Narzędzia wspomagające testowanie

1. Uwarunkowania związane z narzędziami testowymi

Kandydat potrafi sklasyfikować narzędzia testowe według przeznaczenia

i obsługiwanych czynności testowych. Kandydat potrafi zidentyfikować korzyści i czynniki ryzyka związane z automatyzacją testowania. Kandydat pamięta o szczególnych uwarunkowaniach związanych z narzędziami do wykonywania testów i zarządzania testami.

1. Dostępne m.in. narzędzia do testowania:
   1. Bezpośrednie narzędzia do testowania (programy komputerowe, mierniki, wzorce, pozimice itd.)
   2. Narzędzia do zarządzania procesem testowania – analizy wymagań, raportowania, zgłaszania defektów (Bugtracker)
   3. Narzędzia do analizy i oceny systemu testowanego
   4. Inne przydatne (szok że to wymienili w sylabusie😊)
2. Klasyfikacja narzędzi
   1. Zwiększające efektywność czynności testowych – narzędzia do automatyzacji testów (Jenkins, skrypty testowe, Selenium)
   2. Odpowiednie narzędzia przyspieszające testy manualne (dokumentacja, scenariusze, przygotowane i opisane narzędzia)
   3. Narzędzia umożliwiające jak najszybsze odwzorowanie awarii (środowiska testowe)
   4. Automatyzujące wszystko, czego nie da się zrobić ręcznie (np. symulowanie 100 000 zakupów w sklepie internetowym)
   5. Narzędzia do zwiększania niezawodności testowania (automatyzacja porównań – asercji, symulowanie zachowań – np. Selenium )
   6. Efekt próbnika – gdy narzędzie do testowania wpływa na testowaną rzecz (może być na teście)
   7. Niektóre narzędzia mają większą wartość dla Programistów (są wymienione w sylabusie i oznaczone literką D)
3. Narzędzia wspomagające zarządzaniem testowaniem i testaliami
   1. Narzędzia do zarządzania testami i cyklami życia (czyli etapami wytwarzania oprogramowania) – ALM (Application Lifecycle Management)
   2. Narzędzia do zarządzania wymaganiami (np. DOORS, narzędzie które służy do przechowywania i wyszukiwania wymagań)
   3. Narzędzia do zarządzania defektami (Bugtrackery, programy w których zapisane są Defekty, ich opis i aktualny status)
   4. Narzędzia do zarządzania konfiguracją(SCM) – programy do uruchamiania systemu w różnych konfiguracjach (np. program w tunelu aerodynamicznym symulującym pogodę – raz deszcz, raz wiatr, raz śnieg – i dla każdej z tych konfiguracji uruchamia te same testy – np. hamowanie, skręcanie itd.)
   5. Narzędzia wspomagające testowanie statyczne (Lintery, analizatory składni, analizatory martwego kodu, różne programy, które wczytują kod i analizują jego strukturę bez uruchamiania) – przydatne dla programistów
   6. Narzędzia do projektowania i implementacji testów
      1. Narzędzia do testowania opartego na modelu (modele zachowania, tablica stanów, modele niezawodności)
      2. Narzędzia do symulowania danych testowych (np. narzędzie symulujące dane matematyczne do tunelu aerodynamicznego)
   7. Narzędzia wspomagające wykonywanie i logowanie testów
      1. Logowanie – zbieranie tekstowego zapisu działania programu (np. wyciąganie szpuli z kasy fiskalnej z całego dnia – są one też formą logów)
      2. Wykonywanie testów i mierzenie pokrycia (np. JUnit i Eclemma uruchamiane w Jenkinsie – 2 programy które dostarczają metryk i informacji o kodzie)
      3. Jarzma testowe (Test harness) – środowisko przygotowane do wykonania testu (z zaślepkami – np. symulujemy ruch opony po śliskiej drodze symulując (zaślepiając) drogę poprzez ruch 2 wałków pod kołem (z punktu widzenia auta nie ma to znaczenia)
   8. Narzędzia wspomagające pomiar wydajności i analizę dynamiczną
      1. Testy wydajnościowe (obciążeniowe)
      2. Narzędzia do analizy dynamicznej
4. Korzyści w automatyzacji testów
   1. Minimalizujemy ilość pracy którą trzeba by było powtarzać ręcznie
   2. Bardziej spójne testy (błąd ludzki jest minimalizowany przy samym uruchomieniu)
   3. Obiektywne rezultaty testów (miary i pokrycie)
   4. Dostęp do danych statystycznych poprzednich testów
5. Czynniki ryzyka w automatyzacji testów
   1. Jesteśmy ograniczeni do narzędzia (Jenkins, symulatory)
   2. Czas wdrożenia automatyzacji na początku jest dużo dłuższy i droższy niż testy manualne
   3. Narzędzie automatyzujące czasami też narzuca opóźnienia i wymaga utrzymania (koszty utrzymania automatyzacji)
   4. Codziennie generowane raporty są trudne w zarządzaniu (za dużo raportów)
   5. Narzędzie automatyzujące często jest komercyjne i kosztuje
   6. Narzędzie morze nie wspierać nowych platform (np. Linuxa, macOS)
6. Szczególne informacje o narzędziach testowych
   1. Narzędzia do wykonywania testów
      1. Uruchamiają testy
      2. Aby je dobrze skonfigurować musimy odzwierciedlić ręczne wykonanie testu co bywa trudne
      3. Pisanie testów może być zależne od danych jakie dostają – czasami trudno jest to zamodelować
      4. Testowanie oparte na słowach kluczowych – skrypt testowy uruchamia kolejne małe skrypciki wykonujące poszczególne kroki testowe i zbiera ich cząstkowe wyniki
      5. Aby stosować powyższe języki należy dobrze znać dany język skryptowy
   2. Narzędzia do zarządzania testami
      1. Muszą móc się komunikować z innymi narzędziami (np. Excellem) aby powiązać wynik testu z wymaganiem
      2. Muszą zapewniać nam śledzenie (Traceability) testów z wymaganiami
      3. Muszą brać pod uwagę różne konfiguracje
      4. Kwestie te bierzemy pod uwagę mając duże zintegrowane narzędzia do kontroli całego projektu i jego cykli życia (np. JIRA)
7. Skuteczne korzystanie z narzędzi
8. Wybór narzędzi
   1. Oceniamy dojrzałość naszej firmy, dobre słabe strony
   2. Szukamy udoskonaleń które możemy osiągnąć narzędziami
   3. Zapoznajemy się z już używanymi technologiami testowymi (i programistycznymi)
   4. Rozumiemy jak działa nasze narzędzie do ciągłej integracji (CI – Continous Integration – np. Jenkins)
   5. Patrzymy na dostępny czas próbny używania nowego narzędzia
   6. Oceniamy dostawcę (czy to naprawdę trzeba pisać ??? Sylabusie??)
   7. Patrzymy czy mamy specjalistów od narzędzia czy musimy zamówić szkolenie
   8. Analizujemy wady i zalety open source/ komercyjne
   9. Szacujemy całkowite koszty wdrożenia
   10. Na końcu – przeprowadzamy dowód słuszności (PoC – proof of concept) by wykazać efektywność narzędzia
9. PoC – Proof of Concept (były pytania w B i C)
   1. Proof Of Concept – udowadniamy na małym wycinku programu czy cała idea nowego narzędzia ma sens i jest wykonalna
   2. Ma etapy
      1. Projekt pilotażowy który pozwala
         1. Zapoznać się z narzędziem
         2. Ustalić czy pasuje ono do naszej firmy i jej procesów
         3. Ustalić jak będziemy używać tego narzędzia w naszej firmie
         4. Oszacować korzyści czasowo -finansowe względem kosztów zakupu narzędzia
         5. Poznać jakie metryki generuje narzędzie i jak możemy je zbierać
10. Czynniki sukcesu dla narzędzi
    1. Przyrostowe wdrażanie narzędzia
    2. Narzędzie pozwala nam udoskonalać nasz proces
    3. Użytkownicy (Testerzy) mają dostęp do szkoleń z nowego narzędzia
    4. Ustalenie standardów i dobrych praktyk w korzystaniu z narzędzia
    5. Zintegrowanie narzędzia z innymi i zbieranie informacji zwracanych przez nie