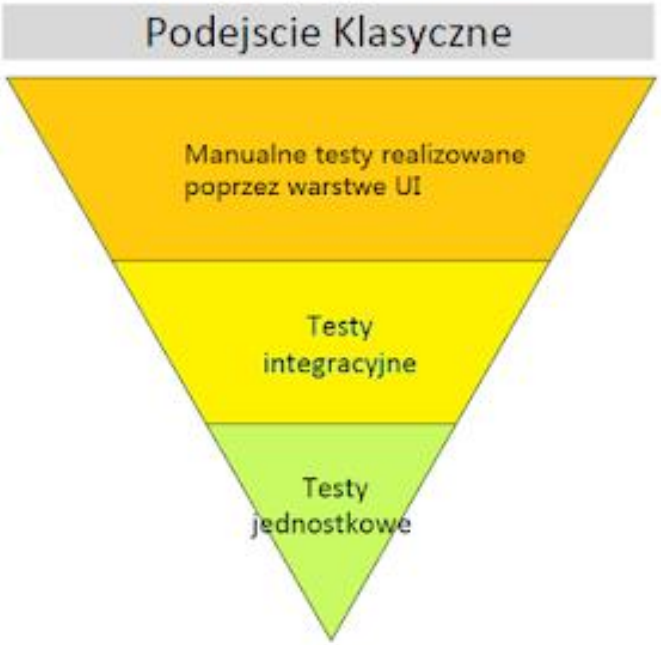
Testy dymne (smoke tests)

- Wybrane testy, które weryfikują działanie podstawowych ścieżek najbardziej krytycznych funkcjonalności (poszukują poważnych problemów)
- Ich celem jest potwierdzenie, że podstawowe funkcjonalności działają i możliwe jest bardziej dogłębne testowanie
- Testujemy bardziej wszerz (szerszy zakres testów) niż w głąb (dokładna weryfikacja modułu/systemu)
- Testy dymne wykonujemy np. po zintegrowaniu kolejnego modułu (poprzednie zostały już zweryfikowane)
- Zajmują mało czasu
- Możemy je wykonywać na wszystkich poziomach testów

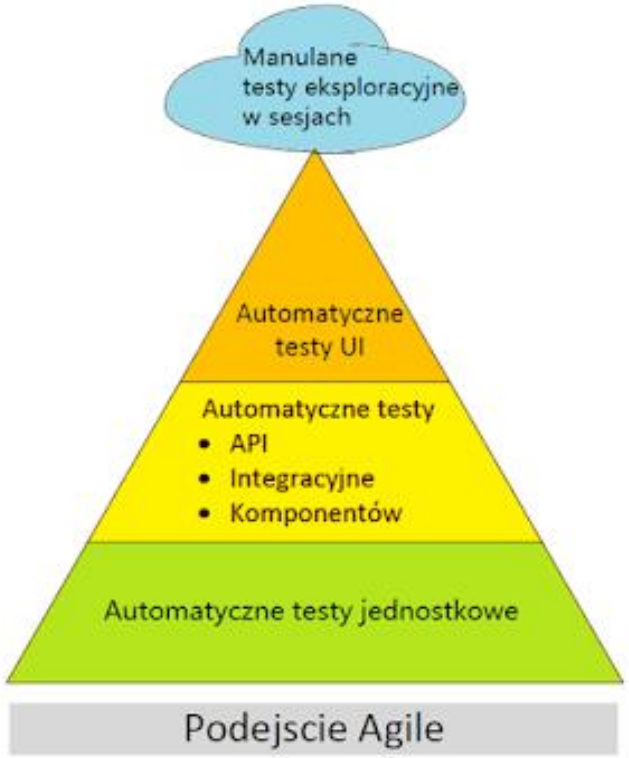
Piramida testów

- Obrazuje ilość zasobów przeznaczonych na testy na danym poziomie (wartość pola figury) oraz kolejność wykonywania testów od dołu do góry
- Elementy na samym dole są najtańsze i najszybsze do zrealizowania
- Przechodząc w górę piramidy, testy są coraz wolniejsze i droższe
- Podejście klasyczne zakłada wysokie zaangażowanie ręcznych testów na wszystkich poziomach poza testami jednostkowymi, których ogólny udział w całym zakresie testów jest niewielki
- Nowoczesne podejście zakłada maksymalną automatyzację testów na wielu poziomach, kładąc duży nacisk na rozbudowane testy jednostkowe (w praktyce przy napiętych terminach potrafi być to trudne do osiągnięcia – czasami nawet najświetniejsze idee potrafią być brutalnie zweryfikowane przez życie)

Piramida testów – podejście klasyczne



Piramida testów – podejście zwinne



Polecane materiały

- Artykuły dotyczące poziomów testów: https://devenv.pl/poziomy-testow/

  http://www.testowanie.net/poziomy-testow/
- Piramida testów: https://ucgosu.pl/2018/09/piramida-testow-do-czego-sluza-poszczegolne-poziomy/