Spis treści

[Podstawy testowania](#_Toc135500624)

[Co to jest testowanie?](#_Toc135500625)

[Po co testujemy?](#_Toc135500626)

[Testowanie w cyklu życiu oprogramowania](#_Toc135500627)

[Modele cyklu życia oprogramowania](#_Toc135500628)

[Poziomy testów](#_Toc135500629)

[Czym jest test jednostkowy?](#_Toc135500630)

[Typy testów](#_Toc135500631)

[Różnica między retestami a testami regresji?](#_Toc135500632)

[Testowanie pielęgnacyjne](#_Toc135500633)

[Testowanie statyczne](#_Toc135500634)

[Techniki testowania](#_Toc135500635)

[Techniki testowania oparte na doświadczeniu](#_Toc135500636)

[Na czym polegają testy eksploracyjne?](#_Toc135500637)

[Zarządzanie testami](#_Toc135500638)

[Planowanie i szacowanie testów](#_Toc135500639)

[Jakie elementy powinien mieć plan testów?](#_Toc135500640)

[Zarządzanie defektami](#_Toc135500641)

[Jak prawidłowo opisać błąd?](#_Toc135500642)

[Narzędzia wspomagające testowanie](#_Toc135500643)

[Inne pytania](#_Toc135500644)

[Co to jest sprint i ile może trwać?](#_Toc135500645)

# Podstawy testowania

## Co to jest testowanie?

### Po co testujemy?

Podstawowe cele testowania to:

* dokonywanie oceny produktów pracy, takich jak wymagania, historyjki użytkownika, projekt, kod;
* sprawdzanie, czy zostały spełnione wszystkie wyspecyfikowane wymagania;
* sprawdzanie, czy przedmiot testów jest kompletny i działa zgodnie z oczekiwaniami użytkowników i innych interesariuszy;
* budowanie zaufania do poziomu jakości przedmiotu testów;
* wykrywanie defektów i awarii;
* zapobieganie awariom;
* dostarczanie interesariuszom informacji niezbędnych do podejmowania świadomych decyzji dotyczących zwłaszcza poziomu jakości przedmiotu testów;
* obniżanie poziomu ryzyka związanego z jakością oprogramowania (ryzyka wystąpienia niewykrytych wcześniej awarii podczas eksploatacji);
* przestrzeganie wymagań wynikających z umów, przepisów prawa i norm/standardów;
* sprawdzanie, czy obiekt testów jest zgodny z tymi wymaganiami lub standardami

### Co to jest przypadek testowy?

przypadek testowy — zestaw warunków wstępnych, danych wejściowych, akcji (w stosownych przypadkach), oczekiwanych rezultatów i warunków końcowych opracowany na podstawie warunków testowych

# Testowanie w cyklu życiu oprogramowania

## Modele cyklu życia oprogramowania

## Poziomy testów

### Jakie znasz poziomy testów?

* Modułowe – czyli moduły, które można przetestować oddzielnie, w izolacji od reszty systemu. Zazwyczaj programista robi to w kodzie.
* Integracyjne – integracje między modułami lub systemami.
* Systemowe – to, co zazwyczaj testuje tester. Testowanie całokształtu zadań, jakie ma wykonywać aplikacja
* Akceptacyjne – zazwyczaj przeprowadza je klient. Często już na bazie dokumentacji. Tutaj testy alpha i beta.

### Czym jest test jednostkowy?

Test jednostkowy to inaczej unit test lub test modułowy. Skupia się na modułach, które można przetestować oddzielnie. Testowany fragment programu poddawany jest testowi, który wykonuje go i porównuje wynik z oczekiwanym.

Cele tego typu testowania to między innymi:

* zmniejszanie ryzyka,
* sprawdzanie zgodności zachowań funkcjonalnych i niefunkcjonalnych modułu z projektem i specyfikacjami,
* budowanie zaufania do jakości modułu,
* wykrywanie defektów w module,
* zapobieganie przedostawaniu się defektów na wyższe poziomy testowania

Testy można podzielić na następujące warianty:

* analiza ścieżek
* użycie klas równoważności
* testowanie wartości brzegowych
* testowanie składniowe (wprowadzane do systemu dane)

Przykładowe produkty pracy, które mogą być wykorzystywane jako podstawa testów w ramach testowania modułowego, to między innymi:

* projekt szczegółowy,
* kod,
* model danych,
* specyfikacje modułów.

Do typowych przedmiotów testów dla testów modułowych zaliczają się:

* moduły
* jednostki lub komponenty,
* kod i struktury danych,
* klasy,
* moduły baz danych

## Typy testów

### Jakie typy testów wyróżniamy?

* funkcjonalne (czarnoskrzynkowe) – czy system robi to, co powinien.
* Niefunkcjonalne – wydajność, bzpieczeństwo
* testowanie białoskrzynkowe – ocena struktury wewnętrznej (kod), np. pokrycie instrukcji
* testowanie powiązane ze zmianami, (testy potwierdzające i testy regresji) – potwierdzające (retesty) oraz regresji

### Różnica między retestami a testami regresji?

Retesty polegają na tym, że Ty jako użytkownik masz sprawdzić, czy defekt (w definicji przypadek testowy) naprawiony przez programistę został poprawiony. W ramach retestu my testerzy sprawdzamy, czy błąd tak jak jest opisany w krokach testowych, występuje nadal w aplikacji. Test regresji to ponowne przetestowanie uprzednio testowanego programu po dokonaniu w nim modyfikacji, w celu upewnienia się, że w wyniku zmian nie powstały nowe defekty lub nie ujawniły się wcześniej nie wykryte w nie zmienionej części oprogramowania. Testy takie są przeprowadzane po zmianach oprogramowania lub jego środowiska pracy. Zajmują dużo czasu i są bardzo kosztowne.

### Różnica między testami funkcjonalnymi a niefunkcjonalnymi

Testy funkcjonalne próbują znaleźć odpowiedź na pytanie „co robi” system, czyli weryfikują system, pod kątem tego, co robi / jakie spełnia funkcjonalności. Pozostając w tej konwencji – testy niefunkcjonalne są próbą odpowiedzi na pytanie „jak działa” system.

## Testowanie pielęgnacyjne

# Testowanie statyczne

# Techniki testowania

## Techniki testowania oparte na doświadczeniu

### Na czym polegają testy eksploracyjne?

testowanie eksploracyjne — podejście do testowania, w którym testerzy dynamicznie projektują i przeprowadzają testy na podstawie swojej wiedzy, badania elementu testowego i wynników z poprzednich testów. Opiera się na testach nieformalnych. Oznacza to, że tester nie wykonuje testów zaplanowanych z góry, w osobnej fazie planowania, lecz każdy jego kolejny krok w bieżącym scenariuszu testowym jest dynamiczny i zależy od:

* wiedzy i intuicji testera,
* doświadczenia z tą lub podobnymi aplikacjami,
* tego, jak system zachował się w poprzednim kroku scenariusza

Warto, gdy:

* specyfikacja testowanego produktu jest niepełna, złej jakości lub jest jej brak,
* istnieje presja czasu, testerzy mają mało czasu na przeprowadzenie testów;
* testerzy znają dobrze produkt i są doświadczeni w testach eksploracyjnych.

# Zarządzanie testami

## Planowanie i szacowanie testów

### Jakie elementy powinien mieć plan testów?

Plan testów to dokumentacja opisująca cele testowe do osiągnięcia oraz środki i harmonogram ich realizacji, zorganizowane tak, by koordynować czynności testowe.

1. **Identyfikator planu testów** – Unikalny identyfikator planu testów umożliwiający jednoznaczne zidentyfikowanie dokumentu.
2. **Wprowadzenie** – Podsumowanie zawartości planu testów, czyli krótki opis celu i zakresu dokumentu, przedsięwzięcia, którego dotyczy plan testów, oraz celów  
   ustalonych dla procesu testowania. W razie potrzeby należy podać odwołania do innych dokumentów.
3. **Opis przedmiotu testów** – Ten punkt planu zawiera listę elementów, które zostaną poddane testom – np. *aplikacje, procedury użytkowania* czy *instalacji, dokumentacja użytkownika, specyfikacje produktu* itd.
4. **Zakres testów (testowane cechy)** – Lista wszystkich cech oprogramowania, które zostaną poddane testom. Może to być lista wymagań wysokopoziomowych uzyskanych od klienta, ze wskazaniem sposobu realizacji testów.
5. **Wyłączenia z zakresu** – Lista wszystkich cech oprogramowania, które nie będą przedmiotem testów – np. wymagania wydajnościowe czy bezpieczeństwa testowane w ramach odrębnej inicjatywy testowej. Decyzja o wyłączeniu z zakresu testów określonych cech powinna być uzasadniona, tzn. należy określić powód wyłączenia.
6. **Podejście do testowania** – Opis podejścia do testowania oprogramowania ze wskazaniem sposobu osiągnięcia celów testowania. Wskazane jest określenie głównych faz czy etapów testów, poszczególnych czynności, wykorzystywanych technik i narzędzi. Może obejmować podział testów na poziomy (np. jednostkowe, integracyjne, systemowe i akceptacyjne) wraz z kryteriami przejścia między poszczególnymi poziomami (kryteria wejścia i wyjścia).
7. **Kryteria zaliczenia/niezaliczenia dla elementów testowych** – Określenie kryteriów, które pozwolą ocenić, czy dany element „przeszedł” testy, czyli został wykonany zgodnie ze specyfikacją i wymaganiami. Kryteria te należy określić dla każdego obiektu testów.
8. **Kryteria zawieszenia i wznowienia testowania** – Warunki i okoliczności powodujące podjęcie decyzji o czasowym zatrzymaniu realizacji wszystkich lub wybranych aktywności testowych zdefiniowanych w planie testów. Przykładowymi kryteriami zawieszenia dla testowania aplikacji może być *negatywny wynik „testów dymnych” dla wersji oprogramowania przekazanej do testów, czy liczba wykrytych usterek o wysokiej krytyczności uniemożliwiająca kontynuację prac*.
9. **Produkty prac (artefakty)** – lista artefaktów będących produktami procesu testowego, np.:  
   ■ *dokumentacja zarządcza* – plan testów, szczegółowe harmonogramy  
   ■ *specyfikacje testów* – przypadki testowe, procedury, raporty, skrypty  
   ■ *wyniki wykonywania testów*  
   ■ *dane testowe*  
   ■ *raporty z postępu realizacji testów*
10. **Czynności i zadania testowania** – w tej części planu należy zdefiniować wszystkie czynności testowe, np.:  
    ■ *planowanie testów*  
    ■ *projektowanie przypadków testowych* (manualnych, automatycznych)  
    ■ *przygotowanie danych testowych i środowisk testowych*, włączając w to instalację, konfigurację i naukę obsługi narzędzi wspierających  
    ■ *wykonanie testów*, z uwzględnieniem testów potwierdzających i regresyjnych  
    ■ *raportowanie wyników* wykonywania testów  
    ■ *monitorowanie i nadzór nad testami*  
    ■ *raportowanie postępu* testów  
    ■ *czynności na zakończenie testów*
11. **Środowiska testowe** – należy określić wymagania wobec środowiska testowego – czyli np.:  
    ■ *wymagania dotyczące sprzętu i oprogramowania* niezbędnego do realizacji procesu testowego  
    ■ *wymagania dotyczące konfiguracji*: stacji roboczych, serwerów, baz danych, systemów operacyjnych itd.  
    ■ *pożądane warunki pracy*, w tym powierzchnia biurowa, telefony, dostęp do Internetu
12. **Role i odpowiedzialności** – wskazanie osób odpowiedzialnych za realizację poszczególnych zadań testowych, stworzenie i utrzymywanie testaliów, konfigurację środowisk, przekazanie oprogramowania i innych produktów do testów. Informacje prezentowane w tym rozdziale można zawrzeć w części dotyczącej zadań testowych i produktów testowania, np. w postaci macierzy RACI przedstawiającej różne poziomy zaangażowania interesariuszy w tworzenie danego produktu czy realizację zadania. Warto pamiętać, że **macierz RACI** znana jest również jako macierz przypisania odpowiedzialności (ang. **responsibility assignment matrix, RAM**), a jej nazwa pochodzi od pierwszych liter angielskich terminów definiujących zaangażowanie różnych ról interesariuszy w określone zadania/produkty.  
    **RACI** oznacza:  
    ■ ***R (ang. responsible)*** – odpowiedzialny – interesariusz realizujący dane zadanie  
    ■ ***A (ang. accountable)*** – rozliczany (również osoby zatwierdzające lub organ dokonujący ostatecznego zatwierdzenia) – ten, który jest ostatecznie odpowiedzialny za prawidłowe i dokładne zakończenie zadania lub dostarczenie produktu oraz ten, który deleguje do pracy osoby odpowiedzialne  
    ■ ***C (ang. consulted)*** – konsultujący (czasami określany jako doradca) – interesariusze proszeni o opinie na temat realizowanej pracy/dostarczanego produktu są zazwyczaj ekspertami przedmiotowymi  
    ■ ***I (ang. informed)*** – informowany – osoba informowana o realizacji prac, aktualnych postępach lub jedynie o zakończeniu zadania lub produktu
13. **Potrzeby szkoleniowe, zapotrzebowanie na zasoby** – lista potrzeb szkoleniowych i kadrowych niezbędnych do realizacji poszczególnych prac testowych. Może przybrać formę opisu kompetencji koniecznych do pełnienia określonych ról testowych, ze szczególnym wskazaniem na kompetencje, których aktualnie brakuje w zespole, co przełoży się na konieczność kształcenia, szkoleń czy zatrudnienia specjalisty.
14. **Harmonogram** – terminy rozpoczęcia i zakończenia wykonywania poszczególnych zadań wraz z określeniem ich pracochłonności, kamieni milowych, głównych produktów prac oraz odpowiedzialności za realizację poszczególnych zadań.
15. **Ryzyka plany awaryjne** – w tej części planu należy określić zagrożenia, które mogą wystąpić podczas planowania i realizacji testowania – czynniki, których wystąpienie może mieć negatywny wpływ na powodzenie przedsięwzięcia testowego. Dla każdego istotnego ryzyka należy podać działania mające na celu łagodzenie ryzyka lub plany awaryjne na wypadek zaistnienia zdarzenia określonego jako ryzyko.
16. **Zatwierdzenie planu** – lista interesariuszy projektu, którzy muszą zatwierdzić plan, by wszedł on w życie.

## Zarządzanie defektami

### Jak prawidłowo opisać błąd?

Raport o defekcie w testach dynamicznych powinien zawierać:

* unikatowy identyfikator,
* tytuł i krótkie podsumowanie zgłaszanego defektu,
* datę zgłoszenia (= datę odkrycia błędu),
* autora,
* identyfikację elementu testowego (testowanego elementu, konfiguracji) i środowiska,
* fazę cyklu życia oprogramowania,
* opis defektu umożliwiający jego odtworzenie i usunięcie, a także dzienniki, zrzuty bazy danych, zrzuty ekranu, nagrania;
* wynik rzeczywisty i oczekiwany,
* priorytet usunięcia defektu,
* status zgłoszenia,
* opis niezgodności ułatwiający określenie jej przyczyny,
* określenie stopnia pilności rozwiązania,
* określenie elementu konfiguracji oprogramowania lub systemu,
* wnioski i zalecenia,
* historię zmian,
* odwołania do innych elementów, w tym do przypadku testowego, dzięki któremu problem został ujawniony

### Opisz cykl życia błędu

Obraz zawierający szkic, krąg, diagram, biały

Opis wygenerowany automatycznie

# Narzędzia wspomagające testowanie

# Inne pytania

### Co to jest sprint i ile może trwać?

Sprint to krótki, ograniczony czasowo okres, w trakcie którego zespół Scrum pracuje nad ukończeniem konkretnej ilości pracy. Powinien trwać maksymalnie miesiąc, jednak sugerowany czas to tydzień lub dwa.