



Certyfikowany Tester

Sylabus poziomu zaawansowanego Kierownik Testów

Wersja 2012.1

Stowarzyszenie Jakości Systemów Informatycznych

International Software Testing Qualifications Board®



Informacje o prawach autorskich

Wszelkie prawa dla wersji angielskiej zastrzeżone dla © International Software Testing Qualifications Board (dalej nazywane ISTQB®).

Prawa autorskie do niniejszego dokumentu zastrzeżone dla © Stowarzyszenie Jakości Systemów Informatycznych (SJSI).

Dokument niniejszy może być kopiowany w całości lub publikowany w wybranych fragmentach z podaniem źródła.

Advanced Level Test Manager Sub Working Group: Rex Black (Chair), Judy McKay (Vice Chair), Graham Bath, Debra Friedenberg, Bernard Homès, Kenji Onishi, Mike Smith, Geoff Thompson, Tsuyoshi Yumoto; 2010-2012.

Tłumaczenie z języka angielskiego oraz udział w przeglądach: Elżbieta Pawełek-Lubera, Tomasz Lubera, Karolina Zmitrowicz, Radosław Smilgin, Jan Sabak, Lucjan Stapp, Joanna Kazun, Tomasz Watras.

Przegląd dokumentu do drugiej publikacji: Monika Petri-Starego, Lucjan Stapp, Joanna Kazun.

Historia zmian

Wersja	Data	Uwagi
ISEB v1.1.	04 września 2001	ISEB Practitioner Sylabus
ISTQB® 1.2E	wrzesień 2003	ISTQB® Sylabus poziomu zaawansowanego od EOQ-SG
V2007	12 października 2007	Certyfikowany tester Poziom zaawansowany program 2007
D100626	26 czerwca 2010	Włączenie zmian przyjętych w 2009 roku, oddzielenie poszczególnych rozdziałów do oddzielnych modułów
D101227	27 grudnia 2010	Akceptacja zmian formatu i poprawek stylistycznych
D2011	31 października 2011	Podział sylabusu, przerobienie efekty kształcenia i zmiany tekstu, aby dopasować efekty uczenia się. Dodanie BO
Alpha 2012	09 lutego 2012	Uwzględnienie wszystkich uwag od NB otrzymanych od wydania październikowego
Beta 2012	26 marca 2012	Włączenie uwag od NB otrzymanych na czas wydania Alpha
Beta 2012	07 kwietnia 2012	Wersja beta zatwierdzona przez GA
Beta 2012	08 czerwca 2012	Przekazanie kopi wersji do NB
Beta 2012	27 czerwca 2012	Włączenie uwagi EWG i dodanie Słownika
RC 2012	15 sierpnia 2012	Finalne zmiany NB - Kandydat do publikacji
GA 2012	19 października 2012	Ostateczne poprawki do wydania GA
PL 2013	14 października 2013	Wersja dokumentu do pierwszej publikacji
PL 2020	06 czerwca 2020	Przegląd dokumentu do drugiej publikacji

Spis treści

Historia zmian	3
Spis treści	4
Podziękowania	6
0. Wprowadzenie do Sylabusu	7
0.1. Cel niniejszego dokumentu	7
0.2. Opis	7
0.3. Cele nauczania podlegające egzaminowi	7
1. Proces testowy – 420 min.	8
1.1. Wstęp	9
1.2. Planowanie testów, monitorowanie i nadzór nad testami	9
1.2.1. Planowanie testów	9
1.2.2. Monitorowanie testów i nadzór nad testami	11
1.3. Analiza testów	12
1.4. Projektowanie testów	13
1.5. Implementacja testów	14
1.6. Wykonywanie testów	15
1.7. Ocena kryteriów zakończenia i raportowanie	15
1.8. Czynności zamykające testy	16
2. Zarządzanie testami – 750 min.	18
2.1. Wstęp	20
2.2. Zarządzanie testami w kontekście	20
2.2.1. Zrozumienie interesariuszy testowania	20
2.2.2. Dodatkowe czynności cyklu życia wytwarzania oprogramowania i produkty pracy	21
2.2.3. Zestwienie czynności testowych i innych czynności cyklu życia	22
2.2.4. Zarządzanie testowaniem niefunkcjonalnym	25
2.2.5. Zarządzanie testowaniem opartym na doświadczeniu	25
2.3. Testowanie oparte na ryzyku i inne podejścia do priorytetyzacji testów i przydzielania prac związanych z testowaniem	26
2.3.1. Testowanie oparte na ryzyku	26
2.3.2. Techniki testowania opartego na ryzyku	31
2.3.3. Inne techniki wyboru testów	34
2.3.4. Priorytetyzacja testów i przydzielanie prac w procesie testowym	35
2.4. Dokumentacja testowa i inne produkty pracy	36
2.4.1. Polityka testów	36
2.4.2. Strategia testów	37
2.4.3. Główny plan testów	39
2.4.4. Plan testów jednego poziomu	40
2.4.5. Zarządzanie ryzykiem projektowym	40
2.4.6. Inne produkty testowania	41
2.5. Szacowanie testów	42
2.6. Definiowanie i wykorzystywanie metryk testowych	43
2.7. Wartość biznesowa testowania	48
2.8. Testowanie rozproszone, zlecone na zewnątrz, zlecone wewnątrz	49
2.9. Zarządzanie z zastosowaniem norm branżowych	50
3. Przeglądy – 180 min.	53
3.1. Wprowadzenie	54
3.2. Przegląd kierowniczy i audyty	55
3.3. Zarządzanie przeglądami	56
3.4. Metryki dla przeglądów	57
3.5. Zarządzanie przeglądem formalnym	58

4. Zarządzanie defektami – 150 min.	60
4.1. Wprowadzenie	61
4.2. Cykl życia defektu i cykl procesu twórczego	61
4.2.1. Cykl życia i stany defektu	62
4.2.2. Zarządzanie błędnymi i zduplikowanymi raportami o defektach	62
4.2.3. Międzyfunkcyjne zarządzanie defektami	63
4.3. Atrybuty defektów	63
4.4. Ocena zdolności procesu testowego przy użyciu atrybutów defektu	65
5. Doskonalenie procesu testowego – 135 min.	66
5.1. Wprowadzenie	67
5.2. Proces doskonalenie testowania	67
5.2.1. Wprowadzenie do doskonalenie procesu	67
5.2.2. Typy doskonalień procesu	68
5.3. Doskonalenie procesu testowego	68
5.4. Doskonalenie procesu testowego z użyciem modelu TMMi	69
5.5. Doskonalenie procesu testowego z użyciem TPI Next	70
5.6. Doskonalenie procesu testowego z użyciem CTP	70
5.7. Doskonalenie procesu testowego z użyciem STEP	71
6. Narzędzia testowe i automatyzacja – 135 min.	72
6.1. Wprowadzenie	73
6.2. Wybór narzędzi	73
6.2.1. Narzędzia o otwartym kodzie	73
6.2.2. Narzędzia własne	74
6.2.3. Zwrot z inwestycji (ROI – Return On Investment)	74
6.2.4. Proces wyboru narzędzia	76
6.3. Cykl życia narzędzia	77
6.4. Metryki związane z narzędziami	78
7. Umiejętności interpersonalne – budowa zespołu – 210 min.	79
7.1. Wprowadzenie	80
7.2. Umiejętności	80
7.3. Dynamika zespołu testowego	82
7.4. Umieszczenie testów w organizacji	83
7.5. Motywacja	85
7.6. Komunikacja	85
8. Odnosińki	87
8.1. Normy	87
8.2. Dokumenty ISTQB®	88
8.3. Znaki handlowe	88
8.4. Książki	88
8.5. Inne odnośniki	89
9. Indeks	90

Podziękowania

Niniejszy dokument sporządzony został przez zespół wchodzący w skład Podgrupy Roboczej International Software Testing Qualifications Board dla poziomu zaawansowanego dla modułu Kierownik Testów: Rex Black (Przewodniczący), Judy McKay (Zastępca Przewodniczącego), Graham Bath, Debra Friedenberg, Bernard Homès, Paul Jorgensen, Kenji Onishi, Mike Smith, Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto.

Autorzy wyrażają podziękowania dla zespołu recenzentów i wszystkich Rad Krajowych za sugestie i wsparcie.

W czasie przygotowywania Sylabusu dla poziomu zaawansowanego Kierownik Testów, w skład Grupy Roboczej Poziomu Zaawansowanego wchodzili (w porządku alfabetycznym):

Graham Bath, Rex Black, Maria Clara Choucair, Debra Friedenberg, Bernard Homès (Zastępca Przewodniczącego), Paul Jorgensen, Judy McKay, Jamie Mitchell, Thomas Mueller, Klaus Olsen, Kenji Onishi, Meile Posthuma, Eric Riou du Cosquer, Jan Sabak, Hans Schaefer, Mike Smith (Przewodniczący), Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto.

Następujące osoby brały udział w przeglądach, dodawaniu komentarzy i opiniowały niniejszy sylabus:

Chris van Bael, Graham Bath, Kimmo Hakala, Rob Hendriks, Marcel Kwakernaak, Rik Marselis, Don Mills, Gary Mogyorodi, Thomas Mueller, Ingvar Nordstrom, Katja Piroué, Meile Posthuma, Nathalie Rooseboom de Vries, Geoff Thompson, Jamil Wahbeh, Hans Weiberg.

Niniejszy dokument został oficjalnie wydany przez Zgromadzenie Ogólne ISTQB® w dniu 19 października 2012 roku.

0. Wprowadzenie do Sylabusu

0.1. Cel niniejszego dokumentu

Niniejszy sylabus stanowi podstawę przygotowania do międzynarodowej certyfikacji w zakresie testowania oprogramowania na poziomie zaawansowanym dla modułu Kierownik Testów. ISTQB® udostępnia niniejszy sylabus na zasadach przedstawionych poniżej:

1. Radom Krajowym w celu jego przetłumaczenia na język lokalny i akredytacji dostawców szkoleń. Organizacje Narodowe mogą dostosować sylabus do wymagań swojego języka i zmodyfikować listę materiałów źródłowych, aby dostosować ją do lokalnych publikacji.
2. Komisjom Egzaminacyjnym w celu ułożenia pytań egzaminacyjnych w ich lokalnym języku, dostosowanych do celów nauki dla każdego modułu.
3. Dostawcom szkoleń w celu przygotowania materiałów szkoleniowych i określenia odpowiednich metod nauczania.
4. Kandydatom w celu przygotowania się do egzaminu (jako część kursu przygotowującego lub w celu samodzielnego przygotowania).
5. Międzynarodowej społeczności zajmującej się inżynierią oprogramowania i systemów informatycznych w celu rozwijania zawodu testera oprogramowania i systemów informatycznych, a także jako materiał wyjściowy do książek i artykułów.

ISTQB® może zezwolić innym podmiotom na wykorzystanie niniejszego sylabusu do innych celów, pod warunkiem uzyskania wcześniejszej pisemnej zgody.

0.2. Opis

Plan poziomu zaawansowanego składa się z trzech odrębnych części:

- Kierownik Testów,
- Analityk Testów,
- Techniczny Analityk Testów.

Opis poziomu zaawansowanego [ISTQB_AL_OVIEW] zawiera następujące informacje:

- rezultaty biznesowe dla każdego sylabusu,
- streszczenie każdego sylabusu,
- zależności pomiędzy sylabusami,
- opis poziomów wiedzy (poziomy K),
- załączniki.

0.3. Cele nauczania podlegające egzaminowi

Cele nauczania są zgodne z celami biznesowymi i wykorzystuje się je w celu ułożenia egzaminu umożliwiającego uzyskanie certyfikatu Kierownika Testów na poziomie zaawansowanym. W sensie ogólnym, wszystkie części niniejszego sylabusu są sprawdzane na poziomie K1, co oznacza, że kandydat rozpozna, zapamięta i powtórzy termin lub koncepcję. Cele nauczania na poziomach K2, K3 i K4 wymienione są na początku odpowiednich rozdziałów.

1. Proces testowy – 420 min.

Słowa kluczowe

kryteria wyjścia, przypadek testowy, zamknięcie testów, warunek testowy, nadzór nad testami, projektowanie testów, wykonanie testów, implementacja testów, dziennik testów, planowanie testów, procedura testowa, skrypt testowy, sumaryczny raport z testów

Proces testowy – cele nauczania

1.2. Planowanie testów, monitorowanie i nadzór nad testami

TM-1.2.1. (K4) Kandydat potrafi przeanalizować wymagania testowe dla systemu w celu zaplanowania czynności testowych i produktów, które osiągną cele testów.

1.3. Analiza testów

TM-1.3.1. (K3) Kandydat potrafi zastosować śledzenia powiązań, aby sprawdzić kompletność i spójność zdefiniowanych przypadków testowych w odniesieniu do celów testów, strategii testów i planu testów.

TM-1.3.2. (K2) Kandydat potrafi opisać czynniki, które mogą mieć wpływ na stopień szczegółowości definiowania warunków testowych oraz wskazać zalety i wady definiowania warunków testowych w stopniu szczegółowym.

1.4. Projektowanie testów

TM-1.4.1. (K3) Kandydat potrafi zastosować śledzenia powiązań, aby sprawdzić kompletność i spójność zaprojektowanych przypadków testowych w odniesieniu do zdefiniowanych warunków testowych.

1.5. Implementacja testów

TM-1.5.1. (K3) Kandydat potrafi użyć ryzyk, priorytetów, środowisk testowych, zależności między danymi oraz ograniczeń dla wytworzenia harmonogramy uruchomienia testów, który będzie kompletny i spójny z celami testowania, strategią testów oraz planem testów

1.6. Wykonywanie testów

TM-1.6.1. (K3) Kandydat potrafi wykorzystać śledzenia powiązań w celu monitorowania postępów w testowaniu pod kątem spełnienia kryterium kompletności i spójności z celami testów, strategią testów oraz planem testów.

1.7. Ocena kryteriów zakończenia i raportowanie

TM-1.7.1. (K2) Kandydat potrafi wyjaśnić znaczenie dokładnego i terminowego gromadzenia informacji podczas procesu testowego w celu wsparcia dokładnego raportowania i oceny na podstawie kryteriów wyjścia.

1.8. Czynności zamykające testy

TM-1.8.1. (K2) Kandydat potrafi scharakteryzować cztery grupy czynności związanych z zakończeniem testów.

TM-1.8.2. (K3) Kandydat potrafi przeprowadzić retrospektywę projektu w celu dokonania ewaluacji procesów i określenia obszarów do poprawy.

1.1. Wstęp

Certyfikowany tester - Sylabus poziomu podstawowego ISTQB® opisuje podstawowy proces testowy, który obejmuje następujące czynności ¹:

- planowanie i nadzór nad testami,
- analiza i projektowanie testów,
- implementacja i wykonanie testów,
- ocena kryteriów zakończenia i raportowanie,
- czynności zamykające test.

Certyfikowany tester - Sylabus poziomu podstawowego ISTQB® podaje, że chociaż w trakcie całego procesu testowego czynności następują w logicznej kolejności, to jednak mogą się one pokrywać lub odbywać się jednocześnie. Zazwyczaj należy dostosować główne czynności związane z testowaniem do kontekstu systemu i projektu.

Jeśli chodzi o sylabusy poziomu zaawansowanego, niektóre z tych czynności rozpatrywane są osobno, aby zagwarantować dalsze udoskonalenie i optymalizację procesu testowego, lepsze dopasowanie czynności testowych do cyklu wytwarzania oprogramowania, a także w celu umożliwienia skutecznego monitorowania testów i nadzoru nad testami. Czynności te obecnie rozpatruje się w następującej kolejności:

- planowanie, monitorowanie i nadzór nad testami,
- analiza,
- projektowanie,
- implementacja,
- wykonanie,
- ocena kryteriów zakończenia i raportowanie,
- czynności zamykające testy.

1.2. Planowanie testów, monitorowanie i nadzór nad testami

W niniejszej sekcji skupiono się na procesach planowania, monitorowania i nadzoru nad testami. Jak przedstawiono w Sylabusie poziomu podstawowego ISTQB®, czynności te należą do obowiązków kierownika testów.

1.2.1. Planowanie testów

Dla każdego poziomu testów, planowanie testów zaczyna się przy inicjacji procesu testowego dla danego poziomu i trwa przez cały projekt do momentu zakończenia czynności zamykających testy dla tego poziomu. Planowanie testów obejmuje zdefiniowanie czynności i zasobów wymaganych dla spełnienia misji i celów określonych w strategii testów. Proces ten obejmuje również określenie metod gromadzenia i śledzenia metryk, które będą stosowane do prowadzenia projektu, określania kryteriów realizacji planu i oceny stopnia realizacji celów. Poprzez określenie przydatnych metryk na poszczególnych etapach planowania można dokonać wyboru narzędzi, ustalić harmonogram szkoleń oraz ustalić wytyczne dotyczące dokumentacji.

Strategie (lub strategia) wybrane dla projektu testowego pomagają określić zadania, jakie powinny być realizowane na poszczególnych etapach planowania. Na przykład, stosując strategię testowania

¹ Podstawowy proces testowy jest zgodny z Sylabusem poziomu zaawansowanego Kierownik Testów 2012. W sylabusie Certyfikowany tester – Sylabus poziomu podstawowego Wersja 2018, V.3.1. dokonano aktualizacji procesu testowego.

opartego na ryzyku (patrz Rozdział 2.) dla kierowania procesem planowania testów (w odniesieniu do czynności łagodzących ryzyko, wymaganych do obniżenia zidentyfikowanych ryzyk produktowych i wsparcia planowania na wypadek awarii), stosuje się metodę analizy ryzyka. W przypadku, gdy zostanie zidentyfikowana pewna liczba prawdopodobnych i ważnych defektów dotyczących bezpieczeństwa, należy dołożyć starań, aby opracować i przeprowadzić testy bezpieczeństwa. Podobnie, jeśli zostanie stwierdzone, że ważne defekty znajdują się w specyfikacji projektowej, możliwym skutkiem procesu planowania testów może być dodatkowe testowanie statyczne (przeglądy) specyfikacji projektowej.

Informacja o ryzyku może zostać wykorzystana również do określenia priorytetów różnych czynności testowych. Na przykład, gdy wydajność systemu stanowi wysokie ryzyko, testowanie wydajności może zostać przeprowadzone, jak tylko dostępny będzie zintegrowany kod. Podobnie, jeśli zastosowana ma być strategia reaktywna, uzasadnione może być planowanie w celu utworzenia kart opisu testów i wyboru narzędzi do technik testowania dynamicznego, takich jak testowanie eksploracyjne.

Ponadto, na etapie planowania testów, Kierownik Testów w jasny sposób określa podejście do testowania, łącznie z tym, które poziomy testów będą uwzględnione, cele każdego poziomu, a także, jakie techniki testów będą stosowane na każdym poziomie testów. Na przykład, podczas testowania niektórych systemów lotniczych przy wykorzystaniu strategii opartej na ryzyku, w ramach oceny ryzyka zaleca się wymagany poziom pokrycia kodu, a w konsekwencji, jakich technik testowych należy użyć.

Pomiędzy podstawą testu (np. konkretnymi wymaganiami lub ryzykami), warunkami testowymi a pokrywającymi je testami mogą występować złożone relacje. Pomiędzy tymi produktami często istnieją relacje typu wiele do wielu. Aby umożliwić skuteczne wykonanie czynności planowania, monitorowania i nadzoru nad testami, należy zrozumieć te relacje. Od zrozumienia relacji pomiędzy tymi produktami mogą również zależeć decyzje dotyczące wyboru narzędzi.

Pomiędzy produktami wytworzonymi przez zespół programistów i zespół testerów również mogą występować zależności. Na przykład, możliwe jest, że macierz śledzenia powiązań musi umożliwić identyfikację powiązań pomiędzy pojedynczymi elementami specyfikacji projektowej ze strony projektantów systemu, wymaganiami biznesowymi ze strony analityków biznesowych a produktami testowania zdefiniowanymi przez zespół testujący. Jeśli będą projektowane i wykorzystywane przypadki testowe niskiego poziomu, na etapie planowania może pojawić się wymaganie stwierdzające, że szczegółowe dokumenty projektowe pochodzące ze strony zespołu programistów muszą zostać zatwierdzone przed rozpoczęciem projektowania przypadków testowych. W przypadku stosowania metodyk zwinnych, przed rozpoczęciem testowania, w celu przekazywania informacji pomiędzy zespołami, można wykorzystać nieformalne spotkania informacyjne.

Plan testów może również zawierać opis określonych cech oprogramowania, które mieszczą się w jego zakresie (w oparciu o analizę ryzyka, jeśli ma to uzasadnienie), a także wyraźnie określać cechy, które nie mieszczą się w jego zakresie. W zależności od poziomu sformalizowania i rodzaju dokumentacji właściwej dla danego projektu, każda cecha, która mieści się w zakresie planu testów, może być powiązana z odpowiednią specyfikacją projektu testów.

Na tym etapie Kierownik Testów może być zobowiązany do pracy z architektami w projekcie rozwiązania, w celu zdefiniowania początkowej specyfikacji środowiska testowego i po to, aby zweryfikować dostępność wymaganych zasobów oraz zapewnić, że osobom, które będą konfigurować środowisko, zadanie to zostało powierzone. Istotne jest także zrozumienie zasadności posienionych kosztów i czasu potrzebnych do wdrożenia produkcyjnego oraz nakładu pracy wymaganego do wykonania i dostarczenia środowiska testowego.

Na koniec, należy zidentyfikować wszystkie zewnętrzne zależności i powiązane umowy o gwarantowanym poziomie usług (ang. SLA – *Service Level Agreement*) i, jeśli jest to wymagane,

należy nawiązać wstępny kontakt z interesariuszami projektu. Przykładami takich zależności są żądania dotyczące zasobów skierowane do grup zewnętrznych, zależności od innych projektów (jeśli ma to miejsce w ramach programu), zewnętrznych sprzedawców lub partnerów w wytwarzaniu oprogramowania, zespołu wdrożeniowego oraz administratorów baz danych.

1.2.2. Monitorowanie testów i nadzór nad testami

Aby Kierownik Testów mógł zapewnić skuteczny nadzór nad testami, należy ustalić harmonogram testów i strukturę monitorowania w celu umożliwienia śledzenia powiązań pomiędzy produktami pracy związanymi z testowaniem a zasobami w odniesieniu do planu testów. Struktura ta powinna obejmować szczegółowe miary i cele, które są wymagane do odniesienia stanu produktów i czynności testowych do planu testów i celów strategicznych.

W przypadku małych i mniej złożonych projektów, stosunkowo łatwo można odnieść czynności testowe i produkty pracy związane z testowaniem do planu i celów strategicznych. Niemniej jednak, aby to osiągnąć, należy zdefiniować bardziej szczegółowe cele. Definicja taka może uwzględniać miary i rezultaty zapewniające spełnienie celów testowania i realizację pokrycia.

Szczególnie istotne jest odniesienie stanu czynności i produktów pracy związanych z testowaniem do podstawy testów w sposób zrozumiały i odpowiedni dla interesariuszy projektu i interesariuszy biznesowych. Definiowanie celów i pomiar zaawansowania prac w oparciu o warunki testowe i grupy warunków testowych może być sposobem osiągnięcia tego celu poprzez odniesienie innych produktów testowych do podstawy testów za pomocą warunków testowych. Właściwe zdefiniowanie śledzenia powiązań, w tym możliwość raportowania stanu śledzenia powiązań, sprawia, że złożone powiązania występujące pomiędzy produktami wytwarzania oprogramowania, podstawą testów a produktami pracy związanymi z testowaniem stają się bardziej przejrzyste i zrozumiałe.

Czasami szczegółowe miary i cele, których monitorowania wymagają interesariusze, nie odnoszą się bezpośrednio do funkcjonalności systemu lub specyfikacji, szczególnie wtedy, gdy formalna dokumentacja nie istnieje lub jest jej niewiele. Na przykład, interesariusz biznesowy może być bardziej zainteresowany ustaleniem poziomu pokrycia w odniesieniu do funkcjonującego cyklu biznesowego, nawet jeśli specyfikacja została zdefiniowana w odniesieniu do funkcjonalności systemu. Zaangażowanie interesariuszy biznesowych we wczesnej fazie projektu może pomóc zdefiniować miary i cele, które nie tylko można wykorzystać w celu zapewnienia skuteczniejszego nadzoru w trakcie projektu, ale też w celu sterowania i oddziaływania na czynności testowe przez cały czas trwania projektu. Na przykład, miary i cele interesariuszy mogą skutkować zaprojektowaniem produktów pracy w postaci przypadków testowych i wykonaniem testów i/lub harmonogramów wykonywania testów, które ułatwią dokładne monitorowanie postępów testów w odniesieniu do tych miar. Cele te ułatwiają również zapewnienie spójności pomiarów dla określonego poziomu testów oraz potencjalnie mogą ułatwić zapewnienie spójności informacji na różnych poziomach testów.

Nadzór nad testami jest działaniem ciągłym. Obejmuje porównywanie faktycznego postępu prac z planem testów i wdrażanie działań naprawczych w razie potrzeby. Nadzór nad testami umożliwia ukierunkowanie procesu testowego tak, aby zrealizować misję, strategię oraz cele, co w razie konieczności pociąga za sobą powrót do czynności planowania testów. Odpowiednia reakcja na dane pochodzące z nadzoru nad testami zależy od szczegółowych informacji dotyczących planowania.

Treść dokumentów dotyczących planowania testów oraz czynności z zakresu nadzoru nad testami omówiono w Rozdziale 2.

1.3. Analiza testów

W sylabusach poziomu zaawansowanego przedstawiono zagadnienia analizy testów oraz projektowania testów jako odrębne działania, aczkolwiek uznając, że mogą być one wdrażane jako czynności wykonywane równolegle, w sposób zintegrowany lub iteracyjny, w celu zaprojektowania produktów procesu testowego.

Analiza testów jest czynnością definiującą w formie warunków testowych to, „co” ma być testowane. Warunki testowe można zidentyfikować poprzez analizę podstawy testów, celów testów oraz ryzyk produktowych. Można je uznać za szczegółowe miary i cele określające powodzenie (np. jako część kryteriów wyjścia), a ich powiązania powinny być możliwe do prześledzenia w odniesieniu do podstawy testów i zdefiniowanych celów strategicznych, w tym celów testów i innych kryteriów osiągnięcia sukcesu po stronie projektu lub interesariuszy. Warunki testowe powinny być również w widoczny sposób powiązane z projektami testów i innymi produktami, w miarę jak produkty te są dostarczane.

Analizę testów dla danego poziomu testów można przeprowadzić wtedy, gdy zostanie ustalona podstawa testów tylko dla tego poziomu testów. Aby zidentyfikować warunki testowe, można zastosować formalne techniki testowania i inne ogólne techniki analityczne (np. strategie analityczne oparte na ryzyku oraz strategie analityczne oparte na wymaganiach). Warunki testowe mogą określać wartości lub zmienne w zależności od poziomu testów, informacji dostępnych w momencie przeprowadzania analizy i wybranego poziomu szczegółowości (tj. szczegółowości dokumentacji).

Czynniki, które należy wziąć pod uwagę podejmując decyzję o poziomie szczegółowości warunków testowych, obejmują:

- poziom testów,
- poziom szczegółowości i jakość podstawy testów,
- złożoność systemu/oprogramowania,
- ryzyko projektowe i ryzyko produktowe,
- związek pomiędzy podstawą testów, tym, co należy przetestować a sposobem, w jaki należy to przetestować,
- zastosowany cykl życia wytwarzania oprogramowania,
- stosowane narzędzie do zarządzania testami,
- poziom, na którym należy sprecyzować i udokumentować projektowanie testów i inne produkty testowania,
- umiejętności i wiedzę Analityków Testów,
- poziom dojrzałości procesu testowego i organizacji (należy zauważyć, że wyższy poziom dojrzałości może wymagać wyższego poziomu szczegółowości lub dopuścić jej niższy poziom),
- dostępność innych interesariuszy projektu w celu konsultacji.

Szczegółowe określenie warunków testowych prowadzi do powstania większej ich liczby. Na przykład może występować pojedynczy, ogólny warunek testowy dla aplikacji e-commerce: „Przetestuj płatność”, natomiast w szczegółowym dokumencie dotyczącym warunków testowych, powyższy warunek może zostać wyrażony w formie wielu warunków testowych według zasady, że jeden warunek testowy przypada na każdą obsługiwaną metodę płatności, jeden na każdy możliwy kraj docelowy i tak dalej.

Szczegółowe definiowanie warunków testowych niesie ze sobą pewne korzyści, między innymi:

- Umożliwia większą elastyczność w odnoszeniu innych produktów testowania (np. przypadków testowych) do podstawy testów i celów testów, tym samym umożliwiając Kierownikowi Testów lepsze i dokładniejsze monitorowanie testów i nadzór nad testami.
- Przyczynia się do zapobiegania defektom, jak opisano w sylabusie Certyfikowany Tester Sylabus poziomu podstawowego ISTQB®, poprzez wystąpienie we wczesnej fazie projektu dla

wyższych poziomów testów, w momencie, gdy zostanie ustalona podstawa testów i zanim potencjalnie będzie dostępna architektura systemu i szczegółowy projekt.

- Umożliwia przedstawienie interesariuszom produktów testowania za pomocą sformułowań dla nich zrozumiałych (często zdarza się, że przypadki testowe i inne produkty pracy związane z testowaniem nie są zrozumiałe dla interesariuszy biznesowych i proste metryki, takie jak liczba wykonanych przypadków testowych, nie mają żadnego znaczenia dla wymagań interesariuszy dotyczących pokrycia).
- Pomaga kształtować i wpływać nie tylko na inne czynności testowe, ale także na czynności z zakresu wytwarzania oprogramowania.
- Ułatwia projektowanie, implementację i wykonanie testów, a wynikowe produkty pracy będą optymalizowane przez bardziej efektywne pokrycie szczegółowych miar i celów.
- Daje podstawę do łatwiejszego poziomu śledzenia w ramach danego poziomu testów.

Niektóre wady szczegółowego określania warunków testowych to:

- potencjalna czasochłonność,
- potencjalnie trudne utrzymanie w zmieniającym się środowisku,
- poziom sformalizowania musi być zdefiniowany i wdrożony w całym zespole.

Określenie szczegółowych warunków testowych może być szczególnie efektywne w następujących sytuacjach:

- nieobciążające (lekkie) metody dokumentacji projektowania testów, takie jak listy kontrolne, używane są w celu złagodzenia negatywnych skutków cyklu wytwarzania oprogramowania, ograniczeń kosztowych i/lub czasowych lub innych czynników,
- niewiele lub brak formalnych wymagań lub innych produktów wytwarzania oprogramowania służących jako podstawa testów,
- projekt jest przedsięwzięciem zakrojonym na szeroką skalę, złożonym lub obciążonym wysokim poziomem ryzyka i wymaga poziomu nadzorowania i monitorowania, którego nie można zapewnić tylko poprzez odnoszenie przypadków testowych do produktów wytwarzania oprogramowania.

Warunki testowe mogą być określone mniej szczegółowo, kiedy podstawę testów można w łatwy i bezpośredni sposób odnieść do produktów projektowania testów. Sytuacja taka jest bardziej prawdopodobna w następujących przypadkach:

- testowanie modułów,
- mniej skomplikowane projekty, w których istnieją proste hierarchiczne powiązania pomiędzy tym, co należy przetestować a sposobem, w jaki ma to zostać przetestowane
- testy akceptacyjne, podczas których przypadki użycia mogą zostać wykorzystane w celu wsparcia definiowania testów.

1.4. Projektowanie testów

Projektowanie testów jest czynnością definiującą, „jak” coś ma być przetestowane. Czynność ta obejmuje identyfikację przypadków testowych poprzez stopniowe rozwijanie zidentyfikowanych warunków testowych lub podstawy testów z zastosowaniem technik określonych w strategii testowej i/lub planie testów.

W zależności od podejść stosowanych do monitorowania testów, nadzoru nad testami i śledzenia powiązań, przypadki testowe mogą być bezpośrednio odniesione (lub pośrednio poprzez warunki testowe) do podstawy testów i zdefiniowanych celów. Cele te obejmują cele strategiczne, cele testów i inne kryteria projektowe bądź kryteria sukcesu określone przez interesariuszy.

Projektowanie testów dla danego poziomu testów może zostać wykonane po tym, jak określone zostaną warunki testowe i dostępna jest wystarczająca ilość informacji, aby umożliwić tworzenie przypadków testowych niskiego lub wysokiego poziomu, zgodnie z zastosowanym podejściem do projektowania testów. W przypadku testowania na wyższych poziomach jest bardziej prawdopodobne, że projektowanie testów jest odrębną czynnością następującą po wcześniejszej analizie testów. W przypadku testowania na niższych poziomach prawdopodobne jest, że analiza i projektowanie testów będą przeprowadzane jako zintegrowane działanie.

Możliwa jest również sytuacja, że niektóre zadania zwykle występujące w trakcie implementacji testów zostaną włączone w projektowanie testów, gdy stosowane jest podejście iteracyjne do tworzenia koniecznych do wykonania testów, np. tworzenie danych testowych. W rzeczywistości podejście takie może zoptymalizować pokrycie warunków testowych poprzez stworzenie przypadków niskiego lub wysokiego poziomu w trakcie projektowania testów.

1.5. Implementacja testów

Implementacja testów jest działaniem, w trakcie którego Analitycy Testów analizują i priorytetyzują testy. W kontekstach formalnie udokumentowanych, implementacja testów jest działaniem, w ramach którego wdraża się projekty testów jako konkretne przypadki testowe, procedury testowe i dane testowe. Niektóre organizacje stosujące normę IEEE 829² [IEEE829] definiują dane wejściowe i powiązane z nimi oczekiwane rezultaty w specyfikacjach przypadków testowych i kroki testowe w specyfikacjach procedur testowych. Częściej jednak jest tak, że dane wejściowe każdego testu, oczekiwane rezultaty i kroki testów są dokumentowane razem. Implementacja testów obejmuje również tworzenie danych testowych (np. w zwykłych plikach lub tabelach relacyjnych baz danych), które są następnie przechowywane.

Implementacja testów obejmuje również końcową kontrolę mającą na celu upewnienie się, że zespół testowy jest gotowy do wykonania testów. Kontrola taka może obejmować zagwarantowanie dostarczenia wymaganego środowiska testowego, danych testowych i kodu (ewentualnie uruchomienie jakiegoś środowiska testowego i/lub testów akceptacyjnych kodu) oraz zapewnienie, że wszystkie przypadki testowe zostały zaprojektowane, sprawdzone i są gotowe do wykonania. Kontrola końcowa może również obejmować sprawdzenie jawnych i niejawnych kryteriów wejścia dla danego poziomu testów (zob. Sekcja 1.7.). Implementacja testów może też obejmować opracowanie szczegółowego opisu środowiska testowego i danych testowych.

Poziom szczegółowości i powiązana z tym złożoność pracy wykonanej w trakcie wdrażania testów mogą być kształtowane przez poziom szczegółowości produktów testowania (np. przypadków testowych i warunków testowych). W niektórych przypadkach, szczególnie, gdy testy mają być archiwizowane w celu długofalowego ponownego wykorzystywania w testowaniu regresji, mogą one zawierać szczegółowe opisy kroków wymaganych do przeprowadzenia testu, tak, aby zapewnić wykonanie testu w sposób wiarygodny i spójny, niezależnie od testera przeprowadzającego test. Jeśli mają zastosowanie przepisy prawa, testy powinny zapewniać potwierdzenie zgodności ze stosownymi normami (zob. Sekcja 2.9.).

W trakcie wdrażania testów ujętych w harmonogramie wykonywania testów należy uwzględnić kolejność, w jakiej mają być przeprowadzone testy manualne i automatyczne. Kierownicy Testów powinni dokładnie sprawdzić ograniczenia, w tym ryzyka i priorytety, zgodnie z którymi może być wymagane wykonywanie testów w określonej kolejności lub z zastosowaniem określonego sprzętu. Zależności dotyczące środowiska testowego lub danych testowych muszą być znane i sprawdzone.

² Standard ISO/IEC/IEEE 29119 zastąpił normę IEEE 829.

Wczesna implementacja testów może mieć kilka wad. Na przykład, w przypadku zwinnego cyklu wytwarzania oprogramowania, kod może ulegać znacznym zmianom pomiędzy kolejnymi iteracjami, co oznacza, że znaczna część pracy wdrożeniowej dezaktualizuje się. Nawet jeśli nie jest stosowany cykl życia oprogramowania tak zmiennego jak w metodykach zwinnych, jakiegokolwiek cykl iteracyjny lub przyrostowy może przynieść istotne zmiany pomiędzy kolejnymi iteracjami, powodując, że testy skryptowe stają się niewiarygodne lub wymagają znacznej pielęgnacji. To samo dotyczy źle zarządzanych sekwencyjnych cykli życia oprogramowania, w których wymagania często się zmieniają, nawet w późnej fazie projektu. Przed rozpoczęciem działań związanych z implementacją testów, rozsądnym jest zrozumienie cyklu życia wytwarzania oprogramowania oraz przewidzenie cech oprogramowania, które będą dostępne do testowania.

Wczesna implementacja testów może mieć też pewne zalety. Przykładem zalet wczesnej implementacji jest fakt, że konkretne testy dostarczają opracowane przykłady tego, jak powinno zachowywać się oprogramowanie, jeśli zostało napisane zgodnie z podstawą testów. Ekspertom dziedzinowym może być łatwiej zweryfikować konkretne testy niż abstrakcyjne reguły biznesowe, dzięki czemu mogą zidentyfikować kolejne słabe punkty w specyfikacjach oprogramowania. Takie zweryfikowane testy mogą dostarczyć projektantom i programistom pouczające przykłady wymaganego zachowania dla projektantów i programistów.

1.6. Wykonywanie testów

Wykonywanie testów rozpoczyna się po dostarczeniu przedmiotu testów i spełnieniu kryteriów wejścia do wykonania testów. Przed wykonaniem testów należy je zaprojektować lub przynajmniej zdefiniować. Powinny być dostępne narzędzia, zwłaszcza te służące do zarządzania testami, śledzenia defektów i (jeśli ma to uzasadnienie) automatyzacji wykonywania testów. Powinna być dostępna możliwość śledzenia rezultatów testów, w tym śledzenia metryk, a śledzone dane powinny być zrozumiałe dla wszystkich członków zespołu. Powinny być dostępne i opublikowane standardy dotyczące rejestracji (logowania) testów i raportowania defektów. Efektywne przeprowadzenie testów wymaga zapewnienia dostępności powyższych elementów przed wykonaniem testów.

Testy powinny być wykonywane zgodnie z przypadkami testowymi, chociaż Kierownik Testów powinien rozważyć dopuszczenie pewnego stopnia dowolności, aby tester mógł pokryć dodatkowe interesujące scenariusze testowe i zachowania zaobserwowane w trakcie testowania. W przypadku stosowania strategii testów, która jest przynajmniej po części reaktywna, należy zarezerwować pewną ilość czasu na sesje testowe z zastosowaniem technik opartych na doświadczeniu i defektach. Oczywiście w każdym raporcie o defekcie wykrytym podczas testowania nieskryptowego muszą być opisane niezbędne dla odtworzenia defektu odstępstwa od zapisanego przypadku testowego. Testy zautomatyzowane będą realizować zdefiniowane dla nich instrukcje bez odstępstw.

Głównym zadaniem Kierownika Testów w trakcie wykonywania testów jest monitorowanie postępu prac zgodnie z planem testów oraz, jeśli jest to wymagane, inicjowanie i wykonywanie czynności kontrolnych w celu doprowadzenia testowania do pomyślnego zakończenia w odniesieniu do misji, celów i strategii. W tym celu Kierownik Testów może prześledzić powiązania pomiędzy rezultatami testów a warunkami testowymi, podstawą testów i, na koniec, celami testów, a także pomiędzy celami a rezultatami testów. Proces ten opisany jest szczegółowo w Sekcji 2.6.

1.7. Ocena kryteriów zakończenia i raportowanie

W Sekcji 2.6. omówiono dokumentowanie i raportowanie dla celów monitorowania postępów testów i nadzoru nad testami.

Z punktu widzenia procesu testowego, ważne jest, aby upewnić się, że są dostępne skuteczne procedury mające na celu dostarczenie informacji źródłowych niezbędnych dla oceny kryteriów wyjścia i raportowania.

Definicja wymagań dotyczących informacji i sposobów ich gromadzenia jest częścią planowania, monitorowania i nadzoru nad testami. Podczas analizy testów, projektowania testów, implementacji testów i wykonywania testów, Kierownik Testów powinien upewnić się, że członkowie zespołu testowego odpowiedzialni za te działania dokładnie i punktualnie dostarczają wymagane informacje, aby umożliwić efektywną ocenę i raportowanie.

Częstotliwość i poziom szczegółowości wymagany dla celów raportowania zależą od projektu i organizacji. Zagadnienia te powinny zostać uzgodnione na etapie planowania testów i powinny obejmować konsultacje z właściwymi interesariuszami projektu.

1.8. Czynności zamykające testy

Po stwierdzeniu ukończenia testowania, należy zapisać kluczowe dane wyjściowe i przekazać je właściwej osobie lub zarchiwizować. W sensie ogólnym, są to czynności związane z zakończeniem testów. Czynności te dzielą się na cztery główne grupy:

1. Kontrola zakończenia procesu testowania – upewnienie się, że testowanie faktycznie zostało zakończone. Na przykład, powinny być wykonane lub celowo pominięte wszystkie zaplanowane testy, a wszystkie znane defekty powinny zostać naprawione i ponownie przetestowane, odroczone do przyszłego wydania bądź zaakceptowane jako stałe ograniczenia.
2. Przekazanie artefaktów testowania – przekazanie wartościowych produktów osobom, które ich potrzebują. Na przykład, znane defekty, odroczone lub zaakceptowane, należy przekazać do wiadomości osobom, które będą korzystać i wspierać pracę z systemem. Testy i środowiska testowe powinny zostać przekazane osobom odpowiedzialnym za testowanie pielęgnacyjne. Zestawy testów regresji (manualnych lub automatycznych) powinny zostać udokumentowane i przekazane zespołowi zajmującemu się pielęgnacją.
3. Wyciągnięcie wniosków – prowadzenie retrospektyw lub uczestnictwo w tych spotkaniach, podczas których mogą zostać udokumentowane ważne wnioski (zarówno pochodzące z projektu testowego, jak i całego cyklu życia wytwarzania oprogramowania). Podczas takich spotkań ustala się plany, aby upewnić się, że dobre praktyki będą powtarzane, a niewłaściwe praktyki nie zostaną powtórzone tam, gdzie nie można rozwiązać problemów, zaś prace nad ich rozwiązaniem powinny być uwzględnione w planach projektu. Należy wziąć pod uwagę następujące zagadnienia:
 - a) Czy przedstawiciele użytkownika podczas sesji dotyczących analizy ryzyka jakościowego byli odpowiednio szeroko reprezentowani? Na przykład, z powodu późnego wykrycia nieoczekiwanych skupisk defektów, zespół mógł stwierdzić, że w sesjach dotyczących analizy ryzyka jakościowego w przyszłych projektach powinna brać udział szersza reprezentacja przedstawicieli użytkownika.
 - b) Czy obliczenia były dokładne? Na przykład, obliczenia mogły być w znacznym stopniu błędne, dlatego też w przyszłości podczas szacowania należy wyjaśnić niedokładność szacunków z podaniem zasadniczych przyczyn, np. czy testowanie było nieskuteczne, czy obliczenia dały faktycznie niższy wynik niż ten, który powinien zostać uzyskany.
 - c) Jakie są tendencje i rezultaty analizy przyczynowo-skutkowej defektów? Na przykład, należy ocenić, czy na jakość analizy i wytwarzania oprogramowania miały wpływ późne żądania zmian, wyszukać tendencje, które wskazują na niewłaściwe praktyki, takie jak pominięcie poziomu testów, na którym wcześniej zostałyby znalezione defekty, co byłoby bardziej opłacalne, gdyż skutkowałoby oszczędnością czasu. Należy sprawdzić, czy można odnieść tendencje w występowaniu defektów do obszarów takich jak nowe technologie, zmiany personelu lub brak umiejętności.

- d) Czy istnieją możliwości udoskonalenia procesu?
 - e) Czy miały miejsce nieoczekiwane odstępstwa od planu, które należy uwzględnić w planowaniu w przyszłości?
4. Archiwizacja rezultatów, dzienników, raportów i innych dokumentów oraz produktów pracy w systemie zarządzania konfiguracją. Na przykład, w archiwum planów powinny być zapisane plan testów i plan projektu z wyraźnymi odniesieniami do systemu i wersji, na której były używane.

Zadania powyższe są istotne, choć często pomijane, i powinny być wyraźnie uwzględnione jako część planu testów.

Często jedno lub więcej z tych zadań jest pomijane, zwykle z powodu przedwczesnego przeniesienia lub zwolnienia członków zespołu związanego z projektem, nacisków na realizację kolejnych projektów w związku z harmonogramem prac lub zasobami bądź też wypalenia zespołu. W projektach realizowanych na podstawie kontraktu, kiedy oprogramowanie jest wytwarzane na zlecenie, kontrakt powinien określać wymagane zadania.

2. Zarządzanie testami – 750 min.

Słowa kluczowe

plan testów jednego poziomu, główny plan testów, ryzyko produktowe, ryzyko projektowe, ryzyko jakościowe, ryzyko, analiza ryzyka, zarządzanie ryzykiem, identyfikacja ryzyka, poziom ryzyka, łagodzenie ryzyka, testowanie oparte na ryzyku, podejście do testowania, warunki testowe, nadzór nad testami, dyrektor testów, szacowanie testów, Kierownik Testów, poziom testowania, zarządzanie testami, monitorowanie testów, plan testów, polityka testów, strategia testów, szerokopasmowa technika delficka

Zarządzanie testami – cele nauczania

2.2. Zarządzanie testami w kontekście

- TM-2.2.1. (K4) Kandydat potrafi poddać analizie interesariuszy, okoliczności i potrzeby projektu lub programu wytwarzania oprogramowania, wraz z modelem cyklu życia wytwarzania oprogramowania oraz zidentyfikować optymalne czynności testowe.
- TM-2.2.2. (K2) Kandydat rozumie, jaki wpływ na testowanie mają czynności i produkty pracy cyklu życia wytwarzania oprogramowania oraz jaki wpływ ma testowanie na czynności i produkty pracy cyklu życia wytwarzania oprogramowania.
- TM-2.2.3. (K2) Kandydat potrafi wyjaśnić sposoby zarządzania problemami związanymi z zarządzaniem testami opartymi na doświadczeniu i testami niefunkcjonalnymi.

2.3. Testowanie oparte na ryzyku i inne podejścia do priorytetyzacji testów i przydzielania prac związanych z testowaniem

- TM-2.3.1. (K2) Kandydat potrafi wyjaśnić różne sposoby reagowania na ryzyko w testowaniu opartym na ryzyku.
- TM-2.3.2. (K2) Kandydat potrafi wyjaśnić, podając przykłady, różne techniki analizy ryzyka produktowego.
- TM-2.3.3. (K4) Kandydat potrafi przeanalizować, zidentyfikować i oszacować ryzyka jakościowe produktu, podsumowując ryzyka i ich szacowany poziom w oparciu o punkty widzenia kluczowych interesariuszy projektu.
- TM-2.3.4. (K2) Kandydat potrafi opisać, jak można złagodzić zidentyfikowane ryzyka jakościowe produktu i jak nimi zarządzać w trakcie cyklu życia i procesu wytwarzania oprogramowania stosownie do ich oszacowanego poziomu.
- TM-2.3.5. (K2) Kandydat potrafi podać przykłady różnych opcji dotyczących wyboru testów, priorytetyzacji testów i przydzielania prac.

2.4. Dokumentacja testowa i inne produkty pracy

- TM-2.4.1. (K4) Kandydat potrafi poddać analizie podane fragmenty polityki testów i strategii testów i stworzyć główny plan testów, plany testów jednego poziomu i inne, kompletne i zgodne z tymi dokumentami produkty pracy procesu testowego.
- TM-2.4.2. (K4) Kandydat potrafi przeanalizować ryzyka projektowe dla danego projektu i wybrać odpowiednie opcje zarządzania ryzykiem (tj. łagodzenie, plany awaryjne, przeniesienie i/lub akceptacja).
- TM-2.4.3. (K2) Kandydat potrafi opisać, podając przykłady, jak strategię testów wpływają na czynności testowe.
- TM-2.4.4. (K3) Kandydat potrafi zdefiniować normy dokumentacji i szablony dla produktów testowania, które pasują do organizacji, cyklu życia wytwarzania oprogramowania i potrzeb projektu, dostosowując szablony udostępnione przez instytucje normatywne, jeśli ma to uzasadnienie.

2.5. Szacowanie testów

- TM-2.5.1. (K3) Kandydat potrafi opracować dla danego projektu szacunki dla wszystkich czynności procesu testowego, korzystając ze wszystkich technik szacowania, które mają zastosowanie.
- TM-2.5.2. (K2) Kandydat rozumie i potrafi podać przykłady czynników, które mogą mieć wpływ na oszacowanie.

2.6. Definiowanie i korzystanie z metryk testowych

- TM-2.6.1. (K2) Kandydat potrafi opisać i porównać typowe metryki związane z testowaniem.
- TM-2.6.2. (K2) Kandydat potrafi porównać różne wymiary monitorowania postępu testów.
- TM-2.6.3. (K4) Kandydat potrafi przeanalizować wyniki testów i przedstawić je w raporcie w kategoriach: pozostałego ryzyka, statusu defektów, statusu wykonania testów, statusu pokrycia testami i pewności w celu przedstawienia spostrzeżeń i rekomendacji, które umożliwiają interesariuszom projektu podjęcie decyzji dotyczących wydania.

2.7. Wartość biznesowa testowania

- TM-2.7.1. (K2) Kandydat potrafi podać przykłady każdej z czterech kategorii określających koszt jakości.
- TM-2.7.2. (K3) Kandydat potrafi oszacować wartość testowania w oparciu o koszt jakości, wraz z innymi czynnikami ilościowymi i jakościowymi oraz przedstawić szacowaną wartość interesariuszom testów.

2.8. Testowanie rozproszone, zlecone innej firmie i prowadzone wewnątrz firmy

- TM-2.8.1. (K2) Kandydat rozumie czynniki wymagane do skutecznego wykorzystania rozproszonej, zleconej na zewnątrz oraz zleconej wewnątrz firmy strategii obsadzenia zespołu testowego.

2.9. Zarządzanie wdrażaniem norm branżowych

- TM-2.9.1. (K2) Kandydat potrafi podsumować informacje pochodzące ze źródeł i wykorzystać standardy dotyczące testowania oprogramowania.

2.1. Wstęp

W karierze profesjonalisty na poziomie zaawansowanym zaczęła pojawiać się specjalizacja w dziedzinie testowania. Niniejszy rozdział poświęcony jest wiedzy wymaganej od profesjonalistów w dziedzinie testowania, gdy przechodzą na stanowiska Lidera Testów i Kierownika Testów oraz Dyrektora Testów. W niniejszym sylabusie odnosimy się wspólnie do tych profesjonalistów nazywając ich Kierownikami Testów, rozumiejąc zarazem, że różne organizacje będą miały różne definicje tych stanowisk i poziomów odpowiedzialności osób na tych stanowiskach.

2.2. Zarządzanie testami w kontekście

Głównym obowiązkiem Kierownika Testów jest zapewnienie i wykorzystanie zasobów (ludzi, oprogramowania, sprzętu, infrastruktury, itd.) w celu przeprowadzenia procesów zwiększających wartość rozumianą jako wkład w powodzenie projektu. Dla kierowników oprogramowania i IT, procesy są zazwyczaj częścią projektu lub programu, który ma na celu dostarczenie oprogramowania lub systemu do wykorzystania wewnętrznego lub zewnętrznego. Dla Kierowników Testów procesy te to procesy wiążące się z testowaniem, a dokładnie z czynnościami podstawowego procesu testowego opisanego w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB® oraz w Rozdziale 1. niniejszego sylabusu. Skoro każdy proces testowy zwiększa wartość jedynie poprzez wniesienie wkładu do ogólnego powodzenia projektu lub programu (lub poprzez zapobieganie awariom bardziej dotkliwego rodzaju), Kierownik Testów musi właściwie zaplanować i sprawować nadzór nad procesem testowym. Innymi słowy, Kierownik Testów musi odpowiednio zorganizować proces testowy, wraz z powiązanymi czynnościami i produktami pracy, stosownie do potrzeb i okoliczności innych interesariuszy, ich czynności (np. cykl życia wytwarzania oprogramowania, w którym występuje testowanie) oraz ich produktów pracy (np. wymagań, specyfikacji).

2.2.1. Zrozumienie interesariuszy testowania

Osoby są interesariuszami testów, gdy są zaangażowane w czynności testowe, tworzenie produktów powstałych w wyniku testowania lub w działania związane z jakością ostatecznego systemu lub dostarczanego elementu. Udział interesariusza może obejmować bezpośrednie lub pośrednie zaangażowanie w czynności testowe, bezpośrednie lub pośrednie otrzymanie produktów powstałych w wyniku testowania lub pozostawanie pod bezpośrednim lub pośrednim wpływem jakości materiałów produkowanych w trakcie prac projektowych lub prac związanych z programem.

Chociaż interesariusze testów są różni w zależności od projektu, produktu, organizacji i innych czynników, zwykle należą do nich:

- Programiści, liderzy i kierownicy zespołów tworzących oprogramowanie. Interesariusze ci wdrażają oprogramowanie podlegające testowaniu, otrzymują wyniki testów i często muszą podejmować działania w oparciu o te wyniki (np. naprawić zgłoszone defekty).
- Architekci baz danych, architekci systemowi i projektanci. Interesariusze ci projektują oprogramowanie, otrzymują wyniki testów, a często podejmują działania w oparciu o te wyniki.
- Analitycy marketingowi i biznesowi. Interesariusze ci określają cechy oraz poziom jakości nieodłączny dla tych cech, które oprogramowanie musi posiadać. Często są również zaangażowani w definiowanie pożądanego pokrycia testami, w przeglądy wyników testów oraz w podejmowanie decyzji w oparciu o wyniki testów.
- Kierownictwo wyższego szczebla, kierownicy produktu i sponsorzy projektu. Interesariusze ci są często zaangażowani w definiowanie wymaganego pokrycia testami, przeglądy wyników testów oraz podejmowanie decyzji w oparciu o wyniki testów.
- Kierownicy projektów. Interesariusze ci są odpowiedzialni za zarządzanie swoimi projektami, aby odniósł sukces, co wymaga zrównoważenia priorytetów jakości, harmonogramu, cech

- i budżetu. Często pozyskują zasoby wymagane do czynności testowych oraz współpracują z Kierownikiem Testów w planowaniu testów i nadzoru nad testami.
- Wsparcie techniczne, wsparcie klienta i pracownicy wsparcia technicznego. Interesariusze ci wspierają użytkowników i klientów, którzy korzystają z cech i charakterystyk jakościowych dostarczonego oprogramowania.
- Użytkownicy bezpośredni i pośredni. Interesariusze ci korzystają bezpośrednio z oprogramowania (tj. są użytkownikami końcowymi) lub otrzymują dane wyjściowe lub usługi dostarczane lub wspierane przez oprogramowanie.

W celu uzyskania dalszych informacji na temat interesariuszy testów zob. Rozdział 2. w [Goucher09].

Powyższa lista interesariuszy nie jest pełna. Kierownicy Testów muszą zidentyfikować konkretnych interesariuszy testowania dla ich projektu lub programu. Kierownik Testów musi również rozumieć dokładny charakter powiązania interesariusza z testowaniem oraz to, jak zespół testowy spełnia potrzeby interesariuszy. Oprócz zidentyfikowania interesariuszy testów, jak opisano powyżej, Kierownik Testów powinien zidentyfikować inne czynności cyklu wytwarzania oprogramowania i produkty, które mają wpływ na testowanie i/lub które pozostają pod wpływem testowania. Bez tego proces testowy może nie osiągnąć optymalnej efektywności i wydajności (zob. Sekcja 2.2.3.).

2.2.2. Dodatkowe czynności cyklu życia wytwarzania oprogramowania i produkty pracy

Skoro testowanie oprogramowania jest oceną jakości jednego lub więcej produktów wytworzonych niezależnie od czynności testowych, zwykle istnieje ono w kontekście szerszego zbioru czynności cyklu wytwarzania oprogramowania. Kierownik Testów musi zaplanować i pokierować czynnościami testowymi ze zrozumieniem, jak inne czynności i ich produkty wpływają na testowanie, jak zostało omówione w sylabusie Certyfikowany tester Sylabus poziomu podstawowego ISTQB® oraz zdawać sobie sprawę z faktu, jak testowanie wpływa na inne czynności i ich produkty.

W organizacjach wykorzystujących zwinne metodyki wytwarzania oprogramowania, programiści często stosują technikę wytwarzania sterowanego testami, projektują automatyczne testy jednostkowe oraz ciągle integrują kod (wraz z testami do tego kodu) z systemem zarządzania konfiguracją. Kierownik Testów powinien współpracować z kierownikiem zespołu tworzącego oprogramowanie w celu upewnienia się, że testerzy są włączeni w te czynności i na nie ukierunkowani. Testerzy mogą wykonywać przeglądy testów jednostkowych zarówno, aby przedstawić swoje sugestie dotyczące zwiększenia pokrycia i efektywności tych testów, jak i w celu uzyskania głębszego zrozumienia zasad działania oprogramowania i jego implementacji. Testerzy mogą oceniać sposoby włączenia własnych automatycznych testów, zwłaszcza funkcjonalnych testów regresji, do systemu zarządzania konfiguracją. [Crispin09]

Choć konkretny związek pomiędzy czynnościami testowymi, innymi interesariuszami testowymi, czynnościami w cyklu wytwarzania oprogramowania a produktami pracy zmienia się zależnie od projektu, wybranego cyklu wytwarzania i szeregu innych czynników, testowanie jest ściśle połączone i powiązane z:

- Inżynierią i zarządzaniem wymaganiami. Kierownik Testów musi dokonać oceny wymagań w trakcie określania zakresu i szacowania pracochłonności testowania, jak również musi być świadomy zmian wymagań i wykonywać czynności nadzoru nad testami w celu dostosowania się do tych zmian. Techniczni Analitycy Testów i Analitycy Testów powinni brać udział w przeglądach wymagań.
- Zarządzaniem projektem. Kierownik Testów, współpracując z Analitykami Testów i Technicznymi Analitykami Testów, musi przedstawić Kierownikowi Projektu harmonogram i wymagania dotyczące wykorzystania zasobów. Kierownik Testów musi współpracować

- z Kierownikiem Projektu w celu zrozumienia zmian w planie projektu i wykonania czynności nadzoru nad testami, aby dostosować się do tych zmian.
- Zarządzaniem konfiguracją, zarządzaniem wydaniem oraz zarządzaniem zmianą. Kierownik Testów, współpracując z zespołem testowym, musi zdefiniować procesy i mechanizmy dostarczania obiektów testów oraz ująć je w planie testów. Kierownik Testów może poprosić Analityków Testów i Technicznych Analityków Testów o zaprojektowanie testów weryfikujących wersję oprogramowania i zapewnienie kontroli wersji w czasie wykonywania testów.
 - Tworzeniem oprogramowania i jego pielęgnacją. Kierownik Testów powinien współpracować z Kierownikami Zespołów tworzących oprogramowanie w celu koordynacji dostarczania obiektów testów, wraz z zawartością i datami każdego wydania testowego, jak również uczestniczenia w zarządzaniu defektami (zob. Rozdział 4.).
 - Wsparciem technicznym. Kierownik Testów powinien współpracować z Kierownikiem Wsparcia Technicznego w celu zapewnienia prawidłowego dostarczenia wyników testów w czasie zamknięcia testów tak, aby osoby zaangażowane we wsparcie produktu po wydaniu wiedziały o znanych awariach i rozwiązaniach tymczasowych. Ponadto, Kierownik Testów powinien współpracować z Kierownikiem Wsparcia Technicznego w celu analizy awarii produkcyjnych, aby wdrożyć udoskonalenia procesu testowego.
 - Tworzeniem dokumentacji technicznej. Kierownik Testów powinien współpracować z Kierownikiem ds. Dokumentacji Technicznej, aby zapewnić dostarczenie dokumentacji do testowania w terminie oraz zapewnić zarządzanie defektami w tych dokumentach.

Oprócz zidentyfikowania interesariuszy testów, jak opisano powyżej, Kierownik Testów musi zidentyfikować inne czynności cyklu wytwarzania oprogramowania i produktów pracy, które mają wpływ na testowanie i/lub na które testowanie wywiera wpływ. W innym przypadku proces testowy nie osiągnie optymalnej efektywności i wydajności.

2.2.3. Zestrojenie czynności testowych i innych czynności cyklu życia

Testowanie powinno być integralną częścią projektu, niezależnie od tego, jakie modele wytwarzania oprogramowania są wykorzystywane. Obejmuje to:

- Modele sekwencyjne, takie jak model kaskadowy, model V oraz model W. W modelu sekwencyjnym wszystkie produkty pracy i czynności charakterystyczne dla danego etapu (np. specyfikacja wymagań, projektowanie, implementacja, testowanie jednostkowe, testowanie integracyjne, testowanie systemowe oraz testowanie akceptacyjne) zostają zakończone zanim rozpocznie się kolejny etap. Planowanie testów, analiza testów, projektowanie testów oraz implementacja testów zazębia się z planowaniem projektu, analizą biznesową / wymagań, projektowaniem oprogramowania i bazy danych oraz programowaniem, a dokładny charakter tego procesu zależy od poziomu testów, o którym mowa. Wykonywanie testów następuje sekwencyjnie zgodnie z poziomami testów omówionymi w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB® oraz w niniejszym sylabusie.
- Modele iteracyjne i przyrostowe, takie jak Rapid Application Development (RAD) oraz Rational Unified Process (RUP). W modelu iteracyjnym lub przyrostowym cechy, które mają zostać zaimplementowane, są grupowane razem (np. według priorytetu biznesowego lub ryzyka), a następnie dla każdej grupy cech są realizowane różne etapy projektu, łącznie z czynnościami charakterystycznymi dla tych etapów i ich produktami pracy. Etapy projektu mogą być realizowane sekwencyjnie albo w sposób zazębiający się, a same iteracje mogą występować w sekwencji lub pokrywać się. W trakcie rozpoczęcia projektu, planowanie testów wysokopoziomowych oraz analiza testów występują równolegle do planowania projektu i analizy biznesowej lub analizy wymagań. Szczegółowe planowanie, analiza, projektowanie oraz wdrożenie testów następują na początku każdej iteracji, w sposób zazębiający się. Wykonanie testów często pociąga za sobą nakładanie się na siebie poziomów testów. Każdy poziom testów rozpoczyna się możliwie najwcześniej i może trwać po rozpoczęciu kolejnych, wyższych poziomów testów.

- Metodyki zwinne, takie jak Scrum lub Extreme Programming (XP). Są to iteracyjne cykle życia wytwarzania oprogramowania, w których iteracje są bardzo krótkie (często dwa do czterech tygodni). Produkty pracy i czynności dla każdej iteracji zostają zakończone przed rozpoczęciem kolejnej iteracji (tj. iteracje są sekwencyjne). Testowanie odbywa się podobnie jak w modelach iteracyjnych, jednak przy wyższym stopniu zazębiania się różnych czynności testowych z czynnościami wytwarzania, łącznie ze znaczącym nakładaniem się wykonywania testów (na różnych poziomach) z czynnościami wytwarzania. Wszystkie czynności wykonywane w danej iteracji, łącznie z czynnościami testowymi, powinny zostać zakończone przed rozpoczęciem kolejnej iteracji. W projekcie realizowanym w metodyce zwinnej, rola Kierownika Testów zwykle zmienia się z bezpośredniej roli kierowniczej w rolę autorytetu technicznego / w rolę doradczą.
- Model spiralny. W modelu spiralnym, na wczesnym etapie projektu, wykorzystuje się prototypy, aby potwierdzić wykonalność i poeksperymentować z decyzjami projektowymi i implementacyjnymi, w oparciu o poziom priorytetu biznesowego i ryzyka technicznego do wybrania kolejności, w jakiej wykonywane są eksperymenty prototypowe. Prototypy testuje się, aby określić, jakie aspekty problemów technicznych pozostają nierozwiązane. Gdy tylko problemy techniczne zostaną rozwiązane, projekt jest kontynuowany zgodnie z modelem sekwencyjnym lub iteracyjnym.

Aby poprawnie zestroić czynności testowe w ramach cyklu życia wytwarzania oprogramowania, Kierownik Testów musi posiadać dokładną wiedzę na temat modeli tego cyklu życia wykorzystywanych w jego organizacji. Na przykład w modelu V podstawowy proces testowy ISTQB® zastosowany do poziomu testów systemowych mógłby zostać dostosowany do wybranego cyklu życia wytwarzania oprogramowania w następujący sposób:

- Czynności planowania testów systemowych występują jednocześnie z planowaniem projektu, a nadzór nad testami jest kontynuowany aż do momentu ukończenia wykonywania i zamknięcia testów systemowych.
- Czynności analizy i projektowania testów systemowych występują jednocześnie ze specyfikacją wymagań, specyfikacją projektu systemowego i architektury (wysokopoziomowego) oraz specyfikacją projektu modułów (niskopoziomowego).
- Czynności wdrożenia testów systemowych mogą rozpocząć się w czasie projektowania systemu, jednakże większość tych czynności zazwyczaj występuje jednocześnie z kodowaniem i testami komponentów, z pracą nad czynnościami wdrożenia testów systemowych kończącą się często zaledwie kilka dni przed rozpoczęciem wykonania testów systemowych.
- Czynności wykonania testów systemowych rozpoczynają się, gdy wszystkie kryteria wejścia testów systemowych zostaną spełnione (lub zrezygnuje się z nich), co zwykle oznacza, że zakończone jest przynajmniej testowanie komponentów, a często także testowanie integracji komponentów. Wykonanie testów systemowych trwa do momentu spełnienia kryteriów wyjścia testów systemowych.
- Ocena kryteriów wyjścia testów systemowych oraz raportowanie wyników testów systemowych następują przez cały okres wykonywania testów systemowych, zwykle z większą częstotliwością i w większym pośpiechu, w miarę zbliżania się terminu zakończenia projektu.
- Czynności zamykające testy systemowe występują po spełnieniu kryteriów wyjścia testów systemowych, a wykonanie testów systemowych zostaje uznane za zakończone; czynności te mogą być czasami odłożone na później, do momentu zakończenia testów akceptacyjnych i zakończenia wszystkich czynności projektowych.

W iteracyjnym lub przyrostowym cyklu życia wytwarzania oprogramowania należy wykonywać te same zadania, ale może się różnić ich koordynacja i zakres. Na przykład, zamiast wdrażać całe środowisko testowe na początku projektu, bardziej efektywne może być wdrożenie jedynie jego części potrzebnej do aktualnej iteracji. W przypadku któregośkolwiek z iteracyjnych lub przyrostowych modeli cyklu życia

wytwarzania oprogramowania, im bardziej dalekosiężne jest planowanie, tym bardziej dalekosiężny może być zakres podstawowego procesu testowego.

Oprócz faz planowania, które występują w każdym projekcie, na wykonanie testów i raportowanie może mieć wpływ również cykl życia wytwarzania oprogramowania wykorzystywany przez zespół. Na przykład w iteracyjnym cyklu życia efektywne może być tworzenie kompletnych raportów i przeprowadzanie sesji przeglądów po iteracjach przed rozpoczęciem kolejnej iteracji. Traktując każdą iterację jako mini-projekt, zespół ma szansę wprowadzenia korekt w oparciu o to, co miało miejsce w czasie poprzedniej iteracji. Ponieważ iteracje mogą być krótkie i czasowo ograniczone, może być uzasadnione skrócenie czasu i ograniczenie pracochłonności raportowania i oceniania, ale zadania należy wykonywać w sposób umożliwiający śledzenie ogólnych postępów testowania oraz możliwie najszybsze zidentyfikowanie jakichkolwiek obszarów problemowych. Problemy związane z procesem, które miały miejsce w jednej iteracji, z dużym prawdopodobieństwem mogą mieć wpływ na kolejną iterację, a nawet mogą ponownie wystąpić w kolejnej iteracji, jeżeli nie zostaną podjęte działania naprawcze.

Ogólne informacje na temat tego, jak dopasować proces testowy do innych czynności cyklu życia wytwarzania oprogramowania, mogą zostać ujęte w strategii testów (zob. Sekcja 2.4.2.). W czasie planowania testów i/lub planowania projektu, Kierownik Testów powinien dopasować do siebie czynności specyficzne dla danego projektu, dla każdego poziomu testów oraz dla wszystkich wybranych kombinacji cyklu życia wytwarzania oprogramowania i procesu testowego.

Zależnie od potrzeb organizacji, projektu i produktu, mogą być konieczne dodatkowe, poza zdefiniowanymi w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB®, poziomy testowania, takie jak:

- testowanie integracyjne sprzętu z oprogramowaniem,
- testowanie integracyjne systemu,
- testowanie interakcji funkcji,
- testowanie integracji produktu klienta.

Każdy poziom testów powinien mieć jasno zdefiniowane następujące elementy:

- cele testów z ich efektami,
- zakres testów i elementy testowe,
- podstawę testów wraz ze sposobem zmierzenia pokrycia tej podstawy (tj. śledzenie powiązań),
- kryteria wejścia i wyjścia,
- produkty testowania wraz z raportowaniem wyników,
- odpowiednie techniki testów wraz ze sposobem zapewnienia stosownych stopni pokrycia przy wykorzystaniu tych technik,
- pomiary i metryki istotne dla celów testowania, kryteriów wejścia i wyjścia oraz raportowania wyników (wraz z pomiarami pokrycia),
- narzędzia testowe, które mają zostać zastosowane do konkretnych zadań testowych (tam, gdzie ma zastosowanie),
- zasoby (np. środowiska testowe),
- odpowiedzialne osoby i grupy, zarówno wewnątrz, jak i na zewnątrz zespołu testowego,
- zgodność ze standardami organizacyjnymi, prawnymi i innymi (tam, gdzie ma zastosowanie).

Jak omówiono w dalszej części niniejszego rozdziału, najlepszą praktyką jest definiowanie tych elementów w sposób spójny na wszystkich poziomach testowania, aby uniknąć nieekonomicznych i niebezpiecznych luk na różnych poziomach podobnych testów.

2.2.4. Zarządzanie testowaniem нефункциональным

Nieprzygotowanie planu dla testów нефункциональных może skutkować wykryciem poważnych, czasami katastrofalnych, problemów związanych z jakością systemu po wydaniu. Jednakże wiele typów testów нефункциональных jest tak drogich, że Kierownik Testów musi w oparciu o ryzyko i ograniczenia dokonać wyboru, które z nich wykonać. Ponadto, istnieje wiele różnych typów testów нефункциональных i niektóre z nich mogą nie być odpowiednie dla danego systemu.

Z uwagi na fakt, że Kierownik Testów może nie posiadać dostatecznej wiedzy specjalistycznej, aby zająć się wszystkimi aspektami planowania, musi przekazać część zadań związanych z planowaniem testów Technicznemu Analitykowi Testów (a w niektórych przypadkach Analitykom Testów) przydzielonemu do czynności testowania нефункционального. Kierownik Testów powinien poprosić analityków o uwzględnienie następujących ogólnych czynników:

- wymagań interesariuszy,
- wymaganych narzędzi,
- środowiska testowego,
- czynników organizacyjnych,
- bezpieczeństwa.

Więcej informacji na ten temat znajduje się w Rozdziale 4. Sylabusu dla poziomu zaawansowanego Techniczny Analityk Testów na Poziomie Zaawansowanym [ISTQB ATTA SYL].

Inną istotną kwestią, którą Kierownicy Testów powinni mieć na względzie, jest to, jak zintegrować testy нефункциональные z cyklem życia wytwarzania oprogramowania. Powszechnym błędem jest oczekiwanie na zakończenie wszystkich testów funkcjonalnych, zanim rozpoczną się testy нефункциональные, co może skutkować późnym odkryciem krytycznych defektów нефункциональных. Zamiast tego, należy nadać priorytet testom нефункциональным i ustawić je w kolejności według ryzyka. Często istnieją sposoby na łagodzenie ryzyka нефункционального na wczesnych poziomach testowania lub nawet w czasie wytwarzania oprogramowania. Na przykład, przeglądy użyteczności prototypów interfejsu użytkownika w czasie projektowania systemu mogą być dość skuteczne w identyfikowaniu defektów użyteczności, które spowodowałyby istotne problemy z realizacją prac zgodnie z harmonogramem, jeżeli zostałyby odkryte pod koniec testowania systemowego.

W iteracyjnych cyklach życia, tempo zmian oraz iteracji może utrudnić zorientowanie na pewne testy нефункциональные, które wymagają zbudowania skomplikowanej infrastruktury testów. Czynności projektowania i implementacji testów, które trwają dłużej niż ramy czasowe pojedynczej iteracji, należy zorganizować jako oddzielne czynności poza iteracjami.

2.2.5. Zarządzanie testowaniem opartym na doświadczeniu

Testowanie oparte na doświadczeniu zapewnia korzyści przez efektywne znajdowanie defektów, które mogą zostać pominięte przy stosowaniu innych technik testowania i funkcjonuje jako mechanizm kontroli kompletności tych technik, niemniej jednak stawia ono pewne wyzwania wobec procesu zarządzania testami. Kierownik Testów powinien mieć świadomość wyzwań, a także korzyści płynących z technik opartych na doświadczeniu, w szczególności z testowania eksploracyjnego. Trudno jest określić pokrycie osiągnięte podczas takiego testowania, przy typowym lekkim stopniu logowania i minimalnym wcześniejszym przygotowaniu testów. Powtarzalność wyników testów wymaga szczególnej uwagi kierownictwa, zwłaszcza, gdy w proces testowania zaangażowanych jest kilku testerów.

Jednym ze sposobów zarządzania testowaniem opartym na doświadczeniu, zwłaszcza testowaniem eksploracyjnym, jest rozdzielenie pracy testowej na krótkie okresy trwające od 30 do 120 minut, zwane czasem sesjami testowymi. Takie zarządzanie czasem ogranicza i orientuje prace, które mają zostać

wykonane w trakcie sesji i zapewnia odpowiedni poziom monitorowania i planowania. Każda sesja jest dokumentowana przy pomocy karty opisu testu, którą Kierownik Testów przedstawia testerowi na piśmie lub ustnie. Karta opisu testu zawiera opis warunku lub warunków testowych, które mają zostać pokryte w trakcie sesji testowej, co dodatkowo pomaga utrzymać orientację i zapobiega nakładaniu się różnych czynności testowych, jeżeli wiele osób jednocześnie wykonuje testowanie eksploracyjne.

Inną techniką zarządzania testowaniem opartym na doświadczeniu jest włączenie takiego samodzielnego i spontanicznego testowania w bardziej tradycyjne, wcześniej zaprojektowane sesje testowe. Na przykład testerzy mogą otrzymać pozwolenie (i przydzielony czas) na eksplorowanie poza wyraźnie zdefiniowane kroki, dane wejściowe i oczekiwane wyniki w predefiniowanych testach. Można również przydzielić testerom samodzielne sesje testowe jako część codziennego testowania przed, w czasie lub po dniu wykonywania predefiniowanych testów. Jeżeli podczas takich sesji testowania zidentyfikowane zostaną defekty lub interesujące obszary do przyszłego testowania, testy predefiniowane mogą zostać uaktualnione.

Na początku sesji testowania eksploracyjnego, tester ustala i wykonuje niezbędne zadania przygotowujące do przeprowadzenia testów. W czasie sesji, tester uczy się aplikacji, która jest przedmiotem testów, projektuje i wykonuje testy zgodnie z wykorzystywaną techniką i tym, czego dowiedział się o aplikacji, bada defekty i zapisuje wyniki testów w logu. (Jeżeli wymagana jest powtarzalność testów, testerzy powinni również logować dane wejściowe testów, czynności i wydarzenia.) Po sesji może nastąpić omówienie wykonanego zadania, co nadaje kierunek kolejnym sesjom.

2.3. Testowanie oparte na ryzyku i inne podejścia do priorytetyzacji testów i przydzielania prac związanych z testowaniem

Powszechnym wyzwaniem dla kierownictwa testów jest odpowiednie wybranie, przydzielenie testów i nadanie im priorytetów. Oznacza to, że spośród praktycznie nieskończonej liczby warunków testowych i kombinacji warunków, które mogą zostać pokryte, zespół testowy musi wybrać skończony zbiór warunków, określić odpowiednią ilość pracy, która ma zostać przydzielona w celu pokrycia każdego warunku przypadkami testowymi i ułożenia wynikających z tych warunków przypadków testowych w hierarchii ważności, co optymalizuje skuteczność i wydajność prac testowych, które mają zostać wykonane. Pomóc w rozwiązaniu tego problemu może zastosowanie przez Kierownika Testów identyfikacji i analizy ryzyka, wraz z innymi czynnikami, jednak wiele wzajemnie oddziałujących na siebie ograniczeń i zmiennych może wymagać rozwiązania kompromisowego.

2.3.1. Testowanie oparte na ryzyku

Ryzyko to możliwość otrzymania negatywnego lub niepożądanego wyniku lub wydarzenia. Ryzyka istnieją zawsze, gdy może pojawić się problem, który obniżyłby postrzeganie jakości produktu lub powodzenia projektu przez klienta, użytkownika, uczestnika lub innego interesariusza. Gdy potencjalny problem zasadniczo wpływa na jakość produktu, problemy takie określa się jako ryzyka jakościowe, ryzyka produktowe lub ryzyka jakości produktu. Gdy potencjalny problem zasadniczo wpływa na powodzenie projektu, problemy takie określa się jako ryzyka projektowe lub ryzyka planowania.

W testowaniu opartym na ryzyku, w czasie analizy ryzyka jakości produktu z interesariuszami, identyfikuje się i ocenia ryzyka jakościowe. Następnie zespół testowy projektu implementuje i wykonuje testy tak, aby złagodzić ryzyka jakościowe. Jakość obejmuje całość funkcji, zachowań, cech i atrybutów, które mają wpływ na zadowolenie klientów, użytkowników oraz innych interesariuszy. Dlatego też ryzyko jakościowe to potencjalna sytuacja, w której w produkcji mogą zaistnieć problemy jakościowe. Przykłady ryzyk jakościowych dla systemu obejmują niepoprawne obliczenia w raportach (ryzyko funkcjonalne związane z dokładnością), wolną reakcję na wprowadzanie danych przez

użytkownika (niefunkcjonalne ryzyko związane z wydajnością i czasem reakcji) oraz trudność w zrozumieniu ekranów i pól (niefunkcjonalne ryzyko związane z użytecznością i zrozumiałością). Gdy testy ujawniają defekty, testowanie zmniejsza ryzyko jakościowe przez dostarczenie wiedzy o defektach i możliwość ich usunięcia przed wydaniem. Gdy w wyniku testowania nie zostaną znalezione defekty, testowanie zmniejsza ryzyko jakościowe poprzez zapewnienie, że w testowanych warunkach system działa poprawnie.

W testowaniu opartym na ryzyku wykorzystuje się ryzyka jakości produktu, aby wybrać warunki testowe, przydzielić do tych warunków prace testowe i nadać priorytety wynikowym przypadkom testowym. Istnieją różne techniki testowania opartego na ryzyku, różniące się typem i poziomem zgromadzonej dokumentacji, a także zastosowanym poziomem sformalizowania. Celem testowania opartego na ryzyku jest pośrednie lub bezpośrednie wykorzystanie testowania do zmniejszenia ogólnego poziomu ryzyka jakościowego, a w szczególności do obniżenia ryzyka do dopuszczalnego poziomu.

Testowanie oparte na ryzyku obejmuje cztery główne czynności:

- identyfikacja ryzyka,
- ocena ryzyka,
- łagodzenie ryzyka,
- zarządzanie ryzykiem.

Czynności te nakładają się na siebie. Każda z nich zostanie omówiona w kolejnych akapitach.

Aby identyfikacja i ocena ryzyka były jak najbardziej efektywne, czynności te powinny obejmować przedstawicieli wszystkich grup interesariuszy projektu i produktu, jednak czasami realia projektu powodują, że niektórzy interesariusze działają jako zastępcy innych interesariuszy. Na przykład, w tworzeniu oprogramowania na rynek masowy, mała grupa potencjalnych klientów może zostać poproszona o pomoc w zidentyfikowaniu potencjalnych defektów, które miałyby największy wpływ na wykorzystanie przez nich oprogramowania; w tym przypadku próbka potencjalnych klientów zastępuje całą bazę możliwych klientów. Z powodu swojej szczególnej wiedzy na temat ryzyk jakości produktu i awarii, testerzy powinni być aktywnie zaangażowani w proces identyfikacji i oceny ryzyka.

2.3.1.1. Identyfikacja ryzyka

Interesariusze mogą identyfikować ryzyka za pomocą jednej lub więcej następujących technik:

- rozmowy z ekspertami,
- niezależne oceny,
- wykorzystanie szablonów ryzyka,
- retrospektywy projektu,
- warsztaty dotyczące ryzyka,
- burza mózgów,
- listy kontrolne,
- odwołanie się do przeszłego doświadczenia.

Włączenie jak najszerszej, reprezentatywnej grupy interesariuszy zwiększa szanse na to, że proces identyfikacji ryzyka umożliwi określenie istotnych ryzyk jakości produktu.

Identyfikacja ryzyka często skutkuje wytworzeniem produktów ubocznych, tj. służy identyfikacji problemów, które nie są ryzykami jakości produktu. Przykłady obejmują ogólne pytania lub problemy dotyczące produktu lub projektu lub problemy w dokumentach, do których są odniesienia, takich jak wymagania i specyfikacje projektu. Produktami ubocznymi identyfikacji ryzyka jakościowego są również ryzyka projektowe, ale nie są one w centrum zainteresowania testowania opartego na ryzyku. Zarządzanie ryzykiem projektowym jest jednakże ważne dla całokształtu testowania, nie tylko testowania opartego na ryzyku i zostało omówione szerzej w Sekcji 2.4.

2.3.1.2. Ocena ryzyka

Po zidentyfikowaniu ryzyka, można dokonać jego oceny, czyli zbadać zidentyfikowane ryzyka. Ocena ryzyka obejmuje w szczególności skategoryzowanie każdego ryzyka i określenie prawdopodobieństwa i wpływu związanego z każdym ryzykiem. Ocena ryzyka może również obejmować ocenę i przypisanie innych cech każdemu ryzyku, takich jak właściciel ryzyka.

Kategoryzacja ryzyka oznacza przypisanie każdego ryzyka do odpowiedniego typu, takiego jak wydajność, niezawodność, funkcjonalność itd. Niektóre organizacje korzystają z cech jakości opisanych w normie ISO 9126 [ISO9126]³ do kategoryzacji ryzyk, ale wiele organizacji korzysta z własnych schematów. Ta sama lista kontrolna, która została wykorzystana do identyfikacji ryzyka, jest często wykorzystywana do kategoryzowania ryzyk. Istnieją ogólne listy kontrolne ryzyka jakościowego i wiele organizacji dostosowuje je do swoich potrzeb. Przy korzystaniu z list kontrolnych jako podstawy do identyfikacji ryzyka, ryzyko jest często kategoryzowane w trakcie identyfikacji.

Dla każdej pozycji ryzyka, określenie poziomu ryzyka zwykle obejmuje ocenę prawdopodobieństwa jego wystąpienia i wpływu, jaki wywrze jego wystąpienie. Prawdopodobieństwo wystąpienia ryzyka to inaczej prawdopodobieństwo, że w systemie będącym przedmiotem testów istnieje potencjalny problem. Innymi słowy, prawdopodobieństwo to ocena poziomu ryzyka technicznego. Czynniki, które mają wpływ na prawdopodobieństwo dla ryzyka produktowego i projektowego, obejmują:

- złożoność technologii i zespołów,
- problemy z personelem i szkoleniami wśród analityków biznesowych, projektantów i programistów,
- konflikt w zespole,
- problemy związane z umowami z dostawcami,
- zespół rozproszony geograficznie,
- podejście tradycyjne kontra nowe podejścia,
- narzędzia i technologia,
- słabe przywództwo kierownicze lub techniczne,
- czas, zasoby, budżet i presja kierownictwa,
- brak wcześniejszych działań związanych z zapewnieniem jakości,
- wysoki współczynnik zmian,
- wysokie wcześniejsze współczynniki defektów,
- problemy dotyczące interfejsu i integracji.

Wpływ ryzyka po jego wystąpieniu to dotkliwość skutku dla użytkowników, klientów lub innych interesariuszy. Czynniki oddziałujące na wpływ defektu w ryzykach projektowych i produktowych obejmują:

- częstotliwość wykorzystania uszkodzonej funkcji,
- krytyczność funkcji dla osiągnięcia celu biznesowego,
- uszczerbek na reputacji,
- straty gospodarcze,
- potencjalne straty lub odpowiedzialność finansowa, ekologiczna lub społeczna,
- sankcje prawne, cywilne lub karne,
- utrata licencji,
- brak rozsądnego rozwiązania tymczasowego,
- widoczność awarii prowadząca do negatywnego rozgłosu,
- bezpieczeństwo.

³ Standard ISO/IEC 25010 zastąpił normę ISO 9126.

Poziom ryzyka można ocenić ilościowo lub jakościowo. Jeżeli prawdopodobieństwo i wpływ ryzyka można ustalić ilościowo, można pomnożyć te dwie wartości, aby obliczyć wartość liczbową priorytetu ryzyka ilościowego. Zwykle jednak poziom ryzyka można ustalić jedynie jakościowo. Oznacza to, że można mówić o prawdopodobieństwie bardzo wysokim, wysokim, średnim, niskim lub bardzo niskim, ale nie można wyrazić go precyzyjnie procentowo; podobnie można mówić o wpływie ryzyka, który jest bardzo wysoki, wysoki, średni, niski lub bardzo niski, ale nie można wyrazić takiego wpływu w kryteriach finansowych w sposób kompletny lub dokładny. Ocena jakościowa poziomów ryzyka nie powinna być postrzegana jako coś gorszego od podejścia ilościowego; w rzeczywistości, gdy oceny ilościowe poziomów ryzyka są wykorzystywane nieodpowiednio, wyniki wprowadzają interesariuszy w błąd w kwestii zakresu, w jakim rozumie się ryzyko, a dzięki ocenie jakościowej można tym ryzykiem zarządzać. Oceny jakościowe poziomów ryzyka są często łączone poprzez mnożenie lub dodawanie, w celu stworzenia sumarycznego wskaźnika ryzyka, który może zostać wyrażony jako priorytetu ryzyka, ale powinna być interpretowana jako relatywny wskaźnik jakościowy na skali porządkowej.

Ponadto, jeżeli analiza ryzyka nie opiera się na dokładnych i statystycznie istotnych danych na temat ryzyka, analiza ryzyka będzie opierać się na subiektywnym postrzeganiu interesariuszy w kwestii prawdopodobieństwa i wpływu ryzyka. Kierownicy projektu, programiści, użytkownicy, analitycy biznesowi, architekci i testerzy zazwyczaj prezentują różne perspektywy postrzegania i dlatego też są możliwe różne opinie na temat poziomu ryzyka dla każdej pozycji ryzyka. Proces analizy ryzyka powinien obejmować pewien sposób osiągnięcia konsensusu lub, w najgorszym przypadku, ustalenia uzgodnionego poziomu ryzyka (np. poprzez nakaz kierownictwa lub poprzez wyliczenie średniej, mediany lub wartości modalnej ryzyka dla pozycji ryzyka). Ponadto należy sprawdzać poziomy ryzyka pod kątem odpowiedniego rozkładu w całym zakresie testów, aby upewnić się, że współczynniki ryzyka zapewniają znaczące wskazówki w kwestii ustalenia kolejności testów, ich priorytetów i przydziału prac związanych z testowaniem. W przeciwnym przypadku, nie można wykorzystać poziomów ryzyka jako wskazówek do czynności związanych z łagodzeniem ryzyka.

2.3.1.3. Łagodzenie ryzyka

Testowanie oparte na ryzyku rozpoczyna się od analizy ryzyka jakościowego (identyfikacji i oceny ryzyk jakości produktu). Analiza ta jest podstawą głównego planu testów i innych planów testów. Jak zostało to określone w planie (planach), testy są projektowane, implementowane i wykonywane w celu pokrycia ryzyk. Pracochłonność opracowania i wykonania testu jest proporcjonalna do poziomu ryzyka, co oznacza, że bardziej drobiazgowo techniki testowania (takie jak testowanie sposobem par) są wykorzystywane przy wyższym poziomie ryzyka, podczas gdy mniej skrupulatne techniki testowania (takie jak klasy równoważności lub testowanie eksploracyjne ograniczone ramami czasowymi) są wykorzystywane dla niższych poziomów ryzyka. Ponadto, na poziomie ryzyka opiera się priorytet zaprojektowania i wykonania testu. Niektóre normy związane z bezpieczeństwem (np. FAA DO-178B/ED 12B⁴, IEC 61508) zalecają techniki testowania i stopień pokrycia w oparciu o poziom ryzyka. Ponadto, poziom ryzyka powinien mieć wpływ na decyzje takie jak wykorzystanie przeglądów produktów pracy projektowej (wraz z testami), poziom niezależności, poziom doświadczenia testera oraz stopień potwierdzenia testowania (ponowne testowanie), a także wykonane testowanie regresji.

W trakcie projektu, zespół testowy powinien mieć świadomość istnienia dodatkowych informacji, które zmieniają zestaw ryzyk jakościowych i/lub poziom ryzyka związany ze znanymi ryzykami jakościowymi. Powinna być wykonywana okresowa korekta rezultatów analizy ryzyka jakościowego, z której wynikają korekty testów. Korekty te powinny mieć miejsce przynajmniej w trakcie osiągania głównych kamieni milowych projektu. Korekty obejmują identyfikację nowych ryzyk, ponowną ocenę poziomu istniejących ryzyk oraz ocenę efektywności działań mających na celu łagodzenie ryzyka. Przykładowo, jeżeli w trakcie fazy pozyskiwania wymagań sesja identyfikacji ryzyka i oceny opierała się

⁴ Standard FAA DO-178B/ED 12B zastąpił normę FAA DO-178C/ED 12C

o specyfikację wymagań, ponowna ocena ryzyk powinna nastąpić po zakończeniu przygotowania specyfikacji projektu. Kolejny przykład - jeżeli w czasie testowania okaże się, że moduł zawiera znacznie większą liczbę defektów, niż oczekiwano, można dojść do wniosku, że prawdopodobieństwo wystąpienia defektów w tym obszarze jest wyższe, niż przewidywano i dlatego też należy zwiększyć prawdopodobieństwo wystąpienia i ogólny poziom ryzyka. Może to skutkować zwiększeniem liczby testów, które mają zostać wykonane na tym module.

Ryzyka jakościowe produktu można również złagodzić przed rozpoczęciem wykonania testów. Na przykład, jeżeli w trakcie identyfikacji ryzyka zostaną znalezione problemy z wymaganiami, zespół projektowy może wdrożyć jako działanie łagodzące ryzyko, dokładne przeglądy specyfikacji wymagań. Może to obniżyć poziom ryzyka, co może oznaczać, że w celu złagodzenia pozostałego ryzyka jakościowego będzie potrzebne wykonanie mniejszej liczby testów.

2.3.1.4. Zarządzanie ryzykiem w cyklu życia

W sytuacji idealnej, zarządzanie ryzykiem odbywa się podczas całego cyklu życia wytwarzania oprogramowania. Jeżeli organizacja posiada dokument polityki testów i/lub dokument strategii testów, wtedy powinny one opisywać ogólny proces, za pomocą którego zarządza się ryzykami produktowymi i projektowymi w testowaniu oraz pokazuje, jak zarządzanie ryzykiem łączy się ze wszystkimi etapami testowania i na nie wpływa.

W dojrzałej organizacji, w której zespół projektowy posiada pełną świadomość ryzyka, zarządzanie ryzykiem odbywa się na wielu poziomach, a nie tylko w odniesieniu do procesu testowania. Ważne ryzyka uwzględnia się nie tylko wcześniej na danych poziomach testów, ale również we wcześniejszych poziomach testów, na przykład, jeżeli jako kluczowy obszar ryzyka jakościowego zostanie zidentyfikowana wydajność, testowanie wydajnościowe nie tylko rozpocznie się wcześniej w testach systemowych, ale też testy wydajnościowe będą wykonywane w czasie testowania modułowego i integracyjnego. Dojrzałe organizacje nie tylko identyfikują ryzyka, ale również źródła ryzyka i konsekwencje tych ryzyk w przypadku ich zaistnienia. W przypadku defektów, które jednak wystąpiły, wykorzystuje się analizę przyczyny podstawowej w celu dokładniejszego zrozumienia źródeł ryzyka i wdrożenia udoskonaleń procesu, które przede wszystkim będą zapobiegać powstawaniu defektów. Łagodzenie ryzyka występuje w całym cyklu życia wytwarzania oprogramowania. Analiza ryzyka jest wykonywana przy pełnej znajomości wyników powiązanych działań, analizy zachowania systemu, oceny ryzyka w oparciu o koszty, analizy ryzyka produktowego, analizy ryzyka użytkownika końcowego oraz analizy ryzyka odpowiedzialności. Analiza ryzyka wykracza poza testowanie, a zespół testowy uczestniczy w analizie ryzyka obejmującej cały program i ma na nią wpływ.

Większość technik testowania opartego na ryzyku obejmuje również techniki służące do wykorzystania poziomu ryzyka przy ustalaniu kolejności testów i nadawaniu im priorytetów, tym samym zapewniając wczesne pokrycie najważniejszych obszarów i wykrycie najważniejszych defektów w czasie wykonywania testów. W niektórych przypadkach wszystkie testy o najwyższym ryzyku są wykonywane przed testami o niższym ryzyku i wykonywane są w ściśle określonej kolejności (podejście to często zwane jest „testowaniem w głąb”); w innych przypadkach stosuje się podejście polegające na badaniu wrywkowym, aby wybrać próbkę testów pochodzących ze wszystkich pozycji zidentyfikowanego ryzyka używając ryzyka do nadania wagi wyborowi, jednocześnie zapewniając pokrycie każdej pozycji ryzyka przynajmniej raz (często nazywane „testowaniem wszerz”).

Bez względu na to, czy testowanie odbywa się w głąb czy wszerz, istnieje ryzyko, że zostanie wykorzystany czas przydzielony na testowanie, a nie wszystkie testy zostaną wykonane. Testowanie oparte na ryzyku pozwala testerom informować kierownictwo w kategoriach pozostałego poziomu ryzyka w danym momencie i umożliwia kierownictwu podjęcie decyzji o tym, czy rozszerzyć testowanie, czy przenieść pozostałe ryzyko na użytkowników, klientów, dział wsparcia technicznego i/lub personel operacyjny.

Podczas wykonywania testów, monitorowanie i kontrolowanie cyklu życia wytwarzania oprogramowania, łącznie z podejmowaniem decyzji o wydaniu w oparciu o pozostały poziom ryzyka, umożliwiają uczestnikom projektu, kierownikom projektu i produktu, dyrektorom, kierownictwu wyższego szczebla oraz interesariuszom projektu zastosowanie najbardziej wyszukanych technik testowania opartego na ryzyku, które nie muszą być bardzo formalne ani najcięższe. Wymaga to, aby Kierownik Testów raportował wyniki testów w kategoriach ryzyka w sposób zrozumiały dla każdego interesariusza testów.

2.3.2. Techniki testowania opartego na ryzyku

Istnieje kilka technik testowania opartego na ryzyku. Niektóre z nich są nieformalne. Przykładem może być podejście, w którym tester analizuje ryzyka jakościowe w trakcie testowania eksploracyjnego [Whittaker09]. Może to ukierunkować testowanie, ale może też prowadzić do zbyt dużego skupiania się na prawdopodobieństwie wystąpienia defektów, a nie na ich wpływie, i nie obejmuje danych wejściowych interesariusza obejmujących więcej niż jedną funkcję. Takie podejście jest ponadto subiektywne i zależy od umiejętności, doświadczenia i preferencji indywidualnych testera. W gruncie rzeczy, stosując takie podejście, rzadko można osiągnąć pełne korzyści płynące z testowania opartego na ryzyku.

Próbując uchwycić korzyści płynące z testowania opartego na ryzyku przy minimalizowaniu kosztów, wielu praktyków stosuje lekkie podejścia oparte na ryzyku. Łączą one szybkość reakcji i elastyczność nieformalnych podejść z siłą i budowaniem konsensusu charakterystycznych dla podejść bardziej formalnych. Przykłady lekkich podejść to między innymi: Pragmatyczna Analiza i Zarządzanie Ryzykiem (PRAM) [Black09], Systematyczne Testowanie Oprogramowania (SST) [Craig02] oraz Zarządzanie Ryzykiem Produktowym (PRisMa) [vanVeenendaal12]. Oprócz zwyczajowych atrybutów testowania opartego na ryzyku, techniki te zazwyczaj mają następujące cechy:

- rozwinęły się stopniowo w oparciu o doświadczenie branżowe z testowaniem opartym na ryzyku, zwłaszcza w branżach, w których istotne są kwestie wydajności,
- opierają się na poważnym zaangażowaniu zespołu interesariuszy obejmującego wiele funkcji, reprezentujących zarówno biznesowe, jak i techniczne punkty widzenia w trakcie początkowej identyfikacji i oceny ryzyka,
- są optymalizowane, gdy zostają wprowadzane w trakcie najwcześniejszych faz projektu, kiedy istnieje najwięcej opcji łagodzenia ryzyk jakościowych i gdy główne produkty pracy i produkty uboczne analizy ryzyka mogą pomóc wpłynąć na specyfikację i wdrożenie produktu w sposób, który zminimalizuje ryzyko,
- korzystają z wygenerowanych danych wyjściowych (z macierzy ryzyka lub z analizy ryzyka), jako podstawy dla planu testów i warunków testowych, a zatem dla wszystkich późniejszych czynności zarządzania testami i analizy testów,
- wspierają raportowanie wyników testów w kategoriach pozostałego ryzyka interesariuszom na wszystkich poziomach testów.

Niektóre z tych technik (np. SST) do analizy ryzyka wymagają specyfikacji jako danych wejściowych wymagań i nie mogą zostać wykorzystane, jeśli specyfikacje wymagań nie są zapewnione. Inne techniki (np. PRisMa oraz PRAM) zachęcają do stosowania połączonej strategii opartej na ryzyku i wymaganiach, wykorzystując jako dane wejściowe do analizy ryzyka wymagania i/lub inne specyfikacje, ale mogą funkcjonować wyłącznie w oparciu o dane wejściowe interesariusza. Wykorzystanie wymagań jako danych wejściowych pomaga zapewnić dobre pokrycie wymagań, ale Kierownik Testów musi zagwarantować, że nie zostaną pominięte ważne ryzyka niezasugerowane wymaganiami, zwłaszcza w obszarach niefunkcjonalnych. Gdy jako dane wejściowe zapewnione są odpowiednie, uszeregowane pod kątem ważności wymagania, zwykle widoczna jest silna korelacja między poziomami ryzyka a priorytetami wymagań.

Wiele z tych technik zachęca również do wykorzystania procesu identyfikacji i oceny ryzyka jako sposobu na zbudowanie konsensusu między interesariuszami w sprawie podejścia do testowania. Jest to ogromna korzyść, ale wymaga poświęcenia przez interesariuszy czasu na uczestniczenie w grupowych sesjach burzy mózgów czy indywidualnych rozmowach. Niewystarczające uczestnictwo interesariuszy prowadzi do luk w analizie ryzyka. Oczywiście, interesariusze nie zawsze zgadzają się w kwestii poziomu ryzyka, więc osoba prowadząca analizę ryzyka jakościowego musi pracować kreatywnie i pozytywnie z interesariuszami, aby osiągnąć możliwie najwyższy poziom porozumienia. Do osoby prowadzącej analizę ryzyka jakościowego zastosowanie mają wszystkie umiejętności wyszkolonego moderatora prowadzącego spotkania przeglądowe.

Podobnie jak bardziej formalne techniki, techniki lekkie umożliwiają wykorzystanie wag dla czynników prawdopodobieństwa i wpływu do podkreślenia ryzyk biznesowych lub technicznych (na przykład, zależnie od poziomu testowania). Aczkolwiek w przeciwieństwie do technik bardziej formalnych, lekkie techniki:

- wykorzystują jedynie dwa czynniki: prawdopodobieństwo i wpływ oraz
- wykorzystują proste oceny jakościowe i skale.

Lekki charakter tych podejść zapewnia elastyczność, możliwość zastosowania w wielu dziedzinach oraz dostępność w zespołach o wszystkich poziomach doświadczenia i umiejętności (nawet z osobami bez wiedzy technicznej i małym doświadczeniu).

Na formalnym i „ciężkim” końcu spektrum, Kierownik Testów ma do dyspozycji kilka opcji:

- Analizę zagrożeń, która rozciąga się w górę procesu analitycznego, starając się zidentyfikować zagrożenia, które leżą u podstaw każdego ryzyka.
- Koszt narażenia na ryzyko, gdzie proces oceny ryzyka obejmuje określenie trzech czynników dla każdej pozycji ryzyka jakościowego: 1) prawdopodobieństwa (wyrażonego jako wartość procentowa) awarii związanej z pozycją ryzyka; 2) kosztu straty (wyrażonego jako wielkość finansowa) związanego z typową awarią związaną z pozycją ryzyka, gdyby awaria wystąpiła w systemie w produkcji; a także 3) kosztu testowania dla tych awarii.
- Analizę Przyczyn i Skutków Awarii (FMEA) i jej odmian [Stamatis03], w których identyfikowane są ryzyka jakościowe, ich potencjalne przyczyny i prawdopodobne skutki, a następnie przypisywane są współczynniki wagi, priorytetu i wykrywalności.
- Rozmieszczenie Funkcji Jakości (QFD), technikę zarządzania ryzykiem jakościowym z konsekwencjami dla testowania, szczególnie poprzez branie pod uwagę ryzyk, które powstają z niepoprawnego lub niewystarczającego zrozumienia wymagań klientów lub użytkowników.
- Analizę Drzewa Usterek (FTA), gdzie różne faktycznie zaobserwowane awarie (z testowania lub po produkcyjnym uruchomieniu systemu) lub potencjalne awarie (ryzyka jakościowe) są poddawane analizie przyczyny podstawowej rozpoczynając od defektów, które mogłyby spowodować awarię, następnie przez pomyłki i defekty, które mogą spowodować wyżej wymienione defekty, do momentu, w którym zostaną zidentyfikowane różne przyczyny podstawowe.

Konkretne techniki, które powinny zostać wykorzystane do testowania opartego na ryzyku oraz stopień sformalizowania każdej z nich, zależą od różnych aspektów projektu, procesu i produktu. Na przykład, nieformalne podejście, takie jak technika eksploracyjna Whittakera, może mieć zastosowanie do łatki czy szybkiej naprawy. W projektach zwinnych, analiza ryzyka jakościowego ściśle łączy się z wczesnym okresem każdego przebiegu, a dokumentacja ryzyk zostaje włączona w śledzenie historyjek użytkownika. Systemy systemów wymagają analizy ryzyka w każdym z systemów, jak również w ogólnym systemie systemów. Projekty krytyczne dla bezpieczeństwa i krytyczne dla realizacji misji projektu wymagają wyższych poziomów formalizacji i dokumentacji.

Dane wejściowe, procesy i dane wyjściowe dla testowania opartego na ryzyku będą określane za pomocą wybranej techniki. Często stosowane dane wejściowe obejmują spostrzeżenia interesariuszy,

specyfikacje i dane historyczne. Powszechnie stosowane procesy to, między innymi, identyfikacja ryzyka, ocena ryzyka i kontrola ryzyka. Powszechnie stosowane dane wyjściowe obejmują listę ryzyk jakościowych (z powiązanym poziomem ryzyka oraz zalecanym przydzieleniem nakładu pracy związanej z testowaniem), wykryte defekty w dokumentach wejściowych takich jak specyfikacje, pytania lub problemy związane z pozycjami ryzyka oraz ryzyka projektowe mające wpływ na testowanie lub na projekt jako całość.

Zwykle najbardziej krytycznym czynnikiem powodzenia w przypadku testowania opartego na ryzyku jest zaangażowanie w identyfikację i ocenę ryzyka właściwego zespołu interesariuszy. Wszyscy interesariusze na własny sposób pojmują, co stanowi o jakości produktu i mają własny zbiór priorytetów i obaw dotyczących jakości. Jednakże, interesariusze zwykle należą do dwóch szeroko rozumianych kategorii: interesariusze biznesowi i interesariusze techniczni.

Interesariusze biznesowi to, między innymi, klienci, użytkownicy, personel operacyjny, działy wsparcia oraz personel wsparcia technicznego. Interesariusze ci rozumieją klienta i użytkownika i dlatego też mogą zidentyfikować ryzyka i ocenić ich wpływ z biznesowego punktu widzenia.

Do grupy interesariuszy technicznych należą programiści, architekci, administratorzy baz danych i administratorzy sieci. Interesariusze ci rozumieją zasadnicze sposoby, w jakie oprogramowanie może zawieść, dlatego też mogą zidentyfikować ryzyka i ocenić prawdopodobieństwo ich wystąpienia z technicznego punktu widzenia.

Niektórzy interesariusze mają zarówno biznesowy, jak i techniczny punkt widzenia. Na przykład eksperci dziedzinowi, którzy pełnią istotną rolę w testowaniu lub analizie biznesowej, często mają szersze spojrzenie na ryzyko z racji wiedzy technicznej i biznesowej.

Proces identyfikacji pozycji ryzyka to proces, który generuje pokaźną listę ryzyk. Nie ma potrzeby, aby interesariusze debatowali na temat pozycji ryzyka; jeżeli jeden z interesariuszy postrzega coś jako ryzyko dla jakości systemu, jest to pozycją ryzyka. Jednakże ważne jest, aby interesariusze osiągnęli konsensus w kwestii klasyfikacji, jeśli chodzi o poziom ryzyka. Na przykład, w lekkich podejściach, które wykorzystują prawdopodobieństwo i wpływ jako czynniki klasyfikacji, część procesu musi obejmować znalezienie wspólnego planu klasyfikacji dla prawdopodobieństwa i wpływu. Wszyscy interesariusze, wraz z grupą testującą, muszą korzystać z tej samej skali i muszą być w stanie osiągnąć porozumienie w kwestii wspólnej klasyfikacji prawdopodobieństwa i wpływu dla każdej pozycji ryzyka jakościowego.

Jeżeli testowanie oparte na ryzyku ma być stosowane długofalowo, Kierownik Testów musi promować i inicjować testowanie oparte na ryzyku wśród interesariuszy. Aby zapewnić ciągłe wykorzystanie tej techniki, interdyscyplinarny zespół musi zrozumieć wartość analizy ryzyka. Konieczne jest, aby Kierownik Testów rozumiał potrzeby interesariuszy, ich oczekiwania i dostępność czasową, aby mogli oni uczestniczyć w tym procesie.

Odpowiednie zaangażowanie interesariusza w proces analizy ryzyka oferuje istotną korzyść Kierownikowi Testów, polegającą na tym, że w przypadku projektów o niedostatecznej specyfikacji, ze słabymi lub brakującymi wymaganiami, interesariusze nadal mogą zidentyfikować ryzyka, gdy zostaną naprowadzeni za pomocą odpowiedniej listy kontrolnej. Korzyść ta staje się widoczna, gdy po zastosowaniu techniki testowania opartego na ryzyku, wzrasta skuteczność wykrywania defektów przez zespół testowy. Dzieje się tak, ponieważ stosowana jest bardziej kompletna podstawa testów, w tym lista elementów ryzyka jakościowego.

W przypadku testowania opartego na ryzyku, w czasie zakończenia testów, zespoły testowe powinny mierzyć zakres, w jakim osiągnięto korzyści. W wielu przypadkach obejmuje to udzielenie odpowiedzi (poprzez metryki i konsultacje z zespołem) na niektóre lub wszystkie z następujących pytań:

- Czy zespół testowy wykrył wyższy odsetek ważnych defektów niż mniej ważnych defektów?

- Czy zespół testowy znalazł najważniejsze defekty na wczesnym etapie, w okresie wykonywania testów?
- Czy zespół testowy był w stanie wyjaśnić wyniki testów interesariuszom w kategoriach ryzyka?
- Czy testy, które zespół testowy pominął (jeżeli miało to miejsce), mają niższy poziom powiązanego ryzyka niż te wykonane?

W większości przypadków, zakończone powodzeniem testowanie oparte na ryzyku skutkuje udzieleniem odpowiedzi twierdzącej na wszystkie cztery pytania wymienione powyżej. W dłuższej perspektywie, Kierownik Testów powinien postawić cele ulepszenia procesu dla tych metryk, a zarazem dążyć do poprawienia efektywności procesu analizy ryzyka jakościowego. Oczywiście, do testowania opartego na ryzyku można zastosować inne metryki i kryteria sukcesu, a Kierownik Testów powinien dokładnie rozważyć związek pomiędzy tymi metrykami, celami strategicznymi, które realizuje zespół testowy a zachowaniami, które wynikną z zarządzania opartego na konkretnym zestawie metryk i kryteriów sukcesu.

2.3.3. Inne techniki wyboru testów

Chociaż wielu Kierowników Testów stosuje technikę testowania opartego na ryzyku jako jeden z elementów swojej strategii testów, to również wielu kierowników korzysta również z innych technik testowania.

Jedną z czołowych alternatywnych technik identyfikacji i priorytetyzacji warunków testowych jest testowanie oparte na wymaganiach. W testowaniu opartym na wymaganiach można wykorzystywać kilka technik, takich jak przeglądy niejednoznaczności, analiza warunków testowych oraz tworzenie grafów przyczynowo-skutkowych. Przeglądy niejednoznaczności umożliwiają zidentyfikowanie i eliminację niejasności w wymaganiach, które służą jako podstawa testów, nierzadko poprzez wykorzystanie listy kontrolnej często spotykanych defektów wymagań (zob. [Wiegiers03]).

Jak opisuje [Craig02], analiza warunków testowych obejmuje bardzo dokładne czytanie specyfikacji wymagań w celu zidentyfikowania warunków testowych, które należy pokryć. Jeżeli wymagania te mają przypisany priorytet, może on zostać wykorzystany, aby rozdzielić prace i nadać priorytet poszczególnym przypadkom testowym. W przypadku braku przypisanego priorytetu, trudno jest określić stosowną pracochłonność i kolejność testowania bez połączenia testowania opartego na wymaganiach z testowaniem opartym na ryzyku.

Tworzenie grafów przyczynowo-skutkowych omówiono w Sylabusie dla poziomu zaawansowanego Analityk Testów w kontekście pokrycia kombinacji warunków testowych jako części analizy testów. Jednakże szersze zastosowanie tej techniki może zredukować szeroki kontekst testów do wykonalnej liczby przypadków testowych i nadal zapewnić 100% funkcjonalnego pokrycia podstawy testów. Tworzenie grafów przyczynowo-skutkowych również umożliwia identyfikację luk w podstawie testów w czasie projektowania przypadków testowych, które z kolei pozwalają na identyfikację defektów wcześniej w cyklu życia wytwarzania oprogramowania, gdy projektowanie testów rozpoczyna się na podstawie projektu wymagań. Jedną z głównych przeszkód dla przyjęcia metody tworzenia grafów przyczynowo-skutkowych jest złożony sposób ich tworzenia. Istnieją narzędzia wspierające tę metodę, a jej ręczne wykonanie może być trudne.

Ogólną przeszkodą do testowania opartego na wymaganiach jest to, że specyfikacje wymagań są często dwuznaczne, nietestowalne, niekompletne lub nie istnieją. Nie wszystkie organizacje mają motywację, aby rozwiązać te problemy, więc testerzy stający w obliczu takich sytuacji muszą wybrać inną strategię testów.

Inną, czasem wykorzystywaną do zwiększenia wykorzystania istniejących wymagań metodą, jest tworzenie profili użycia i profili operacyjnych. Jest to podejście oparte na modelu, które wykorzystuje

połączenie przypadków użycia, użytkowników (czasami zwanych personami), danych wejściowych i danych wyjściowych, tak, aby dokładnie opisać rzeczywiste wykorzystanie systemu. Umożliwia to testowanie nie tylko funkcjonalności, ale również użyteczności, współdziałania, niezawodności, bezpieczeństwa i wydajności.

W trakcie analizy i planowania testów, zespół testowy identyfikuje profile użycia i stara się pokryć je przypadkami testowymi. Profil użycia to oparte na dostępnych informacjach oszacowanie realistycznego użycia oprogramowania. Oznacza to, że podobnie jak w przypadku testowania opartego na ryzyku, możliwe jest, że profile użycia nie będą idealnie odzwierciedlały faktycznego stanu rzeczy. Jeśli jednak dostępna jest wystarczająca ilość informacji i znane jest zdanie interesariuszy, model ten będzie odpowiedni (zob. [Musa04]).

Niektórzy Kierownicy Testów stosują również podejścia metodyczne, takie jak listy kontrolne, aby określić, co testować, jak dużo i w jakiej kolejności. W przypadku produktów, które są bardzo stabilne, wystarczająca może być lista kontrolna głównych obszarów funkcjonalnych i niefunkcjonalnych do testowania, połączona z repozytorium istniejących przypadków testowych. Lista kontrolna dostarcza heurystyk do określenia czasu na działania testowe i ustalenia kolejności testów, zwykle w oparciu o rodzaj i ilość zmian, które miały miejsce. Podejścia takie są raczej mniej istotne, gdy wykorzystuje się je do przetestowania problemów istotniejszych niż małe zmiany.

Inną popularną metodą jest wykorzystanie podejścia reaktywnego. W podejściu reaktywnym, przed wykonaniem testów, ma miejsce bardzo niewiele zadań związanych z analizą, projektowaniem lub wdrożeniem testów. Zespół testowy koncentruje się na reagowaniu na produkt w postaci, w jakiej jest on faktycznie dostarczany. Przedmiotem dalszego testowania stają się skupiska błędów, w miarę ich wykrywania. Nadawanie priorytetu i przydział prac odbywa się w całkowicie dynamiczny sposób. Podejście reaktywne może sprawdzić się jako uzupełnienie innych podejść. Stosowane samodzielnie, podejście to zwykle pomija główne obszary aplikacji, które są ważne, ale nie występuje w nich duża liczba błędów.

2.3.4. Priorytetyzacja testów i przydzielanie prac w procesie testowym

Bez względu na zastosowaną technikę, lub – jeszcze lepiej – kombinację technik, które wykorzystuje Kierownik Testów, musi on włączyć tę technikę do projektu i procesu testowego. Na przykład, w sekwencyjnych cyklach życia (np. model V), zespół testowy wybiera testy, przydziela prace i wstępnie priorytetyzuje testy w czasie fazy definiowania wymagań, z okresowymi korektami, podczas gdy w iteracyjnych lub zwinnych cyklach życia wymagane jest podejście z iteracji na iterację. Na etapie planowania i nadzoru nad testami zespół testowy musi wziąć pod uwagę stopień, w jakim ryzyka, wymagania i/lub profile użycia będą ewoluować i stosownie reagować na te zmiany.

W trakcie analizy, projektowania i implementacji testów, należy zastosować podział pracy i priorytetyzację określone w czasie planowania testów. W procesie testowym często występuje podział na dokładną analizę i/lub modelowanie, jedynie po to, żeby informacje te nie zostały wykorzystane do ukierunkowania dalszego przebiegu procesu testowego. Podział ten zwykle ma miejsce w czasie projektowania i implementacji.

Podczas wykonywania testów należy zrealizować zadania zgodnie z priorytetami ustalonymi podczas planowania testów, aczkolwiek istotną sprawą jest okresowa aktualizacja priorytetów w oparciu o informacje uzyskane po wstępnym zaprojektowaniu planu testów. Podczas oceny i raportowania wyników testów i statusu kryteriów wyjścia, Kierownik Testów musi również je ocenić i złożyć raport z punktu widzenia ryzyk, wymagań, profili użycia, list kontrolnych i innych wskazówek wykorzystanych do wyboru testów i nadania im priorytetu. Jeśli jest to konieczne, wybór testów powinien nastąpić w oparciu o priorytety.

Częścią raportowania wyników i oceny kryteriów wyjścia może być wykonany przez Kierownika Testów pomiar stopnia, w jakim testowanie zostało ukończone. Pomiar powinien obejmować śledzenie powiązań pomiędzy przypadkami testowymi a wykrytymi defektami do stosownej podstawy testów. Na przykład, w testowaniu opartym na ryzyku, testerzy mogą zbadać pozostały poziom ryzyka w miarę wykonywania testów i wykrywania defektów. Pomaga to w wykorzystaniu testowania opartego na ryzyku do określenia odpowiedniego momentu na wydanie. Raportowanie testów powinno odnosić się do pokrytych ryzyk i tych wciąż otwartych, jak również do korzyści już osiągniętych i jeszcze nieosiągniętych. Przykład raportowania wyników testów w oparciu o pokrycie ryzyka można znaleźć w [Black03].

Na koniec, w czasie zakończenia testów, Kierownik Testów powinien przeanalizować metryki i kryteria sukcesu, które odnoszą się do potrzeb i oczekiwań interesariuszy testowania, łącznie z potrzebami i oczekiwaniami klientów i użytkowników dotyczącymi zagadnienia jakości. Jedynie w przypadku, gdy testowanie zaspokaja te potrzeby i oczekiwania, zespół testowy może stwierdzić, że jest naprawdę skuteczny.

2.4. Dokumentacja testowa i inne produkty pracy

Jako część działań związanych z zarządzaniem testami często tworzona jest dokumentacja. Konkretnie nazwy dokumentów z obszaru zarządzania testami i zakres każdego dokumentu zwykle się różnią. Poniższa lista przedstawia typy dokumentów często stosowane przez firmy i organizacje do zarządzania testami:

- polityka testów – opisuje cele organizacji w związane z testowaniem,
- strategia testów – opisuje ogólne, zależne od projektu, metody testowania w danej organizacji,
- główny plan testów (lub plan testów dla projektu) – opisuje wdrożenie strategii testów dla konkretnego projektu,
- plan testów jednego poziomu (lub plan testów dla fazy) – opisuje poszczególne czynności, które mają zostać wykonane w ramach każdego poziomu testów.

Fizyczny układ tego typu dokumentów może się różnić zależnie od kontekstu. W niektórych organizacjach i w niektórych projektach mogą one być połączone w jeden dokument; w innych, można je znaleźć w osobnych dokumentach; z kolei w niektórych organizacjach ich treść może być wyrażona jako intuicyjne, niepisane lub tradycyjne metodyki testowania. W większych i bardziej sformalizowanych organizacjach i projektach dokumenty te mają postać pisemnych produktów pracy, podczas gdy w mniejszych i mniej sformalizowanych organizacjach i projektach takich produktów pracy jest mniej. Niniejszy sylabus opisuje każdy z tych typów dokumentów oddzielnie, jednak w praktyce poprawne wykorzystanie każdego dokumentu określa kontekst organizacyjny i projektowy.

2.4.1. Polityka testów

Polityka testów opisuje, dlaczego dana organizacja prowadzi testy. Definiuje ona ogólne cele testowania, które organizacja chce osiągnąć. Polityka ta powinna zostać opracowana przez personel wyższego szczebla ds. zarządzania testami w organizacji we współpracy z kierownikami wyższego szczebla dla grup interesariuszy procesu testowania.

W niektórych przypadkach, polityka testów będzie uzupełnieniem lub będzie stanowić część szerszej polityki jakości. Polityka jakości zawiera opis ogólnych wartości i celów kierownictwa związanych z jakością.

Spisana polityka testów, o ile istnieje, może to być krótkim dokumentem wysokiego poziomu, który:

- podsumowuje wartość, którą organizacja czerpie z testowania,
- definiuje cele testowania, takie jak budowanie zaufania do oprogramowania, wykrywanie defektów w oprogramowaniu i zmniejszenie poziomu ryzyka jakościowego (zob. Sekcja 2.3.1.),

- opisuje, jak oceniać skuteczność i wydajność testowania w spełnianiu tych celów,
- przedstawia w zarysie typowy proces testowy, być może korzystając z podstawowego procesu testowego ISTQB® jako podstawy,
- określa, jak organizacja będzie ulepszać swój proces testowy (zob. Rozdział 5.).

Polityka testów powinna określać działania w obszarze testowania dla nowo rozwijanych aplikacji, jak również dla pielęgnacji oprogramowania. Może się również odnosić do wewnętrznych i/lub zewnętrznych norm dotyczących testowania produktów pracy i terminologii, która ma być wykorzystywana w całej organizacji.

2.4.2. Strategia testów

Strategia testów opisuje ogólną metodykę testowania w danej organizacji. Obejmuje ona sposób, w jaki testowanie jest wykorzystywane do zarządzania ryzykami produktowymi i projektowymi, podziału procesu testowania na poziomy testów oraz czynności wysokiego poziomu związanych z testowaniem. Ta sama organizacja może posiadać różne strategie dla różnych sytuacji, takich jak różne cykle wytwarzania oprogramowania, różne poziomy ryzyka lub różne wymagania prawne. Strategia testów oraz procesy i czynności w niej opisane powinny być zgodne z polityką testów. Powinna ona przedstawiać ogólne kryteria wejścia i wyjścia testów dla organizacji lub dla jednego lub więcej programów.

Jak wspomniano powyżej, strategie testów opisują ogólne metodyki testowania, które zwykle obejmują:

- Strategie analityczne, takie jak testowanie oparte na ryzyku, w których zespół testowy analizuje podstawę testów w celu zidentyfikowania warunków testowych, które mają zostać pokryte. Na przykład, w testowaniu opartym na wymaganiach, na etapie analizy testów, warunki testowe wyprowadzane są z wymagań, następnie testy są projektowane i implementowane, aby pokryć te warunki. W dalszej kolejności testy są wykonywane, często przy wykorzystaniu parametru priorytet danego wymagania pokrytego każdym testem, w celu określenia kolejności wykonywania testów. Wyniki testów są raportowane w kategoriach statusu wymagań, np. wymaganie przetestowane, które przeszło testy pozytywnie, wymaganie przetestowane, które nie przeszło testów, wymaganie jeszcze nie w pełni przetestowane, testowanie wymagania zablokowane itd.
- Strategie oparte na modelu, takie jak profilowanie operacyjne, w których zespół testowy opracowuje model (oparty na faktycznych lub przewidywanych sytuacjach) środowiska, w którym system istnieje, dane wejściowe i warunki, którym system podlega, a także opisuje, jak system powinien się zachować. Na przykład, w testowaniu wydajnościowym opartym na modelu szybko rozwijającej się aplikacji na urządzenia mobilne, można opracować modele przychodzącego i wychodzącego ruchu sieciowego, aktywnych i nieaktywnych użytkowników oraz wynikającego z tego obciążenia związanego z przetwarzaniem, w oparciu o aktualne użycie i rozwój projektu w miarę upływu czasu. Ponadto, modele można opracowywać biorąc pod uwagę aktualny sprzęt w środowisku produkcyjnym, oprogramowanie, pojemność danych, sieć i infrastrukturę. Modele można również opracowywać dla idealnych oczekiwanych i minimalnych przepustowości, czasów reakcji i alokacji zasobów.
- Strategie metodyczne, takie jak oparte na cechach jakości, w których zespół testowy wykorzystuje z góry ustalony zestaw warunków testowych, takich jak norma jakości (np. ISO 25010 [ISO25010], która zastąpiła ISO 9126 [ISO9126])⁵, lista kontrolna lub zbiór uogólnionych logicznych warunków testowych, które mogą odnosić się do konkretnej dziedziny, aplikacji lub rodzaju testowania (np. testowanie bezpieczeństwa) i wykorzystuje ten zestaw warunków testowych w każdej kolejnej iteracji lub dla każdego kolejnego wydania. Na przykład w testowaniu pielęgnacyjnym prostej, stabilnej strony handlu elektronicznego, testerzy mogą

⁵ Standard ISO/IEC 25010 zastąpił normę ISO 9126.

wykorzystywać listę kontrolną, w której opisano główne funkcje, atrybuty i linki dla każdej strony. Testerzy obejmują testowaniem istotne elementy wspomnianej listy kontrolnej za każdym razem, gdy strona jest modyfikowana.

- Strategie zgodne z procesem lub normą, takie jak dla systemów medycznych podlegających normom Amerykańskiej Agencji ds. Żywności i Leków. Zespół testowy przestrzega wtedy zestawu procesów zdefiniowanych przez komisję normatywną lub inny panel ekspertów, w którym procesy odnoszą się do dokumentacji, właściwej identyfikacji i wykorzystania podstawy testów i wyroczni testowych oraz organizacji zespołu testowego. Na przykład, w projektach prowadzonych metodą zarządzania zwinnego Scrum, testerzy w każdej iteracji analizują historyjki użytkownika opisujące poszczególne funkcje, szacują pracochłonność testowania dla każdej funkcji jako część procesu planowania dla iteracji, identyfikują warunki testowe (zwane często kryteriami akceptacji) dla każdej historyjki użytkownika, wykonują testy, które obejmują te warunki i raportują status każdej historyjki użytkownika (nieprzetestowana, testy zaliczone lub niezaliczone) w czasie wykonania testów.
- Strategie reaktywne, takie jak korzystanie z ataków na podstawie zidentyfikowanych defektów, w których zespół testowy czeka z zaprojektowaniem i wdrożeniem testów do momentu otrzymania oprogramowania, reagując na faktyczne zachowanie systemu poddawanego testowaniu. Na przykład, gdy stosuje się testowanie eksploracyjne aplikacji opartej na menu, można opracować zestaw kart opisów testów odpowiadających funkcjom, wyborom z menu i ekranom. Każdemu testerowi zostaje przydzielony zestaw kart opisów testów, które później wykorzystuje on do zorganizowania swoich sesji testowania eksploracyjnego. Testerzy okresowo raportują wyniki sesji testowania Kierownikowi Testów, który może zrewidować karty opisu testów w oparciu o te wyniki.
- Strategie konsultatywne, takie jak testowanie kierowane przez użytkownika, w których zespół testowy przy identyfikowaniu warunków testowych do pokrycia polega na danych wejściowych dostarczonych przez jednego lub więcej kluczowych interesariuszy. Na przykład, w zleconym zewnętrznemu wykonawcy testowaniu zgodności dla aplikacji internetowej, firma może podać zewnętrznemu dostawcy usług testowych ułożoną według priorytetów listę wersji przeglądarki, oprogramowania chroniącego przed złośliwym kodem, systemów operacyjnych, rodzajów połączenia i innych opcji konfiguracyjnych, których oceny wymagają wobec swojej aplikacji. Dostawca usług testowych może wtedy do projektowania testów wykorzystywać techniki, takie jak testowanie sposobem par (dla opcji o wysokim priorytecie) i klasy równoważności (dla opcji o niższym priorytecie).
- Strategie testowe minimalizujące regresję, takie jak rozległa automatyzacja, w których zespół testowy wykorzystuje różne techniki do zarządzania ryzykiem regresji, automatyzacji testów, zwłaszcza automatyzacji funkcjonalnych i/lub niefunkcjonalnych testów regresji na jednym lub kilku poziomach. Na przykład, podczas testowania regresji aplikacji internetowej, testerzy mogą korzystać z narzędzia do automatyzacji testów w oparciu o interfejs użytkownika, w celu zautomatyzowania typowych i wyjątkowych przypadków użycia dla aplikacji. Testy te są następnie wykonywane za każdym razem, gdy aplikacja jest modyfikowana.

Różne strategie testowe można łączyć ze sobą. Konkretnie wybrane strategie powinny być odpowiednie do potrzeb i środków organizacji, a organizacje mogą dostosowywać strategie do siebie, aby pasowały do konkretnych operacji i projektów.

Strategia testów może opisywać poziomy testów, które mają zostać wykonane. W takim przypadku w dokumencie tym w dokumencie tym, powinny być podane wskazówki dotyczące kryteriów wejścia i wyjścia dla każdego poziomu testów i związków między tymi poziomami (np. podział celów pokrycia testami).

Strategia testów może również zawierać opis następujących kwestii:

- procedury integracyjne,
- techniki specyfikacji testów,

- niezależność testowania, która może się zmieniać zależnie od poziomu,
- normy obowiązkowe i opcjonalne,
- środowiska testowe,
- automatyzację testów,
- narzędzia testowe,
- możliwość wielokrotnego użycia produktów oprogramowania i produktów testowania,
- testowanie potwierdzające (retesty) oraz testowanie regresji,
- kontrolę testów i raportowanie,
- pomiary testów i metryki,
- zarządzanie defektami,
- podejście do zarządzania konfiguracją testaliów,
- role i obowiązki.

Konieczne może okazać się stosowanie różnych krótko- i długoterminowych strategii testowania. Różne strategie testowania są odpowiednie dla różnych organizacji i projektów. Na przykład, gdy chodzi o aplikacje krytyczne ze względów bezpieczeństwa, bardziej prężniejsza strategia może być bardziej odpowiednia niż w innych przypadkach. Ponadto, strategia testowania różni się również dla różnych modeli wytwarzania oprogramowania.

2.4.3. Główny plan testów

Główny plan testów obejmuje całokształt prac w obszarze testowania do wykonania w danym projekcie, wraz z poszczególnymi poziomami testów, na których mają zostać wykonane testy danego poziomu, a związkami między tymi poziomami oraz między poziomami testów a odpowiadającymi im czynnościami wytwarzania oprogramowania. W głównym planie testów powinien zostać omówiony sposób, w jaki testerzy wdrożą strategię testów dla tego projektu (tj. podejście do testowania). Główny plan testów powinien być zgodny z polityką testów oraz strategią, a w konkretnych obszarach, w których się od nich różni, powinien wyjaśnić te odstępstwa i wyjątki, wraz z potencjalnym ich wpływem na projekt, wynikającym z tych odstępstw. Na przykład, jeśli strategia testów organizacji polega na wykonaniu jednej pełnej fazy testowania regresji na niezmiennym się systemie tuż przed wydaniem, ale w bieżącym projekcie nie zostanie wykonane testowanie regresji, plan testów powinien wyjaśniać, dlaczego w taki sposób zaplanowano testy i co zostanie zrobione, aby złagodzić ryzyko spowodowane takim odstępstwem od zwykłej strategii. Plan testów powinien również obejmować wyjaśnienie wszelkich innych rezultatów, których oczekuje się w wyniku tego odstępstwa. Na przykład, pominięcie testowania regresji może oznaczać konieczność zaplanowania wydania pielęgnacyjnego miesiąc po początkowym wydaniu projektu. Główny plan testów powinien uzupełniać plan projektu lub przewodnik eksploatacji w ten sposób, że powinien opisywać prężność testowania będącego częścią większego projektu lub operacji.

O ile konkretna treść i struktura głównego planu testów jest różna zależnie od organizacji, jej norm dokumentacji i stopnia sformalizowania projektu, o tyle typowe tematy dla głównego planu testów obejmują następujące zagadnienia:

- elementy, które mają zostać przetestowane i te, które nie mają zostać przetestowane,
- cechy jakościowe, które mają zostać przetestowane i te, które nie mają zostać przetestowane,
- plan i budżet testów, który powinien być równoległy do budżetu projektu lub budżetu operacyjnego,
- cykle wykonania testów i ich związek z planem wydania oprogramowania/systemu,
- związki i produkty między testowaniem a innymi osobami i działami,
- określenie, jakie elementy testowe znajdują się w zakresie i poza zakresem testowania dla każdego opisanego poziomu testów,
- konkretne kryteria wejścia, kryteria kontynuacji (zawieszenia/wznowienia) oraz kryteria wyjścia dla każdego poziomu testów, a także związki między poziomami,

- ryzyka projektowe testów,
- ogólne zarządzanie pracą związaną z testowaniem,
- obowiązki dotyczące wykonania testów każdego poziomu,
- dane wejściowe i wyjściowe dla każdego poziomu testów.

W mniejszych projektach lub podczas eksploatacji, w których tylko jeden poziom testów jest sformalizowany, główny plan testów oraz plan testów dla tego poziomu będą często połączone w jeden dokument. Na przykład, jeżeli testy systemowe są jedynym sformalizowanym poziomem, przy czym nieformalne testowanie modułów i testowanie integracyjne są wykonywane przez programistów, a nieformalne testy akceptacyjne są wykonywane przez klientów jako część procesu testów beta, wtedy plan testów systemowych może obejmować elementy wymienione w niniejszej sekcji.

Ponadto, testowanie zwykle zależy od innych czynności w projekcie. Gdyby te czynności nie były dostatecznie udokumentowane, zwłaszcza w kategoriach ich wpływu i związków z testowaniem, tematy związane z tymi czynnościami mogłyby zostać ujęte w głównym planie testów (lub odpowiednim planie testów jednego poziomu). Na przykład, jeżeli proces zarządzania konfiguracją nie jest udokumentowany, plan testów powinien określić sposób, w jaki przedmioty testów mają być dostarczane zespołowi testowemu.

2.4.4. Plan testów jednego poziomu

W planie testów jednego poziomu opisane są konkretne czynności, które mają zostać wykonane w ramach każdego poziomu testów lub, w niektórych przypadkach, w ramach danego typu testów. Plany testów jednego poziomu poszerzają, gdy to konieczne, główny plan testów dla konkretnego poziomu lub typu testów, które mają zostać udokumentowane. Przedstawiają szczegóły planu, zadań i kroków milowych niekoniecznie ujętych w głównym planie testów. Ponadto, w zakresie, w jakim różne normy i szablony mają zastosowanie do specyfikacji testów na różnych poziomach, szczegóły te zostałyby ujęte w planach testów jednego poziomu.

W mniej formalnych projektach lub działaniach, plan testów jednego poziomu jest często jedynym pisemnym dokumentem z obszaru zarządzania testami. W takich sytuacjach niektóre z elementów informacyjnych wymienionych powyżej w niniejszej sekcji mogłyby zostać ujęte w dokumencie planu testów, o którym mowa.

W przypadku projektów prowadzonych w metodykach zwinnych, plany testów dla przebiegu lub iteracji mogą zastąpić plany testów jednego poziomu.

2.4.5. Zarządzanie ryzykiem projektowym

Ważną częścią prawidłowego planowania jest zarządzanie ryzykami projektowymi. Ryzyka projektowe można zidentyfikować korzystając z procesów podobnych do opisanych w Sekcji 2.3. Gdy ryzyka projektowe zostaną zidentyfikowane, należy je zgłosić kierownikowi projektu, który powinien podjąć w stosunku do nich stosowne działania. Zespół testowy nie zawsze może zmniejszyć takie ryzyka, jednakże Kierownik Testów powinien i może z powodzeniem łagodzić niektóre ryzyka projektowe, takie jak:

- gotowość środowiska testowego i narzędzi,
- dostępność i kwalifikacje zespołu testowego,
- brak norm, zasad i technik dla zadań w obszarze testowania.

Podejścia do zarządzania ryzykiem projektowym obejmują wcześniejsze przygotowanie testaliów, wstępne testowanie środowisk testowych, wstępne testowanie wczesnych wersji produktu, stosowanie ostrzejszych kryteriów wejścia do testowania, wprowadzanie w życie wymagań dotyczących

testowalności, uczestnictwo w przeglądach wczesnych produktów projektowania, uczestnictwo w zarządzaniu zmianą oraz monitorowanie postępu projektu i jakości.

Po zidentyfikowaniu i przeanalizowaniu ryzyka projektowego, można rozpocząć dalsze prace w oparciu o cztery główne opcje dla zarządzania tym ryzykiem:

1. Złagodzenie ryzyka poprzez środki zapobiegawcze w celu zmniejszenia jego prawdopodobieństwa i/lub wpływu.
2. Tworzenie planów awaryjnych w celu zmniejszenia wpływu ryzyka, jeżeli stanie się ono faktem.
3. Przeniesienie ryzyka na innego adresata ryzyka, aby się nim zajął.
4. Zignorowanie lub akceptacja ryzyka.

Wybór najlepszej opcji zależy od korzyści i możliwości dostępnych w ramach danej opcji, jak również kosztu oraz jakichkolwiek potencjalnych, dodatkowych ryzyk związanych z wybraną opcją. Gdy dla ryzyka projektowego zostanie określony plan awaryjny, najlepszą praktyką jest zidentyfikowanie wyzwalacza, który określi, kiedy i jak plan awaryjny zostanie uruchomiony oraz właściciela (adresata) ryzyka, który wykona plan awaryjny.

2.4.6. Inne produkty testowania

Testowanie wiąże się z tworzeniem pewnej liczby innych produktów pracy, takich jak raporty o defektach, specyfikacje przypadków testowych i logów testowych. Większość tych produktów jest wytwarzana przez Analityków Testów i Technicznych Analityków Testów. W Sylabusie poziomu zaawansowanego Analityk Testów omówiono kwestie związane z tworzeniem i dokumentowaniem tych produktów pracy. Kierownik Testów powinien zapewnić spójność i wysoką jakość produktów pracy, o których mowa, poprzez:

- ustanowienie i monitorowanie metryk, przy pomocy których monitorowana jest jakość tych produktów pracy, takich jak odsetek odrzuconych raportów o defektach,
- współpracę z Analitykami Testów i Technicznymi Analitykami Testów w celu wybrania i dostosowania odpowiednich szablonów dla tych produktów pracy,
- współpracę z Analitykami Testów oraz Technicznymi Analitykami Testów w celu ustanowienia norm dla tych produktów pracy, takich jak stopień szczegółowości niezbędny w testach, logach testów i raportach,
- przeglądy produktów testowania przy wykorzystaniu odpowiednich technik i przez odpowiednich uczestników i interesariuszy.

Zakres, rodzaj i specyfika dokumentacji testowej mogą być kształtowane, między innymi, poprzez wybrany cykl życia wytwarzania oprogramowania, stosowne normy i przepisy oraz ryzyka jakościowe produktu i projektowe związane z konkretnym systemem, który jest rozwijany.

Istnieje wiele źródeł szablonów dla produktów testowania, takich jak np. IEEE 829⁶ [IEEE829]. Kierownik Testów musi pamiętać, że dokumenty IEEE 829 zostały zaprojektowane do wykorzystania w każdej branży. Jako takie, szablony te charakteryzują się wysokim poziomem szczegółowości, który nie zawsze musi mieć zastosowanie do konkretnej organizacji. Najlepszą praktyką jest dostosowanie dokumentów IEEE 829 celem stworzenia standardowych szablonów, które można wykorzystać w konkretnej organizacji. Konsekwentne stosowanie szablonów skutkuje zmniejszeniem wymagań w zakresie szkolenia i pomaga ujednolicić procesy w danej organizacji.

Testowanie wiąże się również z tworzeniem sumarycznych raportów z testów, które zwykle są tworzone przez Kierownika Testów i są opisane w dalszej części rozdziału.

⁶ Standard ISO/IEC/IEEE 29119 zastąpił normę IEEE 829.

2.5. Szacowanie testów

Szacowanie, jako czynność związana z zarządzaniem, jest tworzeniem przybliżonego celu dla kosztów i dat ukończenia związanych z działaniami, które są częścią określonej operacji lub projektu. Najlepsze szacowanie:

- reprezentuje zbiorową wiedzę doświadczonych praktyków i ma wsparcie zaangażowanych uczestników,
- przedstawia konkretne, szczegółowe katalogi kosztów, zasobów, zadań i osób zaangażowanych,
- przedstawia dla każdej czynności najbardziej prawdopodobny koszt, pracochłonność i czas trwania.

Od dawna wiadomo, że szacowanie w dziedzinie inżynierii oprogramowania i systemów jest trudne, zarówno pod względem technicznym, jak i politycznym, jednak istnieją dobrze zdefiniowane najlepsze praktyki zarządzania projektami dotyczące obszaru szacowania. Szacowanie testów jest zastosowaniem tych najlepszych praktyk do czynności testowych związanych z projektem lub działaniem.

Szacowanie testów powinno obejmować wszystkie czynności wchodzące w skład procesu testowego opisanego w Rozdziale 1. Dla kierownictwa często najbardziej interesujący jest szacowany koszt, pracochłonność, a zwłaszcza czas trwania wykonania testów, ponieważ wykonanie testów zwykle znajduje się na krytycznej ścieżce projektu. Szacunki wykonania testów są jednak zwykle trudne do wykonania i zawodne w sytuacji, gdy ogólna jakość oprogramowania jest niska lub nieznana. Ponadto, na jakość oszacowania prawdopodobnie wpłynie zaznajomienie i doświadczenie korzystania z systemu. Częstą praktyką jest szacowanie liczby przypadków testowych wymaganych w czasie wykonania testów, ale działa to jedynie wtedy, gdy można przyjąć założenie, że liczba defektów w oprogramowaniu, które ma być testowane, jest niska. Założenia poczynione w czasie szacowania powinny zostać zawsze udokumentowane jako część szacowania.

Szacowanie testów powinno uwzględniać wszystkie czynniki, które mogą mieć wpływ na koszt, pracochłonność oraz czas trwania czynności testowych. Czynniki te obejmują, między innymi, następujące pozycje:

- wymagany poziom jakości systemu,
- rozmiar systemu, który ma być testowany,
- dane historyczne z testowania dla poprzednich projektów testowych, które mogą zostać poszerzone o dane branżowe lub dane z innych organizacji będące punktem odniesienia,
- czynniki procesowe, obejmujące strategię testów, cykl życia wytwarzania lub utrzymania oprogramowania i dojrzałość procesu, a także dokładność oszacowania projektu,
- czynniki materiałowe, obejmujące automatyzację testów i narzędzia, środowiska testowe, dane testowe, środowiska programistyczne, dokumentację projektu (np. wymagania, projekty, itd.) oraz produkty testowania do ponownego użycia,
- czynnik ludzki, uwzględniający kierowników i liderów technicznych, zaangażowanie i oczekiwania dyrektorów i kierownictwa wyższego szczebla, umiejętności, doświadczenie i nastawienie w zespole projektowym, stabilność zespołu projektowego, relacje w zespole projektowym, wsparcie środowiska testowego i debugowego, dostępność wykwalifikowanych podwykonawców i konsultantów, a także wiedzy dziedzinowej,
- złożoność procesu, technologii, organizacji, liczba interesariuszy testowania, skład i lokalizacja podzespołów,
- istotne zwiększenie tempa pracy, potrzeby szkolenia i orientacji w sposobie działania systemu,
- przyswojenie lub rozwój nowych narzędzi, technologii, procesów, technik, niestandardowego sprzętu lub dużej ilości testaliów,

- wymagania dotyczące wysokiego stopnia szczegółowej specyfikacji testowej, zwłaszcza takiej, która ma być zgodna z nieznanym standardem dokumentacji,
- skomplikowana koordynacja dostarczenia elementów w czasie, zwłaszcza do testowania integracyjnego i rozwoju testów,
- wrażliwe dane testowe (np. dane, które są wrażliwe na upływ czasu).

Głównym czynnikiem, który Kierownicy Testów powinni uwzględnić w swoim szacowaniu, jest również jakość oprogramowania dostarczonego do testowania. Na przykład, jeżeli programiści przyjęli najlepsze praktyki, takie jak zautomatyzowane testowanie jednostkowe i ciągła integracja, wtedy aż 50% defektów zostanie usuniętych przed dostarczeniem kodu zespołowi testowemu (zob. [Jones11], aby dowiedzieć się więcej na temat efektywności tych praktyk w usuwaniu defektów). Niektórzy twierdzą, że metodyki zwinne, łącznie z wytwarzaniem sterowanym testami (ang. *Test Driven Development*, TDD), skutkują wyższym poziomem jakości produktu pracy dostarczanego do testowania.

Szacowanie może być wykonywane wstępująco lub zstępująco. Techniki, które mogą zostać wykorzystane w szacowaniu testów, pojedynczo lub w połączeniu, obejmują:

- Intuicję, zgadywanie i przeszłe doświadczenie,
- strukturę podziału pracy (ang. *Work Breakdown Structure*, WBS),
- sesje szacowania w zespole (np. szerokopasmowa technika delficka ang. *Wideband Delphi*),
- standardy i normy firmowe,
- odsetek ogólnej pracochłonności projektu lub poziomy obsadzenia personelem (np. proporcja testerów do programistów),
- historię organizacyjną i metryki, łącznie z modelami wywodzącymi się z tych metryk, które umożliwiają oszacowanie liczby defektów, liczby cykli testowych, liczby przypadków testowych, związanej z każdym testem średniej pracochłonności oraz liczby cykli regresji,
- średnie branżowe i modele prognozujące, takie jak punkty funkcyjne, linie kodu, szacowany wysiłek programistyczny lub inne parametry projektu (np. zob. [Jones07]).

W większości przypadków po sporządzeniu oszacowanie testów musi być dostarczone wraz z uzasadnieniem do kierownictwa (zob. Sekcja 2.7.). Często odbywają się negocjacje, nierzadko skutkujące zmianą oszacowania. W idealnej sytuacji, ostateczne oszacowanie testów odzwierciedla najlepszy możliwy stan równowagi między celami organizacyjnymi a projektowymi w obszarach jakości, planu, budżetu i funkcji oprogramowania.

Ważne jest, aby pamiętać, że każde oszacowanie jest oparte na informacjach dostępnych w czasie, w którym zostało przygotowane. We wczesnej fazie projektu, informacje mogą być bardzo ograniczone. Ponadto, informacje, które są dostępne we wczesnej fazie projektu, mogą zmienić się po pewnym czasie. W celu zachowania dokładności, oszacowania powinny być aktualizowane, aby odzwierciedlały nowe i zmienione informacje.

2.6. Definiowanie i wykorzystywanie metryk testowych

Utarłe powiedzenie związane z zarządzaniem mówi, że to, co się mierzy, to zostaje zrobione. Ponadto to, co nie jest mierzone, nie zostaje zrobione, ponieważ to, co nie jest mierzone, łatwo jest zignorować. Dlatego też ważne jest stworzenie odpowiedniego zestawu metryk dla każdego przedsięwzięcia, łącznie z testowaniem.

Metryki testowe można sklasyfikować jako należące do jednej lub więcej następujących kategorii:

- metryki projektowe, które mierzą postęp w kierunku ustanowionych kryteriów wyjścia projektu, takich jak odsetek wykonanych przypadków testowych, testy zaliczone i niezaliczone,
- metryki produktowe, które mierzą pewien atrybut produktu, taki jak zakres, w jakim został on przetestowany lub gęstość defektów,

- metryki procesowe, które mierzą wydajność procesu testowego lub rozwoju oprogramowania, takie jak odsetek defektów wykrytych przez testy,
- metryki osobowe, które mierzą wydajność poszczególnych osób lub grup, takie jak implementacja przypadków testowych zgodnie z harmonogramem.

Każda metryka może należeć do dwóch, trzech, a nawet czterech kategorii. Na przykład, wykres trendów pokazujący dzienne tempo przybywania defektów może być powiązany z kryterium wyjścia (nie znaleziono nowych defektów przez tydzień), jakością produktu (testowanie nie skutkuje zlokalizowaniem w nim dalszych defektów) oraz wydajnością procesu testowego (testowanie umożliwia znajdowanie dużej liczby defektów we wczesnej fazie okresu wykonywania testów).

Szczególnie delikatnie należy obchodzić się z metrykami osobowymi. Kierownicy czasami mylą metryki, które są przede wszystkim metrykami procesowymi, z metrykami osobowymi, co prowadzi do katastrofalnych skutków, gdy ludzie działają tak, aby zmienić metryki w korzystny dla siebie sposób. Poprawna motywacja i ocena osób testujących została omówiona w Rozdziale 7. niniejszego sylabusu oraz w Sylabusie dla poziomu eksperckiego Doskonalenie procesu testowego (Wdrażanie doskonalenia i zmian) [ISTQB ETL SYL].

Na poziomie zaawansowanym zajmujemy się głównie wykorzystaniem metryk do mierzenia postępu testowania, tj. metryk projektowych. Niektóre z metryk projektowych wykorzystywanych do pomiaru postępu testów odnoszą się również do produktu i procesu. Więcej informacji na temat wykorzystania metryk produktowych i procesowych przez kierownictwo można znaleźć w Sylabusie dla poziomu eksperckiego Doskonalenie procesu testowego (Wdrażanie doskonalenia i zmian) [ISTQB ITP SYL].

Stosowanie metryk umożliwia testerom raportowanie wyników testów w sposób spójny i umożliwia spójne śledzenie postępów w czasie. Od Kierowników Testów często wymaga się, aby prezentowali metryki na różnych spotkaniach, w których mogą brać udział interesariusze z wielu grup, począwszy od personelu technicznego, aż do kierownictwa wykonawczego. Ponieważ metryki są czasami wykorzystywane do określenia ogólnego powodzenia projektu, należy zachować szczególną staranność podczas określania, co śledzić, jak często to raportować i jaka metoda ma zostać wykorzystana do przedstawienia informacji. Kierownik Testów musi w szczególności wziąć pod uwagę następujące kwestie:

- Definicja metryk. Należy zdefiniować ograniczony zestaw użytecznych metryk. Metryki należy zdefiniować w oparciu o konkretne cele dla projektu, procesu i/lub produktu. Metryki powinny zostać zdefiniowane w równowadze, ponieważ pojedyncza metryka może dać mylące wrażenie statusu lub trendów. Po zdefiniowaniu metryk należy uzgodnić ich interpretację ze wszystkimi interesariuszami, aby uniknąć zamieszania podczas omawiania pomiarów. Często występuje tendencja do definiowania zbyt wielu metryk zamiast do skoncentrowania się na najbardziej użytecznych metrykach.
- Śledzenie metryk. Raportowanie i łączenie metryk powinno być możliwie najbardziej zautomatyzowane, aby ograniczyć czas spędzany na zbieranie i przetwarzanie pomiarów. Wahania pomiarów w czasie dla konkretnej metryki mogą odzwierciedlać informacje inne, niż interpretacja uzgodniona na etapie definiowania metryki. Kierownik Testów powinien być przygotowany do dokładnego przeanalizowania możliwych rozbieżności w pomiarach, różnych od oczekiwań oraz powodów tej rozbieżności.
- Raportowanie metryk. Dla kierownictwa celem jest zapewnienie natychmiastowego zrozumienia informacji. Prezentacje mogą pokazywać wartość metryki w pewnym momencie lub pokazywać ewolucję metryki w czasie, aby można było ocenić trendy.
- Wiarygodność metryk. Kierownik Testów musi również zweryfikować informacje, które są raportowane. Pomiary pobrane dla metryki mogą nie odzwierciedlać prawdziwego statusu projektu lub mogą przekazywać nadmiernie pozytywny lub negatywny trend. Zanim dane zostaną zaprezentowane, Kierownik Testów musi dokonać ich przeglądu zarówno pod względem dokładności, jak i pod względem informacji, które będą z nich wynikać.

Postęp testów monitorowany jest w pięciu podstawowych wymiarach:

- ryzyka produktowe (jakościowe),
- defekty,
- testy,
- pokrycie,
- zaufanie.

Ryzyka produktowe, defekty, testy oraz pokrycie mogą być i często są mierzone i raportowane na określone sposoby w czasie trwania projektu lub działania. Jeżeli pomiary te są związane ze zdefiniowanymi kryteriami wyjścia w planie testów, mogą one stanowić obiektywny standard, według którego będzie oceniane ukończenie prac związanych z testowaniem. Zaufanie jest mierzone poprzez ankiety lub przez wykorzystanie, jako zastępczej metryki, pokrycia; jednakże, zaufanie często raportuje się również subiektywnie.

Metryki związane z ryzykami produktowymi obejmują:

- odsetek ryzyk całkowicie pokrytych testami, które zostały zaliczone,
- odsetek ryzyk, dla których kilka lub wszystkie testy są niezaliczone,
- odsetek ryzyk jeszcze nieprzetestowanych w całości,
- odsetek ryzyk pokrytych, ułożonych według kategorii ryzyka,
- odsetek ryzyk zidentyfikowanych po początkowej analizie ryzyka jakościowego.

Metryki związane z defektami obejmują:

- łączną liczbę zgłoszonych (znalezionych) defektów w stosunku do łącznej liczby rozwiązanych (naprawionych) defektów,
- średni czas pomiędzy awariami lub tempo pojawiania się awarii,
- podział na grupy ilości lub odsetka defektów sklasyfikowanych według następujących kategorii:
 - poszczególne elementy testowe lub komponenty,
 - przyczyny podstawowe,
 - źródło defektu (np. specyfikacja wymagań, nowa funkcja, regresja itd.),
 - wydanie testów,
 - faza, w której defekt wprowadzono, wykryto i usunięto,
 - priorytet / ważność,
 - raporty odrzucone lub zdublikowane.
- trendy w czasie pomiędzy zgłoszeniem a rozwiązaniem defektu,
- liczbę napraw defektów, które wprowadziły nowe defekty (zwane czasami błędami wtórnymi).

Metryki związane z testami obejmują:

- całkowitą liczbę zaplanowanych testów oraz wyspecyfikowanych (zaimplementowanych), wykonanych, zaliczonych, niezaliczonych, zablokowanych i pominiętych,
- status testów regresji i potwierdzających, łącznie z tendencjami i sumami całkowitymi dla niezaliczonych testów regresji i testów potwierdzających,
- godziny testowania zaplanowanego na dzień w stosunku do godzin faktycznie zrealizowanych,
- dostępność środowiska testowego (odsetek planowanych godzin testowania, w których zespół testowy może korzystać ze środowiska testowego).

Metryki związane z pokryciem testami obejmują:

- pokrycie wymagań i elementów projektu,
- pokrycie ryzyka,
- pokrycie środowiska/konfiguracji,

- pokrycie kodu.

Ważne jest, aby Kierownicy Testów wiedzieli, jak interpretować i wykorzystywać metryki pokrycia w celu zrozumienia i raportowania statusu testów. Dla wyższych poziomów testów, takich jak testowanie systemowe, testowanie integracji systemów i testowanie akceptacyjne, główną podstawą testów są zwykle produkty pracy, takie jak specyfikacje wymagań, specyfikacje projektu, przypadki użycia, historyjki użytkowników, ryzyka produktowe, wspierane środowiska oraz wspierane konfiguracje. Do niższych poziomów testów, takich jak testowanie jednostkowe (np. pokrycie instrukcji i gałęzi) oraz testowanie integracyjne komponentów (np. pokrycie interfejsu) w większym stopniu mają zastosowanie metryki strukturalnego pokrycia kodu. Podczas gdy Kierownicy Testów mogą wykorzystywać metryki pokrycia kodu do mierzenia zakresu, w jakim testy sprawdzają strukturę systemu poddawanego testom, raportowanie wyników testów wyższego poziomu zwykle nie wiąże się z metrykami pokrycia kodu. Ponadto, Kierownicy Testów powinni zrozumieć, że nawet jeżeli testowanie jednostkowe i testowanie integracyjne komponentów osiągną 100% swoich celów pokrycia strukturalnego, defektami i ryzykami jakościowymi należy zająć się na wyższych poziomach testów.

Metryki mogą być również połączone z czynnościami podstawowego procesu testowego (opisanego w sylabusie Certyfikowany tester - Sylabus poziomu podstawowego ISTQB® oraz w niniejszym sylabusie). Poprzez takie połączenie, metryki mogą być wykorzystywane w całym procesie testowym w celu monitorowania samego procesu testowego oraz postępu w kierunku osiągnięcia celów projektu.

Metryki służące do monitorowania planowania testów i czynności kontrolnych mogą obejmować:

- pokrycie ryzyka, wymagań i innych elementów podstawy testów,
- wykrywanie defektów,
- planowane a faktyczne godziny na opracowanie testaliów i wykonanie przypadków testowych.

Metryki służące do monitorowania czynności analizy testów mogą obejmować:

- liczbę zidentyfikowanych warunków testowych,
- liczbę defektów znalezionych w czasie analizy testów (np. poprzez zidentyfikowanie ryzyk lub innych warunków testowych przy wykorzystaniu podstawy testów).

Metryki służące do monitorowania czynności projektowania testów mogą obejmować:

- odsetek warunków testowych pokrytych przypadkami testowymi,
- liczbę defektów znalezionych w czasie projektowania testów (np. poprzez opracowywanie testów w odniesieniu do podstawy testów).

Metryki służące do monitorowania czynności implementacji testów mogą obejmować:

- odsetek skonfigurowanych środowisk testowych,
- odsetek załadowanych rekordów danych testowych,
- odsetek zautomatyzowanych przypadków testowych.

Metryki służące do monitorowania czynności wykonania testów mogą obejmować:

- odsetek wykonanych, zaliczonych i niezaliczonych, a zaplanowanych przypadków testowych,
- odsetek warunków testowych pokrytych wykonanymi (i/lub zaliczonymi) przypadkami testowymi,
- planowane a faktyczne zgłoszone/rozwiązane defekty,
- planowane a faktyczne osiągnięte pokrycie.

Metryki służące do monitorowania postępów testów i czynności zamykających testy będą obejmowały mapowanie na kroki milowe, kryteria wejścia i kryteria wyjścia (zdefiniowane i zatwierdzone podczas planowania testów), które mogą obejmować metryki, takie jak:

- liczba warunków testowych, przypadków testowych lub specyfikacji testowych zaplanowanych i wykonanych w podziale na to, czy zostały one zaliczone, czy też nie zostały zaliczone,
- suma defektów, często rozłożona według ważności, priorytetu, aktualnego stanu, podsystemu, na który wywiera wpływ lub innej klasyfikacji (zob. Rozdział 4.),
- liczba zmian wymaganych, zaakceptowanych, zbudowanych i przetestowanych,
- koszt planowany w stosunku do faktycznego,
- planowany czas trwania w stosunku do faktycznego,
- daty planowane a daty faktyczne dla kamieni milowych procesu testowego,
- daty planowane a daty faktyczne dla związanych z procesem testowym kamieni milowych projektu (np. zamrożenie kodu),
- status ryzyka produktowego (jakościowego), często rozłożony na ryzyka złagodzone w stosunku do niezłagodzonych, głównych obszarów ryzyka, nowych ryzyk odkrytych po analizie testów itd.,
- procentowa strata prac związanych z testowaniem, kosztów lub czasu z powodu blokujących wydarzeń lub planowanych zmian,
- status testów potwierdzających i regresji.

Metryki służące do monitorowania czynności zamykających testy mogą obejmować:

- odsetek przypadków testowych wykonanych, zaliczonych, niezaliczonych, zablokowanych i pominiętych w czasie wykonywania testów,
- odsetek przypadków testowych oddanych do repozytorium przypadków testowych do ponownego użycia,
- odsetek przypadków testowych zautomatyzowanych lub planowanych w stosunku do faktycznie zautomatyzowanych przypadków testowych,
- odsetek przypadków testowych zintegrowanych z testami regresji,
- odsetek zgłoszonych defektów rozwiązanych/nierozwiązanych,
- odsetek zidentyfikowanych i zarchiwizowanych produktów pracy procesu testowego.

Ponadto, do monitorowania procesu testowego często są wykorzystywane standardowe techniki zarządzania projektem, takie jak struktury podziału pracy. W zespołach pracujących w metodach zwinnych, testowanie jest częścią postępu historii użytkownika na wykresie spalania. Gdy wykorzystywane są metodyki odchudzone, postęp testowania jest często monitorowany na podstawie poszczególnych historii użytkownika poprzez przemieszczenie karteczki z daną historią pomiędzy kolumnami na tablicy Kanban.

Na bazie zdefiniowanego zestawu metryk można raportować pomiary ustnie (w formie opisowej), numerycznie (w tabelach) lub obrazowo (w postaci wykresów). Pomiary można wykorzystać do kilku celów, łącznie z:

- analizą w celu odkrycia, jakie tendencje i przyczyny można wyróżnić za pomocą wyników testów,
- raportowaniem w celu przedstawienia wyników testowania zainteresowanym uczestnikom projektu i interesariuszom,
- kontrolą w celu zmiany przebiegu procesu testowego lub projektu, jako całości oraz w celu monitorowania wyników tej korekty przebiegu.

Właściwe sposoby zbierania, analizowania i raportowania pomiarów testów zależą od konkretnych potrzebnych informacji, celów i umiejętności osób, które będą wykorzystywać te pomiary. Ponadto, w zależności od odbiorców, konkretna treść sumarycznych raportów z testów powinna się różnić od siebie.

Dla celów nadzoru nad testami konieczne jest, aby poprzez cały proces testowy (po zakończeniu planowania testów) metryki dostarczały Kierownikowi Testów informacji potrzebnych do ukierunkowania

prac związanych z testowaniem w stronę pomyślnego zakończenia misji testowania, zrealizowania strategii i osiągnięcia celów. Dlatego też podczas planowania trzeba koniecznie uwzględnić wymagania dotyczące zakresu informacji, a monitorowanie musi obejmować również zbieranie wszelkich potrzebnych metryk dotyczących produktów pracy. Ilość potrzebnych informacji i nakład pracy włożony w ich zebranie zależy od różnych czynników projektowych obejmujących rozmiar, złożoność i ryzyko.

Na etapie nadzoru nad testami konieczne jest reagowanie na informacje generowane w poszczególnych etapach procesu testowego, jak również na zmieniające się warunki, w jakich projekt lub przedsięwzięcie funkcjonuje. Na przykład, jeżeli w trakcie testowania dynamicznego odkrywane jest skupisko defektów w obszarach, w których występowanie defektów uznano za nieprawdopodobne lub jeżeli okres wykonania testów zostanie skrócony z powodu opóźnień w rozpoczęciu testowania, należy zmienić założenia analizy ryzyka i plan testów. Mogłoby to skutkować ponownym nadaniem priorytetów testom i ponownym przydzieleniem pozostałego zakresu prac na wykonanie testów.

Jeżeli raport z postępów testów ujawni odchylenia od planu testów, należy przeprowadzić kontrolę testów. Kontrola testów ma na celu przekierowanie projektu i/lub testowania w bardziej pomyślnym kierunku. Aby wpłynąć na nadzór nad testami w projekcie lub zmierzyć jego pracochłonność, należy (korzystając z wyników testów) rozważyć następujące opcje:

- zrewidowanie analizy ryzyka jakościowego, priorytetów testów i/lub planów testów,
- dodanie zasobów lub zwiększenie nakładu pracy związanej z projektowaniem lub testowaniem w inny sposób,
- opóźnienie daty wydania,
- rozluźnienie lub wzmocnienie kryteriów wyjścia testów,
- zmiana zakresu (funkcjonalnego i/lub niefunkcjonalnego) projektu.

Wdrożenie tych opcji zazwyczaj wymaga konsensusu pomiędzy interesariuszami projektu lub operacji oraz zgody: kierownika/kierowników projektu lub operacji.

Informacje dostarczone w sumarycznym raporcie z testów w dużej mierze zależą od potrzeb informacyjnych docelowych odbiorców, np. kierownictwa projektu lub kierownictwa biznesowego. W przypadku kierownika projektu interesujące może być posiadanie szczegółowych informacji dotyczących defektów; dla kierownika biznesowego głównym zagadnieniem do raportowania może być status ryzyk produktowych.

2.7. Wartość biznesowa testowania

Kierownik Testów powinien dążyć do takiej optymalizacji procesu testowego, aby dostarczyć najlepszą możliwą do osiągnięcia wartość biznesową. Zbyt obszerne testowanie nie dostarcza dobrej wartości biznesowej, ponieważ będzie powodować nieuzasadnione opóźnienia i spowoduje zwiększenie kosztów, nie generując oszczędności. Niewielkie zaangażowanie w proces testowy skutkuje dostarczeniem użytkownikom produktu o niskiej wartości biznesowej, ze znaczną liczbą defektów. Optimum leży pomiędzy tymi dwiema skrajnościami. Kierownik Testów musi pomóc interesariuszom zrozumieć to optimum i wartość dostarczaną przez testowanie.

Podczas gdy większość organizacji uważa testowanie za wartościowe w pewnym sensie, niewielu kierowników, łącznie z Kierownikami Testów, potrafi tę wartość określić ilościowo, opisać ją lub wyrazić w inny sposób. Ponadto, wielu Kierowników Testów, Liderów Testów i testerów skupia się na taktycznych szczegółach testowania (aspektach specyficznych dla zadania lub poziomu testów), jednocześnie ignorując szersze zagadnienia strategiczne (wyższego poziomu) związane z testowaniem, które interesują innych uczestników projektu, zwłaszcza kierowników.

Testowanie dostarcza wartość dla organizacji, projektu i/lub operacji zarówno w sposób ilościowy, jak i jakościowy:

- Wartości ilościowe obejmują znajdowanie defektów, którym się zapobiega lub które się naprawia przed wydaniem; znajdowanie defektów, które są znane przed wydaniem (nienaprawione, ale udokumentowane, być może z rozwiązaniami tymczasowymi); zmniejszenie ryzyka poprzez wykonanie testów i dostarczenie informacji o statusie projektu, procesu i produktu.
- Wartości jakościowe obejmują lepszą reputację w kwestii jakości, płynniejsze i bardziej przewidywalne wydania, zwiększone zaufanie, ochronę od odpowiedzialności prawnej oraz zmniejszenie ryzyka utraty całego zadania, a nawet życia (w związku z działaniem systemów krytycznych ze względów bezpieczeństwa).

Kierownicy Testów powinni zrozumieć, które z tych wartości mają zastosowanie w ich organizacji, projektu i/lub działań i umieć komunikować się w sprawie testowania w kontekście tych wartości.

Uznana metoda pomiaru wartości ilościowej i wydajności testowania nazywa się kosztem jakości (lub – czasami – kosztem słabej jakości). Koszt jakości obejmuje podział kosztów projektu i kosztów operacyjnych na cztery kategorie związane z kosztami wykrycia i usunięcia defektów w produkcji:

- koszty zapobiegania (działań prewencyjnych), np. szkolenie programistów, aby pisali łatwiejszy do utrzymania i bezpieczniejszy kod,
- koszty wykrycia, np. pisanie przypadków testowych, konfigurowanie środowisk testowych i przeglądy wymagań,
- koszty awarii wewnętrznych, np. naprawiania defektów wykrytych w czasie testowania lub przeglądów przed wydaniem,
- koszty awarii zewnętrznych, np. wsparcie kosztów związanych z wadliwym oprogramowaniem dostarczonym klientom.

Częścią budżetu na testowanie są koszty wykrycia (tj. pieniądze, które zostałyby wydane, nawet gdyby testerzy nie znaleźli żadnych defektów, takie jak pieniądze wydane na zaprojektowanie testów), podczas gdy pozostała część to koszt awarii wewnętrznych (tj. faktyczny koszt związany z wykrytymi defektami). Całkowite koszty wykrycia i wewnętrznych awarii są zwykle o wiele niższe od kosztów awarii zewnętrznej, co nadaje testowaniu wysoką wartość. Poprzez określenie kosztów w tych czterech kategoriach, Kierownik Testów może wykazać przekonującą wartość biznesową testowania.

Więcej informacji na temat wartości testowania, wraz z kosztem jakości, można znaleźć w [Black03].

2.8. Testowanie rozproszone, zlecone na zewnątrz, zlecone wewnątrz

W wielu przypadkach, część lub być może nawet całość prac związanych z testowaniem jest wykonywana przez osoby pracujące w różnych lokalizacjach, zatrudnionych przez różne firmy lub oddzielonych od zespołu projektowego. Jeżeli prace związane z testowaniem wykonywane są w wielu lokalizacjach, prace takie są rozproszone. Jeżeli prace związane z testowaniem są wykonywane w jednej lub wielu lokalizacjach przez osoby, które nie są pracownikami firmy i które nie znajdują się w tej samej lokalizacji, co zespół projektowy, prace zlecone są na zewnątrz. Jeżeli prace związane z testowaniem są wykonywane przez osoby, które znajdują się w tej samej lokalizacji, co zespół projektowy, ale nie są współpracownikami, są one zlecone wewnątrz.

Dla wszystkich rodzajów prac związanych z testowaniem wspólna jest potrzeba posiadania dobrze zdefiniowanych kanałów komunikacji i dobrze zdefiniowanych oczekiwań dla misji, zadań i produktów. Zespół testowy musi mniej polegać na nieformalnych kanałach komunikacji, takich jak dyskusje na korytarzu i koledzy spędzający czas towarzysko. Ważne jest to, aby były zdefiniowane metody

komunikacji, łącznie z zajęciem się takimi tematami jak: eskalacja problemów, rodzaje informacji, które mają być przekazywane oraz metody komunikacji, które mają być wykorzystywane. Aby uniknąć nieporozumień i nierealistycznych oczekiwań, wszyscy, po wszystkich stronach relacji zespołowych, muszą jasno rozumieć swoje role i obowiązki, jak również role i obowiązki innych osób. Pojawienie się problemów z komunikacją i oczekiwaniami jest bardziej prawdopodobne w przypadku różnic w lokalizacji, stref czasowych, różnic kulturowych i językowych.

Dla wszystkich rodzajów prac związanych z testowaniem wspólna jest również potrzeba dopasowania metodyk. O ile w każdym projekcie może wystąpić niedopasowanie metodyk, o tyle bardziej prawdopodobne jest, że będzie miało to miejsce w sytuacjach, gdy praca jest wykonywana w środowisku rozproszonym i/lub wykonywana jest przez zewnętrzne podmioty. Jeżeli dwa zespoły testowe używają różnych metodyk lub zespół testowy korzysta z innej metodyki niż programiści lub kierownictwo projektu, może to spowodować poważne problemy, zwłaszcza w czasie wykonywania testów. Na przykład, jeżeli klient wykorzystuje zwinną metodykę wytwarzania oprogramowania, podczas gdy dostawca usług testowania posiada wcześniej zdefiniowaną metodykę testowania, która zakłada sekwencyjny cykl życia oprogramowania, punktem spornym będzie koordynacja prac w czasie i charakter dostaw przedmiotów testów do dostawcy usług testowania.

W przypadku testowania rozproszonego, musi być wyraźny podział pracy testowej między wieloma lokalizacjami, a decyzje muszą być podejmowane w inteligentny sposób. Bez takiego nakierowania, najbardziej kompetentny zespół może nie wykonać pracy testowej, do której posiada kwalifikacje. Ponadto, jeżeli każdy zespół nie wie, za co jest odpowiedzialny, może nie wykonać tego, co miał wykonać. Oczekiwania w stosunku do każdego zespołu muszą zostać jasno wyrażone. Bez uważnego zarządzania prace testowe, jako całość, mogą uciec z powodu luk, które zwiększają pozostałe ryzyko jakościowe dla systemu i nakładania prac na siebie (co zmniejsza wydajność).

Na koniec, w przypadku wszystkich prac związanych z testowaniem, cały zespół testowy musi bezwzględnie zbudować i utrzymać zaufanie, że wszystkie zespoły testowe wykonają swoje zadania właściwie pomimo barier organizacyjnych, kulturowych, językowych i geograficznych. Brak zaufania prowadzi do niewydolności i opóźnień związanych z czynnościami weryfikacyjnymi, wzajemnego obwiniania za problemy i uprawiania polityki.

2.9. Zarządzanie z zastosowaniem norm branżowych

Zarówno w sylabusach poziomu podstawowego, jak i zaawansowanego, znajdują się odniesienia do kilku norm. Normy, o których mowa, obejmują cykle życia wytwarzania oprogramowania, testowanie oprogramowania, cechy jakości oprogramowania, przeglądy i zarządzanie defektami. Kierownicy Testów powinni posiadać wiedzę na temat tych norm, polityki swojej organizacji w sprawie wykorzystania norm oraz tego, czy w ich przypadku normy są wymagane, konieczne lub użyteczne.

Normy mogą pochodzić z różnych źródeł, takich jak:

- źródła międzynarodowe lub źródła zawierające międzynarodowe cele,
- źródła krajowe, takie jak krajowe zastosowania norm międzynarodowych,
- źródła specyficzne dla dziedziny, takie jak w przypadku, gdy normy międzynarodowe lub krajowe są dostosowywane do konkretnych dziedzin lub są opracowywane dla konkretnych dziedzin.

Międzynarodowe jednostki normatywne obejmują standardy ISO oraz IEEE. ISO to Międzynarodowa Organizacja Normatywna, zwana również IOS, Międzynarodową Organizacją dla Standaryzacji. Składa się ona z członków reprezentujących w imieniu swojego kraju krajową jednostkę najbardziej istotną dla

dziedziny, która jest standaryzowana. Jednostka ta wypromowała kilka norm przydatnych dla testerów oprogramowania, takich jak ISO 9126⁷, ISO 12207 [ISO12207] oraz ISO 15504 [ISO15504].

IEEE to Instytut Inżynierów Elektryków i Elektroników, organizacja profesjonalistów zlokalizowana w Stanach Zjednoczonych, ale posiadająca przedstawicieli krajowych w ponad stu krajach. Organizacja ta zaproponowała kilka norm, użytecznych dla testerów oprogramowania, takich jak IEEE 829 [IEEE829]⁸ oraz IEEE 1028 [IEEE1028]⁹.

Wiele krajów posiada własne normy krajowe. Niektóre z nich są przydatne dla testowania oprogramowania. Przykładem jest brytyjska norma BS 7925-2 [BS7925-2], dostarczająca informacji dotyczących wielu technik projektowania testów opisanych w sylabusach dla poziomu zaawansowanego – Analityk Testów oraz Techniczny Analityk Testowy.

Niektóre normy są specyficzne dla danej dziedziny, a niektóre z nich mają konsekwencje dla testowania oprogramowania, jakości oprogramowania i wytwarzania oprogramowania. Na przykład, w dziedzinie awioniki, do oprogramowania wykorzystywanego w samolotach cywilnych ma zastosowanie norma amerykańskiej Federalnej Administracji Lotnictwa DO-178B (i jej odpowiednik UE ED 12B)¹⁰. Norma ta ustanawia pewne poziomy kryteriów pokrycia strukturalnego w oparciu o poziom krytyczności oprogramowania, które jest testowane.

Inny przykład normy specyficznej dla dziedziny można znaleźć w systemach medycznych w wydanym przez Amerykańską Agencję ds. Żywności i Leków, standardzie Tytuł 21 CFR Część 820 [FDA21]. Standard ten zaleca pewne techniki testowania strukturalnego i funkcjonalnego, a także strategię testów i zasady, które są zgodne z sylabusami ISTQB®.

W niektórych przypadkach na testowanie wpływają normy lub szeroko rozpowszechnione metodyki, które nie zajmują się przede wszystkim testowaniem, ale które mają duży wpływ na kontekst procesu wytwarzania oprogramowania, w którym testowanie się odbywa. Jednym z przykładów jest model doskonalenia procesu wytwarzania oprogramowania CMMI®. Obejmuje on dwa kluczowe obszary procesu: weryfikację i walidację, które często są interpretowane jako odnoszące się do poziomów testów (takich jak, odpowiednio, testowanie systemowe i testowanie akceptacyjne). Model ten dostarcza wniosków dotyczących strategii testów, często interpretowanych jako wymaganie włączenia testowania analitycznego w oparciu o wymagania zawarte w strategii testów.

Trzy inne, istotne przykłady, to PMI's PMBOK, PRINCE2® oraz ITIL®. PMI oraz PRINCE2® to często wykorzystywane modele zarządzania projektami, odpowiednio w Ameryce Północnej i Europie. ITIL® to model, który ma na celu zapewnienie, że grupa IT dostarcza organizacji, w której funkcjonuje, wartościowych usług. Terminologia i czynności określone w tych modelach znacznie się różnią od sylabusu i słownika ISTQB®. Podczas pracy w organizacji, która wykorzystuje PMI's PMBOK, PRINCE2 i/lub ITIL®, Kierownik Testów musi rozumieć wybrane modele, ich implikacje i terminologię wystarczająco dobrze, aby efektywnie pracować w danym kontekście.

Niezależnie od tego, jakie normy lub metodyki są przyjmowane, należy pamiętać, że zostały stworzone przez grupy profesjonalistów. Normy odzwierciedlają zbiorowe doświadczenie i wiedzę grupy źródłowej, ale również jej słabe strony. Kierownicy Testów powinni znać normy, które mają zastosowanie w ich

⁷ Standard ISO/IEC 25010 zastąpił normę ISO 9126.

⁸ Standard ISO/IEC/IEEE 29119 zastąpił normę IEEE 829.

⁹ Standard ISO/IEC 20246 zastąpił normę IEEE 1028.

¹⁰ Standard FAA DO-178B/ED 12B zastąpił normę FAA DO-178C/ED 12C.

środowisku i kontekście, bez względu na to, czy są to normy formalne (międzynarodowe, krajowe lub specyficzne dla dziedziny), czy normy i zalecane praktyki wewnętrzne.

Biorąc pod uwagę stosowanie wielu norm, należy pamiętać, że niektóre normy są niezgodne z innymi normami lub nawet przedstawiają sprzeczne definicje. Kierownik Testów powinien określić przydatność różnych norm dla określonego kontekstu, w którym testowanie się odbywa. Informacje podane w normie mogą być użyteczne dla projektu albo stanowić utrudnienie. Niemniej jednak, normy mogą zapewnić odniesienie do udowodnionych dobrych praktyk i zapewnić podstawę do zorganizowania procesu testowego.

W niektórych przypadkach, zgodność z normami jest zalecana i ma implikacje dla testowania. Kierownik Testów musi mieć świadomość wszelkich takich wymagań, aby przestrzegać norm i upewnić się, że zostanie utrzymana stosowna zgodność.

3. Przeglądy – 180 min.

Słowa kluczowe

audyt, przegląd nieformalny, inspekcja, przegląd kierowniczy, moderator, przegląd, plan przeglądu, przeglądający, przegląd techniczny, przejrzanie

Przeglądy – cele nauczania

3.2. Przegląd kierowniczy i audyty

TM-3.2.1. (K2) Kandydat rozumie kluczowe cechy przeglądów kierowniczych i audytów.

3.3. Zarządzanie przeglądami

TM-3.3.1. (K4) Kandydat potrafi dokonać analizy projektu, wyboru odpowiedniego rodzaju przeglądu i określić plan prowadzenia przeglądów w celu zapewnienia właściwej realizacji, kontynuacji i odpowiedzialności.

TM-3.3.2. (K2) Kandydat rozumie czynniki sukcesu, umiejętności i czas wymagany do udziału w przeglądzie.

3.4. Metryki dla przeglądów

TM-3.4.1. (K3) Kandydat potrafi zdefiniować metryki procesu i produktu, które będą używane w przeglądzie.

3.5. Zarządzanie przeglądem formalnym

TM-3.5.1. (K2) Kandydat potrafi wyjaśnić, podając przykłady, cechy przeglądu formalnego.

3.1. Wprowadzenie

W programie sylabusu Certyfikowany tester – Sylabus poziomu podstawowego ISTQB® przeglądy zostały przedstawione jako statyczne podejście do testowania produktów. Audyt i przegląd kierowniczy koncentrują się bardziej na procesie wytwarzania oprogramowania, niż na produktach pracy wytwórczej.

Ponieważ przeglądy są formą testów statycznych, Kierownicy Testów są również odpowiedzialni za ich sukces, szczególnie w odniesieniu do testowanego produktu. Jednak w szerszym kontekście projektów informatycznych, odpowiedzialność ta powinna być przedmiotem polityki organizacyjnej. Biorąc pod uwagę możliwe powszechne zastosowanie formalnych przeglądów w wielu dyscyplinach, zarówno przed, jak i w ramach realizowanych projektów oprogramowania, stroną, na której spoczywa odpowiedzialność, może być Kierownik Testów, Kierownik Zapewnienia Jakości lub przeszkolony Koordynator Przeglądu. W niniejszym sylabusie strona, na której spoczywa odpowiedzialność (ktokolwiek to jest), jest określana jako Lider Przeglądu.

Lider Przeglądu powinien zapewnić, że istnieje atmosfera, która sprzyja realizacji czynników sukcesu określonych w programie nauczania Sylabusu poziomu podstawowego ISTQB®. Ponadto Lider Przeglądu powinien opracować plan pomiarów w celu zapewnienia, że realizowany przegląd dostarczy wartość dodaną.

Równie ważne jest zaangażowanie testera w proces przeglądu, ponieważ testerzy dobrze rozumieją operacyjne zachowania i wymagane cechy oprogramowania.

W celu zrozumienia swojej roli podczas przeglądu, uczestnicy przeglądu powinni być przeszkoleni z procesu prowadzenia przeglądu. Wszyscy uczestnicy przeglądu powinni być przekonani co do korzyści, jakie wynikają z dobrze przeprowadzonego przeglądu.

Przeglądy, o ile są wykonane prawidłowo, są najważniejszymi i najbardziej opłacalnymi czynnikami, które wpływają na jakość dostarczanego rozwiązania. Zatem niezwykle istotne jest to, aby Liderzy Przeglądów mogli realizować skuteczne przeglądy w swoich projektach i pokazywać płynące z ich korzyści.

Możliwe przeglądy prowadzone w ramach projektu to:

- Przeglądy wymagań, rozpoczęte w momencie, gdy wymagania są dostępne do przeglądu, który umożliwia idealne sprawdzenie zarówno wymagań funkcjonalnych, jak i нефункциональных.
- Przeglądy projektu na wysokim poziomie, rozpoczęte wtedy, gdy całościowy projekt architektoniczny jest dostępny do przeglądu.
- Szczegółowe przeglądy projektu, rozpoczęte, gdy szczegółowy projekt jest dostępny do przeglądu.
- Przeglądy kodu, przeprowadzane dla poszczególnych modułów tworzonego oprogramowania, które mogą obejmować testy jednostkowe i ich wyniki, a także przeglądy samego kodu.
- Przeglądy produktów pracy procesu testowego, które mogą obejmować plan testu, warunki testowe, wynik analizy jakości na podstawie ryzyka, testy, dane testowe, środowiska testowe i wyniki testów.
- Przegląd kryteriów wejścia testów (gotowość testów) oraz kryteriów wyjścia z testów dla każdego poziomu testów, który odpowiednio umożliwia sprawdzenie kryteriów wejścia przed rozpoczęciem wykonywania testów i kryteriów wyjścia przed zakończeniem testów.
- Przeglądy akceptacyjne, wykorzystywane w celu akceptacji systemu przez klienta lub innych interesariuszy.

Oprócz stosowania wielu technik przeglądów produktu, Lider Przeglądu powinien pamiętać, że w trakcie przeglądu w dokumencie można znaleźć defekty w sposób statyczny, zaś przeglądy należy

rozszerzyć o inne techniki testowania statycznego (takie jak analiza statyczna) i testy dynamiczne kodu. Użycie kombinacji tych technik wpływa na poprawę pokrycia testowego i pozwala na wykrycie większej liczby defektów.

Różne techniki mają różne cele. Na przykład, przegląd może skutkować wyeliminowaniem problemu na poziomie wymagań, zanim problem ten zostanie wprowadzony do kodu. Analiza statyczna może pomóc w egzekwowaniu standardów kodowania i pozwoli sprawdzić, czy problemy, które mogą być zbyt pracochłonne dla zespołu, zostały znalezione poprzez rewizję produktów pracy. Inspekcje mogą nie tylko prowadzić do wykrycia i usunięcia defektów, ale także mogą uczyć autorów, w jaki sposób uniknąć tworzenia defektów w produktach swojej pracy.

Sylabus Certyfikowany tester - Sylabus poziomu podstawowego ISTQB® wprowadza następujące rodzaje przeglądów:

- przegląd nieformalny,
- przejrzanie,
- przegląd techniczny,
- inspekcja.

Oprócz nich, Kierownicy Testów mogą być również zaangażowani w:

- przeglądy kierownicze,
- audyty.

3.2. Przegląd kierowniczy i audyty

Przeglądy kierownicze są wykorzystywane do monitorowania postępu, oceny statusu i podejmowania decyzji dotyczących przyszłych działań. Przeglądy te wspierają decyzje o przyszłości projektu, takie jak dostosowanie poziomu zasobów do potrzeb projektowych, wdrażanie działań naprawczych lub zmian mających wpływ na zakres projektu.

Kluczowe cechy przeglądu kierowniczego:

- prowadzony przez lub dla menedżerów mających bezpośrednią odpowiedzialność za projekt lub system,
- prowadzony przez lub dla interesariuszy lub osób podejmujących decyzję np. kierownik wyższego szczebla lub dyrektor,
- sprawdzenie zgodności lub odchyień od planów,
- sprawdzenie adekwatności procedur zarządzania,
- ocena ryzyka projektu,
- ocena wpływu działań i sposoby pomiaru skutków tych działań,
- sporządzenie listy działań, problemów, które należy rozwiązać i decyzji, które należy podjąć.

Integralną częścią czynności doskonalenia procesów są przeglądy kierownicze procesów, takie jak retrospektywy projektu (tj. wyciągnięcie wniosków).

Kierownicy Testów powinni uczestniczyć w przeglądach procesu testowego i mogą je inicjować.

Audyty są wykonywane zwykle w celu wykazania zgodności procesu testowego z określonym zestawem kryteriów, najprawdopodobniej wynikających z zastosowanego standardu, ograniczeń prawnych lub umów. Audyty mają dostarczyć niezależną ocenę zgodności z procesami, normami itp.

Kluczowe cechy audytów:

- prowadzone i moderowane przez Audytora Wiodącego,
- dowody zgodności zebrane poprzez wywiady, świadectwa i sprawdzone dokumenty,

- udokumentowane wyniki zawierają uwagi, rekomendacje, działania korygujące i oceny zaliczenia / niezaliczenia.

3.3. Zarządzanie przeglądami

Przeglądy powinny być planowane po zakończeniu faz projektu lub w kamieniach milowych projektu. Przeglądy powinny się zazwyczaj odbywać po zebraniu wymagań i definicji projektu, poczynając od celów biznesowych, aż do najniższego poziomu szczegółowości projektu. Przeglądy kierownicze powinny zwykle odbywać się dla głównych etapów projektu, często w ramach czynności weryfikacji przed, w trakcie i po wykonaniu testów oraz dla innych istotnych etapów projektu. Strategia przeglądu powinna być skoordynowana z polityką testów i ogólną strategią testów.

Przed sformułowaniem całościowego planu przeglądu na poziomie projektu, Lider Przeglądu (którym może być Kierownik Testów) powinien uwzględnić:

- co powinno być poddane przeglądowi (produkty i procesy),
- kto powinien być zaangażowany w konkretne przeglądy,
- jakie są istotne czynniki ryzyka do pokrycia.

Na początku fazy planowania projektu, Lider Przeglądu powinien określić elementy, które powinny być poddane analizie i wybrać odpowiedni rodzaj przeglądu (przegląd nieformalny, przejrzenie, przegląd techniczny lub inspekcję albo kombinację dwóch lub trzech typów przeglądów) oraz poziom jego sformalizowania. W tym miejscu może być zalecane dodatkowe szkolenie z procesu przeglądu. Od tego momentu może być przydzielony budżet (czas i zasoby) dla procesu przeglądu. Ustalenie budżetu powinno uwzględniać ocenę ryzyka i zwrotu z inwestycji.

Zwrot z inwestycji dla przeglądów jest różnicą pomiędzy kosztem przeprowadzenia przeglądu, a kosztem naprawy defektów na późniejszym etapie (lub braku napraw), jeśli przegląd nie był realizowany. Sposób obliczenia kosztu jakości przedstawiony jest w sekcji 2.7. i może być pomocny w celu ustalenia tej wartości.

Określenie optymalnego czasu przeprowadzenia przeglądów zależy od następujących czynników:

- dostępność różnych elementów będących przedmiotem przeglądu w formie ostatecznej,
- dostępność odpowiednich pracowników,
- termin, w którym powinna być dostępna ostateczna wersja produktu,
- czas potrzebny na realizację procesu przeglądu dla tego konkretnego elementu.

W trakcie planowania testów Lider Przeglądu powinien określić odpowiednie miary do oceny przeglądu. Jeżeli przeprowadzane są inspekcje, to na wniosek autora, po ukończeniu fragmentu dokumentu (np. indywidualnych wymagań lub ich części), powinny być przeprowadzone krótkie inspekcje.

Podczas planowania testów należy określić cele procesu przeglądu. Obejmują one prowadzenie skutecznych i efektywnych przeglądów i podejmowanie decyzji zgodnie z informacją zwrotną z przeglądu.

Przeglądy projektów często dotyczą całego systemu, mogą być konieczne dla podsystemów, a nawet dla poszczególnych elementów oprogramowania. Liczba przeglądów, rodzaje przeglądów, organizacja przeglądów oraz zaangażowane osoby, wszystko to zależy od wielkości i złożoności projektu oraz od ryzyka produktowego.

Aby przeglądy były skuteczne, uczestnicy przeglądów muszą posiadać odpowiedni poziom wiedzy, zarówno technicznej, jak i proceduralnej. Dokładność i dbałość o szczegóły są nieco innymi umiejętnościami wymaganymi od recenzentów, aby realizować skuteczne przeglądy. W komentarzach

recenzentów warto szukać atrybutów takich jak przejrzystość i właściwe priorytety. Może zaistnieć potrzeba przeprowadzenia szkoleń z zakresu wiedzy proceduralnej, aby upewnić się, że recenzenci rozumieją swoje role i odpowiedzialności w procesie przeglądu.

Planowania przeglądu powinno uwzględniać ryzyko związane z czynnikami technicznymi, czynnikami organizacyjnymi i czynnikami ludzkimi przy wykonywaniu przeglądu. Do przeprowadzenia udanego przeglądu krytyczna jest dostępność recenzentów z wystarczającą wiedzą techniczną. W planowanie przeglądów powinny być zaangażowane wszystkie zespoły w ramach projektu, zapewniając wpływ każdego zespołu na sukces procesu przeglądu. Na etapie planowania należy zapewnić, że każda organizacja przeznacza wystarczającą ilość czasu na niezbędne przygotowanie recenzentów i ich udział w przeglądzie, na odpowiednim etapie harmonogramu projektu. Należy również zaplanować czas na wymagane szkolenia techniczne i obowiązki dla recenzentów. Na wypadek niedostępności recenzentów ze względu na zmiany w planach osobistych lub biznesowych, należy wskazać zapasowych recenzentów.

Podczas realizacji formalnych przeglądów, Lider Przeglądu musi zapewnić, że:

- uczestnicy przeglądu realizują odpowiednie metryki umożliwiające ocenę efektywności przeglądu,
- u celu poprawy przyszłych przeglądów tworzone i utrzymywane są listy kontrolne,
- w podejściu do zarządzania defektami podczas przeglądów (patrz Rozdział 4.) zdefiniowano wagę defektów i priorytet.

Po każdym przeglądzie, Lider Przeglądu powinien:

- zbierać metryki z przeglądu i zapewnić, że zidentyfikowane kwestie są odpowiednio wyjaśnione w kontekście specyficznych celów testów,
- użyć metryk przeglądu w celu określenia zwrotu z inwestycji (ROI – ang. *Return on Investment*),
- zapewnić informację zwrotną dla właściwych interesariuszy,
- zapewnić informację zwrotną dla uczestników przeglądu.

Aby ocenić skuteczność przeglądu, Kierowników Testów może porównać faktyczne wyniki realizowanych testów (tj. po przeglądzie) z wynikami raportów z przeglądu. W przypadku, gdy przeglądany produkt został zatwierdzony na podstawie przeglądu, ale później okazał się wadliwy, Lider Przeglądu powinien rozważyć, w jaki sposób w procesie przeglądu ominięto niektóre defekty. Prawdopodobne przyczyny to problemy z procesem przeglądu (np. słabe kryteria wejścia/wyjścia), niewłaściwy skład zespołu recenzującego, nieodpowiednie narzędzia weryfikacji (listy kontrolne itp.), niewystarczające przeszkolenie recenzentów, za mało czasu na przygotowanie oraz realizację przeglądu.

Standardowe wycieki defektów (zwłaszcza głównych defektów), powtarzane w kilku projektach, wskazują, że istnieją znaczne problemy z przeprowadzanymi przeglądami. W takiej sytuacji Lider Przeglądu musi dokonać weryfikacji procesu przeglądu i podjąć odpowiednie działania. Możliwe jest również, że z różnych powodów przeglądy mogły stracić swoją skuteczność w czasie. Efekt ten zostanie ujawniony podczas retrospektyw projektu poprzez stwierdzenie o zmniejszeniu skuteczności wykrywania defektów z przeglądów. Tutaj znowu Lider Przeglądu musi zbadać i ustalić przyczyny. W każdym przypadku metryki z przeglądu nie powinny być wykorzystywane do karania lub nagradzania poszczególnych recenzentów i autorów, ale powinny wskazywać na konieczność skoncentrowania się na samym procesie przeglądu.

3.4. Metryki dla przeglądów

Liderzy Przeglądu, (którymi, jak wspomniano w poprzednich rozdziałach, mogą być Kierownicy Testów) muszą zapewnić metryki dla:

- oceny jakości przeglądanej pozycji,
- oceny kosztów przeprowadzenia przeglądu,
- oceny korzyści z dalszego prowadzenia przeglądu.

Liderzy Przeglądów mogą korzystać z miar w celu określenia zwrotu z inwestycji i efektywności przeglądu. Metryki te mogą być również wykorzystywane do raportowania oraz działań związanych z doskonaleniem procesów.

Dla każdego przeglądanego produktu, następujące metryki mogą być zbierane i zaraportowane w celu oceny produktu:

- rozmiar produktu (strony, linie kodu, itp.),
- czas przygotowania (przed przeglądem),
- czas do przeprowadzenia przeglądu,
- czas naprawy defektów,
- czas trwania procesu przeglądu,
- ilość znalezionych defektów i ich ważność,
- identyfikacja obszarów defektów w produkcie (tj. obszary, które mają wyższą gęstość defektów),
- rodzaj przeglądu (przegląd nieformalny, przejrzenie, przegląd techniczny lub inspekcja),
- średnia gęstość defektów (np. ilość defektów na stronę lub na tysiąc linii kodu),
- szacowanie pozostałych defektów (lub śladowej gęstości defektów).

Dla każdego przeglądu mogą być mierzone i raportowane do oceny procesu następujące metryki:

- skuteczność wykrywania defektów (biorąc pod uwagę defekty znalezione później w cyklu życia),
- udoskonalenie procesu przeglądu, poświęconego czasu i nakładu pracy,
- procent pokrycia planowanych produktów pracy,
- rodzaje defektów wykrytych i ich ważność,
- ankiety uczestnika o skuteczności i efektywności procesu przeglądu,
- koszt metryk defektów z przeglądu w porównaniu do defektów z testów dynamicznych a koszt defektów produkcyjnych,
- korelacja skuteczności przeglądu (typ przeglądu a skuteczność wykrywania defektów),
- liczba przeglądających,
- defekty znalezione na godzinę pracy,
- przewidywany czas trwania projektu,
- średni czas naprawy defektu (np. całkowity czas wykrycia i czas naprawy podzielony przez ilość defektów).

Ponadto, wymienione powyżej metryki do oceny produktu są również użyteczne w ocenie procesu.

3.5. Zarządzanie przeglądem formalnym

Sylabus Certyfikowany tester – Sylabus poziomu podstawowego ISTQB® przedstawia różne kroki przeglądu formalnego: planowanie, rozpoczęcie przeglądu, przegląd indywidualny (indywidualne przygotowanie), przekazanie informacji o problemach i analiza problemów, usunięcie defektów i raportowanie¹¹. Aby prawidłowo przeprowadzić przegląd formalny, Liderzy Przeglądu muszą sprawdzić, czy przestrzegane są wszystkie kroki w procesie przeglądu formalnego.

Przeglądy formalne mają następujące cechy:

- zdefiniowane kryteria wejścia i wyjścia,
- dostarczone listy kontrolne do wykorzystania przez przeglądających,

¹¹ Standard ISO/IEC 20246 zastąpił normę IEEE 1028.

- dostarczone raporty, arkusze ocen lub inne arkusze podsumowujące,
- metryki w celu raportowania skuteczności, wydajności i rozwoju procesu przeglądu.

Przed rozpoczęciem przeglądu formalnego, Lider Przeglądu powinien potwierdzić warunki początkowe (określone w procedurze albo na liście kryteriów wejścia).

Jeśli nie są spełnione warunki początkowe dla formalnego przeglądu, w celu podjęcia ostatecznej decyzji, Lider Przeglądu może zaproponować organowi odpowiadającemu za przegląd jedną z następujących czynności:

- weryfikacja przeglądu ze zmienionymi założeniami,
- działania naprawcze niezbędne do kontynuacji przeglądu,
- odroczenie przeglądu.

W ramach kontroli przeglądu formalnego, przeglądy te są monitorowane w kontekście ogólnego (na najwyższym poziomie) programu i są związane z działaniami zapewnienia jakości w projekcie. Kontrola przeglądów formalnych pozwala na zebranie informacji zwrotnych za pomocą metryk produktu i procesu.

4. Zarządzanie defektami – 150 min.

Słowa kluczowe

Anomalia, defekt/usterka, konsylium nad defektami (komitet zarządzania usterkami), awaria, rezultat fałszywie negatywny (rezultat fałszywie zaliczony), rezultat fałszywie pozytywny (rezultat fałszywie niezaliczony), powstrzymanie fazowe, priorytet, podstawowa przyczyna, krytyczność

Zarządzanie defektami – cele nauczania

4.2. Cykl życia defektu i cykl procesu wytwórczego

- TM-4.2.1. (K3) Kandydat potrafi opracować proces zarządzania defektami dla organizacji testowej, włącznie z obiegiem raportu o defekcie, który może być użyty do monitorowania i kontroli defektów pojawiających się w projekcie w całym cyklu życia wytwarzania oprogramowania.
- TM-4.2.2. (K2) Kandydat potrafi wyjaśnić proces i jego uczestników niezbędnych dla skutecznego zarządzania defektami.

4.3. Informacja w raporcie o defekcie

- TM-4.3.1. (K3) Kandydat potrafi określić dane i informacje klasyfikacyjne, które powinny być zebrane w trakcie procesu zarządzania defektami.

4.4. Ocena zdolności procesu testowego przy użyciu informacji z raportu o defekcie

- TM-4.4.1. (K2) Kandydat potrafi wyjaśnić, w jaki sposób statystyki raportów o defektach mogą być używane do oceny zdolności procesu testowego oraz procesów rozwoju oprogramowania.

4.1. Wprowadzenie

Proces zarządzania defektami w danej organizacji i narzędzie używane do zarządzania defektami ma ogromne znaczenie nie tylko dla pracy zespołu testowego, ale również dla wszystkich zespołów biorących udział w procesie rozwoju oprogramowania. Informacja zebrana w trakcie skutecznego procesu zarządzania defektami pozwala Kierownikowi Testów i innym interesariuszom projektu na ustalenie statusu realizacji projektu w całym cyklu życia wytwarzania oprogramowania, a poprzez gromadzenie i analizowanie danych na przestrzeni czasu, może pomóc zlokalizować obszary potencjalnego doskonalenia testów i procesu wytwórczego.

Oprócz zrozumienia ogólnego cyklu życia defektu i sposobu, w jakim jest on użyty do monitorowania i nadzorowania zarówno testów, jak i procesu rozwoju oprogramowania, Kierownik Testów musi również posiadać wiedzę o tym, jakie kluczowe dane należy gromadzić i musi być zwolennikiem postępowania według zasad prawidłowego procesu rozwoju oprogramowania, jak i zasad poprawnego użytkowania wybranego narzędzia do zarządzania defektami.

4.2. Cykl życia defektu i cykl procesu wytwórczego

Jak wyjaśniono w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB®, defekty są wprowadzane wtedy, gdy osoba popełni błąd podczas tworzenia jakiegoś produktu pracy. Takim produktem może być specyfikacja wymagań, historia użytkownika, dokumentacja techniczna, przypadek testowy, kod programu czy dowolny inny produkt powstały w procesie rozwoju oprogramowania lub na etapie procesu utrzymania.

Defekty mogą być wprowadzane w dowolnym etapie cyklu wytwarzania oprogramowania i w każdym produkcie powstałym w procesie wytwórczym, dlatego każda faza cyklu wytwarzania oprogramowania powinna obejmować działania mające na celu wykrycie i usunięcie ewentualnych defektów. Na przykład, statyczne techniki testowania (tj. przeglądy i analiza statyczna) mogą być używane do weryfikacji specyfikacji projektu, specyfikacji wymagań i kodu przed przekazaniem tych produktów do kolejnych czynności. Im wcześniej defekt jest wykrywany i usuwany, tym niższy jest całkowity koszt jakości dla systemu, zaś koszt jakości dla danego poziomu testów jest minimalizowany wtedy, gdy defekt jest usuwany podczas tej samej fazy, w której został on wykryty, tj. gdy proces wytwarzania oprogramowania osiąga doskonałe powstrzymanie fazowe. Ponadto, jak wyjaśniono w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB®, testy statyczne umożliwiają bezpośrednie znalezienie raczej defektów niż wykrycie awarii, a tym samym koszt usuwania usterki jest niższy, ponieważ nie ma potrzeby wykonywania debugowania w celu odizolowania defektu.

Podczas testów dynamicznych, takich jak testy modułowe, testy integracji i testy systemowe, obecność defektu staje się widoczna w momencie, gdy powoduje on awarię, która skutkuje rozbieżnością pomiędzy rzeczywistymi a oczekiwanymi wynikami testu (tj. anomalia). W niektórych przypadkach pojawia się rezultat fałszywie niezaliczony, gdy tester nie dostrzega anomalii. Jeśli tester dostrzega anomalię, pojawia się sytuacja, która wymaga dalszego zbadania. Badanie to rozpoczyna się od wypełnienia raportu o defekcie.

W wytwarzaniu sterowanym testami - TDD (ang. *Test Driven Development*) jako formy wykonywalnych specyfikacji projektu używa się zautomatyzowanych testów modułowych. Wytwarzany kod jest natychmiast badany za pomocą tych testów. Do momentu pełnego wytworzenia modułu, niektóre lub wszystkie z tych testów nie zostaną zakończone wynikiem pozytywnym. W związku z tym, niezaliczenie takiego testu nie stanowi defektu i zwykle nie jest śledzone.

4.2.1. Cykl życia i stany defektu

Większość organizacji testerskich używa narzędzi do zarządzania raportami o defektach poprzez cykl życia defektu. Raport o defekcie zazwyczaj jest rozwijany poprzez przepływ zadań i w trakcie swojego cyklu życia przechodzi przez sekwencję stanów. W przypadku większości z tych stanów, właścicielem raportu jest jeden uczestnik cyklu życia defektu i to on jest odpowiedzialny za realizację zadania, które po ukończeniu będzie powodować, że raport o defekcie zostanie przeniesiony do następnego stanu (i przypisany do kolejnej odpowiedzialnej osoby). W stanach końcowych, takich jak ten, w którym raport o defekcie jest zamykany (co zwykle oznacza, że defekt został naprawiony i zweryfikowany przez testy potwierdzające), anulowany (co zwykle oznacza, że raport o defekcie jest nieprawidłowy), nieodtworzalny (co zwykle oznacza, że anomalii nie da się już zaobserwować) lub odroczone (co zwykle oznacza, że anomalia dotyczy prawdziwego defektu, ale defekt ten nie będzie naprawiony w trakcie realizacji projektu), raport nie ma właściciela, ponieważ nie są konieczne żadne dalsze działania.

W przypadku defektów wykrytych przez testerów podczas testów, istnieją trzy stany, które wymagają działania zespołu testowego:

- Stan początkowy
 - W tym stanie jeden lub więcej testerów zbiera informacje niezbędne dla osoby odpowiedzialnej za rozwiązywanie defektu, by odtworzyć anomalię (patrz punkt 4.3. dla uzyskania dodatkowych informacji na temat danych, które mają być uwzględnione w raporcie o defekcie).
 - Stan ten może być również określony jako "otwarty" lub "nowy".
- Stan zwrócony
 - W tym stanie odbiorca raportu odrzucił raport lub prosi testera o dostarczenia dodatkowych informacji. Stan ten może wskazywać na braki w procesie zbierania wstępnych informacji lub braki w samym testowaniu, w związku z czym Kierownicy Testów powinni monitorować nadmierne wskaźniki tego stanu. Tester musi dostarczyć dodatkowe informacje lub potwierdzić, że raport rzeczywiście powinien zostać odrzucony.
 - Stan ten może być również określony jako "odrzucony" lub "do wyjaśnienia".
- Stan testu potwierdzającego
 - W tym stanie tester będzie uruchamiać testy potwierdzające (często wykonując kroki do odtworzenia błędu z samego raportu o defekcie), aby stwierdzić, czy poprawka rzeczywiście rozwiązała problem. Jeśli test potwierdzający wskazuje, że defekt został naprawiony, tester powinien zamknąć raport. Jeśli test potwierdzający wskazuje, że defekt nie został naprawiony, tester powinien ponownie otworzyć raport, powodując przypisanie go do poprzedniego właściciela, który może następnie ukończyć prace niezbędne do naprawy defektu.
 - Stan ten może być również określony jako "rozwiązany" lub "weryfikacja".

4.2.2. Zarządzanie błędnymi i zduplikowanymi raportami o defektach

W niektórych przypadkach anomalia pojawia się nie jako symptom defektu, lecz raczej z powodu problemu ze środowiskiem testowym, danymi testowymi, innym elementem testalioń lub z powodu błędnego rozumowania testera. Jeśli tester otworzy raport o defekcie, który następnie okazuje się nie być związany z defektem w produkcie poddawany testom, jest to rezultat fałszywie niezaliczony. Raporty takie są zazwyczaj anulowane lub zamykane jako nieprawidłowe raporty o defektach. W dodatku, w niektórych przypadkach defekt może przejawiać się poprzez różne symptomy, które mogą wydawać się testerowi kompletnie niepowiązane ze sobą. Jeśli stworzono dwa lub więcej raporty o defektach, które następnie okazują się być związane z tą samą przyczyną podstawową, jeden z tych raportów jest zwykle zachowywany, podczas gdy pozostałe są zamykane jako raporty zduplikowane.

Chociaż nieprawidłowe lub zduplikowane raporty o defektach odzwierciedlają pewien poziom nieefektywności, pewna ilość takich raportów jest nieunikniona i powinna być przyjęta jako taka przez Kierowników Testów. Kiedy kierownicy próbują wyeliminować wszystkie nieprawidłowe i zduplikowane raporty o defektach, liczba fałszywie zaliczonych rezultatów zwykle wzrasta, ponieważ testerzy są zniechęceni do zgłaszania raportów o defektach. To zmniejsza skuteczność wykrywania defektów w organizacji testerskiej, która w większości przypadków dotyczy jednego z głównych celów organizacji testerskiej.

4.2.3. Międzyfunkcyjne zarządzanie defektami

Chociaż właścicielami całego procesu zarządzania defektami oraz narzędzia do zarządzania defektami jest zwykle organizacja testerska oraz Kierownik Testów, to za zarządzanie defektami zgłoszonymi w danym projekcie odpowiada międzyfunkcyjny zespół. Oprócz Kierownika Testów, uczestnikami komitetu zarządzania usterkami (lub konsylium nad defektami) są zwykle przedstawiciele zespołu wytwarzającego oprogramowanie, zarządzania projektem, zarządzania produktem oraz inni interesariusze zainteresowani rozwijaniem oprogramowaniem.

Gdy anomalie są wykrywane i zgłaszane w narzędziu do zarządzania defektami, komitet zarządzania usterkami powinien odbywać spotkania mające na celu określenie, czy każdy raport o defekcie zawiera poprawny opis zidentyfikowanego defektu i czy powinien być on naprawiony, czy też odroczone. Decyzja ta wymaga, by komitet zarządzania usterkami rozważył korzyści, ryzyka i koszty związane z naprawą lub brakiem naprawy danego defektu. Jeśli defekt ma zostać naprawiony, zespół powinien ustalić priorytet naprawy defektu względem innych prac projektowych. Kierownik Testów oraz zespół testerski mogą zostać poproszeni o konsultacje dotyczące względnej ważności defektu i należy wówczas dostarczyć dostępne i obiektywne informacje z nim związane.

Narzędzie do śledzenia defektów nie powinno być używane jako substytut dobrej komunikacji, podobnie jak spotkania komitetu zarządzania usterkami nie powinny być substytutem efektywnego wykorzystania dobrego narzędzia do śledzenia defektów. Komunikacja, odpowiednie wsparcie narzędziowe, dobrze zdefiniowany cykl życia defektu oraz zaangażowany komitet zarządzający defektami to elementy konieczne do efektywnego i skutecznego zarządzania defektami.

4.3. Atrybuty defektów

Gdy wykrywany jest defekt (w ramach testów statycznych) lub zostaje wykryta awaria (podczas testów dynamicznych), osoba uczestnicząca powinna zebrać dane i ująć je w raporcie o defekcie. Informacja ta powinna być wystarczająca dla trzech celów:

- zarządzanie raportem podczas cyklu życia defektu,
- ocena statusu projektu, w szczególności w zakresie jakości produktu, oraz ocena postępu testów,
- ocena zdolności procesu (jak opisano poniżej w punkcie 4.4.).

Dane potrzebne do zarządzania raportami o defektach oraz statusem projektu mogą się różnić w zależności od tego, w którym momencie w cyklu życia defekt został wykryty. We wczesnych fazach zazwyczaj potrzebna jest mniejsza ilość informacji (np. przeglądy wymagań i testy modułowe). Jednak zebrane, kluczowe informacje, powinny być spójne w całym cyklu życia wytwarzania oprogramowania, a najlepiej we wszystkich projektach, aby umożliwić miarodajne porównanie informacji o defektach w całym projekcie i pomiędzy wszystkimi projektami.

Zbieranie danych dotyczących defektów może pomóc w monitorowaniu postępu testów, nadzorze nad testami i ocenie kryteriów zakończenia. Na przykład atrybuty defektu powinny wspierać analizę gęstości defektów, analizę trendu wykrywania i usuwania/naprawiania defektów, średni czas od wykrycia defektu

do jego naprawy oraz rozmiar i intensywność awarii (np. analiza średniego czasu pomiędzy awariami – ang. *Mean Time Between Failures*, MTBF).

Można zbierać następujące atrybuty defektów:

- imię i nazwisko osoby, która wykryła defekt,
- rola osoby (np. użytkownik, analityk biznesowy, programista, pracownik wsparcia technicznego),
- typ wykonywanych testów (np. testowanie użyteczności, testowanie wydajności, testowanie regresji),
- podsumowanie informacji o defekcie,
- szczegółowy opis defektu,
- kroki do odtworzenia awarii (dla defektu), wraz z rzeczywistymi i oczekiwanymi rezultatami (określającymi anomalię), w tym zrzuty ekranu, zrzuty baz danych i dzienniki, jeśli ma to uzasadnienie,
- faza cyklu życia wprowadzenia, wykrycia i usunięcia defektu, w tym poziom testów, gdy ma to zastosowanie,
- produkt pracy, w którym defekt został wprowadzony,
- krytyczność wpływu na system i/lub interesariuszy produktu (zazwyczaj określany przez zachowanie techniczne systemu),
- priorytet usunięcia defektu (zwykle określony przez wpływ biznesowy awarii),
- podsystem lub moduł, w którym znajduje się defekt (dla analizy gromadzenia się defektów),
- czynność projektowa występująca w momencie wykrycia defektu,
- metoda identyfikacji, która ujawniła defekt (np. przegląd, analiza statyczna, testy dynamiczne, użycie produkcyjne),
- typ defektu (zwykle odpowiada taksonomii defektów, jeśli jest używana),
- charakterystyka jakościowa dotknięta defektem,
- środowisko testowe, w którym zauważono defekt (dla testów dynamicznych),
- projekt i produkt, w których występuje defekt,
- obecny właściciel, czyli osoba aktualnie przydzielona do pracy nad defektem, zakładając, że raport o defekcie nie jest ukończony,
- obecny status raportu o defekcie (zazwyczaj zarządzany przez narzędzie do śledzenia defektów w ramach cyklu życia defektu),
- konkretne produkty pracy (np. elementy testowe i ich numery wydań), w których zaobserwowano defekt, wraz z poszczególnymi produktami pracy, w których defekt został ostatecznie rozwiązany,
- wpływ na projekt i korzyści interesariuszy produktu,
- wnioski, rekomendacje i akceptacje dla działań podjętych lub niepodjętych w celu rozwiązania defektu,
- ryzyko, koszty, możliwości i korzyści związane z naprawą lub brakiem naprawy defektu,
- daty, w których wystąpiły różne przejścia w cyklu życia defektu, właściciele raportu opartego na każdym przejściu oraz działania podejmowane przez członków zespołu projektowego w celu odizolowania, naprawy i sprawdzenia poprawki defektu,
- opis sposobu, w jaki defekt został ostatecznie rozwiązany oraz zalecenia dotyczące testowania poprawki (jeśli defekt został naprawiony przez zmianę w oprogramowaniu),
- inne odniesienia, takie jak test, który ujawnił defekt oraz ryzyko, wymagania lub inne elementy podstawy testów związane z wadą (dla testów dynamicznych).

Istnieją różne standardy i dokumenty, takie jak ISO 9126 [ISO9126]¹², IEEE 829 [IEEE829]¹³, IEEE 1044 [IEEE1044] i ortogonalna klasyfikacja defektów, które pomagają Kierownikowi Testów określić, jakie informacje należy zbierać w procesie raportowania defektów.

Bez względu na typ informacji określonej jako niezbędna dla raportów o defektach, ważne jest to, by testerzy wprowadzali informacje, które są pełne, zwarte, dokładne, obiektywne, istotne i aktualne. Nawet wtedy, gdy interwencja ręczna i bezpośrednia komunikacja przewyższają problemy z danymi w raportach o defektach w kontekście rozwiązania poszczególnych defektów, problemy z danymi w raportach o defektach mogą przeszkadzać we właściwej ocenie stanu projektu, postępu testów oraz zdolności procesu.

4.4. Ocena zdolności procesu testowego przy użyciu atrybutów defektu

Jak opisano w Rozdziale 2., raporty o defektach mogą być przydatne do monitorowania stanu projektu i raportowania. Choć konsekwencje wykorzystania metryk dla procesu omówione są przede wszystkim w Sylabusie dla poziomu eksperckiego Doskonalenie procesu testowego (Wdrażanie doskonalenia i zmian) [ISTQB ETM SYL] na poziomie zaawansowanym, Kierownicy Testów powinni być świadomi tego, jaką rolę pełnią raporty o defektach w ocenie wydajności procesów testowania i rozwoju oprogramowania.

Oprócz dostarczania informacji potrzebnych do monitorowania postępu testów, o których mowa w Rozdziale 2. oraz w podrozdziale 4.3., atrybuty defektu powinny wspierać inicjatywy doskonalenia procesów. Przykłady obejmują:

- korzystanie z informacji o fazie wprowadzenia, wykrycia i usunięcia w trybie faza-po-fazie, celem oceny powstrzymania fazowego oraz proponowania sposobów zwiększenia skuteczności wykrywania defektów w każdej fazie,
- korzystanie z informacji w fazie wprowadzenia wykonania do analizy Pareto dla faz, w których została wprowadzona największa liczba defektów, aby umożliwić ukierunkowane doskonalenie w celu zmniejszenia całkowitej liczby defektów,
- korzystanie z informacji o przyczynie podstawowej defektu w celu określenia zasadniczego powodu powstania defektu, aby umożliwić poprawę procesów, które zmniejszą liczbę defektów,
- korzystanie z informacji o fazie wprowadzenia, wykrycia i usunięcia defektu do wykonania analizy kosztów jakości, w celu zminimalizowania kosztów związanych z defektami,
- korzystanie z informacji o komponentach związanych z defektami w celu wykonania analizy skupisk defektów, aby lepiej zrozumieć ryzyko techniczne (dla testowania opartego na ryzyku) i umożliwić ponowne zaprojektowanie kłopotliwych komponentów.

Zastosowanie wskaźników do oceny efektywności i skuteczności procesu testowego omówiono w Sylabusie dla poziomu eksperckiego Doskonalenie procesu testowego (Wdrażanie doskonalenia i zmian) [ISTQB ETM SYL].

W niektórych przypadkach zespoły mogą podjąć decyzję, że defekty wykryte podczas niektórych lub wszystkich czynności cyklu życia wytwarzania oprogramowania nie będą śledzone. Chociaż jest to wykonywane często w imię efektywności i ze względu na zmniejszenie narzutu na proces testowy, w rzeczywistości w znacznym stopniu zmniejsza to wgląd w możliwości procesu testowego i rozwoju oprogramowania. To sprawia, że zasugerowane powyżej ulepszenia są trudne do przeprowadzenia ze względu na brak wiarygodnych danych.

¹² Standard ISO/IEC 25010 zastąpił normę ISO 9126.

¹³ Standard ISO/IEC/IEEE 29119 zastąpił normę IEEE 829.

5. Doskonalenie procesu testowego – 135 min.

Słowa kluczowe

zintegrowany model dojrzałości organizacyjnej (CMMI), krytyczne procesy testowania (CTP), proces systematycznego testowania i oceny (STEP), zintegrowany model dojrzałości testów (TMMi), TPI Next

Doskonalenie procesu testowego – cele nauczania

5.2. Proces doskonalenia testowania

TM-5.2.1. (K2) Kandydat potrafi wyjaśnić, podając przykłady, dlaczego ważne jest doskonalenie procesu testowego.

5.3. Doskonalenie procesu testowego

TM-5.3.1. (K3) Kandydat potrafi zdefiniować plan doskonalenia procesu testowego przy użyciu modelu IDEAL.

5.4. Doskonalenie procesu testowego z użyciem modelu TMMi

TM-5.4.1. (K2) Kandydat potrafi podsumować założenia, zakres i cele modelu doskonalenia procesu testowego TMMi.

5.5. Doskonalenie procesu testowego z użyciem TPI Next

TM-5.5.1. (K2) Kandydat potrafi podsumować założenia, zakres i cele modelu doskonalenia procesu testowego TPI Next.

5.6. Doskonalenie procesu testowego z użyciem CTP

TM-5.6.1. (K2) Kandydat potrafi podsumować założenia, zakres i cele modelu doskonalenia procesu testowego CTP.

5.7. Doskonalenie procesu testowego z użyciem STEP

TM-5.7.1. (K2) Kandydat potrafi podsumować założenia, zakres i cele modelu doskonalenia procesu testowego STEP.

5.1. Wprowadzenie

Ustanowiony, obejmujący całą organizację, proces testowy powinien być poddawany ciągłym usprawnieniom. Na początku tego rozdziału omówione zostaną ogólne zagadnienia doskonalenia procesu, a następnie wprowadzone zostaną pewne określone modele doskonalenia procesu testowego. Kierownik Testów powinien zakładać, że będzie siłą napędową zmian i doskonalenia procesu. Powinien być zaznajomiony z uznanymi w branży technikami omawianymi w tym rozdziale. Dalsze informacje o usprawnianiu procesu testowego opisane są w Sylabusie dla poziomu eksperckiego Doskonalenie procesu testowego (Wdrażanie doskonalenia i zmian).

5.2. Proces doskonalenie testowania

Tak jak organizacja wykorzystuje proces testowy do doskonalenia oprogramowania, tak techniki doskonalenia procesów są dobierane i używane do usprawnienia procesów wytwarzania oprogramowania i wynikających z nich wydań oprogramowania. Doskonalenie procesów może również dotyczyć procesów testowania. Dostępne są różne metody i podejścia, by poprawić testowanie systemów z wbudowanym oprogramowaniem. Metody te mają na celu usprawnianie procesów wytwarzania oprogramowania, a przez to i produktów poprzez dostarczanie wskazówek i obszarów do doskonalenia.

Koszt testowania często stanowi dużą część całkowitych kosztów projektu. Niemniej jednak w różnych modelach doskonalenia procesu wytwarzania oprogramowania, takich jak CMMI® (szczegółowy opis poniżej) tylko w ograniczonym zakresie przywiązuje się wagę do procesu testowego.

Modele doskonalenia testowania takie jak zintegrowany model dojrzałości testów (TMMi®), proces systematycznego testowania i oceny (STEP), krytyczne procesy testowania (CTP) i TPI Next® zostały opracowane, aby rozwiązać ogólny problem niezwracania uwagi na testowanie w większości modeli doskonalenia oprogramowania. Poprawnie użyte modele mogą dostarczyć metryk obejmujących całość organizacji, które mogą zostać później wykorzystane jako kryteria do porównań (benchmarki).

Prezentacja modeli w tym sylabusie nie ma na celu zalecenia ich wykorzystania. Zostają one przedstawione jedynie jako reprezentatywny wgląd w to, jak modele te działają i co zawierają.

5.2.1. Wprowadzenie do doskonalenie procesu

Doskonalenie jest istotne zarówno w procesie wytwarzania oprogramowania, jak i w procesie testowania. Uczucie się na własnych błędach umożliwia doskonalenie procesu, który jest stosowany w organizacji do wytwarzania i testowania oprogramowania. Cykl ciągłego doskonalenia Deminga: zaplanuj, wykonaj, sprawdź, zastosuj, jest używany od wielu dekad i jest ciągle aktualny, gdy tester potrzebuje udoskonalić proces, który jest aktualnie stosowany.

Przesłanką do poprawy procesu jest przekonanie o tym, że jakość systemu w znacznym stopniu zależy, od jakości procesu, zgodnie z którym odbywa się wytwarzanie oprogramowania. Wyższa jakość w branży wytwarzania oprogramowania redukuje potrzebę wykorzystania dodatkowych zasobów do utrzymania oprogramowania i dzięki temu zapewnia więcej czasu na stworzenie większej ilości lepszych rozwiązań w przyszłości. Modele procesowe stanowią punkt startowy do doskonalenia poprzez pomiar możliwości procesu w organizacji względem modelu. Modele dostarczają również struktury do udoskonalania procesu w organizacji bazując na wyniku oceny.

Ocena procesu prowadzi do określenia jego zdolności, co z kolei motywuje do jego udoskonalenia. Może to pociągać za sobą kolejną ocenę procesu stanowiącą pomiar efektów usprawnień.

5.2.2. Typy doskonalen procesu

Użycie modelu oceny jest popularną metodą zapewniającą zestandaryzowane podejście do usprawniania procesu testowania przy użyciu wypróbowanych i zaufanych praktyk.

Modele doskonalące procesy dzieli się na dwie kategorie:

1. Model referencyjny procesu pozwala na pomiar dojrzałości jako część oceny zdolności organizacji w porównaniu z modelem, ocenę organizacji w ramach struktury oraz dostarcza mapę drogową dla doskonalenia procesu.
2. Model referencyjny zawartości pozwala na ocenę, z perspektywy biznesowej, okazji organizacji dla doskonalenia oraz, w niektórych przypadkach, pozwala również na porównanie ze średnimi branżowymi z wykorzystaniem obiektywnych pomiarów. Ta ocena może być użyta do zbudowania harmonogramu udoskonalania procesu.

Doskonalenie procesu testowego może być również zrealizowane bez wykorzystania modelu, na przykład przez zastosowanie podejścia analitycznego i spotkań retrospektywnych.

5.3. Doskonalenie procesu testowego

Branża informatyczna może używać modeli doskonalenia procesu testowego dla osiągnięcia wyższego poziomu dojrzałości i profesjonalizmu. Modele standardów branżowych pomagają opracować metryki i miary obejmujące całe organizacje, które mogą zostać wykorzystane do porównań. Na potrzeby udoskonalania procesów w branży testowania powstało kilkanaście zestawów rekomendowanych procesów. W tym zestawie znajdują się również STEP, TMMi, TPI Next oraz CTP. Modele wielostopniowe, takie jak TMMi i CMMI, zapewniają normy pozwalające na porównanie pomiędzy różnymi firmami i organizacjami. Modele ciągłe, takie jak CTP, STEP i TPI Next, umożliwiają organizacji zwrócenie uwagi na zagadnienia o najwyższym prioryecie z większą dowolnością w definiowaniu kolejności wdrożenia. Są one omówione w niniejszym rozdziale.

Wszystkie spośród zaprezentowanych modeli umożliwiają organizacji określenie, gdzie znajdują się obecnie procesy testowe. Gdy dokona się oceny zgodnie z TMMi i TPI Next, to powinna zostać wprowadzona, zgodna z tymi modelami, mapa drogową doskonalenia procesu testowego. Alternatywne modele STEP i CTP dostarczają organizacji metody określenia, gdzie doskonalenie procesu w największym stopniu ma szansę na zwrot z inwestycji i pozostawia organizacji wybór właściwego harmonogramu.

Kiedy zostaje podjęta decyzja o przejrzeniu i udoskonaleniu procesu testowego, to w konsekwencji należy zaadaptować kroki doskonalenia procesu, jak opisano w modelu IDEALSM [IDEAL96]:

- rozpoczęcie procesu doskonalenia (Initiating),
- diagnozowanie obecnej sytuacji (Diagnosing),
- ustanawianie planu doskonalenia procesu testowego (Establishing),
- działanie w celu wdrożenia udoskonalen (Acting),
- uczenie się (Learning).

Rozpoczęcie procesu doskonalenia

Przed rozpoczęciem czynności doskonalenia procesu, cele, zakres i pokrycie usprawnianych procesów muszą zostać ustalone przez interesariuszy. Również wybór modelu doskonalenia procesu jest dokonywany w tym momencie. Model może być wybrany spośród publicznie dostępnych opcji (takich jak CTP, STEP, TMMi czy TPI Next) lub wytworzony wewnętrznie. Dodatkowo należy zdefiniować kryteria sukcesu wraz z metodą pomiarową w czasie trwania czynności doskonalenia.

Diagnozowanie obecnej sytuacji

Zgodnie z uzgodnionym podejściem dokonuje się oceny i tworzony jest raport oceny procesu testowego. Raport ten zawiera ocenę aktualnych praktyk testowania i listę potrzebnych udoskonaleń procesu.

Ustanawianie planu doskonalenia procesu testowego

Do poszczególnych elementów listy możliwych usprawnień przypisywane są priorytety. Powinny one bazować na zwrocie z inwestycji, ryzyku, zgodności ze strategią organizacyjną lub mierzalnymi ilościowymi lub jakościowymi korzyściami. Po ustaleniu priorytetów konstruuje się plan dostarczania usprawnień.

Działanie w celu wdrożenia udoskonaleń

W tym kroku następuje wdrożenie planu doskonalenia procesu testowego dla realizacji udoskonaleń. Plan powinien zawierać każdy wymagany trening lub mentoring, pilotażowe wdrożenie procesów i ostatecznie ich pełne wdrożenie.

Uczenie się

Mając w pełni wdrożone usprawnienia procesu testowego, kluczowe jest zweryfikowanie, które korzyści (z tych, które ustalono wcześniej, oprócz potencjalnie nieoczekiwanych) zostały osiągnięte. Istotne jest, by sprawdzić również, które z kryteriów sukcesu dla czynności usprawnienia procesu zostały osiągnięte.

W zależności od zastosowanego modelu doskonalenia procesu, na tym etapie procesu doskonalenia rozpoczynamy monitorowanie następnego poziomu dojrzałości oraz podejmujemy decyzję o rozpoczęciu procesu doskonalenia kolejny raz lub decyzję o przerwaniu czynności na tym etapie, pozostając w obecnym punkcie.

5.4. Doskonalenie procesu testowego z użyciem modelu TMMi

Zintegrowany model dojrzałości testów (TMMi) składa się z pięciu poziomów dojrzałości i jest pomyślany jako uzupełnienie modelu CMMI. Każdy poziom dojrzałości zawiera zdefiniowane obszary procesowe, które muszą być w 85% określone jako kompletne poprzez osiągnięcie szczegółowych i ogólnych celów, zanim organizacja może przejść na następny poziom dojrzałości testów.

Poziomy dojrzałości w TMMi to:

- Poziom 1: Wstępny (Initial)
Poziom początkowy reprezentuje stan, gdzie nie ma formalnie udokumentowanego lub zdefiniowanego procesu testowego. Testy są tworzone *ad hoc* po kodowaniu, a testowanie jest postrzegane jako równoważne z debugowaniem. Cel testów jest rozumiany jako udowodnienie, że oprogramowanie działa.
- Poziom 2: Zarządzany (Managed)
Drugi poziom zostaje osiągnięty, kiedy proces testowy jest jasno oddzielony od debugowania. Może być to osiągnięte poprzez ustanowienie polityki testów oraz celów testowania, wprowadzenie kroków z podstawowego procesu testowego (np. planowanie testów) i implementacji podstawowych technik i metod.
- Poziom 3: Zdefiniowany (Defined)
Poziom trzeci zostaje osiągnięty, gdy proces testowy zostaje zintegrowany z cyklem życia wytwarzania oprogramowania oraz gdy jest udokumentowany w postaci formalnych standardów, procedur i metodyk. Odbywają się przeglądy oraz powinna istnieć wydzielona rola w obszarze testowania oprogramowania, która może być kontrolowana i monitorowana.
- Poziom 4: Mierzony (Measured)

Poziom czwarty zostaje osiągnięty, gdy testowanie może zostać skutecznie zmierzone i zarządzane na poziomie organizacji w celu osiągnięcia korzyści w konkretnych projektach.

- Poziom 5: Zoptymalizowany (Optimized)
Ostatni poziom reprezentuje stan dojrzałości procesu testowego, w którym dane z testowania mogą być użyte, aby zapobiegać defektom i kiedy organizacja skupia się na optymalizacji wdrożonego procesu.

Więcej informacji o TMMi szukaj w [vanVeenendaal11] oraz [www.tmmi.org].

5.5. Doskonalenie procesu testowego z użyciem TPI Next

Model TPI Next definiuje szesnaście kluczowych obszarów, z których każdy uwzględnia szczegółowe aspekty procesu testowego, takie jak: strategia testów, metryki, narzędzia testowe i środowisko testowe.

W modelu zdefiniowane są cztery poziomy dojrzałości:

- Wstępny,
- Kontrolowany,
- Efektywny,
- Optymalizowany.

Dla oceny każdego z kluczowych obszarów, na każdym poziomie dojrzałości, zdefiniowane są szczegółowe punkty kontrolne. Wyniki audytu są zestawiane i wizualizowane za pomocą matrycy dojrzałości pokrywającej wszystkie obszary kluczowe. Definicja celów usprawniania i ich implementacji może zostać dopasowana do potrzeb i zdolności organizacji testowej.

Ogólne podejście przyjęte w tej metodzie sprawia, że TPI Next jest metodą niezależną od dowolnego modelu usprawniania procesu wytwarzania oprogramowania. Pokrywa ona zarówno aspekty inżynierii testowania, jak i wsparcie dla podejmowania decyzji kierowniczych [deVries09].

Więcej informacji o TPI Next szukaj na [www.tpinext.com].

5.6. Doskonalenie procesu testowego z użyciem CTP

Podstawowym założeniem modelu oceny krytycznych procesów testowania (CTP) jest to, że pewne procesy testowe są krytyczne. Te krytyczne procesy, jeśli zostaną przeprowadzone dobrze, będą przyczyniać się do sukcesu zespołu testowego. Z drugiej strony, jeśli te aktywności zostaną przeprowadzone są nieumiejętnie, nawet utalentowany tester i Kierownik Testów nie będą w stanie osiągnąć sukcesu. Model identyfikuje dwanaście krytycznych procesów testowych. CTP jest przede wszystkim modelem referencyjnym zawartości.

Model CTP jest podejściem silnie zależnym od kontekstu, który pozwala na dostosowanie modelu włączając w to:

- identyfikację konkretnych wyzwań,
- rozpoznanie atrybutów prawidłowych procesów,
- uzgodnienie kolejności i priorytetów wdrażania udoskonaleń procesu.

Model CTP można dostosować do wszystkich modeli cyklu życia oprogramowania.

Oprócz wywiadów z uczestnikami, model CTP obejmuje użycie metryk w celu analizy porównawczej organizacji względem średnich branżowych i najlepszych praktyk.

Więcej informacji o CTP szukaj w [Black03].

5.7. Doskonalenie procesu testowego z użyciem STEP

STEP - proces systematycznego testowania i oceny, podobnie jak CTP i inaczej niż TMMi oraz TPI Next, nie wymaga, aby usprawnienia występowały w określonej kolejności.

STEP jest przede wszystkim modelem referencyjnym zawartości, bazującym na idei, że testowanie jest aktywnością cyklu życia oprogramowania, rozpoczynającą się podczas formułowania wymagań i trwającą aż do wycofania systemu. Metodologia STEP podkreśla metodę "najpierw testuj, potem koduj" poprzez użycie strategii testów opartej na wymaganiach w celu zapewnienia, że wczesne zaprojektowanie przypadków testowych umożliwia walidację specyfikacji wymagań przed projektowaniem i kodowaniem.

Podstawowe założenia tej metodyki to:

- strategia testów jest oparta na wymaganiach,
- testowanie rozpoczyna się już na początku cyklu życia oprogramowania,
- testy wykorzystuje się jako modele wymagań i użycia,
- projektowanie testaliów wyprzedza projektowanie oprogramowania,
- defekty są wykrywane wcześniej lub udaje się im całkowicie zapobiec,
- defekty są systematycznie analizowane,
- testerzy i programiści pracują wspólnie.

W niektórych przypadkach model oceny STEP jest łączony z modelem dojrzałości TPI Next.

Więcej informacji o STEP szukaj w [Craig02].

6. Narzędzia testowe i automatyzacja – 135 min

Słowa kluczowe

narzędzie o otwartym kodzie, własne narzędzie

Narzędzia testowe i automatyzacja – cele nauczania

6.2. Wybór narzędzi

- TM-6.2.1. (K2) Kandydat potrafi opisać problemy dotyczące zarządzania występujące podczas wybierania narzędzia o otwartym kodzie.
- TM-6.2.2. (K2) Kandydat potrafi opisać problemy dotyczące zarządzania występujące podczas podejmowania decyzji o korzystaniu z własnego narzędzia.
- TM-6.2.3. (K4) Kandydat potrafi ocenić przedstawioną sytuację w projekcie w celu opracowania planu wyboru narzędzia z uwzględnieniem ryzyk, kosztów i korzyści.

6.3. Cykl życia narzędzia

- TM-6.3.1. (K2) Kandydat potrafi objaśnić różne etapy cyklu życia narzędzia.

6.4. Metryki związane z narzędziami

- TM-6.4.1. (K2) Kandydat potrafi opisać, w jaki sposób zbieranie i ocena pomiarów może zostać ulepszone dzięki użyciu narzędzi.

6.1. Wprowadzenie

Niniejszy rozdział poszerza treści podane w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB®, podając kilka ogólnych koncepcji, które Kierownik Testów musi uwzględnić przy planowaniu zastosowania narzędzi i automatyzacji.

6.2. Wybór narzędzi

Podczas wybierania narzędzi testowych Kierownik Testów musi wziąć pod uwagę wiele różnych zagadnień.

Historycznie rzecz ujmując, najczęstszym wyborem jest zakup narzędzia od dostawcy komercyjnego. W niektórych przypadkach może to być jedyna opcja. Istnieją też inne możliwości, takie jak narzędzia o otwartym kodzie lub narzędzia własne, które również mogą być dostępne.

Niezależnie od rodzaju narzędzia Kierownik Testów musi uważnie zbadać całkowity koszt posiadania narzędzia przez cały przewidywany okres jego wykorzystania poprzez wykonanie analizy kosztów i korzyści. Temat ten jest omówiony poniżej w podrozdziale "Zwrot z inwestycji".

6.2.1. Narzędzia o otwartym kodzie

Narzędzia o otwartym kodzie są dostępne dla prawie każdego aspektu procesu testowego, od zarządzania przypadkami testowymi, poprzez śledzenie błędów, do automatyzacji przypadków testowych. Ważnym wyróżnikiem narzędzi o otwartym kodzie jest to, że chociaż narzędzia te nie generują wysokich kosztów zakupu, to jednak mogą nie posiadać formalnego wsparcia. Niemniej jednak wiele narzędzi o otwartym kodzie posiada oddanych zwolenników, którzy są gotowi wspierać użytkowników nieformalnie.

W dodatku wiele narzędzi o otwartym kodzie na początku zostało stworzonych dla rozwiązania konkretnego problemu i zaadresowania konkretnej kwestii, może więc nie posiadać funkcji dostępnych w podobnych narzędziach komercyjnych. Z tego powodu, przed podjęciem decyzji o wyborze narzędzia otwartego, należy przeprowadzić uważną analizę faktycznych potrzeb zespołu testowego.

Jedną z korzyści wynikających ze stosowania narzędzi o otwartym kodzie jest to, że zwykle można je zmodyfikować lub rozbudować. Jeżeli organizacja testowa posiada odpowiednie kompetencje, to może zmodyfikować narzędzie tak, żeby współpracowało z innymi narzędziami lub zmienić je zgodnie ze swoimi potrzebami. Można połączyć kilka narzędzi, aby rozwiązać problemy, których rozwiązaniu narzędzia komercyjne nie są w stanie sprostać. Oczywiście im więcej narzędzi zostanie wykorzystanych i im więcej modyfikacji jest zrobionych, tym większa złożoność i wyższe koszty ogólne. Kierownik Testów musi upewnić się, że zespół nie zaczyna używać narzędzi otwartych tylko dla nich samych. Tak jak w przypadku wszystkich innych narzędzi za cel należy przyjąć dodatni zwrot z inwestycji.

Kierownik Testów musi rozumieć schemat licencjonowania wybranego narzędzia. Wiele narzędzi posiada jeden z wariantów GNU (*General Public License*), który wymaga, aby dystrybucja oprogramowania odbywała się na tych samych zasadach, na jakich oprogramowanie zostało pozyskane. Jeżeli oprogramowanie zostało dostosowane do potrzeb zespołu testowego, to może zaistnieć konieczność udostępnienia wykonanych zmian pozostałym użytkownikom tego narzędzia na tych samych zasadach licencyjnych. Kierownik Testów powinien sprawdzić skutki prawne dla swojej organizacji w przypadku redystrybucji wybranego narzędzia.

Organizacje, które rozwijają oprogramowanie krytyczne ze względów bezpieczeństwa lub misji, albo takie, które musi być zgodne z przepisami prawa, mogą mieć problemy z wykorzystaniem

oprogramowania otwartego. Chociaż wiele narzędzi o otwartym kodzie jest bardzo wysokiej jakości, to dokładność działania tych narzędzi nie została potwierdzona certyfikatem. Narzędzia komercyjne często posiadają certyfikaty dotyczące ich dokładności i dopasowania do konkretnego zadania (np. DO-178B)¹⁴. Nawet jeżeli narzędzie o otwartym kodzie jest tak samo dobre jak narzędzie komercyjne, to zespół wykorzystujący to narzędzie może być zobowiązany do jego certyfikacji, co stanowi dodatkowy koszt.

6.2.2. Narzędzia własne

Organizacja testowa może mieć specyficzne potrzeby, których nie spełnia żadne narzędzie komercyjne ani otwarte. Powodem tego może być posiadanie własnej platformy sprzętowej, dostosowanego do własnych potrzeb środowiska lub proces, który został zmodyfikowany w niespotykany gdzie indziej sposób. W takich przypadkach, o ile zespół posiada odpowiednie kompetencje, Kierownik Testów może chcieć rozważyć rozwój własnego narzędzia.

Do korzyści z rozwoju własnego narzędzia można zaliczyć to, że narzędzie jest dokładnie dopasowane do potrzeb zespołu i może efektywnie pracować w wymaganym przez zespół kontekście. Narzędzie może być stworzone tak, żeby współpracowało z innymi używanymi narzędziami i żeby generowało dane dokładnie w takiej formie, jaka jest potrzebna. Ponadto narzędzie może mieć zastosowanie w organizacji poza projektem, w którym zostało stworzone. Niemniej jednak przed planowaniem przekazania narzędzia innemu projektowi, ważne jest przemyślenie celu istnienia narzędzia, korzyści i możliwych wad.

Kierownik Testów rozważający stworzenie własnych narzędzi musi również rozważyć możliwe negatywne aspekty tego przedsięwzięcia. Narzędzia własne są często zależne od ich twórców. Dlatego narzędzia te muszą być odpowiednio udokumentowane, tak, aby możliwe było ich utrzymanie przez inne osoby. Jeżeli się tego nie robi, mogą zostać "osierocone" i wypaść z użycia, kiedy ich twórca odejdzie z projektu. Zastosowanie narzędzia może z czasem zostać rozszerzone poza początkowy cel, co może spowodować problemy z jakością narzędzia prowadząc do fałszywie pozytywnych raportów o defektach lub tworzenia niedokładnych danych.

6.2.3. Zwrot z inwestycji (ROI – Return On Investment)

Obowiązkiem Kierownika Testów jest zapewnienie, żeby narzędzia wdrażane w organizacji testowej zwiększały wartość pracy zespołu i dawały organizacji pozytywny zwrot z inwestycji. W celu upewnienia się, że narzędzie przynosi rzeczywiste i długotrwałe korzyści, należy przed jego pozyskaniem przeprowadzić analizę kosztów i korzyści. Podczas tej analizy powinny zostać uwzględnione koszty powtarzające i niepowtarzające się. Niektóre z tych kosztów to koszty pieniężne, inne stanowią koszt wykorzystania zasobów lub czasu. Powinny też zostać uwzględnione ryzyka, które mogą obniżyć wartość narzędzia.

Do kosztów niepowtarzających się można zaliczyć następujące koszty :

- definiowanie wymagań na narzędzie, tak, aby spełniało ono zdefiniowane cele,
- ocena i wybór odpowiedniego narzędzia i jego dostawcy,
- nabycie, adaptacja lub stworzenie narzędzia,
- przeprowadzenie wstępnego szkolenia z narzędzia,
- integracja narzędzia z innymi narzędziami,
- zakup oprogramowania i sprzętu potrzebnego do wsparcia narzędzia.

Koszty powtarzające się:

- posiadanie narzędzia,

¹⁴ Standard FAA DO-178B/ED 12B zastąpił normę FAA DO-178C/ED 12C

- opłaty za licencję i wsparcie,
 - koszty utrzymania samego narzędzia,
 - utrzymanie artefaktów wytworzonych przez narzędzie,
 - koszty ciągłego szkolenia i mentoring,
- przenoszenie narzędzia na inne środowiska,
- dostosowanie narzędzia do przyszłych potrzeb,
- doskonalenie jakości narzędzia oraz procesu w celu zapewnienia optymalnego wykorzystania wybranych narzędzi.

Kierownik Testów powinien też wziąć pod uwagę koszty utraconych korzyści obecne przy wdrażaniu każdego narzędzia. Czas spędzony na pozyskaniu, zarządzaniu, szkoleniu i używaniu narzędzia mógłby być poświęcony na wykonywanie zadań testowych. Dlatego na początku, do momentu uruchomienia narzędzia, może być potrzebne więcej zasobów.

Istnieje wiele ryzyk związanych z wykorzystaniem narzędzi, nie wszystkie narzędzia dostarczają korzyści, które przeważają nad ryzykami. Ryzyka związane z narzędziami testowymi zostały przedstawione w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB®.

Kierownik Testów powinien wziąć pod uwagę również następujące ryzyka:

- niedostateczną dojrzałość organizacji (nie jest gotowa do wykorzystania narzędzia),
- artefakty tworzone przez narzędzie mogą być trudne w utrzymaniu, wymagając wielu poprawek, gdy zmienia się testowane oprogramowanie,
- zmniejszenie udziału Analityków Testów w zadaniach testowych może spowodować obniżenie wartości testowania (np. skuteczność wykrywania defektów zmniejsza się, gdy wykonywane są jedynie skrypty automatyczne).

Na koniec, Kierownik Testów powinien przyrzeć się korzyściom wynikającym z używania narzędzia. Korzyści z wdrożenia i używania narzędzi mogą być następujące:

- redukcja powtarzającej się pracy,
- skrócenie cyklu testowego (np. przez wykorzystanie automatycznych testów regresji),
- zmniejszenie kosztów wykonania testów,
- zwiększenie ilości pewnych typów testów (np. testów regresji),
- zmniejszenie liczby błędów popełnianych przez ludzi w różnych fazach testów, na przykład:
 - dane testowe mogą być bardziej poprawne, jeżeli zostaną wygenerowane przez narzędzie,
 - porównywanie wyników testów może być dokładniejsze, jeżeli zostanie wykonane przez odpowiednie narzędzie,
 - wprowadzanie danych testowych może być bardziej poprawne, jeżeli zostanie wykonane przez narzędzia skryptowe,
- zmniejszenie wysiłku potrzebnego do skorzystania z informacji o testach, na przykład:
 - raporty i metryki wygenerowane przez narzędzie,
 - powtórne wykorzystanie zasobów testowych, takich jak przypadki testowe, skrypty testowe i dane testowe,
- zwiększenie liczby testów, których nie można wykonać bez narzędzi (np. testów wydajnościowych i obciążeniowych),
- podniesienie statusu testerów, którzy automatyzują, oraz organizacji testowej jako całości przez pokazanie zrozumienia i umiejętności wykorzystania wyszukanych narzędzi.

Ogólnie rzecz ujmując, zespół testowy rzadko używa tylko jednego narzędzia. Całkowite ROI, jakie osiąga zespół testowy, jest zwykle funkcją wszystkich narzędzi, jakich używa. Narzędzia powinny

wymieniać między sobą informacje i współpracować ze sobą. Dobrym pomysłem jest posiadanie długoterminowej, szeroko zakrojonej strategii dotyczącej narzędzi testowych.

6.2.4. Proces wyboru narzędzia

Narzędzie testowe stanowi inwestycję długoterminową, bardzo często rozciągającą się na wiele iteracji w pojedynczym projekcie lub nawet mającą zastosowanie w wielu projektach. Kierownik Testów powinien przyrzeć się narzędziu z kilku różnych perspektyw.

- Z biznesowego punktu widzenia wymagany jest dodatni zwrot z inwestycji. W celu uzyskania wysokich zwrotów z inwestycji organizacja powinna upewnić się, że te narzędzia (zarówno testowe, jak i niezwiązane bezpośrednio z testowaniem), które muszą ze sobą współdziałać, faktycznie współdziałają. W niektórych przypadkach, żeby zapewnić współdziałanie, należy usprawnić procesy i łączność między narzędziami, co może być czasochłonne.
- Z projektowego punktu widzenia narzędzie musi być skuteczne (np. pomagać w unikaniu pomyłek przy manualnym wykonywaniu testów, takich jak literówki przy wprowadzaniu danych). Może upłynąć sporo czasu, zanim narzędzie zacznie wypracowywać dodatni zwrot z inwestycji. W wielu przypadkach dodatni ROI uzyskuje się w drugim wydaniu lub w fazie utrzymania, a nie podczas pierwszego projektu, kiedy automatyzacja zostaje zaimplementowana. Kierownik Testów powinien brać pod uwagę cały cykl życia aplikacji.
- Z punktu widzenia użytkownika narzędzia, musi ono wspierać członków zespołu projektowego w wykonywaniu zadań w sposób bardziej skuteczny i efektywny. Należy brać pod uwagę krzywą uczenia się, tak, aby użytkownicy byli w stanie opanować zasady posługiwania się narzędziem szybko i przy minimalnym stresie. Przy pierwszym wdrożeniu narzędzia testowe wymagają przeprowadzenia szkoleń i wsparcia w formie mentoringu dla użytkowników.

Aby zapewnić, że uwzględnione zostały wszystkie punkty widzenia, należy stworzyć mapę drogową wdrażania narzędzia testowego.

Proces wyboru narzędzi testowych został już przedstawiony w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB® w następujący sposób¹⁵:

- dokonanie oceny dojrzałości organizacji,
- zidentyfikowanie wymagania na narzędzia,
- dokonanie oceny narzędzia,
- dokonanie oceny dostawcy lub dostępnego wsparcia (dla narzędzi testowych lub pisanych na zamówienie),
- zidentyfikowanie wewnętrznych wymagań dotyczących coachingu i mentoringu w zakresie korzystania z narzędzia,
- dokonanie oceny potrzeb szkoleniowych z uwzględnieniem umiejętności zespołu testowego w zakresie automatyzacji testów,
- oszacowanie stosunku kosztów do korzyści (jak przedstawione w podrozdziale 6.2.3.).

Dla każdego typu narzędzia, niezależnie od tego, w jakiej fazie testów ma być ono używane, Kierownik Testów powinien określić możliwości narzędzia według poniższej listy:

- Analiza:
 - Czy narzędzie „zrozumie” podane dane wejściowe?
 - Czy narzędzie nadaje się do tego, do czego chcemy go użyć?
- Projektowanie:
 - Czy narzędzie pomoże projektować testalia na podstawie istniejących informacji (np. narzędzie do projektowania przypadków testowych utworzy przypadki testowe z wymagań)?

¹⁵ Więcej szczegółów w nowym sylabusie Certyfikowany tester poziom podstawowy 2018

- Czy projektowanie będzie automatyczne?
- Czy kod testaliów zostanie wygenerowany w całości lub częściowo w formacie używalnym i utrzymywalnym?
- Czy potrzebne dane mogą zostać wygenerowane automatycznie (dane generowane z analizy kodu)?
- Dane i wybór testów:
 - Jak narzędzie wybiera potrzebne mu dane (np. który przypadek testowy wykonać z którym zestawem danych)?
 - Czy narzędzie przyjmuje kryteria wyboru podane automatycznie albo ręcznie?
 - Czy narzędzie jest w stanie określić, jak „wyczyścić” dane produkcyjne bazując na wybranych danych wejściowych?
 - Czy narzędzie jest w stanie określić, które testy są potrzebne bazując na kryteriach pokrycia (np. mając podany zbiór wymagań prześledzić powiązania, żeby określić, które testy należy wykonać)?
- Wykonanie:
 - Czy narzędzie wykonuje testy automatycznie, czy wymagana jest interwencja testera?
 - Jak narzędzie zatrzymuje się i restartuje?
 - Czy narzędzie powinno być w stanie „odbierać” istotne zdarzenia (np. czy narzędzie do zarządzania testami powinno automatycznie zmieniać status przypadku testowego, gdy zostanie zamknięty raport o defekcie zgłoszony z powodu tego przypadku)?
- Ocena:
 - Jak narzędzie oceni, czy dostało poprawny wynik (np. czy narzędzie wykorzysta wyrocznie testową do ustalenia, czy odpowiedź systemu jest prawidłowa)?
 - Jakie możliwości reakcji na błąd posiada narzędzie?
 - Czy narzędzie ma odpowiednie możliwości logowania i raportowania?

6.3. Cykl życia narzędzia

Istnieją cztery różne etapy w cyklu życia użytecznego narzędzia, którymi Kierownik Testów musi zarządzać:

1. Pozyskanie:

Narzędzie musi zostać pozyskane tak, jak zostało to opisane powyżej (patrz podrozdział 6.2. Wybór narzędzi). Po dokonaniu wyboru narzędzia Kierownik Testów powinien wyznaczyć innego pracownika (zwykle Analityka Testów lub Technicznego Analityka Testów) i powierzyć jemu obowiązki administratora narzędzia. Ta osoba będzie podejmowała decyzje, kiedy i jak narzędzie ma zostać użyte, gdzie mają być składowane artefakty wytworzone przez narzędzie, jakie mają być konwencje nazewnictwa itd. Podjęcie tych decyzji na samym początku i brak zgody na to, aby były one podjęte *ad hoc*, może wpłynąć na znaczącą różnicę w zwrocie z inwestycji. Najprawdopodobniej będzie też potrzebne przeprowadzenie szkolenia dla użytkowników narzędzia.
2. Wsparcie i utrzymanie:

Narzędzie wymaga ciągłego procesu wsparcia i utrzymania. Odpowiedzialność za utrzymanie narzędzia może spoczywać na jego administratorze lub na dedykowanej grupie narzędziowej. Jeżeli narzędzie ma współpracować z innymi narzędziami, należy rozważyć sposoby wymiany danych i procesy współpracy. Należy też podjąć decyzję co do sposobu wykonywania i odtwarzania kopii zapasowych narzędzia i jego produktów.
3. Ewolucja:

Trzeba wziąć pod uwagę możliwość wprowadzenia zmian w narzędziu. Wraz z upływem czasu środowisko, potrzeby biznesowe lub problemy z dostawcą mogą wymusić duże zmiany w samym narzędziu i sposobie jego wykorzystania. Na przykład dostawca może wymóc

aktualizację narzędzia, która spowoduje problemy we współpracy z innymi narzędziami. Zmiana w środowisku, konieczna z powodów biznesowych, może spowodować problemy z narzędziem. Im bardziej złożone jest środowisko, w którym narzędzie działa, tym bardziej nawet mała zmiana może zakłócić jego pracę. W tym momencie, w zależności od roli, jaką narzędzie pełni w testach, Kierownik Testów może być zmuszony do zapewnienia ciągłości pracy narzędzia.

4. Wycofanie:

Nadejdzie czas, w którym narzędzie przekroczy czas użytecznego działania. Należy je wtedy zezłomować i zastąpić funkcjonalność narzędzia, a dane trzeba zabezpieczyć i zarchiwizować. Może się tak stać, ponieważ narzędzie kończy swój cykl życia albo po prostu dlatego, że osiągnięty został punkt, w którym korzyści i perspektywy wynikające z wymiany narzędzia przewyższają koszty i ryzyka.

W całym cyklu życia narzędzia to Kierownik Testów jest odpowiedzialny za zapewnienie bezproblemowego funkcjonowania i kontynuacji świadczenia usług przez narzędzie zespołowi testowemu.

6.4. Metryki związane z narzędziami

Kierownik Testów może zaprojektować i zbierać metryki dostarczane przez narzędzia używane przez Analityków Testów i Technicznych Analityków Testów. Różne narzędzia mogą zbierać cenne dane w czasie rzeczywistym i zmniejszyć pracochłonność zbierania tych danych. Dane te mogą zostać wykorzystane przez Kierownika Testów w zarządzaniu pracami testerskimi.

Różne narzędzia potrafią zbierać różne typy danych, na przykład:

- Narzędzia do zarządzania testami mogą dostarczyć wiele różnych metryk. Śledzenie powiązań pomiędzy wymaganiami a przypadkami testowymi oraz automatyczne skrypty pozwalają na badanie pokrycia. Można w dowolnym momencie sprawdzić, jakie są aktualnie dostępne testy, testy planowane oraz bieżący stan wykonania testów (zaliczone, niezaliczone, pominięte, zablokowane, w kolejce).
- Narzędzia do zarządzania zgłoszeniami błędów dostarczają wiele informacji na temat defektów: aktualny status, wagę i priorytet, rozkład w systemie itp. Inne potrzebne dane, takie jak fazy, w jakich błąd został popełniony i znaleziony, współczynnik usterek uciekinierów itd. pomagają Kierownikowi Testów w doskonaleniu procesów.
- Narzędzia do analizy statycznej mogą pomóc w wykryciu i zaraportowaniu problemów z utrzymaniem.
- Narzędzia do testów wydajnościowych dostarczają cenne dane na temat skalowalności systemu.
- Narzędzia do mierzenia pokrycia mogą pomóc Kierownikowi Testów w zrozumieniu, jak dokładnie system został przetestowany.

Wymagania dotyczące raportowania powinny zostać zdefiniowane podczas wyboru narzędzia. Wymagania te muszą być odpowiednio zaimplementowane podczas konfiguracji narzędzia, tak żeby zapewnić, że informacje śledzone przez narzędzia mogą być ujęte w raportach w sposób zrozumiały dla interesariuszy.

7. Umiejętności interpersonalne – budowa zespołu – 210 min.

Słowa kluczowe

niezależność testowania

Umiejętności interpersonalne – budowa zespołu – cele nauczania

7.2. Umiejętności

- TM-7.2.1. (K4) Kandydat potrafi używać arkusza oceny umiejętności, analizować mocne i słabe strony członków zespołu testowego dotyczące umiejętności korzystania z oprogramowania, wiedzy dziedzinowej i biznesowej, różnych obszarów rozwoju systemów, testowania oprogramowania oraz umiejętności interpersonalnych.
- TM-7.2.2. (K4) Kandydat potrafi przeanalizować daną ocenę umiejętności zespołu w celu określenia planu rozwoju umiejętności członków zespołu i szkoleń.

7.3. Dynamika zespołu testowego

- TM-7.3.1. (K2) Kandydat potrafi omówić potrzebne w opisanej sytuacji umiejętności twarde i miękkie wymagane do przeprowadzenia zespołowi testowemu.

7.4. Umiejscowienie testów w organizacji

- TM-7.4.1. (K2) Kandydat potrafi objaśnić alternatywy dotyczące niezależnego testowania.

7.5. Motywacja

- TM-7.5.1. (K2) Kandydat potrafi podać przykłady czynników motywujących i demotyujących testerów.

7.6. Komunikacja

- TM-7.6.1. (K2) Kandydat potrafi wyjaśnić czynniki, które mają wpływ na skuteczność komunikacji w zespole projektowym, a także pomiędzy zespołem testowym a jego interesariuszami.

7.1. Wprowadzenie

Skuteczni Kierownicy Testów zatrudniają i utrzymują zespoły posiadające odpowiedni wachlarz różnych umiejętności. Wymagania w zakresie umiejętności posiadane przez zespół mogą się zmieniać w czasie, więc żeby utrzymać zespół testowy i umożliwić mu pracę na najwyższym poziomie wydajności, oprócz zatrudniania odpowiednich osób, ważne jest również ich szkolenie i zapewnianie im możliwości rozwoju. Aby uzupełnić umiejętności zespołu, Kierownik Testów musi również rozwijać własne kompetencje pozwalające jemu na skuteczne funkcjonowanie pod presją, w szybko zmieniającym się otoczeniu.

W rozdziale tym opisano sposoby oceny umiejętności i wypełniania luk kompetencyjnych tak, aby powstawały synergiczne zespoły, które są wewnętrznie spójne i potrafią skutecznie działać w organizacji. W rozdziale jest również mowa o tym, jak motywować zespoły i jak się skutecznie komunikować.

7.2. Umiejętności

Zdolność testera do testowania oprogramowania może być nabyta przez doświadczenie lub szkolenia z różnych obszarów. Każdy z poniżej wymienionych elementów może przyczynić się do rozwoju wiedzy testera:

- używanie oprogramowania,
- wiedza dziedzinowa lub biznesowa,
- wykonywanie czynności w różnych etapach rozwoju oprogramowania np. analizie, kodowaniu, wsparciu technicznym,
- wykonywanie czynności związanych z testowaniem oprogramowania.

Użytkownicy końcowi systemów dobrze wiedzą, jak system powinien działać, kiedy awarie byłyby najgroźniejsze, jak system powinien reagować w różnych sytuacjach. Eksperti dziedzinowi wiedzą, które obszary są najważniejsze dla biznesu i jak one wpływają na możliwość spełnienia wymagań określonych przez biznes. Ta wiedza może wspomagać priorytetyzację zadań testowych, tworzenie realistycznych danych testowych oraz weryfikację i spisywanie przypadków użycia.

Wiedza z zakresu procesu produkcji oprogramowania (analizy wymagań, projektowania, kodowania) daje pogląd na to, jak wprowadzane są błędy do kodu, gdzie można je znaleźć i jak im zapobiegać. Doświadczenie we wsparciu technicznym daje wiedzę na temat wrażeń użytkowników, ich oczekiwań oraz wymagań związanych z użytecznością. Doświadczenie w kodowaniu jest ważne przy pracy z narzędziami do automatyzacji testów, które to narzędzia wymagają umiejętności programowania i projektowania, a także w przypadku uczestniczenia w statycznej analizie kodu, przeglądach kodu, testach jednostkowych oraz technicznych testach integracyjnych.

Umiejętności bezpośrednio związane z testowaniem to umiejętności opisywane w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB®, Sylabusie poziomu zaawansowanego Analitik Testów i Sylabusie poziomu zaawansowanego Techniczny Analitik Testowania, takie jak analiza specyfikacji, branie udziału w analizie ryzyka, projektowanie przypadków testowych oraz sumienne w wykonywanie testów i zapisywanie ich wyników.

Dla Kierowników Testów ważne jest posiadanie wiedzy, umiejętności i doświadczenia z zakresu zarządzania projektem, ponieważ na zarządzanie testami składa się wiele czynności związanych z zarządzaniem projektami np. tworzenie planów, śledzenie postępów oraz raportowanie do interesariuszy. W przypadku braku kierownika projektu, Kierownik Testów może objąć zarówno rolę Kierownika Testów, jak i kierownika projektu, w szczególności podczas końcowej fazy projektu. Umiejętności te powinny uzupełniać kompetencje opisane w sylabusie Certyfikowany tester – Sylabus poziomu podstawowego ISTQB® i niniejszym sylabusie.

Znaczącą rolę w testowaniu odgrywają również umiejętności interpersonalne, takie jak umiejętność konstruktywnej krytyki i reakcji na krytykę, wywieranie wpływu i negocjacje. Tester posiadający umiejętności techniczne nie spełni swojego zadania, o ile nie posiada i nie wykorzysta umiejętności interpersonalnych. Poza skuteczną współpracą z innymi, skuteczny tester musi być również dobrze zorganizowany, musi zwracać uwagę na szczegóły i musi umieć dobrze komunikować się w mowie i na piśmie.

Idealny zespół testowy posiada zróżnicowane umiejętności i poziom doświadczenia, a jego członkowie powinni wykazywać wolę i zdolność do przekazywania wiedzy i uczenia się od innych. W niektórych sytuacjach jedna umiejętność będzie ważniejsza i bardziej doceniana niż inne. Na przykład w środowisku, gdzie dominują testy techniczne i wymagane są umiejętności testowania API oraz programowania, umiejętności techniczne będą wyżej cenione niż wiedza biznesowa. Z kolei przy testach czarnoskrzynkowych, wiedza dziedzinowa może być bardziej ceniona. Należy pamiętać, że środowiska i projekty podlegają zmianom.

Podczas tworzenia arkusza oceny umiejętności Kierownik Testów powinien wziąć pod uwagę zarówno umiejętności potrzebne do dobrego wykonania pracy, jak i odpowiadające danej roli. Po utworzeniu listy potrzebnych umiejętności można ocenić każdego członka zespołu (nadając mu ocenę od 1 do 5, gdzie 5 oznacza najwyższy oczekiwany poziom danej umiejętności). Ocena poszczególnych osób pokazuje ich mocne i słabe strony i na podstawie tej informacji można sformułować plany szkoleniowe dla zespołów lub ich członków. Kierownik Testów może ustanowić cele rozwojowe dla konkretnych osób polegające na rozwijaniu umiejętności w określonych obszarach. Powinien też zdefiniować konkretne kryteria oceny poszczególnych umiejętności.

Ludzie powinni być zatrudniani na dłuższy czas, a nie na czas pojedynczego projektu. W sytuacji, kiedy Kierownik Testów inwestuje w testerów i stworzy klimat do ciągłego uczenia się, członkowie zespołu będą mieli motywację do doskonalenia swoich umiejętności i poszerzania wiedzy i będą otwarci na nowe perspektywy.

Sytuacja, kiedy Kierownik Testów będzie w stanie zatrudnić idealnych członków do swojego zespołu, rzadko ma miejsce, a nawet, jeżeli okaże się, że będą oni idealni do obecnego projektu, to może się okazać, że nie są idealni do następnego. Kierownik Testów powinien zatrudniać osoby, które są inteligentne, ciekawe i łatwo się adaptują, chcą pracować, umieją pracować skutecznie jako członkowie zespołu oraz chcą i mogą się uczyć. Nawet jeżeli idealni członkowie zespołu nie są dostępni, można zbudować mocną ekipę przez zrównoważenie mocnych i słabych stron poszczególnych osób.

Kierownik Testów może zidentyfikować silne i słabe strony swojego zespołu korzystając z arkusza oceny umiejętności. Ta informacja stanowi podstawę dla planu szkoleń i rozwoju umiejętności. Począwszy od niedostatków w umiejętnościach, które mają największy wpływ na skuteczność i efektywność zespołu, Kierownik Testów powinien zdecydować, jak podejść do tych obszarów. Jedno z możliwych podejść stanowią szkolenia, np. wysłanie współpracowników na szkolenie, zorganizowanie szkoleń na miejscu u siebie, stworzenie dedykowanego szkolenia, wykorzystanie kursów elektronicznych. Innym podejściem jest samodoskonalenie, np. książki, webinaria, zasoby Internetu. Jeszcze innym podejściem jest szkolenie wzajemne, np. wspólna praca kogoś, kto chce osiągnąć daną umiejętność, z kimś, kto posiada już tę umiejętność. Dzięki wspólnej pracy nad zadaniem wymagającym wykorzystania danej umiejętności, a także dzięki udziałowi w krótkich prelekcjach prowadzonych przez lokalnych ekspertów, pracownicy uczą się od siebie nawzajem itp. (Mentoring stanowi podobne podejście, w którym osoba nowa w jakiejś roli pracuje w parze z osobą doświadczoną, która już daną rolę pełniła i która stanowi dla niej ciągłe źródło wsparcia i porady). Oprócz eliminowania słabych stron zespołu, Kierownik Testów powinien pamiętać o wykorzystaniu jego silnych stron, które zostały zidentyfikowane podczas oceny umiejętności, a następnie zostały uwzględnione w planie szkoleń i rozwoju umiejętności. Więcej o planach rozwoju zespołu można przeczytać w [McKay07].

7.3. Dynamika zespołu testowego

Dla stworzenia możliwie najlepszego zespołu, nabór członków do zespołu jest jedną z kluczowych funkcji zarządczych w organizacji. Należy przy tym uwzględnić wiele dodatkowych czynników poza konkretnymi indywidualnymi umiejętnościami potrzebnymi do wykonania pracy. Podczas naboru nowych członków do zespołu należy wziąć pod uwagę również dynamikę zespołu i ocenić, czy dana osoba dobrze uzupełnia umiejętności i typy osobowości już obecne w zespole. Ważne jest również rozważenie korzyści płynących z posiadania różnych typów osobowości w zespole, jak również szerokiego wachlarza różnych umiejętności technicznych. Silny, pokrywający wszystkie kompetencje zespół testowy, poradzi sobie w różnych projektach o różnym stopniu złożoności, jednocześnie skutecznie porozumiewając się z pozostałymi członkami zespołu projektowego.

Testowanie jest często zadaniem związanym z wysoką presją. Harmonogramy projektów są często napięte, a nawet nierealistyczne. Interesariusze oczekują od zespołu testowego bardzo wiele, czasami bezzasadnie wiele. Kierownik Testów musi zatrudnić osoby, które poradzą sobie z sytuacjami pracy pod wysoką presją, które umieją rozładować frustrację i które potrafią skupić się na pracy, nawet jeżeli harmonogram wydaje się niemożliwy do utrzymania. Do Kierownika Testów należy załatwianie problemów z harmonogramem i oczekiwaniami, ale Kierownik Testów musi też zrozumieć, że presja jest odczuwana również przez zespół. Kiedy Kierownik Testów kompletuje zespół, ważne jest, aby rozpoznać środowisko pracy i dopasować osobowości członków zespołu do tego środowiska. W przypadku, gdy pracy jest więcej niż czasu, Kierownik Testów powinien szukać pracowników, którzy po wykonaniu zadania przychodzą i pytają, za co mają się wziąć w następnej kolejności.

Członkowie zespołu będą ciężiej pracowali i bardziej dbali o to, co robią, jeżeli będą czuli się docenieni i potrzebni. Muszą rozumieć, że każdy z nich jest ważny i wkład każdego z nich jest niezbędny do osiągnięcia sukcesu przez cały zespół. W momencie, gdy taka dynamika zostanie stworzona, szkolenia wzajemne będą odbywały się nieformalnie, podział pracy będzie wykonywany przez samych członków zespołu, a Kierownik Testów będzie miał więcej czasu, żeby zmierzyć się z problemami zewnętrznymi.

Nowi członkowie zespołu muszą być szybko asymilowani i muszą dostać odpowiednie wsparcie i nadzór. Każda osoba powinna mieć przydzieloną konkretną rolę w zespole, która może zostać wyznaczona po indywidualnej ocenie danej osoby. Celem jest umożliwienie każdemu z członków zespołu osiągnięcia sukcesu osobistego równolegle z zapewnieniem osiągnięcia sukcesu przez cały zespół. Można to osiągnąć w większej części przez dopasowanie typów osobowości do ról w zespole oraz rozwijanie wrodzonych i wykształcanie nowych umiejętności członków zespołu.

W ustalaniu, kogo zatrudnić i dołączyć do zespołu testowego, może być pomocna obiektywna ocena umiejętności. Można ją przeprowadzić w formie wywiadów, testowania kandydata, przeglądania próbek jego pracy czy weryfikację referencji. Do umiejętności, które należy ocenić, należą umiejętności techniczne oraz umiejętności interpersonalne.

Umiejętności techniczne (umiejętności twarde) można pokazać poprzez:

- projektowanie testów na podstawie dokumentu wymagań,
- przeglądanie dokumentacji technicznej (wymagania, kod itd.),
- pisanie komentarzy do przeglądu w sposób jasny, zrozumiały i obiektywny,
- zastosowanie różnych technik testowania w zależności od przedstawionego scenariusza (np. użycie tablic decyzyjnych do przetestowania zbioru reguł biznesowych),
- ocena awarii i dokładne jej udokumentowanie,
- pokazanie zrozumienia informacji o klasyfikacji defektów,
- pokazanie zrozumienia pierwotnych przyczyn defektów,

- użycie narzędzia do przetestowania podanego API, z wykonaniem testów pozytywnych i negatywnych,
- użycie języka SQL do wyszukania i zmiany informacji w bazie w celu przetestowania podanego scenariusza,
- zaprojektowanie jarzma testowego, które wykona kilka testów i zbierze wyniki testów,
- wykonanie testów automatycznych i rozwiązanie problemów przy awariach,
- pisanie planów i projektów testów,
- pisanie sumarycznych raportów z testów, które zawierają ocenę wyników testów,
- prezentowanie informacji dotyczących opóźniających się testów,
- wyjaśnienie zgłoszenia błędu deweloperowi, który myśli, że nie ma defektu,
- szkolenie współpracownika,
- zaprezentowanie kierownictwu problemu z nieskutecznym procesem,
- przejrzanie przypadku testowego zaprojektowanego przez współpracownika i przekazanie jemu komentarzy na temat tego przypadku,
- przeprowadzenie wywiadu ze współpracownikiem,
- komplementowanie deweloperów.

Powyższych list nie należy traktować jako zamknięte, ponieważ potrzeby zmieniają się pomiędzy środowiskami i organizacjami. Niemniej jednak listy te zawierają zestaw umiejętności, które są potrzebne najczęściej. Dzięki opracowaniu pytań wykorzystywanych podczas rozmów kwalifikacyjnych i umożliwiających skuteczną weryfikację wiedzy, doświadczenia i posiadanych umiejętności kandydatów, Kierownik Testów może ocenić umiejętności kandydatów i określić ich mocne i słabe strony. Gdy powstaną skuteczne narzędzia oceny, to powinny one zostać również zastosowane w stosunku do już zatrudnionych pracowników w celu ustalenia obszarów rozwoju i szkolenia.

Poza umiejętnościami koniecznymi dla poszczególnych członków zespołu, Kierownik Testów musi również posiadać doskonałe umiejętności komunikacyjne i dyplomatyczne. Kierownik Testów musi być w stanie rozładowywać napięcia, rozstrzygać w sytuacjach kontrowersyjnych oraz musi wiedzieć, jakiego środka i technik komunikacji użyć w danej sytuacji i musi umieć skupić się na budowaniu właściwych relacji wewnątrz organizacji.

7.4. Umieszczenie testów w organizacji

Istnieje wiele sposobów na umiejscowienie testów w strukturze organizacji. Mimo, że za jakość są odpowiedzialni wszyscy uczestnicy cyklu życia wytwarzania oprogramowania, niezależny zespół testowy może znacząco przyczynić się do wytworzenia produktu o wysokiej jakości. Niezależność testowania może się w praktyce bardzo różnić w organizacjach i projektach, tak jak to zostało wyszczególnione na poniższej liście uporządkowanej od najniższego do najwyższego poziomu niezależności:

- brak niezależnych testerów:
 - w tym przypadku nie ma niezależności, a kod jest testowany przez programistę, który go napisał,
 - programista, o ile ma czas na przeprowadzenie testów, sprawdzi, czy kod działa tak, jak to było zamierzone, co może (lub nie) odpowiadać rzeczywistym wymaganiom,
 - programista może szybko poprawić znalezione błędy,
- testowanie jest wykonywane przez innego programistę niż ten, który napisał kod:
 - istnieje niewielka niezależność pomiędzy programistą a testerem,
 - programista testujący kod innego programisty może nie chcieć raportować usterek,
 - programista jest zwykle nastawiony na wykonanie pozytywnych przypadków testowych,
- testy są wykonywane przez testera lub zespół testowy, który stanowi część zespołu tworzącego oprogramowanie, tester (lub zespół testowy) podlega kierownikowi projektu lub kierownikowi zespołu tworzącego oprogramowanie:

- tester jest zwykle nastawiony na weryfikację stopnia spełnienia wymagań,
 - ponieważ tester jest członkiem zespołu tworzącego oprogramowanie, może mieć również zlecone zadania programistyczne (oprócz zadań testerskich),
 - szef testera może bardziej skupiać się na dotrzymaniu harmonogramu, niż na osiągnięciu celów jakościowych,
 - testowanie może mieć niższy status w zespole niż programowanie,
 - tester może nie mieć uprawnień do wpływania na cele jakościowe i uprawnień do trzymania się tych celów,
- testy są wykonywane przez specjalistów z departamentów biznesowych, społeczność użytkowników lub inną organizację techniczną niezwiązaną z tworzeniem oprogramowania:
 - wyniki testów są raportowane interesariuszom w sposób obiektywny,
 - jakość jest priorytetem dla takich zespołów,
 - rozwój umiejętności i szkolenia są skoncentrowane na testowaniu,
 - testowanie jest postrzegane jako możliwa ścieżka kariery,
 - jest wyznaczony kierownik grupy testującej, który koncentruje się na jakości,
- zewnętrzni specjaliści wykonują niektóre typy testów:
 - wiedza ekspercka jest stosowana w konkretnych obszarach, gdzie ogólne podejście do testowania będzie najprawdopodobniej niewystarczające,
 - do tych typów testów mogą należeć: testy użyteczności, zabezpieczeń, wydajności i innych obszarów, w których konieczna jest specjalizacja,
 - ci specjaliści powinni skupiać się głównie na jakości, ale ich postrzeganie jest ograniczone do ich własnej specjalności; produkt, który jest wydajny, może nie spełniać wymagań funkcjonalnych, co może nie zostać zauważone przez specjalistę od wydajności,
- testy są wykonywane przez organizację zewnętrzną w stosunku do firmy:
 - ten model umożliwia osiągnięcie największej niezależności testera od programisty,
 - przekaz wiedzy może nie być wystarczający do tego, aby tester mógł testować kompetentnie,
 - potrzebne są precyzyjne wymagania i dobrze zdefiniowana struktura komunikacji,
 - trzeba regularnie audytować jakość zewnętrznej organizacji.

Powyższa lista bazuje na typowych celach różnych osób, ale może nie sprawdzić się w konkretnej organizacji. Stanowisko i tytuł niekoniecznie determinują cele osoby pełniącej określoną rolę. Kierownicy zespołu tworzącego oprogramowanie mogą się skupiać na jakości tak samo, jak dobrzy Kierownicy Testów. Niezależny Kierownik Testów może podlegać kierownictwu, któremu zależy na dotrzymaniu harmonogramu, może więc przez to skupiać się bardziej na harmonogramie niż na jakości. Aby ustalić, jakie jest najlepsze umiejscowienie testów w organizacji, Kierownik Testów musi rozumieć cele organizacji.

Istnieje wiele poziomów niezależności deweloperów od organizacji testowej. Ważne jest zrozumienie, że możliwy jest kompromis, w którym większa niezależność skutkuje większą izolacją i mniejszym transferem wiedzy. Niższa niezależność może sprawić, że wiedza będzie miała szerszy zakres, ale może też wprowadzić konflikt interesów. Poziom niezależności będzie również zależał od wykorzystywanego modelu rozwoju oprogramowania, np. w metodykach zwinnych testerzy będą najczęściej częścią zespołu deweloperskiego.

W organizacji może funkcjonować każda z powyższych opcji. Testowanie może być wykonywane w organizacji deweloperskiej, jak również przez niezależną organizację testową i na koniec może być przeprowadzana certyfikacja przez organizację zewnętrzną. Ważne jest zrozumienie odpowiedzialności i oczekiwań dla każdego z etapów testowania i takie ustawienie wymagań, aby osiągnąć maksymalną jakość końcowego produktu jednocześnie trzymając się uzgodnień związanych z ograniczeniami terminowymi i kosztowymi.

7.5. Motywacja

Istnieje wiele sposobów na motywowanie pracowników na stanowisku testera. Są to między innymi:

- uznanie za ukończoną pracę,
- aprobata kierownictwa,
- szacunek w zespole projektowym i pomiędzy współpracownikami,
- większa odpowiedzialność i autonomia,
- odpowiednie wynagrodzenie za pracę (pensja, większa odpowiedzialność, uznanie).

Istnieją pewne czynniki projektowe, które mogą spowodować, że będzie trudno zastosować te narzędzia motywacyjne. Na przykład, tester może pracować bardzo ciężko w projekcie, w którym terminowe ukończenie pracy jest niemożliwe. Tester może robić wszystko, co możliwe, aby zwrócić uwagę zespołu na jakość, pracować w nadgodzinach, a jednak w wyniku działania czynników zewnętrznych, produkt może zostać wydany przedwcześnie. Skutkiem takiego działania może być wydanie produktu niskiej jakości, pomimo najlepszych starań testera. Taka sytuacja może demotywować, jeżeli wkład testera nie zostanie zrozumiany i zmierzony niezależnie od tego, czy produkt w końcu odniósł sukces.

Zespół testowy musi upewnić się, że śledzi odpowiednie miary, tak, by udowodnić, że dobrze wykonał testy, łagodził ryzyko i precyzyjnie raportował wyniki. Zespół testowy może się łatwo zdemotywować, jeżeli nie uzyska uznania, które w jego odczuciu należy się jemu za dobrze wykonaną pracę.

Uznanie nie przejawia się tylko poprzez nienamacalny szacunek i aprobatę. Uwidacznia się również w okazjach do promocji, powierzaniu większej odpowiedzialności i zwiększaniu zasług. Jeżeli grupa testerów nie jest szanowana, to może nie mieć takich możliwości.

Uznanie i szacunek pojawiają się wtedy, gdy widoczny jest wkład testera w wartość projektu. W pojedynczym projekcie można to najlepiej osiągnąć przez zaangażowanie testerów od samego początku projektu i utrzymanie tego zaangażowania przez cały cykl życia. Z czasem testerzy przez swój wkład w projekt uzyskują szacunek pozostałych interesariuszy projektu i członków zespołu projektowego. Ten wkład powinien zostać wyliczony przez pomiar zmniejszenia kosztów jakości i łagodzenia ryzyka.

Kierownik Testów odgrywa znaczącą rolę zarówno w motywacji członków zespołu testowego, jak i w działaniu jako orędownik zespołu testowego w stosunku do reszty organizacji. Kierownik Testów powinien chwalić poszczególnych testerów za ich osiągnięcia, ale musi też uczciwie i szczerze oceniać ich pomyłki. Sprawiedliwy i konsekwentny Kierownik Testów motywuje testerów swoim przykładem.

7.6. Komunikacja

Komunikacja w zespole testowym realizuje się poprzez:

- dokumentację produktów testowania – strategię testów, plan testów, przypadki testowe, sumaryczne raporty z testów, zgłoszenia błędów itd.,
- informację zwrotną na temat przeglądanych dokumentów – wymagań, specyfikacji funkcjonalnych, przypadków użycia, dokumentacji testów modułowych itd.,
- zbieranie i przekazywanie informacji – interakcja z deweloperami, pozostałymi członkami zespołu testowego, kierownictwem itp.

Komunikacja pomiędzy testerami a innymi interesariuszami projektu musi być profesjonalna, obiektywna i skuteczna, tak żeby budowała i wspierała szacunek dla zespołu testowego. Przy przekazywaniu informacji zwrotnych, a w szczególności konstruktywnych uwag, na temat produktów innych osób, wymagane są dyplomacja i obiektywizm. Ponadto komunikacja musi być nastawiona na

osiąganie celów testowania i na poprawę jakości zarówno produktów, jak i procesu wykorzystywanego do produkcji oprogramowania.

Kierownicy Testów komunikują się z wieloma osobami: użytkownikami, członkami zespołu projektowego, kierownictwem, zewnętrznymi grupami testującymi i klientami. Komunikacja ta musi być skuteczna w odniesieniu do docelowych odbiorców. Na przykład, raport stworzony dla zespołu deweloperskiego, który zawiera trendy i tendencje w liczbie i wagach zgłoszonych błędów, może być zbyt szczegółowy dla kierownictwa wysokiego szczebla. Ważne jest, żeby podczas każdej formy komunikacji, a w szczególności podczas prezentacji, rozumieć przekazywaną informację, sposoby, w jakie ta informacja może zostać odebrana i dodatkowe wyjaśnienia potrzebne do podjęcia decyzji o akceptacji przekazywanej informacji. Ponieważ Kierownik Testów często musi prezentować informacje dotyczące statusu projektu, ważne jest, żeby informacje te były na odpowiednim poziomie szczegółowości (np. kierownicy wolą raczej oglądać trendy związane z błędami niż poszczególne błędy) i były prezentowane w sposób jasny i zrozumiały (np. proste wykresy i kolorowe diagramy). Sprawna komunikacja pomaga w utrzymaniu uwagi odbiorców z równoczesnym przekazywaniem poprawnej informacji. Każda prezentacja powinna być postrzegana przez Kierownika Testów jako okazja do promowania jakości i procesów jakościowych.

Kierownik Testów komunikuje się nie tylko z osobami zewnętrznymi w stosunku do danego działu (komunikacja zewnętrzna). Ważną częścią pracy Kierownika Testów jest efektywna komunikacja z grupą testerską (komunikacja wewnętrzna) celem przekazania wiadomości, instrukcji, zmian w priorytetach i innej standardowej informacji będącej elementem typowego procesu testowego. Kierownik Testów może się również porozumiewać z poszczególnymi osobami stojącymi wyżej (komunikacja wzwyż) lub niżej (komunikacja w dół) w strukturze zarządzania w organizacji. Bez względu na kierunek komunikacji, mają zastosowanie te same zasady: komunikat powinien być odpowiedni dla odbiorców, efektywny, a jego zrozumienie powinno być potwierdzone.

Kierownicy Testów muszą być mistrzami w stosowaniu różnych form komunikacji. Wiele informacji jest przekazywanych przez e-maile, komunikację słowną, w trakcie spotkań formalnych i nieformalnych, poprzez formalne i nieformalne raporty, a nawet przez wykorzystywanie narzędzi do zarządzania testami takich jak narzędzia do zarządzania zgłoszeniami błędów. Cała komunikacja musi być profesjonalna i obiektywna. Korekta, zarówno jakości, jak i zawartości komunikatu, stanowi konieczny krok w przygotowaniu nawet najpilniejszego komunikatu. Komunikacja w formie pisemnej często żyje dużo dłużej niż projekt, w ramach którego powstała. Ważne jest, żeby Kierownik Testów tworzył dokumentację o profesjonalnej jakości, która może być wizytówką organizacji promującej jakość.

8. Odnosiniki

8.1. Normy

- Standard ISO/IEC/IEEE 29119 zastąpił normę IEEE 829.
- Standard ISO/IEC 25010 zastąpił normę ISO 9126.
- Standard ISO/IEC 20246 zastąpił normę IEEE 1028.
- [BS7925-2] BS 7925-2, Software Component Testing Standard.
Rozdział 2
- [FDA21] FDA Title 21 CFR Part 820, Food and Drug Administration, USA
Rozdział 2
- [IEEE829] IEEE Standard for Software and System Test Documentation
Rozdziały 2 oraz 4

Zastąpiony przez:

- ISO/IEC/IEEE 29119-1 (2013) Software and systems engineering — Software testing — Part 1: Concepts and definitions
- ISO/IEC/IEEE 29119-2 (2013) Software and systems engineering — Software testing — Part 2: Test processes
- ISO/IEC/IEEE 29119-3 (2013) Software and systems engineering — Software testing — Part 3: Test documentation
- ISO/IEC/IEEE 29119-4 (2015) Software and systems engineering — Software testing — Part 4: Test techniques
- [IEEE1028] IEEE Standard for Software Reviews and Audits
Rozdział 2

Zastąpiony przez:

- ISO/IEC 20246 (2017) Software and systems engineering — Work product reviews
- [IEEE1044] IEEE Standard Classification for Software Anomalies
Rozdział 4
- [ISO9126] ISO/IEC 9126-1:2001, Software Engineering - Software Product Quality,
Rozdziały 2 oraz 4

Zastąpiony przez:

- ISO/IEC 25010 (2011) Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) System and software quality models
- [ISO12207] ISO 12207, Systems and Software Engineering, Software Life Cycle Processes
Rozdział 2

- [ISO15504] ISO/IEC 15504, Information Technology - Process Assessment
Rozdział 2
- [RTCA DO-178B/ED-12B]: Software Considerations in Airborne Systems and Equipment Certification, RTCA/EUROCAE ED12B.1992.
Rozdział 2

Zastąpiony przez:

- [RTCA DO-178C/ED-12C]: Software Considerations in Airborne Systems and Equipment Certification, RTCA/EUROCAE.2012.

8.2. Dokumenty ISTQB®

- [ISTQB_AL_OVIEW] ISTQB® Advanced Level Overview, Version 1.0
- [ISTQB_ALTM_SYL] ISTQB® Advanced Level Test Manager Syllabus, Version 1.0
- [ISTQB_ALTTA_SYL] ISTQB® Advanced Level Technical Test Analyst Syllabus, Version 1.0
- [ISTQB ETM SYL] ISTQB® Expert Test Management Syllabus, Version 1.0
- [ISTQB_FL_SYL] ISTQB® Foundation Level Syllabus, Version 2011
- [ISTQB_GLOSSARY] Standard glossary of terms used in Software Testing, Version 2.2, 2012
- [ISTQB ITP SYL] ISTQB® Expert Improving the Test Process Syllabus, Version 1.0

8.3. Znaki handlowe

W dokumencie użyto następujących zarejestrowanych znaków handlowych i towarowych:

- CMMI®, IDEALSM, ISTQB®, ITIL®, PRINCE2®, TMMi®, TPI Next®
- CMMI jest zarejestrowane w U.S. Patent and Trademark Office przez Carnegie Mellon University.
- IDEAL jest znakiem handlowym Software Engineering Institute (SEI), Carnegie Mellon University
- ISTQB® jest zarejestrowanym znakiem firmowym International Software Testing Qualifications Board
- ITIL jest zarejestrowanym znakiem firmowym oraz jest zarejestrowanym znakiem społecznościowym Office of Government Commerce, zarejestrowanym w U.S. Patent and Trademark Office
- PRINCE2 jest zarejestrowanym znakiem firmowym Cabinet Office
- TMMi jest zarejestrowanym znakiem firmowym TMMi Foundation
- TPI Next jest zarejestrowanym znakiem firmowym Sogeti Nederland B.V.

8.4. Książki

Adam Roman, „Testowanie i jakość oprogramowania. Modele, techniki, narzędzia”, Wydawnictwo Naukowe PWN, 2015, ISBN 978-83-01-18160-4

[Black03]: Rex Black, “Critical Testing Processes,” Addison-Wesley, 2003, ISBN 0-201-74868-1

- [Black09]: Rex Black, "Managing the Testing Process, third edition", John Wiley & Sons, 2009, ISBN 0-471-22398-0
- [Black11]: Rex Black, Erik van Veenendaal, Dorothy Graham, "Foundations of Software Testing", Thomson Learning, 2011, ISBN 978-1-84480-355-2
- [Craig02]: Rick Craig, Stefan Jaskiel, "Systematic Software Testing," Artech House, 2002, ISBN 1-580-53508-9
- [Crispin09]: Lisa Crispin, Janet Gregory, "Agile Testing: A Practical Guide for Testers and Agile Teams", Addison-Wesley, 2009, ISBN 0-321-53446-8
- [de Vries09]: Gerrit de Vries, et al., "TPI Next", UTN Publishers, 2009, ISBN 90-72194-97-7
- [Goucher09]: Adam Goucher, Tim Riley (editors), "Beautiful Testing", O'Reilly, 2009, ISBN 978-0596159818
- [IDEAL96]: Bob McFeeley, "IDEAL: A User's Guide for Software Process Improvement", Software Engineering Institute (SEI), 1996, CMU/SEI-96-HB-001
- [Jones07]: Capers Jones, "Estimating Software Costs, second edition", McGraw-Hill, 2007, ISBN 978-0071483001
- [Jones11]: Capers Jones and Olivier Bonsignour, "Economics of Software Quality", Pearson, 2011, ISBN 978-0132582209
- [McKay07]: Judy McKay, "Managing the Test People", Rocky Nook, 2007, ISBN 978-1933952123
- [Musa04]: John Musa, "Software Reliability Engineering, second edition", Author House, 2004, ISBN 978-1418493882
- [Stamatis03]: D.H. Stamatis, "Failure Mode and Effect Analysis", ASQC Quality Press, 2003, ISBN 0-873-89300
- [vanVeenendaal11]: Erik van Veenendaal, "The Little TMMi", UTN Publishers, 2011, ISBN 9-490-986038
- [vanVeenendaal12]: Erik van Veenendaal, "Practical Risk-based Testing", UTN Publishers, 2012, ISBN 978-9490986070
- [Whittaker09]: James Whittaker, "Exploratory Testing", Addison-Wesley, 2009, ISBN 978-0321636416
- [Wiegers03]: Karl Wiegers, "Software Requirements 2", Microsoft Press, 2003, ISBN 978-0735618794

8.5. Inne odnośniki

Poniższe referencje odnoszą się do informacji dostępnej w Internecie.
Odnosińki te zostały sprawdzone przed publikacją sylabusu dla poziomu zaawansowanego.

<http://www.istqb.org>
<http://www.sei.cmu.edu/cmml/>
<http://www.tmmi.org/>
<http://www.tpinext.com/>
<http://www.sjsi.org>

9. Indeks

- Agile, *patrz*: metodyki zwinne
- Analiza Drzewa Usterek (FTA), 32
- analiza gęstości defektów, 63
- Analiza Przyczyn i Skutków Awarii (FMEA), 32
- analiza ryzyka, 29, 30
- analiza ryzyka jakościowego, 33
- analiza średniego czasu pomiędzy awariami (ang. *Mean Time Between Failures* MTBF), 64
- analiza testów, 12, 22
- analiza warunków testowych, 34
- analiza zagrożeń, 32
- anomalie, 61, 62
- audyt, 54, 55
- awaria, 63, 64błąd, 61
- budowa zespołu, 79, 80, 81
- CMMI, 52, 66-69
- CTP (Krytyczne Procesy Testowania), 67, 68, 70, 71
- cykl doskonalenia Deminga, *patrz*: model Deminga
- cykl życia defektu, 53
- cykl życia
- zwinny, 15
 - iteracyjny, 15, 23
 - modelu V, 23
 - kaskadowy, 23
- cykl życia narzędzia, 77
- czynności zamykające testy, 9, 16, 23
- defekt
- atrybuty, 63
 - cykl życia i stany, 62
 - międzyfunkcjonalne zarządzanie defektami, 63
- dopasowanie czynności testowych do cyklu życia oprogramowania, 9
- dopasowanie narzędzi do zadań i potrzeb, 74
- doskonalenie procesu testowego, 66, 68
- doskonalenie procesu testowego z CTP, 70
- doskonalenie procesu testowego z STEP, 67, 68, 71
- doskonalenie procesu testowego z TMMi, 67-70
- doskonalenie procesu testowego z TPI
- Next, 67, 68, 70
- dynamika zespołu testowego, 72
- Extreme Programming (XP), 23
- główny plan testów, 16, 34
- IDEAL, 68
- identyfikacja ryzyka, 18, 27, 28, 33
- IEEE, 51
- IEEE 1028, 51
- IEEE 829, 14, 42, 65
- implementacja testów, 15, 16
- inspekcja, 55, 58
- interesariusze, 11, 20, 21
- interesariusze biznesowi, 33
- interesariusze techniczni, 33
- ISO, 51
- ISO 25010, 38
- ISO 9126, 28, 38, 51, 65
- ITIL, 52
- Kierownik Testów, 10, 15, 16, 20-23
- komitet zarządzania usterkami, 63
- komunikacja, 85, 86
- komunikacja do góry, 86
- komunikacja na zewnątrz, 86
- komunikacja w górę, 86
- komunikacja zstępująca, 86
- konsylium nad defektami, *patrz*: komitet zarządzania usterkami
- koszt awarii wewnętrznej, 49, 50
- koszt awarii zewnętrznej, 49
- koszt jakości, 49, 50
- koszt wykrycia, 49
- koszt zapobiegania, 49
- kryteria wejścia i wyjścia, 24, 37, 40, 45, 47, 59
- kryteria zakończenia, 9, 16
- Krytyczne Procesy Testowania (CTP), 67, 68, 70, 71
- krytyczność, 28, 60, 64
- Lean, *patrz*: metodyki odchudzone
- Lider Testów, 20, 49
- log testów, 15, 26, 41
- łagodzenie ryzyka, 18, 25, 27, 29, 30, 41
- łączna ocena ryzyka, 25
- metodyki odchudzone (Lean), 48
- metodyki zwinne (Agile), 10, 15, 21, 23, 33, 35, 38, 40, 43, 48, 50, 84
- metryki dla przeglądów, 58
- metryki osobowe, 44
- metryki procesowe, 44
- metryki produktowe, 44
- metryki projektowe, 44, 45
- metryki dotyczące narzędzi, 69
- metryki testowe, 24

metryki związane z testowaniem, 13, 19, 24, 44, 45
misja, 9, 11, 15, 33
model ciągłego doskonalenia Deminga, 67
model IDEAL, *patrz:* IDEAL
model sekwencyjny, 22
moderator, 32, 53
monitorowanie testów, 11, 12
motywacja, 85
możliwość śledzenia, 15
MTBF (ang. *Mean Time Between Failures*)
patrz: analiza średniego czasu pomiędzy awariami
nadzór nad testami, 9, 11, 12, 23, 48
narzędzia testowe (narzędzia wspierające testy), 72, 73, 74
ocena kryteriów zakończenia i raportowanie, 9, 16
ocena ryzyka, 27, 28, 33, 55
ocena umiejętności, 82
Ogólna Publiczna Licencja GNU, 73
otwarty kod, 73
planowanie ryzyka, 31
plan przeglądu, 54
planowanie testów, 9, 10
planowanie, monitorowanie i nadzór nad testami, 11, 12
plan testów, 16, 19, 26, 39, 40
plan testów jednego poziomu, 40
PMI, 52
podejścia metodyczne, 35
podejście do testowania, 10, 39, 84
podejście reaktywne, 35
podstawa testów, 12, 13, 34
podstawowy proces testowy, 9
polityka testów, 18, 36, 37
powstrzymanie fazowe, 61
poziom ryzyka, 29, 30, 31, 33, 36
poziom testów, 12, 23, 24, 40
prawdopodobieństwo, 28, 29, 30, 33
PRINCE2, 52
priorytet 26, 27, 29, 30, 32, 34, 64, 69, 70
priorytetyzacja testów, 35
procedura testowa, 8, 14
Proces Systematycznego Testowania i Oceny (STEP), 67, 68, 71
profile operacyjne, 35, 37
profile użycia, 35, 36
projektowanie testów, 12, 13
przewodzenie przeglądów formalnych, 57
przegląd kierowniczy, 54, 55
przegląd, 55
przegląd nieformalny, 55, 56, 58

przeglądy niejednoznaczności, 34
przegląd techniczny, 55, 56, 58
przejrzanie, 55, 56, 58
przyczyna podstawowa, 30, 32, 45, 60, 62
przypadek testowy, 8, 10-15, 61, 77
Quality Function Deployment (QFD), *patrz:* Rozmieszczenie Funkcji Jakości (QFD)
raportowanie, 16
recenzent, 57
rezultat fałszywie niezaliczony, 61, 62
rezultat fałszywie zaliczony, 63
Rozmieszczenie Funkcji Jakości (QFD), 32
ryzyko jakościowe, 18, 27, 50
ryzyko jakości produktu, 26, 27, 28, 29
ryzyko produktowe, 12, 18, 26, 28, 30
ryzyko projektowe, 12, 18, 26, 28, 33, 40, 41
ryzyko, 25, 26, 29
ryzyko techniczne, 23
Scrum, 23, 38
sesje testowe (testowania), 22
skrypt testowy, 8, 75
status testów, 46, 47
strategia testów, 18, 36, 37, 39
standardy
 BS 7925-2, 51
 CMMI, 52, 66-69
 DO-178B, 29, 51, 74, 92
 ED-12B, 29, 51, 74
 IEC 61508, 29
 Tytuł 21 CFR Część 820, 52
STEP, 67, 68, 71
strategie
 analityczne, 12, 37
 konsultatywne, 38
 metodyczne, 33
 oparte na modelu, 37
 zgodne z procesem lub normą, 38
 reaktywne, 38
 testowe minimalizujące regresję, 38
Struktura podziału pracy Work Breakdown Structure (WBS), 43
szacowanie testów, 18, 19, 42, 43
szerokopasmowa technika delficka, 18, 43
sumaryczny raport z testów, 8, 18, 43
tablica Kanban, 48
Testing Maturity Model integration (TMMi), 67-70
testowanie integracji produktu klienta, 24
testowanie integracyjne systemu, 24
testowanie interakcji cech, 24
testowanie oparte na doświadczeniu, 25
testowanie oparte na ryzyku, 18, 26, 27, 29, 31, 33, 34, 37

testowanie oparte na wymaganiach, 34
testowanie rozproszone, 19, 50
testowanie eksploracyjne, 10, 26, 29, 38
testowanie potwierdzające (retesty), 39
testowanie regresji, 14, 16, 21, 29, 38, 39, 46, 75
testowanie w głąb, 30
testowanie wrzecz, 30
testowanie zlecane na zewnątrz, 50
testy niefunkcjonalne, 25
TMMi, *patrz*: Testing Maturity Model integration 67-70
TPI Next, 70, 59, 60, 62
typy doskonałości procesu, 67, 68
umiejętności interpersonalne, 82
umiejętności miękkie, 79
umiejętności indywidualne, 82
umiejętności twarde, 79
umowy o gwarantowanym poziomie usług (SLA Service Level Agreement), 11
wartości ilościowe, 49
wartości jakościowe, 49

warunki testowe, 11-15, 27, 37, 38, 54
wpływ (na testowanie), 21, 22
wykonanie testów, 8, 9, 13, 22, 23, 24, 42, 48
wytworzenie sterowane testami (*Test Driven Development* - TDD), 43, 61
zakończenie testowania (czynności zamykające testy), 16, 17
zarządzanie przeglądami, 53, 56
zarządzanie ryzykiem, 18, 27, 30
zarządzanie ryzykiem projektowym, 41
zarządzanie testowaniem niefunkcjonalnym, 25
opartym na doświadczeniu, 25
zewnętrzni specjaliści testowi, 84
zwrot z inwestycji (ROI), 57, 74, 76