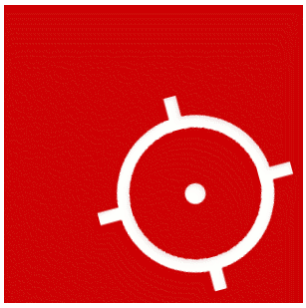

testerzy.pl

Podstawą dla tego rozdziału jest Foundation Level Syllabus wydany przez ISTQB

Automatyzacja

Narzędzia wspierające testowanie

Rodzaje narzędzi

Istnieje wiele narzędzi wspierających różne aspekty testowania. Narzędzia zostały podzielone zgodnie z obszarami, jakie wspierają. Niektóre z nich wspierają pojedyncze czynności, inne więcej niż tylko jedną – wtedy przypisane są do tej im najbliższej. Niektórzy komercyjni dostawcy dostarczają pełne środowiska testowe.
Narzędzia testowe mogą poprawić wydajność czynności testowych poprzez automatyzację powtarzalnych czynności. Mogą również znacznie poprawić wiarygodność testów przykładowo poprzez automatyzację porównań dużej ilości danych lub symulowanie pewnych zachowań.
Negatywne aspekty to przykładowo wpływ narzędzi testowych na wynik końcowy testu. Czas odpowiedzi środowiska może być różny w zależności od tego, jakich narzędzi do mierzenie wydajności się używa. Podobnie w przypadku pomiarów pokrycia kodu. Taki wpływ narzędzia nazywa się efektem sondy. Narzędzia oznaczone w poniższym tekście literką „D” służą przede wszystkim programistom.

Zarządzanie

Narzędzia zarządzające testowaniem mogą zostać zastosowane w każdej fazie tworzenia oprogramowania.
Na charakterystykę narzędzi składa się:

- wsparcie zarządzania testami oraz prowadzonych czynności testowych
- interfejs do narzędzi wykonujących testy, tropiących defekty oraz zarządzających wymaganiami
- niezależne zarządzanie wersjami lub interfejs do zewnętrznego narzędzia zarządzającego konfiguracją
- wspieranie śledzenia testów, rezultatów testów oraz zmian w dokumentach źródłowych takich jak specyfikacja wymagań
- zapisywanie rezultatów testów i generowanie raportów z postępów
- analiza ilościowa (metryki) wynikająca z zagadnień testowych (np cykle testowe, ilość przypadków testowych z pozytywnym wynikiem) oraz badanego obiektu (analizowanych przypadków), w celu przekazania informacji na temat testowanego obiektu oraz kontroli i poprawy procesu testowego.

Narzędzia wspierające zarządzanie wymaganiami

Narzędzia te przechowują deklarację wymagań, sprawdzają logikę i niezdefiniowane (brakujące) wymagania, pomagają definiować priorytety wymagań i wspierają śledzenie poszczególnych testów przez pryzmat wymagań, funkcji i/lub możliwości. Zdolność do śledzenia może być raportowana w dokumencie opisującym postęp testowy. Pokrycie wymagań, funkcji i/lub możliwości również może zostać zaraportowane.

Narzędzia wspierające zarządzanie przypadkami

Narzędzia te pomagają przechowywać i zarządzać raportami z przypadków takich jak defekty, problemy z oprogramowaniem spowodowane przez błędny kod oraz przewidywać problemy i anomalie. Wspomagają zarządzanie przypadkami poprzez:

- ułatwiają ustalanie ważności

- przypisanie akcji do pracownika (np. Naprawę błędu lub wykonanie testów potwierdzających)
- opis atrybutów statusu (np. Odrzucony, gotowy do testów)

Narzędzia wspierające zarządzanie przypadkami monitorują postęp rozwiązywania przypadków, dostarczają statyczną analizę i raporty o przypadkach. Często nazywane są Narzędziami do śledzenia defektów.

Narzędzia wspierające zarządzanie konfiguracją

Precyzyjnie mówiąc nie są to narzędzia testowe, ale są one niezbędne do śledzenia różnych wersji oprogramowania będącego testowanym. Mogą one:

- przechowywać informacje o wersjach oprogramowania lub narzędzi testowych
- umożliwić śledzenie pomiędzy narzędziami testowymi i wariantami produktów
- pomóc w przypadku rozwoju oprogramowania na więcej niż jednym środowisku sprzętowym/informatycznym (np. różne systemy operacyjne, biblioteki, kompilatory, przeglądarki, komputery) .

Statyczne testowanie

Narzędzia wspierające proces przeglądu

Narzędzia tego typu przechowują informacje na temat procesu przeglądu, komentarzy będących wynikiem przeglądu, raportów z defektów i poświęconego wysiłku, referencji do zasad przeglądów i/lub listę punktów kontrolnych. Utrzymują zdolność śledzenia pomiędzy dokumentacja a kodem źródłowym.
Narzędzia wspierające proces przeglądu mogą również dostarczać pomoc do przeglądu dokumentacji współdzielonej przez oddziały będące ulokowane w różnych miejscach geograficznych.

Narzędzia analizy statycznej (D)

Narzędzia te wspierają programistów, testów i personel odpowiedzialny za zapewnienie jakości w znajdywaniu defektów zanim nastąpi faza dynamicznych testów. Główne cele to:

- wprowadzanie w życie standardów tworzenia oprogramowania
- analiza struktur i zależności (np. linków na stronach internetowych)
- pomoc w zrozumieniu kodu źródłowego.

Narzędzia modelujące (D)

Narzędzia modelujące są zdolne do walidacji modeli oprogramowania. Przykładowo: aplikacja sprawdzająca model bazy danych może
znaleźć defekty i nielogiczność w modelu danych; inne narzędzia modelujące mogą znaleźć defekty w modelu stanów lub w modelach
obiektów. Narzędzia te mogą często pomagać generować niektóre przypadki testowe bazując na modelu.

Specyfikacja testowa

Narzędzia projektowania testów

Narzędzia te generują dane wejściowe lub właściwe testy bazując na wymaganiach z graficznego interfejsu użytkownika lub z zaprojektowanych modeli (stanów, danych lub obiektów). Tego typu narzędzia definiują oczekiwany rezultat testu. Testy wygenerowane z modelu stanów lub obiektów mogą służyć weryfikacji implementacji modelu w oprogramowaniu, ale rzadko kiedy są wystarczające dla weryfikacji wszystkich aspektów oprogramowania lub systemu. Mogą one oszczędzać cenny czas i polepszać dokładność testowania, ponieważ zapewniają kompletność testów mogących być stworzonymi przez to narzędzie.
Inne narzędzia z tej kategorii mogą pomagać w wsparciu tworzenia testów poprzez dostarczanie ustrukturyzowych wzorców, nazywanych czasami ramami testów, służących do generowania testów i przyspieszania procesu tworzenia testów.

Narzędzia przygotowania danych testowych

Obsługują one bazy danych, pliki lub transmisję danych by przygotować dane testowe używane podczas wykonywania testów. Zaletą tych narzędzi jest

zapewnienie, że dane będą transferowane w czasie rzeczywistym do środowiska testowego z zachowaniem ich pełnej anonimowości, co z kolei zapewnia ich ochronę.

Wykonanie i rejestrowanie

Narzędzia wykonujące testy

Narzędzia te służą do automatycznego lub pół-automatycznego wykonywania przypadków testowych. Posiadając dane wejściowe i informację o oczekiwanym rezultacie końcowym testy wykonuje się bezpośrednio używając zazwyczaj języków skryptowych. Języki skryptowe umożliwiają obsługę testów przy minimalnym nakładzie sił, na przykład, gdy musimy powtórzyć ten sam test mając różne dane lub przetestować inną część systemu używając tej samej procedury. Zazwyczaj narzędzia wykonujące testy mają wbudowane moduły dynamicznego porównania danych oraz rejestrowania wyników.
Tych narzędzi możemy używać również do „nagrywania" testów poprzez moduły narzędziowe „przechwyć i odtwórz". Zapamiętywanie przebiegu testów podczas badań oraz ich wykonywania bez skryptów, może być przydatne do zreprodukowania i/lub udokumentowania testów, gdy na przykład defekt spowoduje problemy.

Symulatory testowe/narzędzia testowania komponentów (D)

Symulatory mogą ułatwić testowanie komponentów lub części systemu symulując środowisko, w którym testowany obiekt będzie pracował. Narzędzia te można stosować, kiedy inne komponenty tego systemu nie zostały jeszcze dostępny i zostaje on zastąpiony przez sterowniki i zaślepki ("stub" słowa angielskie – nieprzetłumaczalne, oznaczające szkieletowe lub specjalnie spreparowane oprogramowanie używane do rozbudowy lub testowania komponentu, który wywołują lub, od którego jest zależny; zastępuje wywoływany komponent). Mogą również służyć do dostarczenia przewidywalnego i kontrolowalnego środowiska, w którym błędy mogą zostać zlokalizowane.
Szkielet może zostać stworzony, gdy część kodu źródłowego, obiektu, metody lub funkcji, podstawowego elementu systemu lub komponentu może zostać wykonana poprzez wywołanie testowego obiektu i/lub dostarczyć informacji zwrotnej do tego obiektu. Może to być zrobione poprzez dostarczenie sztucznie spreparowanych danych wejściowych do testowego obiektu i/lub wstawić zaślepkę w miejsce rzeczywistego obiektu, dla uzyskania danych wyjściowych.
Symulatory mogą służyć także do dostarczania struktur wykonujących testy dla półproduktów gdzie testujemy języki, systemy operacyjne lub osprzęt muszą być testowane wspólnie. Mogą być nazywane narzędziami do testowania komponentów, gdy ich celem jest wykonywanie testów komponentowych równolegle z tworzeniem kodu źródłowego.

Komparatory testowe

Narzędzia używane do określenia różnic pomiędzy plikami, bazami danych oraz rezultatami testów. Są często częścią narzędzi do automatycznego wykonywania testów, ale stanowią również osobną grupę testów. Komparator może używać wróżki testowej szczególnie, gdy jest ona zautomatyzowana.

Narzędzia mierzące stopień pokrycie (D)

Narzędzia te mogą mieć wpływ lub, niemieć wpływu na wyniki, zależnie od użytych technik pomiaru, od tego, co jest mierzone i jaki język programistyczny jest używany. Narzędzia mierzące stopień pokrycia kodu źródłowego wskazują, jaki procent struktury kodu zostały sprawdzone (np. Deklaracji, gałęzi, decyzji, modułów lub wywołań funkcji).

Narzędzia sprawdzające bezpieczeństwo

Narzędzia te sprawdzają poziom zabezpieczeń przed wirusami komputerowymi lub atakami hakerów. Przykładowo firewall nie jest narzędziem stricte testowym, ale może zostać użyty do tego typu testów. Inne narzędzia sprawdzają funkcjonowanie systemu pod wpływem dużego obciążenia, szukając w nim słabych punktów.

Wydajność i monitorowanie

Narzędzia analizy dynamicznej (D)

Podczas analizy dynamicznej wykrywane są te błędy, które można znaleźć jedynie poprzez wykonanie programu tj. zależności czasowe czy wycieki pamięci.

Używamy ich zazwyczaj podczas testów komponentowych, integracyjnych i testów półproduktów.

Wydajność/obciążenie/stres

Narzędzia wydajnościowe monitorują i raportują jak system zachowuje się w różnych symulowanych warunkach. Symulują obciążenie aplikacji, bazy danych lub systemów takich jak sieci lub serwery. Narzędzia noszą swoje nazwy zgodnie z wydajnością, jaką mierzą, czyli obciążeniowe lub stresowe. Często są zautomatyzowanymi powtarzalnymi procedurami testowymi, opartymi na zadanych parametrach.

Monitory

Narzędzia monitorujące nie są narzędziami testowymi, ale mogą dostarczać informacji, które nie są dostępne w inny sposób.
Narzędzia te analizują, weryfikują i raportują użycie specyficznych zasobów systemu. Mogą również ostrzegać o potencjalnych problemach w serwisach. Przechowują informację o wersji oprogramowania lub środowiska testowego i umożliwiają odtwarzanie tych informacji.

Inne

Poszczególne przykłady tego typu narzędzi mogą być wyszczególnione w zależności od aplikacji, jakie testują np. istnieją specjalne narzędzia wydajnościowe do testowania aplikacji sieciowych, narzędzia analizy statycznej dla różnych platform sprzętowych czy narzędzia analizy dynamicznej dla sprawdzenia aspektów bezpieczeństwa. W tym obszarze możemy również uwzględnić narzędzia nie-testowe, ale używane przez testerów takie jak formularze, SQL, debuggery.

Efektywne użycie narzędzi

Kupienie lub leasingowanie narzędzi nie gwarantuje natychmiastowego sukcesu. Każde narzędzie wymaga dodatkowych nakładów by osiągało pokładane w nim nadzieje i dawało trwały efekt. Należy uważać by oprócz analizy zysków sprawdzić także potencjalne ryzyko.
Zalety używania narzędzi:

- powtarzalna praca może zostać zredukowana do minimum (np. Testy regresji, wprowadzanie tych samych danych wejściowych czy sprawdzanie standardów kodowania)
- większa powtarzalność i logiczność (np. Testy wykonywane przez narzędzia czy testy otrzymywane bezpośrednio z wymagań)
- obiektywne szacowanie (np. statyczne miary, pokrycie i zachowanie systemu)
- ułatwiony dostęp do informacji o testach i testowaniu (np. statystyki i grafy postępu testów, ilość błędów czy wydajność)

Niebezpieczeństwa:

- nierealne oczekiwania względem narzędzi, które się w większości nie spełniają (włączając w to funkcjonalność i ułatwienia testowania)
- niedoszacowanie czasu, wydatków i wysiłku, jaki trzeba poświęcić, aby wprowadzić narzędzie do organizacji (włączając w to treningi czy zewnętrznych ekspertów)
- niedoszacowanie czasu potrzebnego na osiągnięcie widocznych postępów wynikających z wprowadzania narzędzia (włączając w to konieczność zmian procesów i ciągłego wprowadzania poprawek w sposób użycia narzędzia)
- niedoszacowanie wysiłku potrzebnego do analizy wyników wypływających z narzędzia testowego
- zbytnie zaufanie do narzędzia (w szczególności w miejscach gdzie manualne testowanie mogłoby by być bardziej korzystne).

Narzędzia wymagające specjalnej uwagi

Narzędzia wykonujące testy

Tego typu narzędzia powtarzają skrypty odtwarzające elektronicznie przechowywane testy. Wymagają one dużego nakładu sił, aby osiągnąć znaczne korzyści.
Założenia zarządzania danymi wymaga oddzielenia danych wejściowych (zazwyczaj w formularzach) i używania ogólnych skryptów mogących odczytywać dane testowe by wykonywać te same testy z różnymi danymi. Testerzy nie znający języków skryptowych mogą wprowadzać dane wejściowe do tych predefiniowanych skryptów.
W założeniach zarządzania słowami kluczowymi zawarte są formularze zawierające słowa kluczowe opisujące akcję, jaka musi być podjęta oraz dane testowe. Testerzy (nie znający języków skryptowych) mogą definiować testy poprzez użycie słów kluczowych, które mogą być dopasowane do testowanej aplikacji.

Techniczne ekspertyzy w językach skryptowych są potrzebne dla oby dwóch założeń (zarówno dla testerów, jaki i specjalistów automatyzacji).
Bez względu, która technika skryptowa jest używana, oczekiwany rezultat dla każdego testu musi być przechowywany dla późniejszych porównań.

Narzędzia wydajnościowe

Wymagają one eksperta potrafiącego wspierać ich projektowanie i interpretującego ich wyniki.

Narzędzia analizy statycznej

Narzędzia te używane do kodu źródłowego mogą wymuszać stosowanie standardów tworzenia kodu, jeśli jednak analizują istniejący kod mogą wygenerować długą listę ostrzeżeń. Takie wiadomości nie blokują kodu przed pełnym skompilowaniem się (przetłumaczeniem kodu na wykonywalny program), ale informują i adresują problemy, których rozwiązanie może w przyszłości zaowocować łatwiejszym utrzymaniem kodu. Implementacja filtrująca problemy wykluczy niektóre ostrzeżenia w bardzo efektywny sposób.

Narzędzia zarządzające

Dla najlepszego użycia narzędzi zarządzających muszą one być wyposażone w interfejsy do z innymi narzędziami testowymi oraz z formularzami. Pomaga to stworzyć w organizacji najlepszy format raportów. Muszą one być zaprojektowane i monitorowane tak by przynosiły korzyści.

Wprowadzenie narzędzi do organizacji

Główne zasady wprowadzania narzędzi do organizacji:

- ocena dojrzałości organizacji, jej słabe i silne punkty oraz identyfikacja szans do poprawy procesu testowego wspieranego przez narzędzia
- sprawdzenie czytelności wymagań i obiektywnych kryteriów
- dowodzenie przydatności testowej dla konkretnej funkcjonalności i zdeterminowanie czy produkt wypełnia założenia
- sprawdzenie dostawcy i jego oferty (treningi, wsparcie techniczne i aspekty komercyjne)
- identyfikacja wewnętrznych potrzeb dla treningów i szkoleń mentorskich w użyciu narzędzia.

Dowodzenie przydatności może odbywać się w małej skali jako projekt pilotowy. Dzięki temu minimalizuje się negatywny wpływ na projekt, gdy pojawią się poważne przeszkody dla wprowadzanie narzędzia.

Celem projektu pilotowego jest:

- nauczenie się szczegółów narzędziami
- sprawdzanie jak narzędzie pasuje do istniejącego procesu i praktyk lub jak powinny one ulec modyfikacji.
- Decyzja w sprawie standardów użycia narzędzia, zarządzania nim, przechowywania i zarządzanie samym narzędziem i jego wynikami (np. konwencja nazywania plików i testów, stworzenie bibliotek i zdefiniowanie modularności zestawów testowych)
- ocena korzyści płynących z narzędzia w porównaniu z nakładami.

Sukces wprowadzenia narzędzia do organizacji pociąga za sobą:

- rozwój narzędzia i jego implementację w reszcie organizacji
- adaptowanie i poprawiania procesów pod kątem narzędziami
- dostarczenie treningów do organizacji
- zdefiniowanie przewodnika jak używać narzędzia
- implementacja metody „wyciągniętych lekcji” podczas użycia narzędzi
- monitorowanie użycia narzędzia.

Narzędzia wspierające testowanie - przykłady

Niekomercyjne

- Bugzilla

Komercyjne

- Rational – IBM
  - ClearQuest
  - ClearCase
- Mercury
  - TestDirector
  - LoadRunner
  - WinRunner