Testowanie oprogramowania- Karol Kuźniak

Zad.1 Aktualne zachowanie oprogramowania można sprawdzić, iż jest poprawne i nie jest to błąd przez kilka sposobów:

1. Analizę statyczną lub dynamiczną

Testowanie statyczne

* odbywa się na etapie tworzenia programu (przed jego uruchomieniem).
* sprawdza poprawność kodu pod kontem *syntaktycznym* i *semantycznym*.

Testowanie dynamiczne.

* odbywa się po stworzeniu programu i jego uruchomieniu.
* sprawdza działanie programów w przypadkach szczególnych oraz jego ogólną funkcjonalność.

1. Metody czarnej i białej skrzynki

Metoda białej skrzynki.

* w ten sposób testujemy określone fragmenty programów w celu znalezienia przypadkach szczególnych, które powodują błędy.
* zajmujemy się jedynie fragmentami kodu w których występuje ryzyko wystąpienia takich przypadków pomijając te których poprawnego działania jesteśmy pewni.

Metoda czarnej skrzynki.

* odbywa się w oparciu jedynie o znajomośc interfejsów konkretnych modułów/funkcji programu lub interfejsu całego systemu oraz ogólnego przeznacznia testowanej aplikacji.
* testuje ogólną funkcjonalność i poprawność jednak cieżko jest w ten sposób wychwycić przypadki szczególne.

1. Metodologia wykonywania testów funkcjonalnych/strukturalnych
2. Testowanie na różnych etapach projektu

* Testowanie jednostek - sprawdzanie poprawności implementacji każdej jednostki programu (np. funkcji, metody, struktury danych).
* Testowanie zintegrowane - sprawdzanie poprawności komponentów (klasy, moduły) i zależności między nimi.
* Testowanie systemu - sprawdzenie aplikacji jako całość

1. Testowanie mutacyjne (testowanie testów)
2. Automatyzacja przeprowadzania testów.

Zad.2 Przy zgłaszaniu błędów należy zwrócić uwagę na to, iż:

* Raportowanie błędu musi być poprzedzone analizą i upewnieniem się, czy na pewno mamy do czynienia z błędem oprogramowania. Często mogą być raportowane błędy, które nie są błędami oprogramowania, które wynikają z niezrozumienia kontekstu.
* Jeden błąd to jeden raport.
* Bardzo ważne jest tytuł raportu, w którym zgłaszany jest błąd.
* Warto zapoznać się z wcześniejszymi zaraportowanymi błędami, aby nie raportować tych samych błędów.
* Ważne jest opisywanie błędów w jak najprostszych słowach, w których powinno zostać przekazane co się wydarzyło.
* Powinna zostać wskazana waga błędu i wersja oprogramowania

Zd.3 Jak postępować z informacją, że u mnie działa?

Kluczowym ogniwem w procesie tworzenia jest upewnienie się czy tester oprogramowania dąży krok po kroku zgodnie z wskazówkami w instrukcji i czy powiela ten sam problem w tym samym środowisku. Programista przez owe działanie może sprawdzić, czy niewielkie różnice, które mogą występować w wcześniej wskazanych źródłach (tzn. w innych środowiskach, inne podejście do wskazówek w instrukcji) może wpłynąć na finalny rezultat. Ważne jest także, aby tester oprogramowania testował programy na środowisku i na wersji oprogramowania, która najbardziej podobna jest do tej, w której pracuje jego klient.

Zad.4 Jakie wyróżniamy typy testów, czym się różnią?

STQB dzieli testowanie na funkcjonalne, testowanie charakterystyk niefunkcjonalnych jakości, testy strukturalne lub testowanie powiązane ze zmianami, np. potwierdzenie, że defekt został naprawiony (testy potwierdzające) i szukanie niepożądanych skutków zmian (testy regresji).

Testy funkcjonalne oparte są na tych funkcjonalnościach lub funkcjach (opisanych w dokumentacji lub zrozumiane przez testera) i mogą być wykonane na każdym poziomie testowania (np. testy komponentów oparte są na specyfikacji komponentów).Testowanie funkcjonalne analizuje zewnętrzne zachowanie oprogramowania, traktując ją jak czarną skrzynkę.  
  
Testowanie niefunkcjonalne - określanie parametrów. Testy niefunkcjonalne obejmują, ale nie są ograniczone do; testy wydajności, testy obciążeń, testy przeciążające, testy użyteczności, testy zdolności do operowania, testy utrzymaniowe, testy niezawodności i testy zdolności do pracy na różnych platformach. Ten typ testowania określa JAK system działa.Testy niefunkcjonalne mogą być wykonane na wszystkich poziomach testowych.

Testowanie strukturalne. Testy strukturalne mogą być wykonane na wszystkich poziomach testowych. Technik strukturalnych używa się w celu wsparcia pomiarów dokładności testowania poprzez ocenę pokrycia danego typu struktury. Pokrycie jest mierzone wykonaniem kodu za pomocą szeregu testów i wyrażone w procentach pokrytych elementów. Testowanie strukturalne może być oparte na architekturze systemu np. hierarchii wywołań. Testowanie strukturalne może również użyte na innych poziomach w tym dla systemu, integracji systemu i poziomu testów akceptacyjnych (np. modeli biznesowych lub struktury menu).  
  
Testowanie regresji/potwierdzające  
Kiedy defekt zostaje znaleziony i naprawiony wtedy oprogramowanie powinno być ponownie przetestowane, aby upewnić się, że defekt został usunięty. Taką czynność nazywamy testami potwierdzającymi. Testy regresji są powtarzalnymi testami na już przetestowanym wcześniej programie, po modyfikacjach, w celu wykrycia innych defektów wprowadzonych lub nie odkrytych podczas "naprawy". Mogą one znajdować się w testowanym oprogramowaniu jak i w innym powiązanym lub niepowiązanym z nim komponencie. Testy regresji zostają wykonane, gdy oprogramowanie lub środowisko, zostaje zmienione. Testy regresji mogą zostać wykonane na wszystkich poziomach testowych i zajmują się funkcjonalnością, parametrami niefunkcjonalnymi i testowaniem strukturalnym.

Zad.5 Od czego zależy ryzyko projektowe i co się na nie składa?

Analityk testów na za zadanie ocenę wpływu, a ocena możliwości wystąpienia błędu to główne praca należąca do technicznego analityka testów. Występują dwa ryzka, ryzyko w projekcie i ryzyku produktu. Ryzyko projektowe jest ryzykiem otaczającym zdolność projektu do dostarczenia określonych dla niego celów.

**Wyróżniamy:**

* czynniki dostawcy (niemożność dostarczenia produktu/podzespołu przez zewnętrzną grupę; czynniki kontraktowe)
* czynniki organizacyjne (brak umiejętności i ludzi; czynniki osobiste i treningi; czynniki polityczne takie jak problem z komunikacją, niemożność uczenia się na własnych błędach; niepoprawny odbiór lub oczekiwania względem testowania, np. niedocenianie wartości błędów znalezionych podczas testowania)
* czynniki techniczne (problem ze zdefiniowaniem właściwych wymagań; zakres wymagań, który może zostać osiągnięty dla istniejących ram; jakość projektów, kodu i testów).

Zad.6 Co zrobić gdy nie jesteśmy pewni czy mamy do czynienia z błędem?

Przy zgłaszaniu błędów dotyczących oprogramowania, ważne jest mieć na uwadze iż dobrym fundamentem jest posiadanie twardego punktu odniesienia, do którego można powołać się. Warto również w takich przypadkach zgłosić się do innych programistów lub osób, które działają nad danym oprogramowaniem w celu zasięgnięcia porady u kogoś kto zna się na danym programie. Ważne jest również zanotowanie „potencjalnego błędu”, mimo swoich niepewności.

Zad.7 Różnica między weryfikacją a walidacją

Weryfikacja – egzaminowanie poprawności i dostarczenie obiektywnego dowodu, że produkt procesu wytwarzania oprogramowania spełnia zdefiniowane wymagania.

Walidacja – sprawdzenie poprawności i dostarczenie obiektywnego dowodu, że produkt procesu wytwarzania oprogramowania spełnia potrzeby i wymagania użytkownika.

Z powyższych definicji możemy wysnuć proste założenie, że walidacja to spełnienie wymagań użytkownika. Weryfikacja to sprawdzenie, czy produkt jest zgodny z architekturą. Możemy wyróżnić dwa sposoby weryfikacji, statyczną i dynamiczną. Weryfikacja statyczna jest wykonywana przed skompilowaniem kodu i może nią być np. inspekcja kodu. Weryfikacja dynamiczna jest wykonywana na działającym oprogramowaniu, z używając danych testowych.

Zad.8 Czy tester jest odpowiedzialny za jakość oprogramowania?

Jednym z głównych zadań testera oprogramowania jest sprawdzenie czy w programie można znaleźć błędy. Jednak w 100% nie zgodziłbym, iż tester jest odpowiedzialny, gdyż, pomimo że wykrywa on błędy to jego zadaniem nie jest ich naprawa, a także nie podejmuje on decyzji związanych z wypuszczeniem danego oprogramowania na rynek. Pomimo tego, uważam, że za finalny produkt, który zostanie wypuszczony na rynek odpowiadają wszyscy, którzy brali w nim udział, także programiści i testerzy oprogramowania.

Zad.9 Kiedy należy skończyć testować oprogramowanie? Podaj przykłady

* kiedy jest wystarczające pokrycie wymagań – kiedy większość testów pokrywających wszystkie wymagania, zostały wykonane i były poprawne,
* zagęszczenie defektów - występuje nie więcej niż „x” defektów przypisanych do każdego z rozdziału wymagań,
* koszt – koszty mogą być również wyznacznikiem zakończenia testowania oprogramowania, gdyż dalsze testowanie może wiązać się z dalszymi kosztami, na które dana firma nie może sobie pozwolić
* ramy czasowe - testy są przewidziane na dany okres, bez możliwości wydłużenia terminu
* ryzyko sprowadzone jest do akcpetowalnego poziomu - znamy błędy występujące w oprogramowaniu i je akceptujemy

Zad 10. Jaka jest różnica pomiędzy firmware, a software

Software należy kojarzyć z czymś wirtualnym, może być skopiowany, usunięty, uruchomiony.

Ze względu na to, że nie jest to nic fizycznego, nie dotyczą go fizyczne ograniczenia związane z jego produkcją.Przykładami software’u mogą być:

* przeglądarki internetowe
* pakiet Microsoft Office
* Oprogramowanie antywirusowe

Firmware to taki specjalny rodzaj software’u dedykowany dla określonego hardware’u. Mimo że nie jest aż tak popularnym pojęciem jak hardware czy software, firmware jest wszędzie począwszy od mikrofalówki, pralki, a skończywszy na komputerach i smartfonach. Firmware ma bardzo wąskie przeznaczenie dla konkretnego typu hardware’u, zazwyczaj nie jest aktualizowany przez użytkownika końcowego, a raczej producenta, serwis. Firmware to oprogramowanie wbudowane i nie może być bezpośredio odinstalowane tak jak każde inne oprogramowanie. W przypadku komputerów firmwarem jest BIOS (Basic Input/Output System).

Zad. 11 Czym różni się tester manualny od automatyzującego?

Testowanie manualne bazuje głównie na zasobach ludzkich, przez co jest czasochłonne. Testowanie automatyczne pozwala na wykonywanie testów w krótszym czasie niż manualne. Większe ryzyko ludzkiego błędu jest przy testerze manualnym .Testowanie automatyczne jest przeprowadzane za pomocą skryptów i narzędzi, dzięki którym jest rzetelne i dokładniejsze, ale – jednocześnie – skrypty działają wyłącznie tak, jak zostały zaprojektowanie – mogą pominąć błędy, które są oczywiste dla doświadczonego testera manualnego. Podejście manualne pozwala dokładniej obserwować i ocenić, czy aplikacja jest łatwa w użyciu. Używa się również specjalistycznego oprogramowania/stron wspierających proces testowania i rejestrujących błędy. Proces testowania nie angażuje ludzkich umiejętności, przez co aplikacje mogą być mniej przyjazne użytkownikowi, a ocena może być niedokładna. Wyniki testów są zapisywane i łatwo dostępne z poziomu systemu do automatyzacji. Umiejętności programistyczne nie są wymagane w testowaniu manualnym. Znajomość języków programowania jest niezbędna przy tworzeniu skryptów z użyciem większości narzędzi. W testowaniu manualnym nie używa się frameworków, proces testowania obejmuje użycie list kontrolnych, reguł i procesów. Testowanie automatyczne oparte jest na frameworkach takich, jak: Selenide, Serenity, Data Driven, Hybrid, BDD.

Zad. 12 Co to jest ISTQB

International Software Testing Qualifications Board (ISTQB) to międzynarodowa organizacja non-profit, założona w 2002 roku, która opracowała najbardziej znaną obecnie ścieżkę certyfikacji dla testerów. Do połowy roku 2015 ISTQB wydała ponad 400 tysięcy różnych certyfikatów testerskich dla osób z ponad 100 krajów świata. Jednym z głównych celów stowarzyszenia jest promowanie wartości testowania oprogramowania. Kwalifikacje oparte są na programie nauczania, istnieje hierarchia kwalifikacji i wytycznych dotyczących akredytacji i egzaminów.  Co daje certyfikat?

* Przyswojenie podstawowej wiedzy z zakresu rzemiosła testerskiego i roli QA w rozwoju oprogramowania
* Świadectwo posiadania określonych kompetencji, co zwiększa możliwości u aktualnego pracodawcy i na rynku pracy
* Systematyzacja i ujednolicenie terminologii (choć bardziej w języku angielskim niż polskim)