CZĘSTO ZADAWANE PYTANIA I POPEŁNIANE BŁĘDY

Ogólne zagadnienia dla całej książki

1. Proszę spróbować tak odpowiadać, aby każda odpowiedź miała strukturę raportu gotowego do przekazania klientowi albo projektowi.
   - Raporty tekstowe czyta się ciężko, więc trzeba próbować je uatrakcyjniać, stosując na przykład formę tabelaryczną, dodając zrzuty ekranów z opisem tekstowym, dodając strzałkę wskazującą defekt itp.
   - Raport powinien mieć czytelną strukturę – pomocne mogą być nagłówki i sekcje, które pokazują, jak są zaadresowane poszczególne polecenia zadania.
2. Proszę dbać o jakość języka i wystrzegać się błędów. Wszyscy mamy z tym problem, ale w pracy jest ważne, aby nie popełniać prostych błędów, szczególnie takich, których nie wychwyci autokorekta, na przykład „po przez" zamiast „poprzez" czy „było by" zamiast „byłoby".
3. Proszę też dbać o dobrą składnię, co zwykle sprowadza się do „pisz prosto". Unikaj długich zdań, wielokrotnie złożonych czy nawiasów większych niż treść zdania.
4. Raporty testerskie powinny być możliwie jak najbardziej pozbawione emocji i obiektywne. Jest to szczególnie trudne na przykład w przypadku testowania użyteczności, które jest mocno subiektywne.
5. Stwierdzenia typu „nie podoba mi się" mogą być postrzegane jako kontrowersyjne i konfrontacyjne. Proszę pamiętać, że to nie tester zamawia oprogramowanie, a programista może po prostu powiedzieć –

„ma się podobać klientowi". Dodatkowe akcentowanie subiektywnych opinii jest zachętą do polemiki – na przykład zamiast „bardzo dokuczliwe" wystarczy napisać „dokuczliwe". Lepszą formą jest próba poczucia się użytkownikiem i wypowiadania się w jego imieniu, na przykład „użytkownikom może się nie podobać" lub „użytkownicy mogą to uznać za dokuczliwe". Nie zmienia to faktu, że jest to ciągle pewna opinia, ale jej forma zachęca do rozważenia, czego potrzebuje nasz końcowy odbiorca.

6. Skąd mam wiedzieć, czy to co znalazłem, jest defektem, czy nie?
   To bardzo ważny temat dla całej książki. Jeśli jesteśmy przekonani, że coś jest defektem, musimy go zaraportować, a potem często również bronić przed zamknięciem bez rozwiązania. Tester powinien być przekonany, że to, co znalazł jest defektem i musi znaleźć dowody na to, że ma rację. Podstawową informację o zachowaniu oprogramowania daje specyfikacja wymagań. W wielu przypadkach trzeba sięgać do innych źródeł, jak np. zachowanie podobnych aplikacji czy standardów wskazujących, jak w danych okolicznościach powinno zareagować oprogramowanie. Dobry tester musi wskazać źródła, które potwierdzają tezę o problemie. Jeśli takich nie ma, można powołać się na swoje doświadczenie (jeśli się je ma).
7. W pracy często posługujemy się różnego rodzaju analizatorami, które zwracają nam informacje na wysokim poziomie ogólności, na przykład:
   – „wydajność aplikacji to 20/100",
   – „aplikacja jest niebezpieczna",
   – „użyteczność strony jest na poziomie – słaba".
   Sama informacja z takiego narzędzia nie jest podstawą do stworzenia raportu defektu. Kroki reprodukcji nie mogą się sprowadzać do „uruchom analizator i sprawdź negatywny wynik". Jest to tylko wstęp do poważniejszej analizy, która może nam pokazać prawdziwe problemy. Przykładowo, czas ładowania się strony może zależeć od wielu czynników, jak na przykład brak zapisu danych do pamięci podręcznej (ang. *cache*) czy brak kompresji obrazów. Dobry raport problemów powinien zawierać szczegóły wyników pracy analizatora.
8. Powielanie zbędnych informacji w raporcie często zamazuje jego obraz. Dodanie nadmiarowych elementów lub długich opisów niewiele wnosi. Co więcej, może generować dodatkowe pytania.

9. Stosowanie metody kopiuj-wklej zawsze będzie generowało dużo treści, należy jednak zastanowić się nad jej wartością. Raporty muszą być tworzone inteligentnie, a to znaczy, że nie mogą być ani zbyt długie, ani zbyt krótkie. Szukamy w nich skrótów i uproszczeń. Często jednak lepsza jest jedna rozbudowana tabelka niż dziesięć, do których trzeba przewijać strony, aby zestawić sobie dane.
10. Czasami możecie nie znać odpowiedzi na pytanie postawione w tej książce. Wtedy nie bójcie się poprosić o pomoc, przyznając, że nie macie wiedzy lub umiejętności, aby rozwiązać dane zadanie. Pomocą mogą służyć fora lub bardziej doświadczeni testerzy.
11. W raportach należy unikać kolokwializmów i sformułowań potocznych. Ich pojawienie się może obniżać ocenę profesjonalizmu autora.
12. W raportach należy unikać rozbudowanych historii, narracji i kwiecistych opisów. Język raportu musi być prosty, zwięzły.
13. Obszar testowania nie jest zbyt ustandaryzowany i definicje nie będą miały większego znaczenia, jeśli tylko nie będziemy nadpisywać powszechnie przyjętych reguł. Przykładowo – „scenariusz testowy" prawie w każdym źródle ma trochę inną definicję. Są jednak miejsca, gdzie nie warto nadpisywać przyjętego nazewnictwa – „test negatywny" nie oznacza przypadku testowego zakończonego niepowodzeniem, a użycie w testach danych, które powinny zostać odrzucone lub nie powinny być w aplikacji przetworzone.