PROGRAMIŚCI DLA SZCZECINA

Zachodniopomorska Szkoła Biznesu w Szczecinie

Teoria testowania oprogramowania

Część 1 – Wprowadzenie do IT i podstawy testowania

Mariusz Łazor / 22.08.2019

Fundusze Europejskie Program Regionalny

Rzeczpospolita Polska

Pomorze Zachodnie

Unia Europejska Europejski Fundusz Społeczny

Czego nauczymy się na szkoleniu?

- Poznamy podstawowe zasady, którymi rządzi się rynek IT
- Poznamy zagadnienia związane z teorią testowania oprogramowania i zarządzania jakością
- Nauczymy się przeprowadzać testy eksploracyjne oraz projektować scenariusze i przypadki testowe
- Poznamy podstawy technologii wykorzystywanych przy tworzeniu aplikacji internetowych
- Nauczymy się podstaw tworzenia testów automatycznych
- Dowiemy się czym są ciągła integracja i ciągłe dostarczanie i jak one wpływają na testowanie
- Nauczymy się wykonywania operacji na bazach danych
- Poznamy zagadnienia związane z bezpieczeństwem aplikacji internetowych
- Poznamy metodykę SCRUM

Czym jest IT?

„Dziedzina działalności gospodarczej związana z produkcją komputerów i ich oprogramowania, budową systemów informatycznych i ich zastosowaniami w gospodarce."

Charakterystyka pracy w IT

- Wielokulturowe środowisko (konieczna znajomość j. angielskiego)
- Praca zespołowa (najczęściej 4 – 10 osób)
- Styczność z nowoczesnymi technologiami
- Ciągły rozwój (pośrednio zależy to też od postawy pracownika)
- Jasno określona ścieżka rozwoju i awansu (najczęściej)
- Elastyczne godziny pracy
- Możliwość pracy zdalnej
- Miłe środowisko pracy (brak mobbingu, rzadko zdarzają się naciski ze strony przełożonych)
- Samodzielność (sami organizujemy swoją pracę)
- Duża odpowiedzialność (tworzymy projekty warte dziesiątek i setek tysięcy złotych)
- Benefity (karta sportowa, prywatna opieka medyczna, ubezpieczenie grupowe itd.)
- Atrakcyjne wynagrodzenie

Czym zajmujemy się w IT?

- Projektowanie oprogramowania
- Analiza biznesowa procesów
- Analiza oprogramowania
- Tworzenie oprogramowania
- Rozwój i utrzymanie oprogramowania
- Testowanie oprogramowania
- Zapewnianie jakości

Co to jest oprogramowanie?

- Programy komputerowe
- Procedury
- Dokumentacja
- Dane
- Infrastruktura

Jakie oprogramowanie tworzymy?

- Aplikacje internetowe (działające w przeglądarce internetowej)
- Aplikacje mobilne (smartfony, tablety, TV)
- Aplikacje desktopowe (np. Windows, macOS)
- Systemy back-endowe (API)
- Oprogramowanie różnych urządzeń (np. AGD)
- Systemy „automotive” (oprogramowanie samochodów)

Weryfikacja a walidacja

- Weryfikacja - egzaminowanie poprawności i dostarczenie obiektywnego dowodu, że produkt procesu wytwarzania oprogramowania spełnia zdefiniowane wymagania (Czy produkt tworzony jest prawidłowo?)
- Walidacja - sprawdzenie poprawności i dostarczenie obiektywnego dowodu, że produkt procesu wytwarzania oprogramowania spełnia potrzeby i wymagania użytkownika (Czy tworzony produkt jest prawidłowy?)

Co to jest testowanie oprogramowania”

- Czynność weryfikująca poprawność działania oprogramowania i służąca do zebrania informacji na jego temat
- Testowanie służy do zapewnienia Klienta, że otrzyma On produkt zgodny z jego oczekiwaniami (walidacja produktu)
- Podstawą testowania jest tzw. „wyrocznia” (ang. oracle), którą może być dokumentacja, oprogramowanie, sam tester (testy eksploracyjne, wiedza ekspercka), właściciel produktu/systemu (Product Owner)

Co podlega testom?

- Sam pomysł na oprogramowanie
- Dokumentacja projektowa i produktowa
- Kod źródłowy
- Integracja między modułami systemu i różnymi systemami
- Funkcjonalne i niefunkcjonalne cechy systemu

Dlaczego testowanie jest ważne?

- Każdy z nas jest człowiekiem i popełnia błędy, niektóre z nich są trywialne, ale zdarzają się też takie, które powodują duże straty bądź są niebezpieczne dla życia i zdrowia
- Testy pomagają w zmniejszeniu ryzyka wystąpienia problemów podczas użytkowania oprogramowania i przyczyniają się do podniesienia jakości sytemu
- Znalezienie błędu dopiero po oddaniu systemu do użytkowania powoduje powstanie znacznego kosztu i ma skutki wizerunkowe dla samego klienta jak i dostawcy

Czym w sumie zajmuje się tester?

- Projektowanie testów (scenariusze i przypadki testowe)
- Weryfikacja i walidacja systemu
- Walidacja wymagań, czasami również ich analiza
- Zgłaszanie defektów
- Przygotowanie danych testowych
- Tworzenie i utrzymanie środowiska testowego
- Wdrażanie systemu na środowisku produkcyjnym
- Wsparcie klienta w przypadku problemów
- Przygotowanie testów automatycznych

Testowanie a zapewnianie jakości (QA)

Testowanie skupia się na:

- skupia się na produkcie
- wyszukiwaniu defektów
- walidacji produktu pod kątem zgodności z wymaganiami klienta
- jest czynnością detekcyjną
- odpowiedzialność testera

Testowanie a zapewnianie jakości (QA)

Zapewnianie jakości to:

- definiowanie i poprawianie procesów (strategia testów)
- pilnowanie zgodności działań projektowych z procesem
- definiowanie miar do oceny procesu
- nie weryfikuje jakości produktu, skupia się na procesach
- jest czynnością prewencyjną
- odpowiedzialność inżyniera (kierownika/managera) zapewnienia jakości
- w praktyce rola testera i QA często jest współdzielona przez tę samą osobę

Cechy dobrego Testera/QA

- Umiejętność analitycznego myślenia
- Dokładność
- Samoorganizacja
- Komunikatywność
- Cierpliwość i empatia
- Nieugiętość, asertywność i determinacja
- Chęć do rozwoju (szczególnie cenne jest pozyskiwanie wiedzy technicznej)
- Znajomość języka angielskiego

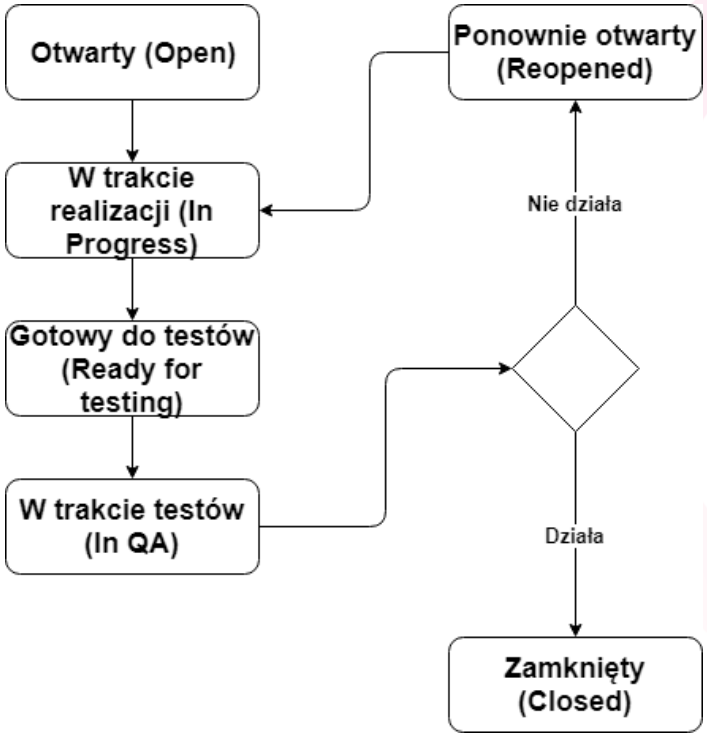
Błąd, awaria, defekt

- Błąd – czynnik w 100% zależny od człowieka (np. działanie programisty)
- Defekt – cecha oprogramowania, np. nieprawidłowa składnia kodu źródłowego czy implementacja funkcjonalności niezgodna z wymaganiami związana z popełnieniem błędu
- Awaria – działanie oprogramowania niezgodne z oczekiwaniami lub wymaganiami, nie zawsze jest ona wynikiem defektu w aplikacji, awaria może być również spowodowana defektem środowiska – awaria objawia się dopiero podczas użytkowania systemu
- Człowiek popełnia błąd, który powoduje powstanie defektu, który z kolei może doprowadzić do wystąpienia awarii

Jak zgłaszać defekty?

- Do zgłaszania defektów używamy systemów, tzw. Trackerów, np. Jira.
- Tytuł – ogólny jednozdaniowy opis problemu
- Krytyczność – wielkość wpływu defektu na całe oprogramowanie
- Priorytet – określa planowaną kolejność rozwiązywania defektów
- Dane testowe – wskazanie danych testowych, które należy wykorzystać do odtworzenia awarii (często awaria objawia się przy wykorzystaniu konkretnych danych testowych)
- Warunki początkowe – warunki, jakie należy spełnić w celu odtworzenia awarii (np. konkretna konfiguracja systemu lub konkretny stan systemu, np. „Użytkownik zalogowany do systemu)
- Kroki do wykonania – lista kroków, jakie należy wykonać w celu odtworzenia problemu
- Opis awarii – wskazanie, co konkretnie nie zadziałało
- Oczekiwany rezultat – opis oczekiwanego stanu systemu po wprowadzeniu poprawki (lub gdyby oprogramowanie nie posiadało defektu)
- Załączniki (np. zrzuty ekranów, logów)

Cykl życia defektu



Zgłaszanie defektów - ćwiczenie

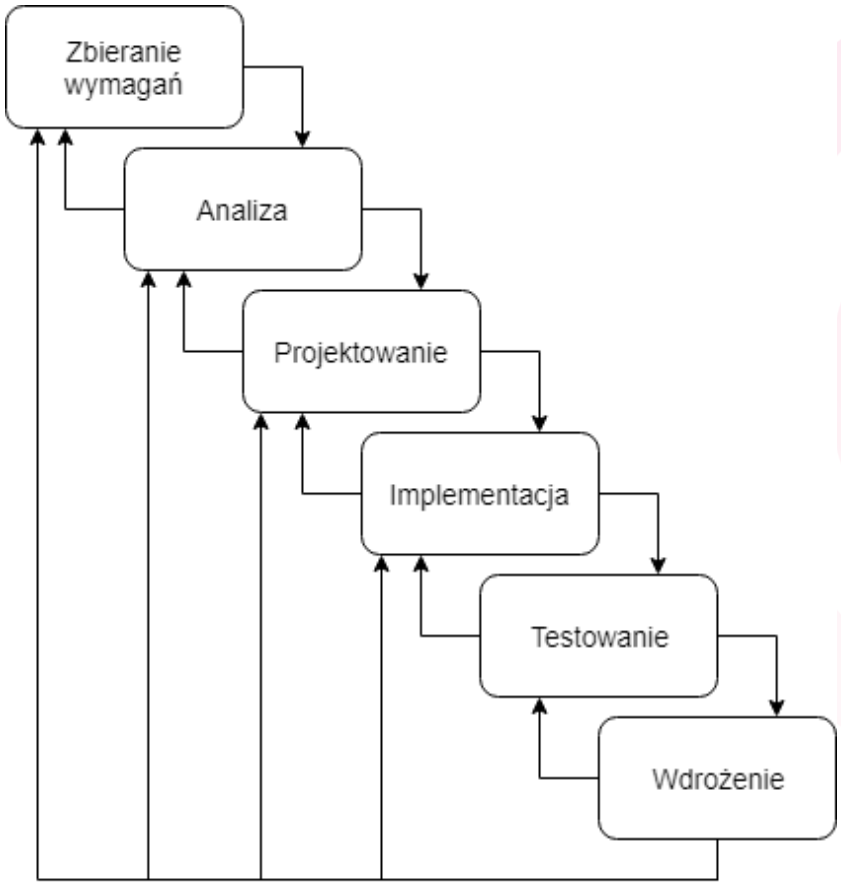
Opis wymagania:

Strona logowania do aplikacji http://demo.nopcommerce.com dostępna w ścieżce „/login" powinna posiadać pola **Email** i **Password**. Adres email powinien być zgodny z formatem xx*@xx*.xx*. W przypadku podania adresu w nieprawidłowym formacie, pod polem **Email** powinien zostać wyświetlony komunikat „**Wrong email**" w kolorze czerwonym. W przypadku niepodania adresu, po kliknięciu przycisku **LOG IN** powinien zostać wyświetlony komunikat „**Please enter your email**". Walidacja poprawności formatu adresu powinna następować dynamicznie, bez konieczności kliknięcia przycisku **LOG IN**.

Zakres ćwiczenia:

1. Zweryfikuj wyżej opisane wymaganie skupiając się tylko na polu **Email**, całkowicie pomiń w testach pole **Password**.
2. Zidentyfikuj ewentualne defekty i zgłoś je używając dowolnego edytora tekstu (np. Word, OpenOffice, Notatnik) tworząc tabelę zawierającą odpowiednie elementy.

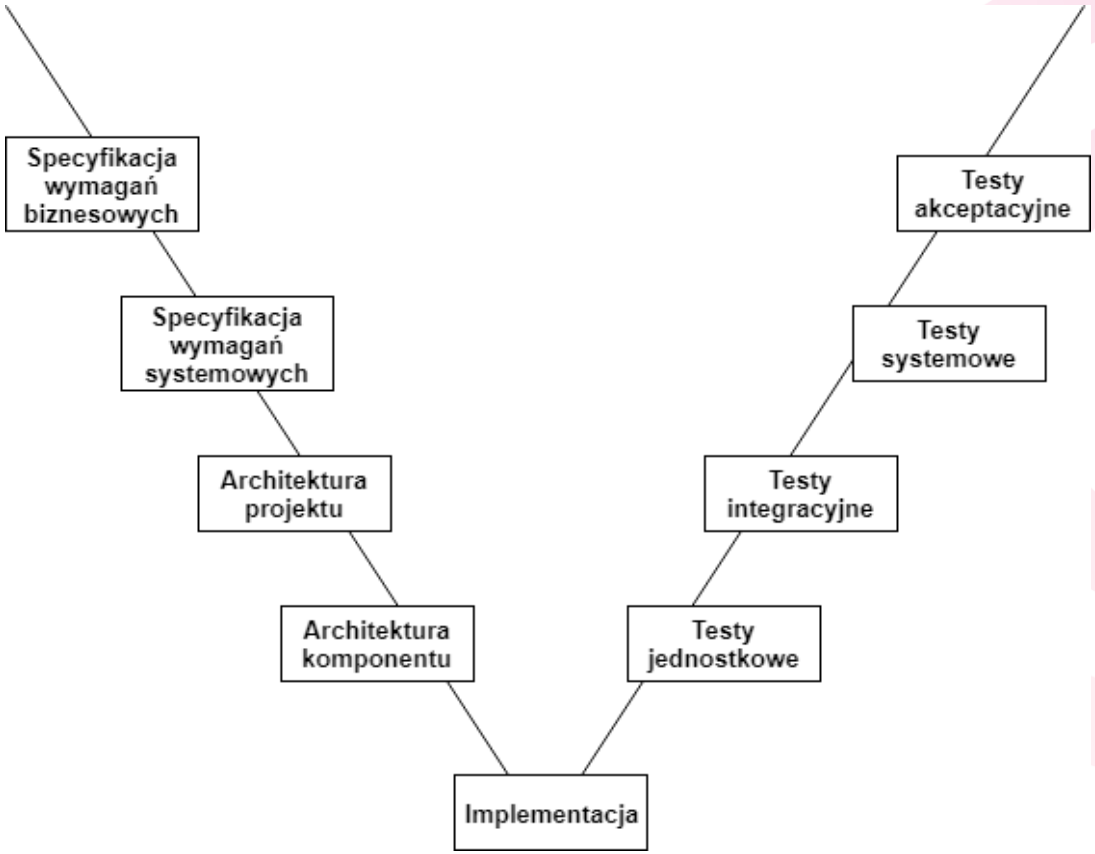
Kaskadowe wytwarzanie oprogramowania



Kaskadowe wytwarzanie oprogramowania

- Proces podzielony na jasno sprecyzowane fazy
- Rozpoczęcie kolejnej fazy jest możliwe dopiero po zakończeniu poprzedniej
- Zakończenie bieżącej fazy powoduje przejście do kolejnej
- W swojej podstawowej postaci nie umożliwia powrotu
- Wymaga kompletu i niezmienności wymagań
- Testy na samym końcu cyklu
- Bardzo wysoki koszt błędów i zmiany wymagań

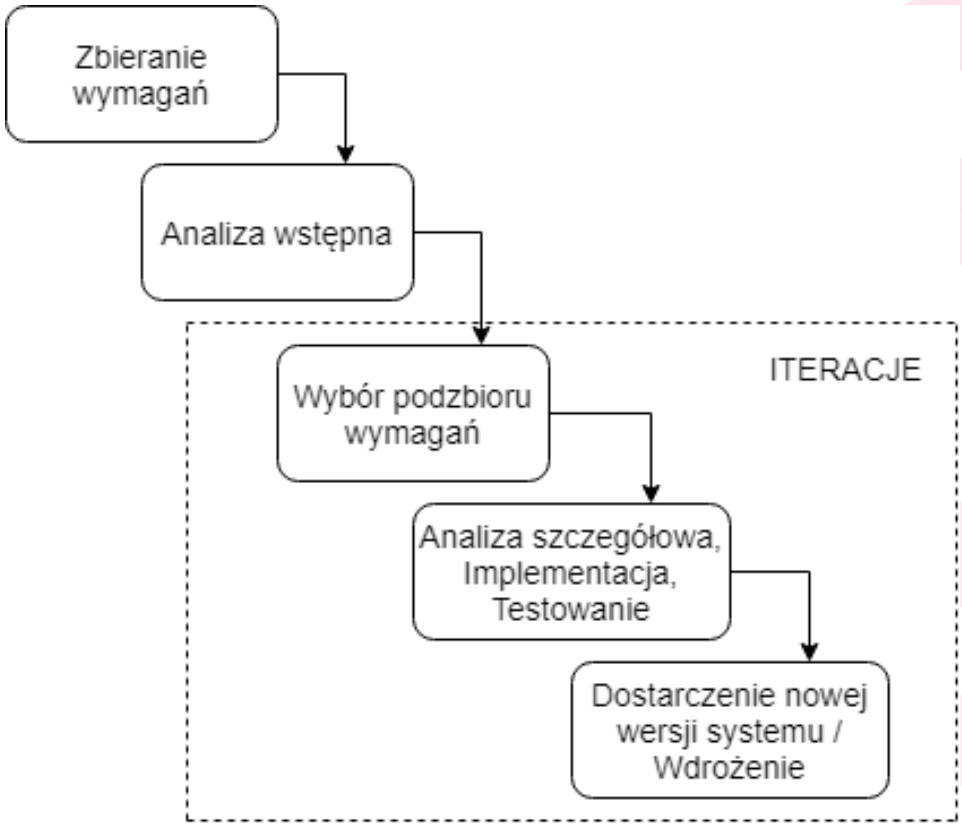
Model V (sekwencyjny)



Model V (sekwencyjny)

- Reprezentuje powiązania między różnymi etapami projektowania systemu (lewa strona) oraz jego weryfikacji (prawa strona)
- Implementacja konkretnego etapu projektu jest równocześnie pokrywana testami
- Testy są planowane już na etapie projektowania
- Szybsze, niż w modelu kaskadowym, wykrywanie defektów

Iteracyjne wytwarzanie oprogramowania



Iteracyjne (zwinne) wytwarzanie oprogramowania

- Proces wytwarzania podzielony jest na następujące po sobie iteracje
- Przed iteracjami ma miejsce jedynie wstępne zdefiniowanie i analiza wymagań, do konkretnej iteracji włączany jest określony zakres wymagań
- Niższa niż w modelu kaskadowym wrażliwość na błędy i zmiany w wymaganiach (np. nowe wymagania są dokładane do kolejnych sprintów, defekty wykryte w jednym sprincie są naprawiane w kolejnym)
- Możliwość szybkiego wdrożenia systemu i czerpania z niego korzyści
- Testowanie ma miejsce w każdej iteracji przez cały czas jej trwania (walidacja wymagań, planowanie testów, weryfikacja systemu)
- Wysokie znaczenie testów automatycznych
- Popularne jest łączenie modelu iteracyjnego z kaskadowym (np. zapętlone „waterfall'e")

Polecane materiały

- Słownik pojęć testerskich - https://sjsi.org/download/6347/
- Sylabus ISTQB poziomu podstawowego - https://sjsi.org/download/6351/
- Sylabus „Tester zwinny" – rozszerzenie ISTQB poziomu podstawowego - https://sjsi.org/download/3343/
- Modele wytwarzania oprogramowania - https://pl.scribd.com/document/30335749/Modele-procesu-wytwarzania-oprogramowania