

Zapewnienie jakości oprogramowania

wykład 1

Podstawy testowania

dr inż. Luiza Fabisiak



1. Podstawy testowania

2. Testowanie w cyklu życia oprogramowania
3. Techniki projektowania testów
4. Zarządzanie testowaniem
5. Testowanie wspierane narzędziami



1. Podstawy testowania

1. Dlaczego testowanie jest niezbędne?

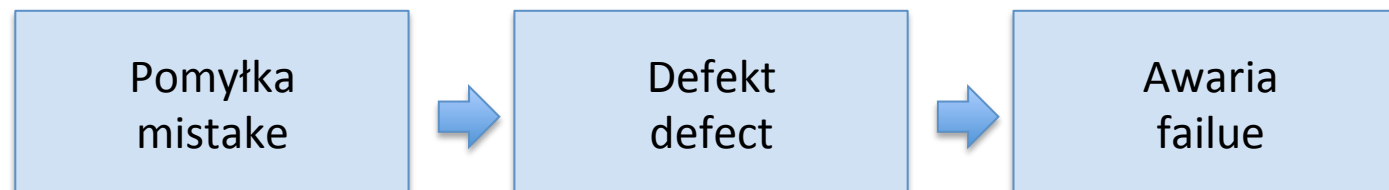
- Co to jest testowanie?
- Podstawowe zasady testowania
- Podstawowy proces testowy
- Psychologia testowania

1.1 Dlaczego testowanie jest niezbędne?



Podstawowe definicje:

- **pomyłka, błąd (mistake, error):** Działanie człowieka powodujące powstanie nieprawidłowego wyniku
- **defekt, usterka, pluskwa (defect, faul, bug):** Skutek błędu twórcy oprogramowania. Usterka może, ale nie musi spowodować awarii
- **awaria (failur, problem, incident):** Odchylenie od spodziewanego zachowania wyniku od działania

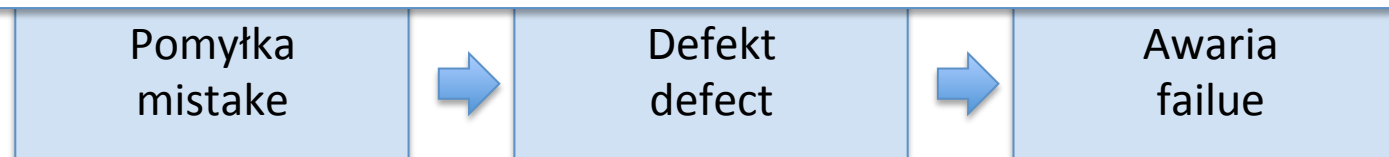


1.1 Dlaczego testowanie jest niezbędne?



Podstawowe definicje:

-
- **Tylko wspólne zaistnienie tych trzech czynników powoduje nieprawidłowe działanie Testowanego Produktu (TP)**
-



1.1 Dlaczego testowanie jest niezbędne?



Defekt (usterki) istnieją, ponieważ:

1. ludzie są z natury omylni;
2. wytwórcy działają:
 - Pod presją czasu;
 - Na skomplikowanym kodzie;
 - W skomplikowanej infrastrukturze;
 - Przy zmieniających się technologiach;
 - wytwórcy działają:
3. Jest wiele **interakcji** wewnątrz systemy i pomiędzy **współpracującymi** systemami.

1.1 Dlaczego testowanie jest niezbędne?



Awarie mogą być wywoływane także przez czynniki środowiska:

- promieniowanie;
- pola magnetyczne;
- pola elektryczne;
- skażenia.

Może to powodować **awarie** w oprogramowaniu wbudowanym lub wpływać na działanie oprogramowania przez **zmianę warunków** działania sprzętu

1.1 Dlaczego testowanie jest niezbędne?



- **Rygorystyczne** testowanie systemów o dokumentacji może zredukować ryzyko wystąpienia **awarii** w środowisku produkcyjnych i przyczynić się do osiągnięcia wysokiej jakości systemu.
- Konieczne jest, aby znalezione **defekty** były poprawione przed dopuszczenie systemu do działania w środowisku produkcyjnym

Uwaga: Testowanie oprogramowania może być również wymagane przez zapisy kontraktowe, wymogi prawne lub standardy przemysłowe

1.1 Dlaczego testowanie jest niezbędne?



Testowanie a jakość produktu (1/3)

- Testowanie umożliwia:
 - Określenie jakości produktu:
 - **Atrybuty funkcjonalne** (kolejne wykłady)
 - **Atrybuty нефunkcjonalne** (kolejne wykłady)
- oraz cechy systemu
 - Niezawodność,
 - Użyteczność,
 - Efektywność,
 - Utrzymywalność
- Testowanie podnosi zaufanie co do jakości oprogramowania, jeżeli znaleziono mało lub nie znaleziono błędów

1.1 Dlaczego testowanie jest niezbędne?



Testowanie a jakość produktu (2/3)

- Testowanie umożliwia:
 - **Poprawienie jakości**, gdy błędy są znalezione i usunięte;
 - **Zwiększenie** zaufania do produktu:

- Testowanie ułatwia **zapobieganie** awariom:
 - Prawidłowo zaprojektowany test, który przechodzi bez znalezienia błędu, redukuje poziom ryzyka w systemie;
 - Jeśli testowanie znajduje błędy i błędy te zostają naprawione, jakość systemu informatycznego wzrasta

1.1 Dlaczego testowanie jest niezbędne?



Testowanie a jakość produktu (3/3)

- Samo testowanie **nie podnosi** jakości oprogramowania i dokumentacji;
- Testowanie ułatwia zapobieganie awariom

ALE

- Testowanie jest tylko **jednym z wielu środków służącym** zapewnianiu jakości, inne to:
 - Standardy kodowania;
 - Szkolenia;
 - Analiza błędów.

1.1 Dlaczego testowanie jest niezbędne?



- Decyzja o tym, ile trzeba testować, powinna brać pod uwagę **poziom ryzyka**, włączając w to ryzyko techniczne, biznesowe i projektowe oraz ramy projektu takie jak czas i budżet
- Testowanie powinno dostarczyć interesariuszom informacji wystarczającej do podjęcia decyzji o:
 - Dopuszczaniu testowanego produktu do następnej fazy produkcji
 - Przekazania go klientowi.



1. Podstawy testowania

1. Dlaczego testowanie jest niezbędne?

- Co to jest testowanie?
- Podstawowe zasady testowania
- Podstawowy proces testowy
- Psychologia testowania

1.2 Co to jest testowanie?



Debugowanie, Wymaganie, Przegląd, Przypadek
testowy, Cel testu, Testowanie

1.2 Co to jest testowanie?



Testowanie to proces składający się z wszystkich czynności cyklu życia oprogramowania – zarówno **statycznych** jak i **dynamicznych**; skoncentrowany na planowaniu, przygotowaniu; ewaluacji oprogramowania oraz powiązanych produktów w celu określenia, czy spełniają one wyspecjalizowane wymagania

1.2 Co to jest testowanie?



Testowanie to technologiczne badanie pozwalające otrzymać informację o jakości Testowanego produktu (TP)

Technologiczne – bo używamy technologicznego podejścia, wykorzystujemy eksperymenty, matematykę, logikę, narzędzia (programy wspomagające), pomiary itp..
Badanie- bo zorganizowane poszukiwanie informacji

1.2 Co to jest testowanie?



Testowanie to:

- **Wykrywanie** podstawowych błędów w celu ich usunięcia;
- **Sprawdzenie** zgodności aplikacji z innymi aplikacjami;
- **Ułatwienie** interesariuszom podjęcie decyzji o wypuszczaniu/ niewypuszczaniu produktu
- **Minimalizacja kosztów** pomocy technicznej
- Kontrola **zgodności z wymaganiami** i uregulowaniami prawnymi.

1.2 Co to jest testowanie?



Testowanie obejmuje również:

- **Przeglądy** dokumentów i kodu (reviews)
- **Analizę** statystyczną

Podstawowe definicje:

- Testowanie dynamiczne

Testowanie, podczas którego wykonuj się kod modułu lub systemu

- Testowanie statyczne

Testowanie bez wykonywania kodu: techniki ręczne (przeglądy) zautomatyzowane (analiza statystyczna)

1.2 Co to jest testowanie?



Cele testowania:

- Określenie poziomu ryzyka związanego z Testowanym Produktem (TP);
- Znajdowanie defektów (poprzez uruchomienie TP i odnotowanie awarii);
- Ocena jakości TP (poprzez uruchamianie TP i wykonywanie scenariuszy testowych);
- Budowanie zaufania odnośnie jakości oraz dostarczanie informacji o jakości (poprzez uruchamianie TP i raportowanie wyników testów);
- Zapobieganie awariom w produkcji (poprzez uruchamianie TP i raportowanie postępu napraw);
- Zapobieganie powstawania błędów (poprzez analizę TP i dokumentacji)

1.2 Co to jest testowanie?



WERYFIKACJA

(verification)



WALIDACJA

(validation)

Proces kontroli systemu lub modułu polegający na sprawdzeniu, czy produkt danego etapu produkcji spełniają warunki zadane na początku tego etapu:

- Sprawdzenie, czy poprawnie zbudowano aplikację;
- Aplikacja jest zgodna z jej specyfikacją.

Określenie poprawności produktów procesu tworzenia oprogramowania pod względem potrzeb i wymagań użytkownika:

- Sprawdzenie, czy aplikacja jest poprawna;
- Sprawdzenie, czy spełnione są życzenia klienta.

1.2 Co to jest testowanie?



Walidacja i Weryfikacja – nie są etapami procesu testowania lecz są jego ogólnymi celami

Walidacja: [por. z ang. „*valid*”] = ważny, obowiązujący, przekonujący.

Weryfikacja: [por. z łac. „*verum*”] = prawda

1.2 Co to jest testowanie?



Czynności związane z testowaniem:

- Planowanie
- Nadzorowanie
- Ustalenie warunków testowych
- Projektowanie zadań testowych
- Porównywanie wyników
- Ocena kryteriów zakończenia
- Raporty dotyczące procesu testowego i testowanej aplikacji
- Zakończenie i zamykanie testów (po zakończeniu fazy testów)

Występują zarówno przed jak i po wykonaniu testów



Różne rodzaje testów i ich cele (1/2):

- **Testy modułowe (jednostkowe),**
integracyjne i systemowe (*module (unit, component), integration, system testing*)
 - Podstawowy cel: identyfikacja defektów w oprogramowaniu i ich poprawienie
- **Testy regresywne (regresyjne),** (*regression testing*)
 - Podstawowy cel: upewnienie się, że nie wprowadzono żadnych nowych błędów podczas implementacji zmian.
- **Testy potwierdzające (retesty),** (*re-testing*)
 - Podstawowy cel: potwierdzenie dokonania naprawy ponownym wykonaniem testu, który wykrył błąd.



Różne rodzaje testów i ich cele (2/2):

- **Testy produkcyjne**, (*operation tests*)
 - Podstawowy cel: ocena modułu lub systemu w jego środowisku produkcyjnym.
- **Testy pielęgnacyjne**, (*maintenance tests*)
 - Podstawowy cel: testowanie zmian w wdrożonym systemie, bądź wpływu zmienionego środowiska na wdrożony system
- **Testy akceptacyjne**, (*acceptance tests, UAT*)
 - Podstawowy cel: formalne potwierdzenie działania systemu zgodnie z oczekiwaniami i wymaganiami użytkowników



Debugowanie i testowanie to nie to samo:

- Testerzy testują a programiści debugują
- Testowanie ma ujawnić awarie spowodowane defektami

Debugowanie jest czynnością programistyczną wykonaną w celu zidentyfikowania przyczyny defektu, poprawienie kodu i sprawdzenia czy defekt został poprawnie naprawiony.

- Późniejsze testowanie potwierdzające wykonywanie przez testera upewnia, że poprawka rzeczywiście usunęła awarię.



1. Podstawy testowania

1. Dlaczego testowanie jest niezbędne?

- Co to jest testowanie?
- Podstawowe zasady testowania
- Podstawowy proces testowy
- Psychologia testowania



7 podstawowych zasad testowania:

- I. Testowanie ujawnia usterki (a nie ich nieobecność)
- II. Testowanie gruntowe jest niewykonalne
- III. Wczesne testowanie
- IV. Kumulowanie się błędów
- V. „Paradoks pestycydów”
- VI. Testowanie jest zależne od kontekstu
- VII. Mylne przekazanie o braku błędów



Testowanie ujawnia obecność defektów (a nie ich nieobecność)

- Testowanie może wykazywać, że TP zawiera usterki
- Testowanie nie może wykazywać, że TP nie zawiera usterek

Testując nie jesteśmy w stanie udowodnić, że system nie ma usterek. Jedynie, co możemy osiągnąć testując, to zmniejszyć prawdopodobieństwo, że w systemie pozostały niezidentyfikowane błędy

- Niewykrycie defektów nie jest dowodem na poprawność TP



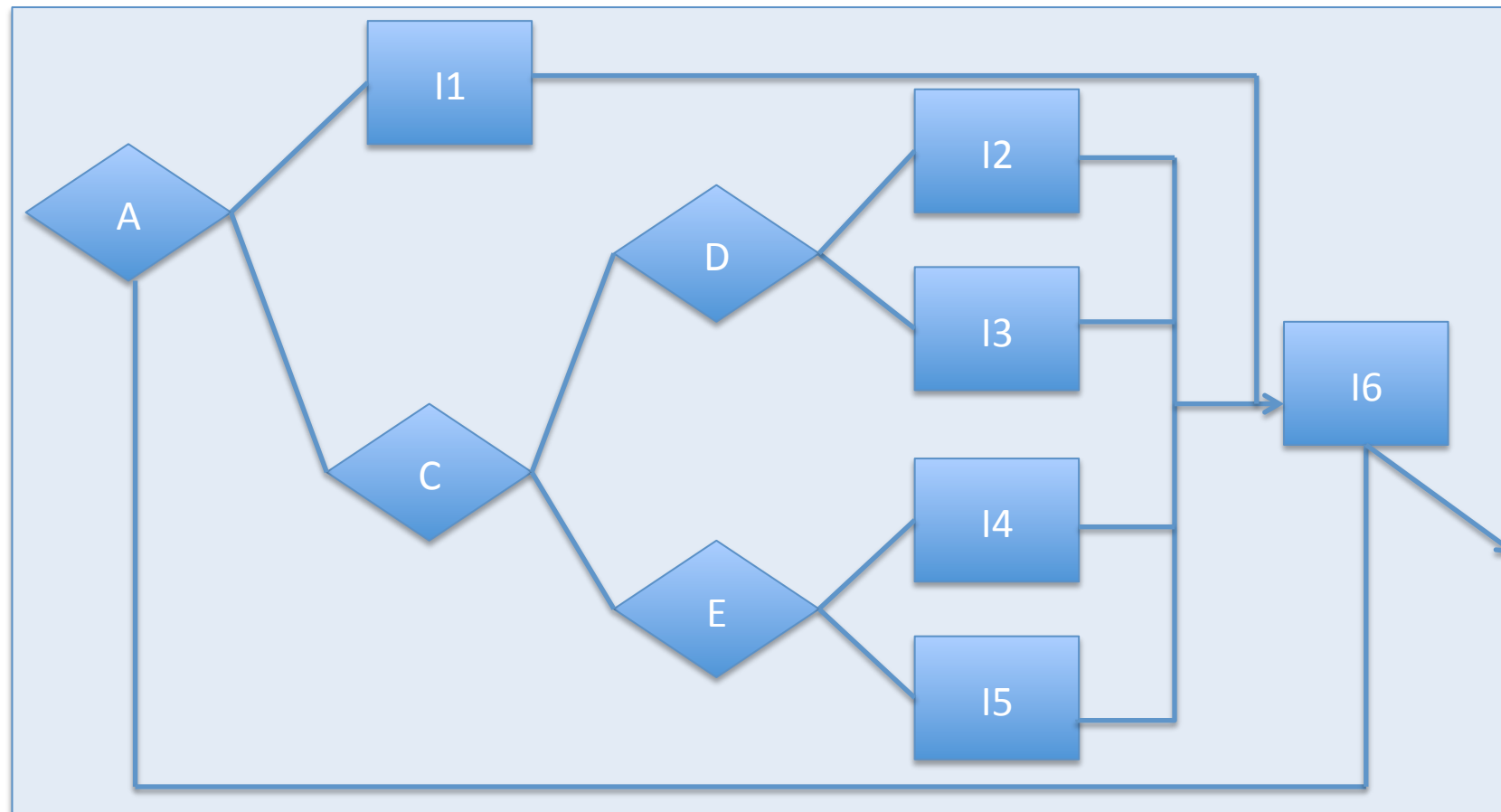
Testowanie gruntowe jest niewykonalne

W ramach testu kompletnego należy przetestować:

- Każdą możliwą wartość wejściową dla każdej zmiennej (włączając zmienne wynikowe i robocze)
- Każdą możliwą sekwencję wykonania programu;
- Każdą konfigurację sprzętu i oprogramowania włączając konfigurację np. serwerów – gdzie niczego nie możemy zmodyfikować;
- Wszystkie możliwe – ale na ogół niewyobrażalne użycia TP przez użytkownika końcowego



Testowanie gruntowe



1.3 Podstawowe zasady testow

W każdej pętli jest 5 ścieżek wykonania:

A I1 I6

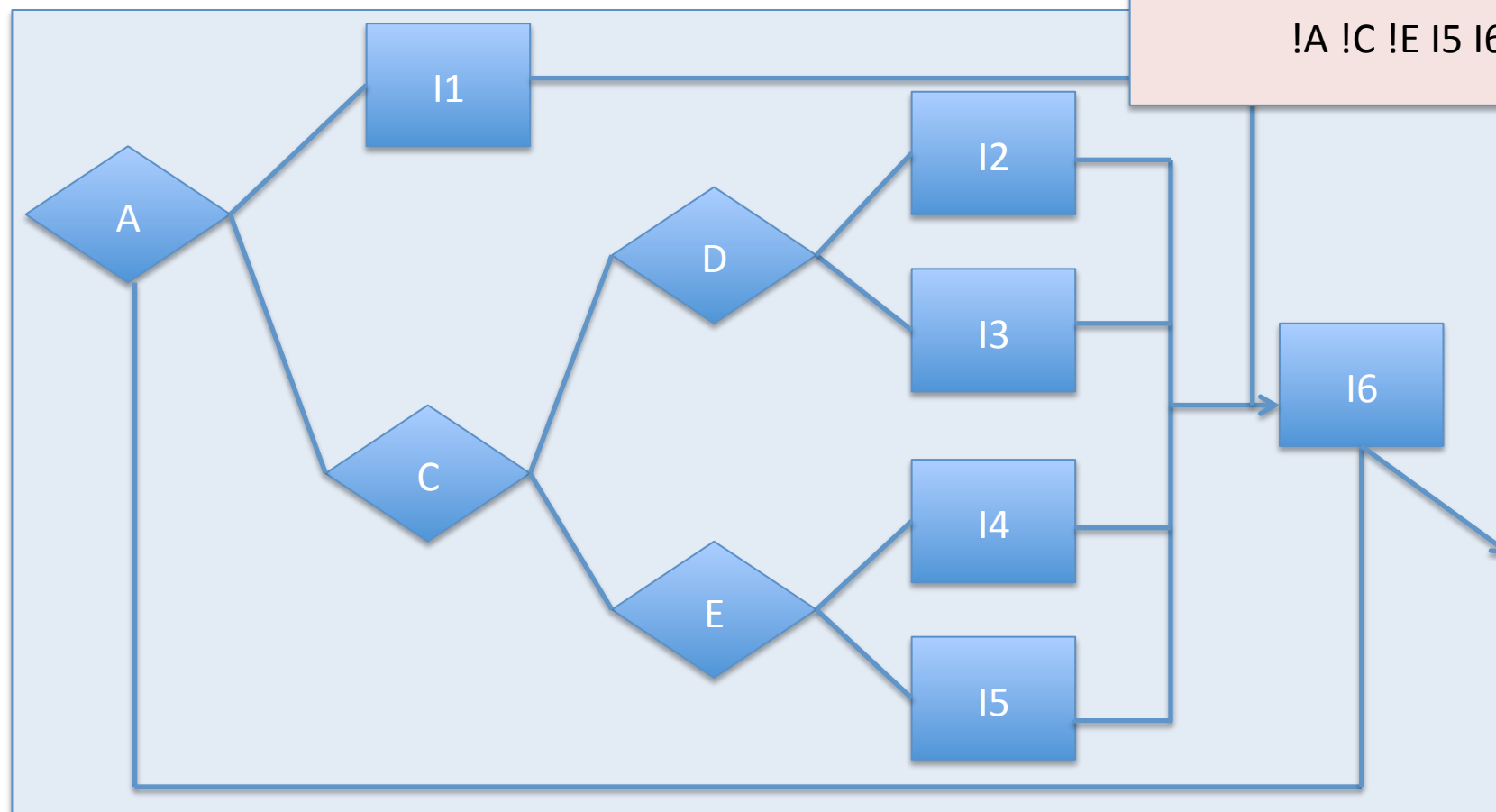
!A C D I2 I6

!A C !D I3 I6

!A !C E I4 I6

!A !C !E I5 I6

Testowanie gruntowe





Testowanie powinno zacząć się jak najwcześniej

Czynności testowe powinny rozpoczynać się:

- Jak najwcześniej (im szybciej wykryjemy błąd tym mniejszy będzie koszt jego usunięcia);
- Jak tylko jest możliwe w przypadku danego oprogramowania

1.3 Podstawowe zasady testowania



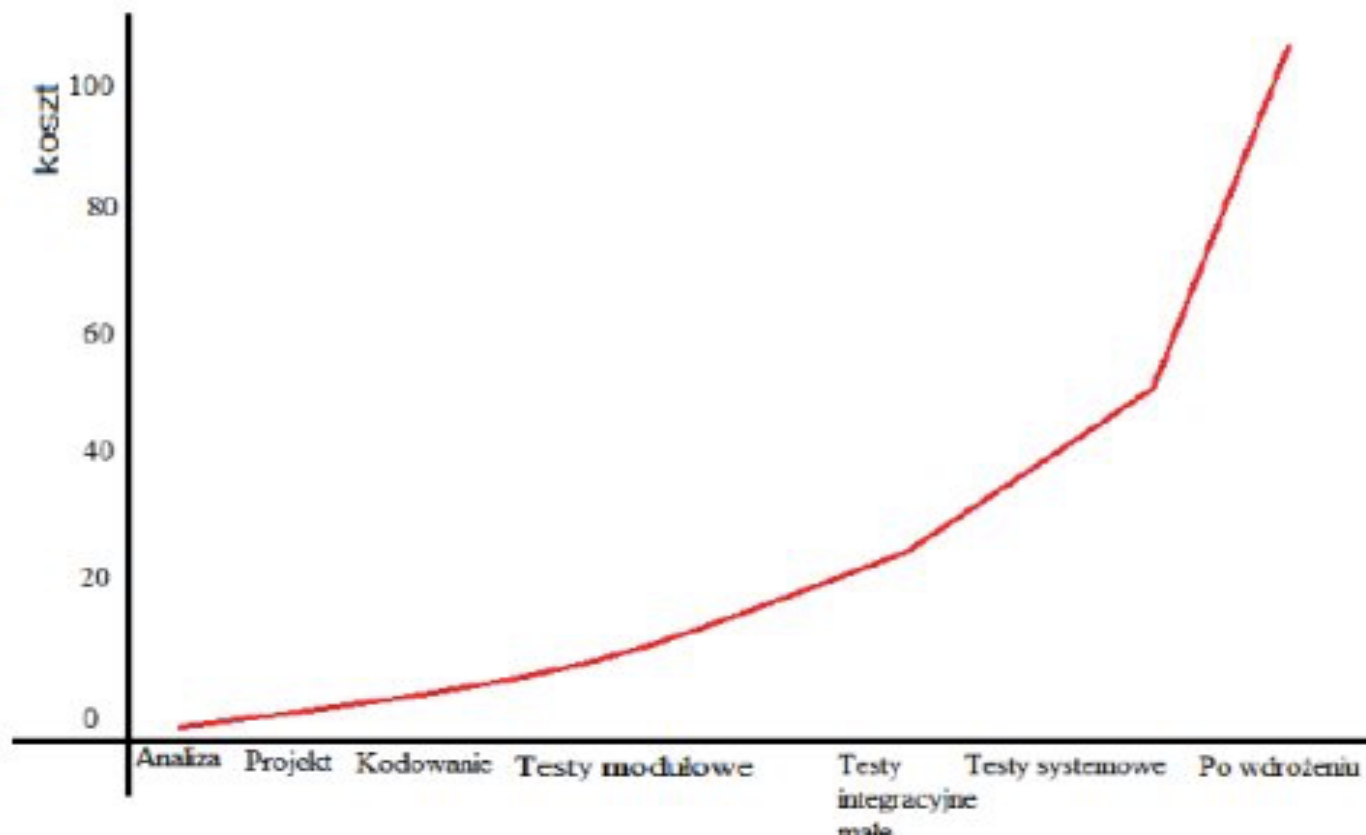
Testowanie powinno zacząć się jak najwcześniej

Faza	Koszt usunięcia błędu	
Analiza	1	6 min
Projekt	5	0,5h
Kodowanie	10	1h
Testy modułowe	15	1,5h
Testy integracyjne	22	2,2h
Testy systemowe	50	5h
Po wdrożeniu	100+	10h +

1.3 Podstawowe zasady testowania



Testowanie powinno zacząć się jak najwcześniej





Kumulowanie się defektów blisko siebie (A tendency to cluster together)

- Większość defektów znalezionych podczas testowania przed wypuszczeniem oprogramowania lub powodujących awarie produkcyjne znajdują się w małej liczbie modułów
- Błędy prawie nigdy nie są równo rozmieszczone
- Znalezienie błędu w danym module powinno skupić uwagę testera na tym module na dłużej



„Paradoks pestycydów”

- Jeśli te same rodzaje testów będą wielokrotnie powtarzane to w końcu ten sam zestaw przypadków testowych nie będzie już w stanie znaleźć żadnych nowych błędów.
- W celu uniknięcia wspomnianego paradoksu bardzo ważne jest, aby regularnie dokonywać przeglądu przypadków testowych i odpowiednio je modyfikować.



Testowanie jest zależne od kontekstu

Testowanie zawsze zależne jest od kontekstu czyli powinno być wykonywane w różny sposób w różnych sytuacjach.

Konieczne jest indywidualne podejście do każdego systemu. Na przykład inaczej będziemy testować

- systemy krytyczne (systemy od których zależy bezpieczeństwo i życie ludzkie),
- systemy bankowe
- gry komputerowe.



Mylne przekonanie o bezbłędności

Testowany produkt (TP) wolny od defektów może nie spełniać wymagań użytkownika:

- Poprawna weryfikacja
ALBO
- Niepoprawna walidacja

Konieczne jest testowanie w oparciu o wymagania – regularna kontrola spełniania wymagań



1. Podstawy testowania

1. Dlaczego testowanie jest niezbędne?

- Co to jest testowanie?
- Podstawowe zasady testowania
- Podstawowy proces testowy
- Psychologia testowania

1.4 Podstawowy proces testowy



podstawa testu, warunek testowy, pokrycie testowe,
dane testowe, wykonanie testów, log (dziennik) testowy,
plan testów, testowanie potwierdzające retesty,
testowanie potwierdzające kryterium zakończenia,
Incydent, Procedura testowa, polityka testów, zestaw
testów, strategia testów, sumaryczny raport z testów,
testalia

1.4 Podstawowy proces testowy



Proces testowy jest częścią procesu wytwórczego oprogramowania

Podlega ograniczeniom wynikającym
podporządkowania procesowi wytwórczemu


1.4 Podstawowy proces testowy



Proces testowy:

Może - ale nie musi – być zdefiniowany formalnie;

Składa się z minimum 5 czynności (faz)

- 
- Planowanie i nadzór
 - Analiza i projektowanie
 - Implementacja i wykonanie
 - Ocena stopnia spełnienia warunków zakończenia i raportowania
 - Czynności zamykające testy

W niektórych organizacjach proces jest bardziej złożony.
Fazy procesu testowego są logiczne sekwencyjne, w procesie wytwórczym mogą się zazębiać lub występować jednocześnie



PLANOWANIE I NADZÓR

- W ramach **planowania** określa się:
cel zakres testów, metodologia testów, strategia testów, zasoby
- Buduje się **harmonogramy** dla:
analizy testów, implementacji, wykonania, oceny testów)
- Dokonuje się **analizy ryzyka** związanego z testami
- Określa się **kryteria zakończenia**



PLANOWANIE I NADZÓR

- W ramach **nadzoru** określa się:

Mierzenie i analiza rezultatów

- Monitorowanie i dokumentowanie
- Postęp prac;
- Pokrycie testowe
- Kryterium zakończenia

Podejmowanie decyzji



ANALIZA I PROJEKTOWANIE

- Ogólne cele testowania zostają przekształcone w konkretne warunki testowe i przypadki testowe

GŁÓWNE ZADANIA:

- Zdefiniowanie i przeglądanie podstawy testów
- Identyfikacja i priorytetyzacja (warunki, wymagania testowe)
- Projektowanie i priorytetyzacja przypadków testowych wysokiego poziomu
- Ocena testowalności wymagań i systemu
- Projektowanie środowiska testowego
- Identyfikacja całej potrzebnej infrastruktury i narzędzi



IMPLEMENTACJA I WYKONANIE

- Warunki testowe (*test conditions*) są przekształcane:
 - W przypadki testowe (*test cases*)
 - Testalia (*test ware*)
- Tworzone jest środowisko testowe (*test environment*)



IMPLEMENTACJA I WYKONANIE

- Tworzenie **danych** testowych
- *Pisanie **procedur** testowych*
- *Przygotowanie **jarzma testowego** (test harness)* i pisanie skryptów automatyzujących testy (opcjonalnie);*
- *Tworzenie scenariuszy testowych na podstawie przypadków testowych celem efektywnego wykonania testu*
- *Weryfikacja poprawności utworzonego środowiska testowego*

Jarzmo testowe – środowisko testowe złożone z zaślepek i sterowników koniecznych do wykonania testów

**wykonanie przypadków testowych ręcznie lub przy pomocy narzędzie, w zaplanowanej kolejności*



OCENA STOPNIA SPEŁNIENIA KRYTERIÓW ZAKOŃCZENIA RAPORTOWANIA

- Wykonanie testu jest oceniane względem zdefiniowanych celów.
- *Ocena powinna być wykonywana dla każdego poziomu testów*
- *Oceniany jest stopień spełnienia kryteriów zakończenia*
- *Ocena składa się z następujących głównych czynności:*
 - *sprawdzenia dziennika testów pod względem warunków zakończenia określonych podczas planowania testów*
 - *Określenia (czy potrzebne jest więcej testów; czy należy zmieniać warunki zakończenia)*
 - *Napisanie końcowego raportu testowania dla interesariuszy*



CZYNNOŚCI ZAMYKAJĄCE TESTOWANIE

Zbierane są dane z zakończonych czynności testowych celem zgromadzenia doświadczenia faktów, testaliów i metryk

- *Czynności zamykające:*

Zamknięcie raportów incydentów lub zgłoszenie zmian do tych które zostały otwarte; wyliczenie potrzebnych metryk; udokumentowanie akceptacji systemu; zakończenie i archiwizacja; przekazanie testaliów zespołowi; analiza wniosków na potrzeby przyszłych projektów.



CZYNNOŚCI ZAMYKAJĄCE TESTOWANIE

Zbierane są dane z zakończonych czynności testowych, które stanowią podstawę do zgromadzenia danych do analizy.

- Zarządca projektu, który jest odpowiedzialny za zakończenie projektu, musi wykonać następujące czynności zamykające testy wykonywane są przy kamieniach milowych projektu takich jak: wydawanie systemu, zakończenie (lub anulowanie) projektu testowego, osiągnięcia kamienia milowego, lub zakończenia wydania serwisowego



1. Podstawy testowania

1. Dlaczego testowanie jest niezbędne?

- Co to jest testowanie?
- Podstawowe zasady testowania
- Podstawowy proces testowy
- Psychologia testowania



Cechy dobrego testera:

1. ciekawość;
2. „profesjonalny pesymizm”
3. Krytyczne spojrzenie;
4. Przywiązywanie wagi do szczegółów;
5. Czujność utrzymywana mimo monotonii testów;
6. Doświadczenie, na którym można oprzeć zgadywanie błędów;
7. Dobra komunikacja z programistami

1.5 Psychologia testowania



Kodeks etyczny:

- Kodeks etyczny ogłoszony przez ISTQB jest zgodny z kodeksem ACM i IEEE
- Głównym celem stworzenia kodeksu etycznego testerów jest możliwy ich dostęp do poufnych danych

Wykład opracowano na podstawie:



1. L. Rosenfeld, P. Morville, *Architektura informacji w serwisach internetowych*, Helion, Gliwice 2003.
2. J. Kalbach, *Projektowanie nawigacji strony WWW. Optymalizacja funkcjonalności witryny*, Helion, Gliwice 2008.
3. M. Pear, *Funkcjonalność serwisów internetowych*, PWN 2013.
4. L. Flordi, *The Blackwell Guide to the Philosophy of Computing and Information*, Blackwell 2002.
5. K. Visocky O'Grady, J. Visocky O'Grady, *The Information Design Handbook*, F+W Media 2008.
6. <http://www.merixstudio.pl/blog/7-zlotych-regul-testowania-stron-i-aplikacji-internetowych/> (dostęp 9.10.16)
7. <http://www.istqb.org/downloads/glossary-current.pdf>, słownik terminów, ISTQB Standard Glossary of Terms used in Software Testing V.2.0, Dec, 2nd 2007 (dostęp 1.10.15)
8. http://www.dmoz.org/Computers/Programming/Software_Testing/Products_and_Tools/, katalog narzędzi i produktów związanych z testowaniem oprogramowania (dostęp 1.10.15)
9. <http://www.nist.gov/director/prog-ofc/report02-3.pdf>, The Economic Impacts of Inadequate Infrastructure for Software Testing, NIST (dostęp 1.10.15)
10. <http://sunnyday.mit.edu/papers/therac.pdf>, artykuł o Therac 25 (błędy oprogramowania spowodowały śmierć pacjentów), Medical Devices: The Therac 25 – Nancy Leveson (dostęp 1.10.15)
11. <http://www.bcs.org/iseb>, <http://www.istqb.org>, strony organizacji certyfikujących w zakresie kompetencji związanych z testowaniem oprogramowania. (dostęp 1.10.15)

KONIEC

