Metodyka Testów

Metodyka testów

Właściciel procesu: Kierownik Wydziału Testów - Rafał Cebera

Koordynator procesu: Agata Parzych

Wersja: 2, Wrzesień 2020

plik do pobrania: Metodyka w pdf

Spis treści

1. Wprowadzenie

1.1. Cel dokumentu

Celem niniejszego dokumentu jest:

- przedstawienie wytycznych w zakresie przygotowania, realizacji oraz udokumentowania testów w projekcie informatycznym.
- Systematyzacja i uporządkowanie podejścia do realizacji testów w projektach CeZ.

Został on opracowany na podstawie Metodyki prowadzenia testów przygotowanej na potrzeby projektu P1 - projektu „Elektroniczna Platforma gromadzenia, analizy i udostępniania zasobów cyfrowych o zdarzeniach medycznych" (P1) – faza 2.

1.2. Elementy metodyki testów

Na metodykę testów składają się:

- Niniejszy opis ogólny
- Szablony dokumentacji testów zaimplementowane w narzędziu wspierającym testowanie:
  - Plan testów
  - Scenariusz testów
  - Raport z testów
  - Raport zbiorczy z testów
  - ...../
- Opisy ról biorących udział w testach:
  - Lider testów
  - Tester
  - Uwaga: dopuszcza się dostosowanie ról w zależności od wielkości i specyfiki projektu. Dostosowanie rozumiane jest jako wskazanie osób pełniących daną rolę, zdefiniowanie kolejnych ról.
- Narzędzia wspierające realizację i dokumentowania testów - xRay

1.3. Zasady utrzymania i rozwoju metodyki

Odpowiedzialnym za utrzymanie i rozwój niniejszej metodyki testów jest Kierownik Wydziału Testów CeZ we współpracy z jednostkami odpowiedzialnymi za rozwój, utrzymanie i bezpieczeństwo systemów teleinformatycznych w CeZ.

Metodyka Przeprowadzenia Testów podlega przeglądom w cyklu półrocznym. Jej aktualizacja odbywać się będzie na podstawie wniosków z przeglądu lub w przypadku zgłoszenia potrzeby zmiany.

2. Kontekst Projektowy

W ramach każdego projektu obejmującego swoim zakresem budowę, zmianę systemu informatycznego konieczne jest zaplanowanie, przygotowanie, wykonanie testów, udokumentowanie ich wyników i zapewnienie wymaganych akceptacji jako podstawowy element przygotowania do wdrożenia systemu lub jego kolejnej wersji.

Planowanie testów w każdym z projektów musi być poprzedzone identyfikacją optymalnej listy testów, które potwierdzą gotowość przygotowanych zmian do wdrożenia.

Podejście opisane niniejszą Metodyką Przeprowadzenia Testów ma zastosowanie dla wszystkich projektów informatycznych realizowanych w CeZ.

3. Proces Testowy

Proces testowy obejmuje następujące etapy:

- Planowanie testów
- Realizacja testów
- Zakończenie realizacji testów

3.1. Planowanie Testów

W projekcie planowanie testów powinno być realizowane w zależności od przyjętej metody realizacji projektu: realizacja projektu metodą tradycyjną (kaskadową, waterfall) lub metodą przyrostową (zwinną). W każdym z przypadków, planowanie testów, z reguły, wykonuje się 2 etapowo: planowanie ogólne i planowanie szczegółowe.

Metoda tradycyjna:

Planowanie ogólne – wykonywane na etapie Analizy

Planowanie szczegółowe – na etapie produkcji rozwiązania

Metoda przyrostowa:

Oba etapy wykonywane dla każdego wydania osobno.

Planowanie ogólne – najpóźniej w ramach prac analitycznych do ostatniego sprintu wydania

Planowanie szczegółowe – na etapie produkcji rozwiązania

Produktem planowania testów dla każdego z poziomów jest dokument Plan Testów, którego format powinien być zgodny z obowiązującym w CeZ szablonem.

Dokumentacja testów przygotowywana jest w narzędziu wspierającym testy x-Ray. Należy zapewnić powiazanie przestrzeni projektu w Confluence z dokumentacją testów w x-Ray.

3.1.1. Planowanie testów dla etapu wdrożenia projektu

Kluczowymi celami planowania etapu wdrożenia projektu są:

- identyfikacja typów i poziomów testów, które powinny być zrealizowane w ramach etapu wdrożenia oraz zależności pomiędzy nimi,
- zbudowanie harmonogramu testów w etapie (inicjatywa testowa),
- identyfikacja zagrożeń dla realizacji testów podczas danego etapu wdrożenia.

Na potrzeby planowania testów niezbędne jest udostępnienie dokumentacji etapu, a w szczególności:

- harmonogramu etapu wdrożenia,
- zakresu etapu wdrożenia, w tym dokumentacji rozwiązania i założeń realizacyjnych,
- wymagań biznesowych i prawnych względem zakresu zawartego w etapie wdrożenia.

3.1.2. Planowanie testów inicjatywy testowej w ramach etapu wdrożenia projektu

Każda inicjatywa testowa uruchamiana jest dla weryfikacji specyficznego, konkretnego aspektu jakościowego rozwiązania. Celem planowania testów na tym poziomie jest uzgodnienie kwestii operacyjnych umożliwiających realizację zadań stawianych przed daną inicjatywą testową oraz zidentyfikowanie potencjalnych zagrożeń dla inicjatywy.

Wszelkie uzgodnienia i założenia realizacyjne powinny zostać zawarte w dokumencie Plan Testów, w zgodzie z szablonem obowiązującym w CeZ. Możliwe są modyfikacje struktury dokumentu, jednak poniższe aspekty obowiązkowo muszą być zaadresowane:

- Cel i zakres testów.
- Zespół testowy.
- Podejście do raportowania i komunikacji.
- Wymagania oraz podejście do zapewnienia środowisk, narzędzi i danych testowych.
- Ryzyka związane z realizacją testów.
- Kryteria poprawności testów.
- Odstępstwa od Metodyki Przeprowadzenia Testów – jeśli mają miejsce.
- Modyfikacje standardów i procedur obowiązujących w CeZ – jeśli mają miejsce.

3.2. Realizacja Testów

Realizacja testów obejmuje następujące podprocesy:

- Implementacja Testu
- Zapewnianie Środowisk, Narzędzi i Danych Testowych
- Wykonanie Testu
- Raportowanie Błędu

3.2.1. Implementacja Testu

Na proces ten składają się zarówno czynności implementacyjne, jak również poprzedzające je czynności projektowe.

Działania projektowe polegające na przygotowaniu Przypadków i Scenariuszy testowych powołują się na Plan Testów, który jest produktem Planowania testów oraz na podstawę testów. Zgodnie ze standardami ISTQB podstawa testów powinna być adekwatna do poziomu i rodzaju testów dobranych do specyfiki systemu. Przykładami takiej podstawy testów są:

- projekt oprogramowania i systemu,
- architektura,
- przepływy procesów,
- wymagania użytkownika,
- wymagania systemowe,
- przypadki użycia,
- procesy biznesowe,
- raporty z analizy ryzyka.

Przygotowując Przypadki i Scenariusze testowe należy stosować techniki projektowania adekwatne do realizowanego typu i poziomu testów. Należy rozważyć stosowanie następujących technik:

- Klasy równoważności
- Analiza wartości granicznych
- Analiza tablicy decyzji
- Analiza przejścia stanów
- Scenariusze oparte o przypadki użycia
- Scenariusze oparte o user stories – dla testów systemowych

- Zgadywanie błędów
- Heurystyka RCRCRC – przy planowaniu testów regresji

Lista stosownych technik może być zmodyfikowana, na bazie doświadczenia osoby projektującej i implementującej testy. Może to być wymagane ze względu na typ/zakres testu, np. dla testów bezpieczeństwa.

Przypadki i Kroki Testowe powinny być opisane zgodnie z szablonami obowiązującymi w CeZ. Rozpoczynając projektowanie testów dla kolejnych inicjatyw testowych, należy dokonywać przeglądu szablonów i Przypadków i Kroków Testowych z perspektywy specyfiki projektowanego testu oraz na podstawie wniosków z wcześniej realizowanych prac. Aktualizacje szablonu/ Przypadków i Kroków Testowych są możliwe w trakcie trwania całego projektu, jednak powinny być poprzedzone analizą wpływu zmian na realizowane prace.

3.2.2. Zapewnianie Środowisk, Narzędzi i Danych Testowych

Wymagania względem środowisk, narzędzi oraz danych testowych identyfikowane są w procesie Planowania testów. W procesie Zapewniania Środowisk, Narzędzi i Danych Testowych wymagania te są uszczegóławiane oraz uruchamiane są procedury, umożliwiające realizację zmodyfikowanych/nowych wymagań.

Aktywności realizowane w ramach procesu Zapewniania Środowisk, Narzędzi
i Danych Testowych:

- szczegółowe opisanie wymagań względem środowiska, danych i narzędzi potrzebnych do wykonania testów, w kontekście informacji pozyskanych w procesie implementacji testów,
- odebranie przygotowanego środowiska testowego oraz podejścia do jego utrzymania,
- odebranie bądź przygotowanie danych testowych w zależności od sposobu ich pozyskiwania:
    - import ze środowisk produkcyjnych lub innych zewnętrznych źródeł (plików, roboczych baz danych), a w razie potrzeby ich anonimizacja,
    - utworzenie z wykorzystaniem skryptów testów automatycznych z poziomu GUI,
    - utworzenie ręcznie z poziomu GUI,
- instalacja i konfiguracja narzędzi wsparcia testów.

3.2.3. Wykonanie Testu

Celem procesu Wykonanie testu jest realizacja zaprojektowanych na etapie Implementacji testu scenariuszy testowych na przygotowanych środowiskach testowych
i zapisanie wyników.

Aktywności podejmowane na tym etapie sprowadzają się do wypełnienia
w odpowiedniej kolejności poniższych czynności:

1. Wykonania – manualne bądź automatyczne – kroków opisanych w przypadkach testowych scenariuszy testowych.
2. Porównanie wyników przeprowadzonego testu z oczekiwanymi wartościami przypadków testowych oraz ustalenia rezultatu testu.
3. Zapisanie wykonania testów wraz z jego rezultatem w odpowiednim narzędziu wsparcia testowania.

Dane zgromadzone w ramach procesu Wykonanie Testu, są wykorzystywane
w procesie Monitorowania i Kontroli Realizacji Testów.

3.2.4. Raportowanie Błędu

Celem procesu Raportowania błędu jest powiadomienie Interesariuszy
(w usystematyzowany i spójny sposób) o zaistnieniu błędu oraz o potrzebie inicjalizacji prac nad jego rozwiązaniem. Za błąd powinno być uznane każde odchylenie/odstępstwo od oczekiwanych wyników przeprowadzanych przypadków testowych jak również inne niespodziewane zdarzenie, które mogło mieć miejsce w trakcie wykonywania testu.

Planując kolejne inicjatywy testowe, należy dokonywać przeglądu zidentyfikowanych błędów i szablonu defektu z perspektywy specyfiki planowanego testu oraz na podstawie wniosków z wcześniej realizowanych prac. Zmiany w podejściu do zarządzania błędami, w tym ich raportowania są możliwe, jednak każdorazowo
powinny być poprzedzone analizą wpływu zmian na realizowane prace. Z tego powodu rekomendowane jest wdrażanie zmian przed lub po zamknięciu inicjatywy testowej.

Raport błędu powinien być utworzony możliwie jak najszybciej od momentu wykrycia rozbieżności. Opóźnienie w zgłoszeniu błędu jest jednak rekomendowane, gdy istota błędu wymaga dłuższej analizy bądź wielokrotnej weryfikacji. Raport błędu można aktualizować – przy zachowaniu historii zmian – w następujących przypadkach:

- przekazania dodatkowych informacji o błędzie,
- odrzuceniu błędu,
- realizacji retestu.

Zalecana klasyfikacja błędów:

A	Blokujący	blocker	Blokuje wykonania dalszych testów systemu, aplikacji
B	wysoki	high	Znaczący problem, dotyczy kluczowych funkcji systemu, aplikacji. Jednak można kontynuować dalsze testy bądź możliwe jest obejście.
D	Średni	Medium	Średni problem, nie dotyczy kluczowych funkcji systemu, aplikacji, możliwe jest kontynuowanie dalszych testów-, możliwe jest wykonanie stosownego obejścia
E	Niski	Low	Niewielka usterka, minimalny wpływ na system.

3.2.5. Monitorowanie i Kontrola Realizacji Testów

Ustalenie mierników i zakresu ich stosowania jest niezbędne dla zrozumienia jakości badanego rozwiązania oraz efektywnego zarządzania realizacją testów. Zestaw
i charakter metryk może być specyficzny dla danej inicjatywy testowej.

Zaleca się rozważenie:

- metryk dotychczas wykorzystywanych w narzędziach CeZ,
- SLA uczestników procesu testowego, w szczególności czasu dostarczania poprawek dla błędów krytycznych.

Sposób raportowania metryk powinien być uzgodniony podczas planowania testów. Podejście do raportowania może różnić się w zależności od realizowanej inicjatywy testowej, jednak za każdym razem musi umożliwiać:

- ocenę postępu prac testowych względem założonego harmonogramu
- jednoznaczną identyfikację błędów blokujących postęp testów bądź krytycznych z perspektywy produkcyjnego użytkowania rozwiązania.

W celu ciągłej, codziennej kontroli nad realizacją testów rekomendowane jest stworzenie raportów generowanych automatycznie i dostępnych bezpośrednio w narzędziu
do zarządzania testami (np. przez odpowiednio skonfigurowane dashboard'y).

Monitorowanie statusu testów i zagrożeń dla realizacji prac całego etapu wdrożeniowego odbywa się za pomocą raportów generowanych w trakcie poszczególnych inicjatyw testowych.

3.3. Zakończenie Realizacji Testów

Po zakończeniu wykonywania testów danej inicjatywy testowej lub etapu wdrożeniowego należy wykonać następujące czynności:

- Końcowy rezultat testów powinien być zarejestrowany i zakomunikowany do interesariuszy, najlepiej w postaci Raportu Końcowego z testów.
- Produkty powstałe w trakcie procesu (Plany Testów, scenariusze testowe, skrypty testowe, narzędzia i dane testowe) należy zarchiwizować i zabezpieczyć do ewentualnego ponownego użycia.
- Środowisko i narzędzia testowe powinny być pozostawione w stanie umożliwiającym ich ponowne użycie dla kolejnych inicjatyw testowych/etapów wdrożeniowych.
- Przeanalizować projekt testowy i na podstawie wyciągniętych wniosków podjąć decyzje o ewentualnych usprawnieniach realizacji kolejnych inicjatyw testowych lub etapów wdrożeniowych.

4. Podejście do testów

4.1. Role i odpowiedzialności

W realizację zadań procesu testowego zaangażowany jest nie tylko zespół testowy ale również inni członkowie zespołu projektowego. Podział ról i obowiązków przedstawia macierz RACI, w zgodzie z poniższą definicją:

R – Realizuje. Rola odpowiedzialna za wykonanie zadania i rozliczana z realizacji zadań

A – Akceptuje. Rola odpowiedzialna za zatwierdzenie realizowanych zadań, nadzorująca .

W – Współpracuje. Rola wykonujące zadanie w zakresie własnej odpowiedzialności - współwykonawca

We – Weryfikacja. Rola weryfikująca, doradzająca, pełniąca rolę konsultanta.

I – Informowana. Rola informowana o danym zadaniu.

Tabela 1. Interfejsy Organizacyjne – Macierz odpowiedzialności

Działanie / czynność	Lider Testów	Tester	Kierownik projektu	Biznes = WB, użytkownicy	Utrzymanie / w tym Infrastruktura	Bezpieczeństwo	Dostawca oprogramowania / deweloper DevOps	Analityk Biznesowy	Uwagi
Pozyskanie zespołu do testów			R	W	W	W			
Przekazanie istniejących dokumentów w zakresie metodyki testów, takich jak: podejścia głównych udziałowców, procesów Zamawiającego, Polityk i Strategii do Lidera Testów	W*		R						przekazanie do Lidera testów zarówno w Centrum i/lub do Dostawcy
Przekazanie dokumentacji (harmonogram Projektu, dokumentacja Projektu, w zależności od fazy realizacji: specyfikacja wymagań, specyfikacja rozwiązania, dokument CR, instrukcje i procedury biznesowe, rejestr ryzyk) do Lidera Testów	We		R						przekazanie do Lidera testów zarówno w Centrum i/lub do Dostawcy
Identyfikacja testów do wykonania	R	W	W			W	W	W	
wybór technik testowania	R	W			W	W			
Przygotowanie, uzgodnienie i publikacja Planu Testów	R	W	We	W	W	W			obejmuje planowanie testów dla każdego wdrożenia /wydania osobno. Przed każdym kolejną grupą testów należy zweryfikować metody testowania. Czynność wielokrotna
Przygotowane i uzgodnienie scenariuszy	R	W		W*	W*	W*	W*	A	
Przekazanie wypełnionych scenariuszy, przypadków i kroków testowych	R	W	I	I				We	
Zebranie i przekazanie wymagań na Środowiska, Dane i Narzędzia testowe	R	W	We/I	W*	W	W			
Dostarczenie Środowisk	We	We	W		R				Obowiązkiem kierownika projektu jest wystąpić do Infrastruktury o dostarczenie środowisk
Przygotowanie Danych i Narzędzi testowych	R	W	W		W	W	W	W	Kierownik projektu uczestniczy w zakresie pozyskania wkładu od Dostawcy oprogramowania
Realizacja testu	W	R		W	W	W	W	w	realizacja testów zgodnie z przygotowanym planem obejmuje testy automatyczne - jeśli zidentyfikowane do wykonania w projekcie

Wsparcie merytoryczne testów.	W	W		W	W	W	W	w	wsparcie rozumiane jako dostarczenie dodatkowych informacji niezbędnych do realizacji testów
Przekazywanie Statusu Testów do kadry zarządzającej	R	W	I	W	W	W			Przekazywanie informacji o stopniu wykonania testów np. na bazie raportów z narzędzia. Cykl do ustalenia w projekcie.
Przekazanie Raportu Błędu do kadry zarządzającej	R	W	I	W	I/We	W	I/We		
Przygotowania i Przekazanie Raportu Końcowego z Testów	R	W	A	W		W			
potwierdzenie wyników testów przez biznes - testy UAT *	R			A					tylko w przypadku gdy testy UAT nie były wykonywane przez Biznes
Przygotowanie wniosków na przyszłość (Lessons Learned) - Retrospektywa	R	W	W	W*	W	W	W		wykonywane po zakończeniu wszystkich testów w projekcie oraz po zakończeniu testów dla konkretnego wdrożenia/wydania
Archiwizacja i zabezpieczanie produktów testów	R	W	I						archiwizacja planu testów, scenariuszy, raportów z testów, akceptacji wyników z testów repozytorium

4.2. Narzędzia testowe

Dla efektywnego zarządzania testami oraz problemami testowymi, należy – w miarę możliwości – na przestrzeni trwania całego projektu stosować te same narzędzia testowe. Repozytorium testów i defektów powinno być wspólne dla kolejnych faz i etapów projektu, oraz poszczególnych typów i poziomów testów. Jednocześnie produkty prac każdej
z inicjatyw testowych powinny być jednoznacznie opisane w narzędziu, umożliwiając ich łatwą i szybką identyfikację.

Realizując prace testowe należy zastosować narzędzia istniejące w CeZ. Jednocześnie, planując kolejne inicjatywy testowe, należy dokonywać przeglądu konfiguracji narzędzi testowych z perspektywy specyfiki planowanego testu oraz wniosków z wcześniej realizowanych prac. Aktualizacje konfiguracji narzędzia są możliwe w trakcie trwania całego projektu, jednak powinny być poprzedzone analizą wpływu zmian na realizowane prace. Z tego powodu rekomendowane jest wdrażanie zmian przed lub po zamknięciu inicjatywy testowej.

Ewentualna zmiana narzędzi testowych powinna być skonsultowana z Właścicielem Metodyki Testów – kierownik Wydziału Testów. W przypadku uzasadnionej potrzeby zmiany narzędzi, należy zadbać o to, by produkty prac jednej zamkniętej inicjatywy testowej były przechowywane w jednym narzędziu.

4.3. Podejście do dokumentacji testowej

W ramach każdej z inicjatyw testowych uruchamianych w projekcie powinien powstawać zestaw następujących dokumentów:

- Plan testów
- Scenariusze testowe
- Przypadki testowe
- Raporty – końcowy i cykliczne

Powyższe dokumenty mogą się różnić formą i strukturą, poziomem szczegółowości, w zależności od realizowanej inicjatywy testowej.

Proces powstawania i utrzymywania poszczególnych dokumentów testowych jest opisany w odpowiednich rozdziałach Metodyki Przeprowadzenia Testów. Powstałe dokumenty muszą być zgodne z Zasadami Zarządzania Konfiguracją w projekcie opisanymi w Dokumentacji Zarządzania Projektem (DZP).

Plan Testów oraz Raport Końcowy zawsze powinny stanowić odrębne dokumenty.
Gdy scenariusze testowe i przypadki testowe są ewidencjonowane w narzędziu do zarządzania testami, nie ma konieczności tworzenia odrębnych dokumentów zawierających tę samą treść.

Wszystkie dokumenty powstałe w wyniku procesu są dokumentami specjalistycznymi.

4.4. Automatyzacja testów

4.4.1. Cel i zakres automatyzacji testów

Celem automatyzacji testów przede wszystkim jest zwiększenie pokrycia testami budowanego systemu oraz skrócenie czasu trwania testów dla obszarów kluczowych
z biznesowego punktu widzenia. Dodatkowo automatyzacja testów pozwala odciążyć osoby wykonujące testy manualne dzięki temu zwiększana jest szczelność testów.
W procesie tworzenia skryptów testów automatycznych dla projektów rekomendowane jest zastosowanie techniki Behavior Driven Development oraz integracja testów automatycznych z programistycznym środowiskiem ciągłej integracji (Continuous Integration). W ramach automatyzacji testów funkcjonalnych przewidziane są następujące rodzaje testów:

- Smoke testy (testy dymne) – zestaw testów automatycznych przeznaczony do szybkiej i podstawowej weryfikacji kluczowych funkcjonalności systemu. Testy typu „Smoke" powinny być uruchamiane po każdej zmianie na środowisku testowym, takiej jak zmiana wersji systemu, zmiany konfiguracyjne, modyfikacje bazy danych etc. Status wykonania zestawu testów dymnych pozwala na szybką i podstawową ocenę jakościową zmian wprowadzonych na środowisku testowym. Dodatkowo wynik tych testów umożliwia podjęcie decyzji dotyczącej o ewentualnym wycofaniu wprowadzonych zmian.
- Testy regresji – zestaw testów automatycznych obejmujący ustalony zakres testów regresyjnych dla funkcjonalności, które nie podlegają zmianom w kolejnych wersjach systemu. Zaleca się uruchamianie zestawu testów zgodnie z wcześniej ustalonym harmonogramem (odpowiednim do cyklu wytwórczego) lub „na żądanie" w zależności od zaistniałych potrzeb. Automatyczne testy regresyjne obejmują szerszy zakres testów niż w przypadku testów dymnych („smoke"), ale nie muszą pokrywać całego zakresu testów regresyjnych przewidzianych w projekcie/etapie

4.4.2. Podejście do automatyzacji testów

Weryfikacją za pomocą testów automatycznych powinny zostać objęte wszystkie środowiska nieprodukcyjne. W pierwszej kolejności powinny powstać testy automatyczne dla zakresu „Smoke", następnie w miarę przyrostu wytworzonych funkcjonalności testy automatyczne dla zakresu regresji. Połączone testy regresji i „Smoke" powinny być jednym
z wyznaczników jakościowych dla decyzji o migracji kodu systemu na kolejne środowiska testowe.

4.4.3. Produkty specyficzne dla automatyzacji testów

W wyniku realizacji testów automatycznych – obok standardowych produktów procesu testowego – powstają następujące produkty:

- Środowisko Testów Automatycznych.
- Repozytorium Automatycznych Skryptów Testowych.
- Raport z testów automatycznych wraz z logami oraz np. zrzutami ekranów.
- Raport zgłoszonych i otwartych błędów.

5. Poziomy i typy testów

5.1. Testy systemowe

5.1.1. Cel i zakres testów systemowych

Celem testów systemowych jest weryfikacja systemu jako całości po zaimplementowaniu zmian opisanych wymaganiami systemowymi. Weryfikacja powinna uwzględniać zarówno sprawdzenie poprawności realizacji nowych wymagań jak również zachowanie systemu w obszarach niemodyfikowanych. Testy systemowe zwykle są najbardziej rozbudowanym poziomem testów, gdyż mogą obejmować następujące rodzaje testów:

- Testy funkcjonalne (w tym scenariusze negatywne)
- Testy wydajności (różnego rodzaju)
- Testy bezpieczeństwa
- Testy migracji
- Testy operacyjne
- Inne rodzaje testów uzasadnione wymaganiami stawianymi przed systemem.

Z racji potencjalnej złożoności tych testów, etap planowania testów systemowych,
w trakcie którego uzgadnia się zakres realizacji testów, jest szczególnie istotny.

5.1.2. Warunki realizacji testów

Podczas planowania testów systemowych, dla każdego systemu niezależnie, musi nastąpić ocena zasadności realizacji różnych rodzajów testów. Ocena ta odbywa się na podstawie dostępnych wymagań systemowych. Uzgodniony zakres testów obowiązkowo musi być udokumentowany w Planie Testów systemowych. W przypadku realizacji więcej niż jednego rodzaju testów, koniecznie jest zidentyfikowanie zależności między nimi w obszarze gotowości funkcjonalnej rozwiązania, wpływu implementacji poprawek do zgłoszonych błędów oraz dostępności i stabilności środowiska testowego.

Różne rodzaje testów wymagają specyficznych umiejętności technicznych, mogą więc wymagać utworzenia dodatkowych zespołów testowych, bądź przeprowadzenia adekwatnych szkoleń. Dlatego podczas tworzenia harmonogramu etapu wdrożeniowego, niezbędne jest zagwarantowanie odpowiedniego czasu na planowanie testów systemowych.

W projektach realizowanych przyrostowo (np. zgodnie z metodyką SCRUM), weryfikacja systemu powinna odbywać po zaimplementowaniu user stories oraz zaraz po dostarczeniu funkcjonalności, tak aby istotne uwagi i błędy były rozwiązywane w kolejnych iteracjach. Dlatego testowanie systemu może być realizowane jednocześnie przez:

1. Testera w ramach zespołu developerskiego – który eksploracyjnie testuje tworzoną historyjkę oraz weryfikuje zgodność zaimplementowanego rozwiązania z kryteriami akceptacji. Błędy zgłaszane przez testera powinny być rozwiązywane na bieżąco, co najmniej w zakresie umożliwiającym spełnienie kryteriów akceptacji danej user story. Testy eksploracyjne mogą odbywać się na środowisku developerskim, jednak potwierdzenie spełnienia kryteriów akceptacji musi być zrealizowane na środowisku zasilonym główną gałęzią kodu systemu.
2. Zespół testowy – który w usystematyzowany sposób weryfikuje funkcjonalności systemu zaraz po zakończeniu sprintu/wydania, w zakresie prezentowanym podczas demo oraz w zakresie regresji. Testy powinny być realizowane na środowisku testowym zasilonym wersją systemu z ostatniego zakończonego sprintu/wydania. Środowisko może być aktualizowane poprawkami dla błędów z perspektywy realizacji testów.

5.2. Testy akceptacyjne systemu

5.2.1. Cel i zakres testów akceptacyjnych

Testy akceptacyjne są przeprowadzane dla umożliwienia Odbiorcy (CeZ i użytkownik końcowy) oprogramowania zaakceptowania bądź odrzucenia dostarczonego rozwiązania. Ich celem jest potwierdzenie, że system realizuje postawione przed nim zadania, a wymagania zostały spełnione na uzgodnionym poziomie. Dlatego zakres testów powinien być formalnie uzgodniony pomiędzy dostawcą oprogramowania i odbiorcą oprogramowania.

5.2.2. Warunki realizacji testów akceptacyjnych

Testy akceptacyjne powinny być realizowane przez dedykowany zespół testowy, przy minimum współudziale albo wsparciu użytkownika końcowego rozwiązania bądź przedstawiciela biznesu, reprezentującego interesy użytkownika końcowego. Test należy realizować na stabilnym środowisku testowym, umożliwiającym realizację zaplanowanych scenariuszy.
By możliwe było rozpoczęcie testów akceptacyjnych, musi być zakończony development obszaru podlegającego testom, a wszelkie niezbędne aktualizacje powinny być objęte obowiązkowym monitorowaniem wersji oprogramowania.

5.2.3. Produkty specyficzne dla testów akceptacyjnych

W przypadku testów akceptacyjnych, Scenariusze realizowanych testów powinny być uzgodnione z użytkownikiem końcowym rozwiązania, bądź przedstawicielem biznesu reprezentującym interesy użytkownika końcowego. Wyniki testów powinny być potwierdzone przez Właściciela Biznesowego.

5.3. Testy regresji

5.3.1. Cel i zakres testów regresji

Celem testów regresji jest zweryfikowanie poprawności działania niemodyfikowanych części oprogramowania i sprawdzenie, czy system nie uległ degradacji (regresji) po wprowadzeniu zmian. Zakres testów regresji powinien być poprzedzony analizą wpływu zmian w systemach na istniejące funkcjonalności biznesowe. Musi także uwzględniać priorytety biznesowe procesów wspieranych przez weryfikowany system oraz ryzyka techniczne związane z budową rozwiązania.

5.3.2. Warunki realizacji testów regresji

Testy regresji mogą być wykonywane na każdym poziomie testów. Konieczność ich realizacji powinna być rozważana podczas planowania każdej inicjatywy testowej.
W zależności od poziomu i rodzaju testów, testy regresji mogą być uruchamiane:

- cyklicznie, w zgodzie z ustalonym harmonogramem
- na żądanie
- na zakończenie fazy testów

Ze względu na powtarzalność wykonywania scenariuszy testowych i możliwość ich wielokrotnego użycia na różnych poziomach testów, zakres testów regresji jest dobrym kandydatem do automatyzacji.

5.4. Testy End-to-End

5.4.1. Cel i zakres testów E2E

Celem testów „End to End" (E2E) jest sprawdzenie kompletnego procesu biznesowego. Realizuje się je zazwyczaj dla procesów przebiegających przez wiele systemów lub wymagających przetwarzania manualnego poza systemami informatycznymi.

5.4.2. Warunki realizacji testów

Zakres testów E2E powinien być uzgodniony z przedstawicielem biznesu, który musi określić priorytety procesów biznesowych podlegających weryfikacji. Testy E2E powinny odbywać się po zakończeniu testów systemowych oraz integracyjnych systemów wspierających weryfikowane procesy biznesowe. W zależności od charakteru weryfikowanych procesów, może być konieczne zapewnienie dodatkowych narzędzi
(np. drukarki, skanery, itp) bądź osób (np. kurier) dla realizacji kolejnych etapów testu. Potrzeby te powinny być zidentyfikowanie na etapie planowania testów a następnie uzgodnione z Kierownikiem Projektu.

5.5. Testy wydajnościowe

5.5.1. Cel i zakres testów wydajnościowych

Celem testów wydajnościowych jest weryfikacja spełnienia przez system stawianych przed nim wymagań wydajnościowych podczas obciążenia go odpowiednią liczbą wywołań (określoną w projekcie w grupie wymagań niefunkcjonalnych). Testy wydajnościowe mogą obejmować weryfikacje systemu pod kątem:

- wydajności podstawowej (Baseline Test),
- wydajności systemu przy zakładanym obciążeniu produkcyjnym (Load Test),
- obciążenia systemu przez dłuższy czas (Soak Test),
- maksymalnego dopuszczalnego obciążenia systemu przez jego zwiększanie, aż do momentu wystąpienia jego awarii (Stress Test),
- wydajności systemu w sytuacji konieczności obsługi dużych wolumenów danych lub operacji (Volume Test).

Zakres testów wydajnościowych oraz ich typ należy każdorazowo definiować na podczas przygotowania planu testów wydajnościowych dla kolejnych etapów wdrożeniowych projektu.

5.5.2. Warunki realizacji testów

Podczas planowania testów wydajnościowych etapu wdrożeniowego powinna nastąpić ocena potrzeby weryfikacji aspektów wydajności wymienionych w wymaganiach niefunkcjonalnych projektu. By taka ocena była możliwa i wiarygodna, dokumentacja projektowa oraz techniczna zakresu wdrażanego danym etapem powinna zostać udostępniona zespołowi testowemu.

Wszystkie cykle testów wydajnościowych należy realizować na środowisku akceptacyjnym, które najlepiej odzwierciedla środowisko produkcyjne pod względem wydajności i konfiguracji. Środowisko akceptacyjne powinno być dostępne dla realizacji testów wydajnościowych podczas każdego z etapów wdrożeniowych projektu.

W celu realizacji testów wydajnościowych niezbędne jest zapewnienie narzędzi umożliwiających przygotowanie automatycznych skryptów testowych oraz przygotowanie środowiska uruchomieniowego. Implementacja skryptów bazuje na odzwierciedleniu rzeczywistej interakcji użytkownika z systemem oraz żądań generowanych w ramach integracji pomiędzy systemami. W trakcie implementacji skrypty są odpowiednio parametryzowane dzięki czemu możliwa jest symulacja wykorzystania aplikacji poprzez określoną liczbę użytkowników.

Decyzje związane z wyborem narzędzi do automatyzacji testów wydajnościowych powinny być podjęte podczas planowania testów etapu wdrożeniowego.

Testy wydajnościowe należy wykonywać iteracyjnie zgodnie z uzgodnionym profilem ruchu. W ramach pojedynczego testu należy wykonać kilka uruchomień w celu osiągnięcia wyników referencyjnych. By test był wiarygodny, w czasie wykonywania testów na środowisku testowym nie mogą być wykonywane żadne operacje obciążające to środowisko, które nie zostały przewidziane profilu ruchu (np. inne testy, masowe przetwarzanie danych, prace administracyjne itp.).

W ramach realizacji testów wydajnościowych dla każdego z etapów projektu przewidywane są minimum 2 cykle testów wydajnościowych.

- Cykl 1 – weryfikacja pod kątem wydajności nowo wdrożonej na środowisko akceptacyjne funkcji systemu objętej koniecznością weryfikacji wydajności,
- Cykl 2 do n – weryfikacja wydajności po wdrożeniu poprawek lub zmian konfiguracji

5.5.3. Produkty specyficzne dla testów wydajnościowych

W wyniku realizacji testów wydajnościowych – obok standardowych produktów procesu testowego (plan testów, scenariusze i przypadki testowe, raporty błędów, raporty z postępu prac oraz raport końcowy) – powstają następujące produkty:

- Profil Ruchu, zawierający następujące informacje:
  - cele weryfikacji wydajności,
  - oczekiwane poziomy wydajności,
  - charakterystyki monitoringu środowiska (indykatory),
  - procentowy udział poszczególnych scenariuszy testowych w teście wydajnościowym
  - planowaną liczbę wirtualnych użytkowników wykonujących poszczególne skrypty testowe.
- Automatyczne skrypty testów wydajnościowych

- Środowisko uruchomieniowe, dane i narzędzia testowe
- Logi wykonania testów

5.6. Testy bezpieczeństwa

Testy bezpieczeństwa – przedstawiają rzeczywisty obraz bezpieczeństwa procesu, aplikacji bądź systemu i infrastruktury IT. Polegają one na możliwie całościowym spojrzeniu na problem bezpieczeństwa systemu oraz wykonaniu kontrolowanych prób ataku. Pozwalają na ocenę bezpieczeństwa poprzez symulację ataku prawdziwego włamywacza komputerowego lub złośliwego użytkownika sieci. W porównaniu ze standardowymi metodami ochrony – sprawdzają realne, a nie potencjalne zabezpieczenia systemu.

5.6.1. Cel testów

Głównym celem testów bezpieczeństwa jest identyfikacja podatności. Działania są ukierunkowane na identyfikację luk, a nie próby ich wykorzystania. W zgłoszeniu błędu opisuje się przykładowe scenariusze wykorzystania podatności. Jeśli scenariusz wykorzystania podatności stwarza ryzyko zakłócenia działania usługi, wówczas nie jest on wykonywany. Scenariusze, które nie wpływają na działanie usługi, są – jako przykładowe – wykonywane.

5.6.2. Zakres testów

Zakres testów bezpieczeństwa oraz ich typ należy każdorazowo definiować podczas przygotowania planu testów bezpieczeństwa dla kolejnych etapów wdrożeniowych projektu. Zgodnie z dobrymi praktykami przeprowadzania testów bezpieczeństwa rekomendowane jest przeprowadzenie następujących usług bezpieczeństwa.

5.6.2.1. Testy penetracyjne aplikacji

W zakresie testów penetracyjnych przyjęto metodykę black-box z elementami grey-box. Oznacza to podejście, w którym zespół testowy przystępując do testów ma poziom wiedzy o aplikacji na poziomie analogicznym jak inni jej użytkownicy.

Aplikacja powinna być przetestowana w oparciu o aktualne standardy OWASP (Open Web Application Security Project), a w szczególności o klasyfikację OWASP Top 10, OWASP ASVS (Application Security Verification Standard) oraz OWASP Testing Guide (zawierający najlepsze praktyki w zakresie testów bezpieczeństwa).

Zadaniem testera bezpieczeństwa jest wykonywanie działań analogicznych, jakie wykonać może potencjalny intruz. Działania testowe wykonywane są w oparciu o narzędzia i z wykorzystaniem technik hackerskich. Podczas testów wykorzystuje się zarówno ręczne jak i automatyczne techniki testowania.

5.6.2.2. Testy penetracyjne aplikacji mobilnych

W zakresie testów penetracyjnych aplikacji mobilnych przyjęto metodykę black-box z elementami grey-box. Oznacza to podejście, w którym zespół testowy przystępując do testów ma poziom wiedzy o aplikacji na poziomie analogicznym jak inni jej użytkownicy.

Testy penetracyjne aplikacji mobilnych wykonywane będą w oparciu o klasyfikację podatności i zagrożeń z listy TOP 10 Mobile Risks organizacji OWASP (Open Web Application Security Project).

W ramach testów bezpieczeństwa sprawdzone powinny zostać zarówno komponenty aplikacyjne i serwerowe (backend z którym połączona jest aplikacja).

Testy mogą być wykonywane z wykorzystaniem emulatorów oraz na fizycznych urządzeniach mobilnych.

5.6.2.3. Testy penetracyjne API

W testowaniu penetracyjnym mechanizmu API podejście do realizacji testów jest analogiczne do podejścia do testów penetracyjnych aplikacji, opisanych w rozdziale 5.6.2.1.

5.6.2.4. Testy penetracyjne infrastruktury

Podatności w systemie mogą wynikać zarówno z defektów aplikacji jak i defektów środowiska serwerowo-sieciowego aplikacji. W celu wykrycia i eliminacji luk
w infrastrukturze, należy zrealizować testy penetracyjne infrastruktury teleinformatycznej.

Skanowanie sieciowe, zarówno automatyczne, jak i ręczne, ma za zadanie wykrycie
i potwierdzenie występowania podatności bez dodatkowych prób przełamywania zabezpieczeń. Na życzenie i za zgodą Kierownika Projektu, bądź innej upoważnionej osoby, możliwe jest przeprowadzenie dodatkowej weryfikacji poprzez próbę wykorzystania wykrytych podatności do głębszej eksploracji środowiska. Tego typu techniki, ze względu na możliwość dokonania zniszczeń w badanym środowisku, powinny być stosowane z zachowaniem najwyższego stopnia ostrożności. Zalecane jest przeprowadzanie ich na środowisku nieprodukcyjnym.

Celem opisanego powyżej rozpoznania topologii sieci i serwerów jest stworzenie pełnego obrazu zasobów IT z perspektywy sieci teleinformatycznej. Zebrane informacje są niezbędne do weryfikacji możliwości przełamania zabezpieczeń lub znalezienia błędów konfiguracji.

5.6.2.5. Audyt bezpieczeństwa konfiguracji systemu

Audyt konfiguracji systemu jest przeprowadzany przez testera bezpieczeństwa za pomocą technik ręcznych oraz przy użyciu narzędzi automatycznych. Konkretna lista komponentów, dla których może mieć zastosowanie audyt konfiguracji jest każdorazowo przedmiotem analizy wstępnej. Jest to czynność specyficzna dla potrzeb danego projektu,
a zakres audytu konfiguracji jest mocno związany ze specyfiką konkretnego rozwiązania IT oraz wynika z kontekstu jego operacyjnego i biznesowego wykorzystania.

5.6.3. Warunki realizacji

Wszystkie wymienione w pkt. 5.6.2 Zakres testów, rodzaje usług bezpieczeństwa należy realizować na środowisku akceptacyjnym, które najlepiej odzwierciedla środowisko produkcyjne i zapewnia stabilną (nie podlegającą modyfikacjom) wersje badanego systemu IT.

Przeprowadzenie testów bezpieczeństwa wymaga udostępnienia stabilnego systemu będącego przedmiotem testów po adresie IP lub po nazwie domenowej (wraz z wymaganym zestawem danych testowych). Na czas przeprowadzania testów, testowane środowisko powinno zostać zamrożone. To znaczy, że żadne zmiany nie powinny być wdrażane. Przed rozpoczęciem testów bezpieczeństwa powinny zostać zakończone wszystkie inne testy, w szczególności testy funkcjonalne i wydajnościowe. Brak zachowania powyższych wymogów może skutkować uzyskaniem niewiarygodnych wyników testów.

Wymagane jest zapewnienie kont dla użytkowników o wszystkich przewidywanych profilach uprawnień dla każdej z aplikacji (w liczbie co najmniej dwa konta dla danej roli). Jeśli nie wszystkie testy interfejsów aplikacyjnych będą wymagały udostępnienia kont testowych, bądź Kierownik Projektu uzna, że system nie będzie testowany z ujęciem profili, wówczas zespół testów przeprowadzi testy bez posiadania użytkowników w systemie.

Na potrzeby przeprowadzenia testów penetracyjnych API wymagane jest udostępnienie przykładowych zapytań SOAP UI wraz z dokumentacją metod.

Na potrzeby przeprowadzenia audytu konfiguracji systemu konieczne jest udostępnienie pełnego dostępu do badanej infrastruktury wraz z dokumentacją opisującą architekturę.

W celu realizacji testów bezpieczeństwa niezbędne jest zapewnienie dla każdego rodzaju usługi adekwatnych narzędzi komercyjnych i opensource. Decyzje związane
z wyborem konkretnych narzędzi są podejmowane podczas planowania testów.

5.7. Testy/badania użyteczności

5.7.1. Cel i zakres testów/badań użyteczności

Testy użyteczności wykonuje się w celu weryfikacji efektywności i satysfakcji użytkowników, z jaką realizują wspierane przez system zadania. Zakres testów użyteczności może obejmować następujące aspekty:

- efektywność wykonywania zadań przez użytkownika,
- liczbę popełnianych błędów,
- łatwość nauczenia się obsługi systemu,
- łatwość zapamiętania jego obsługi,
- satysfakcję z użytkowania.

W celu oceny użyteczności wykorzystuje się trzy główne grupy działań:

- audyt ekspercki,
- testy z użytkownikami,
- analiza porównawcza rozwiązań z rynku.

Zakres testów użyteczności oraz narzędzia wykorzystywane do ich realizacji należy każdorazowo definiować na podczas przygotowania planu testów użyteczności dla kolejnych systemów i etapów wdrożeniowych projektu.

5.7.2. Warunki realizacji testów

Testy użyteczności systemu powinny być przeprowadzone w oparciu o wytyczne dotyczące ergonomii systemów. W wynikach testów powinny zostać zidentyfikowane obszary, gdzie możliwe jest rozszerzenie funkcjonalności systemu i poprawienie przejrzystości oraz funkcjonalności oczekiwanej systemu (UsabilityTesting). By test był wiarygodny, konieczne jest udostępnienie stabilnej wersji oprogramowania, w której zakończone zostały prace rozwojowe obszaru podlegającego testom użyteczności.

5.7.3. Produkty specyficzne dla testów użyteczności

W wyniku realizacji testów użyteczności – obok standardowych produktów procesu testowego – powstają następujące produkty:

- Plan Audytu eksperckiego
- Plan Testów z Użytkownikiem
- Heurystyki i listy kontrolne

5.8. Testy integracyjne

5.8.1. Cel i zakres testów integracyjnych

Celem testów integracyjnych jest wykrycie defektów i problemów w interfejsach
i interakcjach pomiędzy systemami. W zakresie testów integracyjnych znajdują się interfejsy systemu, które zostały opracowane, zmodyfikowane lub skonfigurowane dla potrzeb projektu. Funkcjonalności systemu, wykorzystujące dane wymieniane między systemami,
nie są przedmiotem testów integracyjnych.

5.8.2. Warunki realizacji testów integracyjnych

Wytypowanie interfejsów, które powinny zostać objęte testami integracyjnymi, będzie wymagało wcześniejszego dostarczenia dokumentacji projektowej i technicznej. Pozwoli to zidentyfikować nowe i zmodyfikowane interfejsy kwalifikujące się do weryfikacji.

Testy integracyjne powinny być realizowane na zintegrowanym środowisku testowym, umożliwiając weryfikację realnej komunikacji między systemami. Jeśli na środowisku testowym brakuje niektórych systemów, których obecność jest wymagana do przeprowadzenia testów integracyjnych, konieczne będzie opracowanie zaślepek realizujących potrzebne funkcjonalności tych systemów. W przypadku braku możliwości ich przygotowania lub skonfigurowania w zintegrowanym środowisku testowym, interfejsy wymagające tych zaślepek będą wyłączone z zakresu testów integracyjnych.

Na etapie przygotowania do testów wymagane też będzie dostarczenie dla każdego interfejsu, w zależności od jego typu:

- pliku WSDL i przykładowych plików XML,
- specyfikacji interfejsu REST oraz przykładowych plików JSON,
- specyfikacji formatu pliku (dla interfejsu plikowego).

5.9. Testy Trial run

5.9.1. Cel i zakres testów

Celem testów Trial Run jest zweryfikowanie poprawności działania krytycznych biznesowo części oprogramowania i sprawdzenie ich poprawnego działania po wprowadzeniu zmian. Zakres testów Trial Run powinien być ograniczony zarówno do krytycznych procesów/funkcjonalności biznesowych nie podlegających zmianie jak i krytycznych procesów/funkcjonalności wdrażanych w wydaniu.

5.9.2. Warunki realizacji testów Trial Run

Testy Trial są obligatoryjne dla systemów produkcyjnych (oddanych do użytkowania). Dla realizacji testów Trial Run muszą być zdefiniowane funkcjonalności systemu, bez działania których system nie będzie realizował krytycznych procesów biznesowych.
Testy Trial prowadzone są na środowisku produkcyjnym po wgraniu wersji oprogramowania na środowisko produkcyjne, a przed oddaniem oprogramowania do użytkowania i są integralną części procedury dopuszczenia systemu do użytkowania.

Ze względu na powtarzalność wykonywania scenariuszy testowych i możliwość ich wielokrotnego użycia zakres testów Trial Run jest dobrym kandydatem do automatyzacji.

5.9.3. Produkty specyficzne dla testów

Raport z testów jest podstawą do podjęcia decyzji do uruchomienia produkcyjnego systemu.

6. Procesy Wspierające

6.1. Zarządzanie Ryzykiem

Testowanie jest etapem procesu wytwarzania oprogramowania szczególnie podatnym na występowanie ryzyk projektowych i produktowych. Ryzyka te mogą mieć wpływ zarówno na realizację testów z zachowaniem odpowiedniego standardu, jak również na ostateczną jakość oprogramowania, a w konsekwencji na niezawodność produktu w trakcie jego użytkowania. Dlatego podczas realizacji testów, identyfikacja ryzyk następuje wielokrotnie:

1. Podczas planowania testów – identyfikowane są głównie ryzyka dla przygotowania i realizacji testów, bądź niemożności utrzymania odpowiedniego standardu prac.
2. Podczas wykonania testów – identyfikowane są głównie ryzyka związane z utrudnieniami bądź niemożnością realizacji testów poszczególnych obszarów oraz zagrożenia niedotrzymania terminu zakończenia testów.
3. Podczas podsumowania testów – identyfikowane są ryzyka związane z produkcyjnym wykorzystaniem przetestowanego rozwiązania.

Wszelkie ryzyka zidentyfikowane w trakcie wyżej wymienionych prac są obsługiwanie zgodnie ze strategią zarządzania ryzykiem w projekcie.

6.2. Zarządzanie Błędami

Realizując prace testowe w ramach projektu należy wypracować metodę usuwania błędów w projekcie.

Planując kolejne inicjatywy testowe, należy dokonywać przeglądu podejścia do zarządzania błędami i rozwiązywania problemów (problem solving) z perspektywy specyfiki planowanego testu oraz na podstawie wniosków z wcześniej realizowanych prac. Aktualizacja podejścia do zarządzania błędami jest możliwa w trakcie trwania całego projektu, jednak powinna być poprzedzona analizą wpływu zmian na realizowane prace. Z tego powodu rekomendowane jest wdrażanie zmian przed lub po zamknięciu inicjatywy testowej.

6.3. Zarządzanie i Utrzymanie Środowiska Testowego

Środowisko testowe powinno być skonfigurowane tak, by spełniało wymagania stawiane przez cel realizowanych na tym środowisku testów. Po konfiguracji, a następnie odbiorze środowiska testowego, istotne jest skoordynowane i usystematyzowane podejście do jego utrzymania i aktualizacji. By proces dostarczania aktualizacji oprogramowania był możliwie szybki, a jednocześnie jak najmniej ingerujący w proces wykonania testów oraz zapewniał utrzymanie jakości wcześniej zweryfikowanych obszarów i kontrolę działań realizowanych w środowisku, rekomendowane jest wdrożenie narzędzi DevOps, wspierających w zautomatyzowany sposób deployment oprogramowania. Dostępny jest szeroki wybór narzędzi, zarówno komercyjnych jak i open source.

Wśród narzędzi typu open source wiodącym rozwiązaniem jest Jenkins, który wspiera proces automatycznego budowania aplikacji z kodu źródłowego oraz proces jej testowania, dostarczania i wgrywania na wskazane w konfiguracji środowiska informatyczne. Dzięki bardzo dobrze rozwiniętej społeczności oraz znacznej ilości wtyczek, możliwe jest dostosowanie serwera ciągłej integracji Jenkins do niemalże każdego projektu opartego o ciągłą integrację (Continuous Integration). Zaletami rozwiązania są:

- Łatwa instalacja,
- Łatwa konfiguracja,
- Bogaty oferta wtyczek i rozszerzeń,
- Możliwość pisania własnych rozszerzeń,
- Licencja open source.

Wartym rozważenia narzędziem komercyjnym jest IBM UrbanCoe. Umożliwia on automatyzację wydań na wielu środowiskach. Głównym zadaniem IBM UrbanCode jest wspieranie natychmiastowego i częstego wdrażania nowych wersji oprogramowania poprzez proces ciągłej integracji przy jednoczesnym zapewnieniu audytowalności, wersjonowania. IBM UrbanCode umożliwia:

- automatyzację wydań z możliwością natychmiastowego przywrócenia poprzedniej wersji (rollback),
- automatyzacja dostaw oprogramowania na środowiska chmurowe,
- orkiestracja zmian między różnymi serwerami i komponentami,
- pracę na wielu środowiskach z uwzględnieniem różnic w konfiguracji i regułach bezpieczeństwa,
- wizualizację - co jest dostarczone? Na jakim środowisku? Przez kogo ?
- integrację z oprogramowaniem pośredniczącym (middleware).

7. Słowniki i definicje

Tabela 2. Słownik i definicje

Terminologia	Objaśnienie
Automatyzacja testów	Użycie oprogramowania do wykonania lub wspierania czynności testowych.
Błąd	Działanie oprogramowania niezgodne ze specyfikacją Przypadku / Kroku Testowego.
CeZ	Centrum e-Zdrowia
Implementacja Testu	Implementacja scenariuszy i przypadków testowych w narzędziu testowym.
Inicjatywa testowa	Zestaw zidentyfikowanych testów do wykonania dla wdrażanego zakresu. W przypadku projektu realizowanego metodą przyrostową inicjatywę testową definiuje się dla każdego wydania osobno.
Interesariusz	Podmioty (osoba, instytucja, organizacja, urząd), który może wpływać na projekt CeZ i/lub pozostaje pod jego wpływem.
Krok testowy	Opis konkretnej czynności do wykonania w ramach Przypadku Testowego.
Monitorowanie i Kontrola Realizacji Testów	Weryfikacja przeprowadzonych testów

Narzędzia testowe	Oprogramowanie wspierające czynności testowe.
Plan Testów	Dokument definiujący podejście do testów w danym projekcie.
Podstawa testów	Wszystkie dokumenty, z których można wywnioskować wymagania dla systemu.
Przypadek testowy	Sekwencja czynności do wykonania z użyciem przedmiotu testów.
Ryzyko	Czynnik, który w przyszłości może skutkować negatywnymi lub pozytywnymi konsekwencjami.
Ryzyko produktowe	Ryzyko związane bezpośrednio z przedmiotem testów.
Ryzyko projektowe	Ryzyko związane z zarządzaniem i kontrolą.
Scenariusz testowy	Dokument określający ciąg akcji umożliwiający wykonanie testu.
SCRUM	Metodyka zwinna do wytwarzania oprogramowania.
SLA, Service Level Agreement	Umowa między klientem a usługodawcą dotycząca poziomu jakości usług.
Sprint	Przedział czasowy, w którym pracuje zespół deweloperski, w celu dostarczenia kolejnej działającej wersji produktu.
Środowisko testowe	Środowisko wykorzystywane do przeprowadzenia testów.
Testy akceptacyjne	Testy formalne, przeprowadzane w celu umożliwienia użytkownikowi, klientowi lub innemu uprawnionemu podmiotowi zaakceptowanie bądź odrzucenie dostarczonego rozwiązania.
Testy bezpieczeństwa	Testy sprawdzające skuteczność zabezpieczeń systemu.
Testy End to End, E2E	Testy weryfikujące pełne procesy biznesowe.
Testy integracyjne	Testy sprawdzające interfejsy pomiędzy modułami, interakcje z innymi częściami systemu oraz interfejsy pomiędzy systemami.
Testy regresji	Testy sprawdzające, czy system nie uległ degradacji (regresji) po wprowadzeniu zmian.
Testy systemowe	Testy zajmujące się zachowaniem systemu/produktu.
Testy użyteczności	Testy użyteczności wykonuje się w celu weryfikacji wydajności i satysfakcji użytkowników, z jaką realizują wspierane przez system zadania.
Testy wydajnościowe	Testy sprawdzające, jak „szybko" działa system oraz jakie obciążenie jest on w stanie stabilnie obsłużyć.
Wykonanie Testu	Realizacja zaprojektowanego na etapie Implementacji testu scenariusza testowego na przygotowanym środowisku testowym i zapisanie jego wyniku.

Role i odpowiedzialności w procesie testowym

Tester	· Przygotowuje scenariusze, przypadki i dane testowe. · Wykonuje i dokumentuje przebieg testów. · Analizuje poprawność działania aplikacji poprzez realizację zakresu testów. · Zgłasza defekty, · wykonuje retesty poprawek dla zgłoszonych defektów. · Raportuje do Lidera Testów postęp z realizacji prac. Ma praktyczną znajomość w wykorzystywaniu narzędzi wsparcia testów. Posiada podstawową wiedzę i doświadczenie w używaniu języków skryptowych. Zależnie od specjalizacji może badać poprawność w zakresie funkcjonalności, integracji systemów, wykonywać automatyczne skrypty testów funkcjonalnych.
Starszy Tester	· Przygotowuje scenariusze, przypadki i dane testowe. · Wykonuje i dokumentuje przebieg testów. · Analizuje poprawność działania aplikacji poprzez realizację zakresu testów. · Zgłasza defekty, wykonuje retesty poprawek dla zgłoszonych defektów. · Raportuje do Koordynatora Testów postęp z realizacji prac. · Ma praktyczną znajomość w wykorzystywaniu narzędzi wsparcia testów. · Zależnie od specjalizacji może badać poprawność w zakresie funkcjonalności, integracji systemów, wykonywać automatyczne skrypty testów funkcjonalnych, ma doświadczenie w używaniu języków skryptowych, posiada sporą wiedzę w dziedzinie biznesowej.
Analityk Testów / Specjalista Testów	· Posiada kompetencje i doświadczenie praktyczne w analizie i weryfikacji dokumentacji technicznej projektu oraz dokumentacji wymagań biznesowych (np. ABT). · Współpracuje z Liderem Testów w opracowywaniu planów oraz harmonogramów testów. · Opracowuje podejście i zakres testów - definiuje scenariusze i przypadki testowe oraz wymagania do ich realizacji. · Zna techniki projektowania testów funkcjonalnych i niefunkcjonalnych, wspiera analitycznie proces testowy. · Wybiera optymalne środki do osiągania celów.
Koordynator Testów	· Ma wiedzę i doświadczenie w zakresie metodyki testów, wsparcia procesu testowego. · Koordynuje prace zespołu testowego. · Bierze aktywny udział w procesie rozwiązywania problemów testowych. · Posiada kompetencje i doświadczenie praktyczne w opracowywaniu planów oraz harmonogramów testów. · Wspiera proces identyfikacji, analizy i zarządzania ryzykami w obszarze testów.
Lider Testów	· Ma wiedzę i doświadczenie w zakresie metodyki testów, wsparcia procesu testowego. · Organizuje zespół testowy oraz zarządza jego pracą. · Zarządza procesem rozwiązywania problemów testowych. · Posiada kompetencje i doświadczenie praktyczne w opracowywaniu polityki, strategii, planów oraz harmonogramów testów. · Analizuje ryzyka i zarządza nimi w obszarze testów.

Specjalista Testów Bezpieczeństwa	· Zna narzędzia do realizacji testów bezpieczeństwa. · Ma wiedzę o różnych technologiach tworzenia aplikacji. · Potrafi posługiwać się skryptowymi językami programowania. · Posiada minimum 1 roczną praktykę w realizacji testów bezpieczeństwa. · Zna standardy testowania bezpieczeństwa IT (m.in. OWASP).
Ekspert Testów Bezpieczeństwa	· Posiada szeroką wiedzę z zakresu projektowania i implementacji zabezpieczeń w zasobach IT. Zna bardzo dobrze narzędzia do realizacji testów bezpieczeństwa. Na poziomie eksperckim potrafi przeprowadzić analizę ryzyka i opracować rekomendacje poprawy bezpieczeństwa systemów IT. Zna i biegle wykorzystuje standardy testowania bezpieczeństwa IT (m.in. OWASP).
Audytor IT	· Posiada kompetencje i praktyczne doświadczenie w audytowaniu systemów informatycznych. Zna i potrafi wykorzystać w praktyce standardy zarządzania IT (t.j. COBIT), normy zarządzania systemami bezpieczeństwa informacji (z rodziny ISO 27000), zarządzania usługami (ISO 20000), inne normy z zakresu inżynierii oprogramowania.
Inżynier Automatyzacji Testów	· Zna narzędzia do automatyzacji testów. · Ma praktykę w automatyzacji testów. · Potrafi posługiwać się skryptowymi językami programowania. · Przygotowuje w zautomatyzowany sposób scenariusze, przypadki i dane testowe. · Wykonuje i dokumentuje przebieg testów. · Analizuje poprawność działania aplikacji poprzez realizację zakresu testów. · Ma praktyczną znajomość w wykorzystywaniu narzędzi wsparcia testów oraz w zawansowanym tworzeniu skryptów z wykorzystaniem technik programistycznych. · Zależnie od specjalizacji może wykonywać automatyzację testów funkcjonalnych oraz badanie wydajności systemów.
Ekspert Testów Technicznych	· Posiada ekspercką wiedzę i wieloletnie doświadczenie w zakresie metodyki prowadzenia testów automatycznych, badania wydajności oraz wdrażania narzędzi wsparcia procesu testowego. · Potrafi organizować pracę zespołu oraz rozwiązywać wszelkie pojawiające się kwestie techniczne. · Zna narzędzia do automatyzacji testów, badania wydajności oraz narzędzia w spierające testy na poziomie umożliwiającym samodzielne realizowanie projektów, wdrażanie tych narzędzi w organizacji od podstaw oraz w zakresie eksperckim umożliwiającym prowadzenie zaawansowanych szkoleń. · Posiada bardzo dobrą, popartą praktyką wiedzę z zakresu wytwarzania oprogramowania z użyciem różnych języków oprogramowania.

8. Indeks tabel i rysunków

Spis tabel

Metryka			
Właściciel	CeZ		
Wersja	2	Status dokumentu	

Historia zmian			
Data zmiany	Wersja	Autor zmiany	Opis wprowadzonej w dokumencie zmiany
2020 09	2		Dostosowanie zapisów metodyki celem objęcia zakresem całego Centrum e-Zdrowia