6.6. Trudne aspekty pracy testera

Z testowaniem oprogramowania wiąże się wiele przyjemnych elementów, ale może ono być również ciężkim kawałkiem chleba. Właśnie trudne aspekty podnoszone są przez przeciwników osobnej roli testera lub przez osoby, które chcą deprecjonować rolę testów w organizacji. We współczesnych firmach takich trolli antytesterskich jest coraz mniej, ale nie znaczy to, że na nich nie traficie. Część z nich zostało jedynie zagłuszonych przez poprawność polityczną i czeka na swoją okazję, aby wypunktować testerów za brak skuteczności lub jakościowo słabą pracę. W mojej opinii najgorsze, co może spotkać testera w organizacji, to:

- Próba testowania wszystkiego i zawsze, bez względu na zakres zmian w wersji czy bez względu na ograniczenia czasowe.
- Próba ślepego klikania oprogramowania, które samo z siebie nie pozwala nam zrozumieć logiki swojego działania albo po prostu mamy na to za niskie kompetencje.
- Testowanie w zmieniającym się środowisku, gdzie najgorszym przypadkiem jest testowanie czegoś, co właśnie jest modyfikowane, np. przez programistów.
- Wielokrotne testowanie tego samego bez widocznego celu.
- Praca w sytuacji stresowej, gdy próbuje się nam wmówić, że wszystko zależy od nas, a w dodatku niepowodzenie będzie naszą winą.
- Praca w atmosferze obwiniania testera za niską jakość.

Uda się uniknąć przez dłuższy czas złych zadań testerskich, jeśli trafimy na dobrego lidera. Większość problemów testerskich rozwiązuje się po prostu dobrym zarządzaniem. Dobry kierownik testów będzie chronił swoich ludzi przed wykonywaniem „małpiej" (*monkey testing)* czy monotonnej pracy. Przykładowo, minimalizacja nakładu pracy na regresję jest wynikiem dobrej współpracy testera z programistą. Współpraca ta jest jednak zazwyczaj wbudowana w organizację jako proces lub jako szkielet wzajemnego zaufania.

Wrogiem testera oprogramowania może też być nuda lub nudne zadania. Może to wynikać ze specyfiki pracy testerskiej oraz słabego zarządzania zadaniami. Konsekwencje nudy bywają krytyczne zarówno dla testerów, jak i dla organizacji. Na przykład może ona prowadzić do wypalenia zawodowego lub naturalnego zmęczenia powtarzalnymi zadaniami. Problem nudy

jest mocno powiązany z testowaniem regresji czy też z manualnym testowaniem. Każda organizacja dbająca o motywowanie swoich pracowników powinna dostarczać narzędzia, które są w stanie nudne i powtarzalne czynności automatyzować albo pomóc w ich zredukowaniu. Możliwym rozwiązaniem jest też rotacja zadań wśród większej grupy pracowników.

Nuda jednak może być również wynikiem złego dopasowania zadania do naszych kompetencji. Istnieje ryzyko pojawienia się znudzenia u osób o przeciętnych kompetencjach i dostających łatwe zadania. Rozwiązaniem jest więc dostarczanie zadań dopasowanych do poziomu wiedzy, doświadczenia i kompetencji.

Spotkałem się z przypadkami wymuszania na testerach ręcznego uruchomienia każdego przypadku testowego na każdej nowej wersji oprogramowania. Jest to naturalna droga do zabicia kreatywności i doprowadzenia do zmęczenia pracą testera. Wykonywanie tych samych przypadków testowych bez odpowiedniego uzasadnienia ich uruchomienia może przypominać pracę Charliego Chaplina w filmie „Dzisiejsze czasy". Ciągłe przykręcanie jednej nakrętki na taśmie, przez którą przewijają się ich tysiące. Jeden popełniony błąd i cała ta misterna (lub szatańska) maszyneria przestaje poprawnie działać. Czy aby na pewno wykonanie każdego przypadku testowego jest złe? Uważam, że tak, ponieważ sygnalizuje poważne problemy organizacyjne. Przykładowo, jesteśmy zmuszani do uruchomienia wszystkich przypadków, ponieważ kierownictwo nie wie, co znajduje się w poszczególnych wersjach oprogramowania. Wykonanie przypadków staje się więc dla niego potwierdzeniem, czy coś w nim jest, czy też nie. Aby tego uniknąć, należy już na poziomie planowania wersji definiować, co ma się znaleźć w oprogramowaniu, a później w notach wydania (*release notes*) opisywać, co rzeczywiście zostało do oprogramowania wprowadzone. Czasami jesteśmy zmuszani do uruchomienia wszystkich przypadków, ponieważ kierownictwo nie potrafi odpowiedzieć na pytanie, jakie są powiązania wewnątrz oprogramowania. Aby zapewnić, że drobna poprawka nie uszkodzi (nie wpłynie na działanie) jakiegoś istotnego obszaru aplikacji, testujemy wszystko. Wtedy kluczem do rozwiązania jest analiza wpływu realizowana przez wszystkich członków zespołu, od programisty przez architekta, aż po klienta, dzięki której oceniamy, jaki wpływ na poprawność działania będzie miała dana poprawka. Natomiast testerzy muszą opracować zestaw testów regresywnych (różny od zestawu wszystkich przypadków testowych) uruchamianych na każdej nowej wersji. Zestaw ten warto zautomatyzować. W niektórych