Aleksandra **Kunysz**

Helion

Kierunek jakość

Jak unikać błędów w projekcie

Redaktor prowadzący: Szymon Szwajger
Projekt okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki
Zdjęcie na okładce za zgodą Shutterstock.com

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/jaktob

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-7202-3

Printed in Poland

Spis treści

2. Zacznijmy od podstaw

Znamy przypadki projektów, które się nie udały. Znamy też liczne wyjścia na produkcję, które skończyły się wycofaniem zmian lub naprawianiem błędów na szybko. To, co zawodzi w większości przypadków, to nie maszyny, technologie, frameworki czy biblioteki. To my, ludzie. Ludzie, którzy nie przyznają się do błędu. Ludzie, którzy nie proszą o pomoc. Ludzie, którzy nie dzielą się informacją zwrotną. Gdybyśmy szczerze i bez zbędnych emocji potrafili się dogadać, udałoby się uniknąć wielu błędów i wszystko przebiegałoby sprawniej. Dlatego rozważania o jakości chcę zacząć od komunikacji.

Jest wiele problemów, które powtarzają się w wielu projektach. Zbyt często dajemy się ponieść swojej wizji. Zbyt często ignorujemy zdanie innych. Zbyt często zabijamy nawzajem swój entuzjazm. Niestety nie uczą nas w szkole ani na studiach, jak dobrze współpracować. W branży IT ciągle najbardziej kuleją umiejętności miękkie. Nie dlatego, że nie

mamy do nich predyspozycji, ale dlatego, że gloryfikujemy umiejętności techniczne, zapominając o tym, że większość pracy wykonujemy jako członkowie zespołu.

2.1. Wybujałe ego

Pracując w IT, mamy czasem wrażenie, że jesteśmy superbohaterami. Firmy się o nas biją, a z każdej strony słychać potwierdzenia tego, jak ważna jest nasza praca. Dodatkowo staramy się owiać swoje obowiązki wątkiem tajemnicy, żeby „zwykli śmiertelnicy" jak najlepiej o nas myśleli. Często uważamy się za lepszych, a prawda jest taka, że nasza branża przeżywa renesans i rynek pracownika za bardzo nas rozpieścił. Nam jednak ciągle gdzieś w głowie krąży myśl, że zajmujemy się magią. Skoro już jesteśmy tymi mądrymi czarownikami, którzy posiadają supermoce, to jak możemy się mylić? Jak możemy czegoś nie wiedzieć? To oczywiście nie dotyczy wszystkich, ale zbyt często widziałam sytuacje, w których pewnych problemów dałoby się łatwo uniknąć, gdyby ktoś poprosił o pomoc. To dobre i ambitne, że chcemy sobie ze wszystkim radzić sami, ale czasem mamy po prostu za mało wiedzy. Jest to zupełnie naturalne, bo branża rozwija się tak szybko, że ciężko jest nadążyć za wszystkim.

Skutkiem ubocznym działania wybujałego ego są też wojny. Panuje powszechna opinia, że testerzy nie mogą się dogadać z programistami. Jeszcze częściej zdarza się, że zespół techniczny nie lubi się z „biznesem" i wzajemnie uważają się za niekompetentnych. Wystarczy jedno ziarenko wypuszczone przypadkiem, żeby ruszyła lawina, która obrzuci błotem cały zespół i przy okazji produkt. Potem trudno tę lawinę zatrzymać, bo wszyscy mają już rany, o których ciężko zapomnieć. Nadszarpnięte zaufanie ciężko się zabliźnia. Łatwiej na długie tygodnie

obrazić się na kogoś niż wyciągnąć rękę na zgodę. Im dłużej trwa ten impas, tym trudniej jest zrobić pierwszy krok w stronę rozejmu. To niestety dzieje się bardzo często.

Kiedy występuję na spotkaniach dla programistów, zawsze słyszę jakąś historię, w której ktoś inny jest winien słabej jakości produktu. Na wydarzeniach dla testerów uczestnicy trochę się przy mnie hamują, ale też czasem opowiadają, czego to programiści u nich w projekcie nie spartaczyli. Jeśli zaczynamy myśleć w kategoriach „czyja to jest wina", ciężko nam będzie z tego kiedykolwiek wyjść, zwłaszcza jeśli kilka osób wspólnie się nakręca.

Ego może być cichym, ale potężnym wrogiem na drodze ku jakości. Potrafi być podwaliną słabego kodu, dziurawego procesu *Continuous Integration* czy kiepsko opisanych wymagań. Najgorsze jest to, że ciężko się wyszukuje źródła problemów spowodowanych przez zbyt wybujałe wyobrażenie o sobie i swojej roli w projekcie. Techniką, której często używam, kiedy czuję wewnętrzny opór przed zrobieniem czegoś, jest zadawanie sobie pytań. Jeśli coś może być korzystne dla projektu, ale z jakiegoś powodu nie chcę tego zrobić, to zadaję pytanie: „Co najgorszego może się stać?" albo „Co na tym ucierpi?". Jeśli jedyną rzeczą na liście jest moje ego, to znaczy, że trzeba to po prostu zrobić.

KROK W STRONĘ JAKOŚCI

Zastanów się, czy możesz i chcesz podjąć się pierwszego zadania na liście. Jeśli nie, to co stoi na drodze?

2.2. Blame game

Kiedy w projektach dzieje się dobrze, większość zespołów nieźle radzi sobie z komunikacją. Wyzwania pojawiają się, kiedy zaczyna się palić. Wtedy na wierzch wychodzą prawdziwe intencje i zamiary. Naszym najważniejszym celem powinno być dobro projektu, jednak jesteśmy

tylko ludźmi. Często górę biorą nasze przywary, ukryte plany albo po prostu instynkt przetrwania. Nikt nie lubi być w centrum wydarzeń, jeśli te wydarzenia są niekorzystne.

Kiedy pojawia się błąd na produkcji lub negatywna informacja zwrotna od klienta albo przestaje działać jakiś serwis, mamy dwie drogi. Pierwsza z nich to skupienie się na rozwiązaniu, opracowanie planu działania, zaangażowanie właściwych osób i jak najszybszy powrót do normalności. To nic innego jak postawienie pytania: „Co możemy z tym zrobić?". Druga droga to szukanie winnych. W tym podejściu rozwiązanie to sprawa drugorzędna, najważniejsze jest znalezienie kozła ofiarnego. Kogoś, kogo można obwinić — zrzucić na niego odpowiedzialność i iść dalej bez wnikania w szczegóły. Tutaj pytanie brzmi: „Czyja to wina?". Takie podejście na krótkoterminowo da nam spokój ducha, ale długoterminowo zabija zespół.

Szukanie winnych to w wielu firmach codzienność. Nie musi chodzić o poważne konsekwencje, jak brak awansu czy premii. Może to przyjmować znacznie lżejsze formy, jak wyznaczenie naprawy problemu osoby „winnej" powstania błędu albo wysłanie personalnej wiadomości. Oczywiście osoba znająca najlepiej obszar, w którym powstał błąd, będzie najlepszym kandydatem do jego naprawy. Chęć działania powinna jednak wyjść od tej osoby, nie powinien to być nakaz z góry.

Tutaj dochodzimy do źródła problemu. W wielu zespołach są osoby zaangażowane i osoby unikające wyzwań. Żeby zmobilizować te drugie, zespół tworzy procesy, które wylewają dziecko z kąpielą. Zamiast zachęcić osoby mniej aktywne, wprowadzają podziały i grę zrzucania odpowiedzialności (ang. *blame game*[1]). Wiele razy doświadczyłam, że

[1] *Blame game* to gra w obwinianie się nawzajem. Chciałabym, żeby takiego rozdziału wcale nie było. Jednak wiem, że takie sytuacje niestety ciągle występują w projektach, dlatego muszę o tym napisać.

wprowadzano nowe procesy, żeby zaktywizować konkretnych członków zespołu, zamiast zwyczajnie z nimi porozmawiać. Najczęściej bierne zachowanie nie wynika ze złych intencji, tylko z trudnej sytuacji prywatnej, niepewności w danym obszarze projektu czy nieśmiałości. Nieoficjalna rozmowa albo zaproponowanie pomocy może przynieść lepsze owoce niż posługiwanie się metodą kija i marchewki.

Kiedy zamiast wyciągać wnioski i iść dalej, skupiamy się na tym, kto zawinił, zabijamy innowacyjność. Po co próbować, skoro nagroda jest marna, a karą za niepowodzenie może być wstyd przed całym zespołem? Nawet GIT ma funkcję *blame*, z której często korzystamy. Mówi się, że junior różni się od seniora tym, że na widok brzydkiego kodu zaczyna strasznie narzekać. Natomiast senior najpierw sprawdzi, czy sam nie napisał tego kodu kilka miesięcy wcześniej. Jest coś uwalniającego w zrzucaniu winy na kogoś innego. Dużo trudniej jest pomagać bez obwiniania i narzekania.

Jarek Ratajski podczas mojego webinaru „Rozmowy o jakości" opowiadał o kulturze błędu, jaką stworzyli w firmie. Chodziło o to, że każdy przyznawał się do porażek na forum grupy. Na początku było to dość sztuczne i niekomfortowe, bo uczestnicy szukali tych sytuacji trochę na siłę. Po jakimś czasie wszyscy się do tego przyzwyczaili i przyznawanie się również do poważnych błędów nie stanowiło dla nikogo problemu. Chodzi tutaj o zaufanie i bezpieczeństwo psychologiczne[2]. Jeśli wierzę w to, że zespół nie ukarze mnie (pogardą, wyśmianiem, karą materialną) za popełniony błąd, to ujawniając go, tylko zyskuję. Jeśli natomiast wiem, że coś złego mnie spotka, to go ukrywam, ryzykując poważne problemy w działaniu aplikacji.

[2] Bezpieczeństwo psychologiczne, czyli przekonanie, że nikt w zespole nie będzie ukarany lub upokorzony, gdy zdecyduje się podzielić pomysłem, poprosić o pomoc, przyznać do błędu lub wątpliwości.

KROK W STRONĘ JAKOŚCI

Kiedy następnym razem popełnisz błąd, spróbuj się do niego przyznać. Bez samobiczowania, tłumaczenia się czy użalania. Potraktuj go jako przestrogę dla innych.

2.3. Nowinki

Kiedy wracamy z konferencji, często mamy głowę pełną pomysłów. Od razu dostrzegamy możliwości rozwoju i odświeżenia naszego projektu. Może to być spowodowane troską o projekt, o łatwość jego utrzymania i rozwój. Jest też szansa, że mamy czysto egoistyczne pobudki i chcemy tylko podrasować nasze CV (ang. *resume driven development*[3]). W obu przypadkach należy wziąć pod uwagę szereg czynników, zanim radośnie wrzucimy nowe technologie na produkcję.

Po pierwsze: jak stabilna jest ta biblioteka, ten framework czy język? W tej branży wszystko zmienia się szybko i wiele niedojrzałych technologii zostaje z entuzjazmem przyjętych przez społeczność. Nowinki są wychwalane na konferencjach, nawet gdy nikt jeszcze nie przetestował ich w boju. Jeszcze ważniejsze od poziomu stabilności jest to, czy nam taki poziom odpowiada. Jeśli mamy do czynienia z projektem, który prawdopodobnie umrze młodo, eksperymentowanie jest nawet wskazane. Zwłaszcza jeśli może przyspieszyć pracę. Jeśli natomiast pracujemy nad oprogramowaniem dla banków lub instytucji medycznych, lepiej trzymać się bardziej sprawdzonych narzędzi.

Po drugie: czy jesteśmy gotowi na bycie ambasadorem i opiekunem tej technologii przez dłuższy czas? Na konferencyjnych prezentacjach

[3] *Resume driven development* to praktyka, która ma na celu głównie budowanie naszego CV. Nie cieszy się dobrą sławą, chociaż jednym z jej plusów jest wprowadzanie do projektu innowacji; *https://medium.com/2-minute-madness/why-you-should-think-about-resume-driven-development-f68d1ab55d82*.

najczęściej wszystko działa dobrze i zgodnie z planem. W prawdziwych projektach mamy meandry zależności i często nietypową infrastrukturę. Na pewno więc pojawią się nietypowe problemy. Ich rozwiązania można będzie znaleźć na *StackOverflow*, albo i nie. Czy kiedy będzie ciężko, będziemy gotowi na wzięcie odpowiedzialności za naprawianie błędów? Nie chodzi o to, że zawsze już będziemy jedynymi odpowiedzialnymi za daną technologię. Chodzi raczej o poczucie tej odpowiedzialności. Jeśli nie mamy ochoty mierzyć się z konsekwencjami, może lepiej „pobawić się" z daną technologią w swoim pobocznym projekcie albo podczas hackathonu.

Po trzecie: czy zespół chce tej technologii i jest gotów ją utrzymywać przez długi czas? Najlepiej, jeśli zespół jest zróżnicowany i mamy w nim zarówno entuzjastów nowinek, jak i sceptyków. Bez tych pierwszych używamy przez całe lata tych samych narzędzi. Bez tych drugich możemy nieustannie wchodzić w ślepe uliczki. Dobrze, jeśli ktoś zada kilka kontrolnych pytań, sprawdzi opinie i zagra rolę adwokata diabła. To może nam pomóc wycofać się z użycia narzędzia, zanim przyniesie nam kłopoty. Nie chodzi więc o to, żeby cały zespół entuzjastycznie przyjmował każdy pomysł, ale żeby wyraził chęć używania tego narzędzia po wstępnej analizie. Żeby większość była gotowa tworzyć nowe funkcjonalności, ale też utrzymywać je po dłuższym czasie.

Wreszcie po czwarte: czy stać nas na eksperymenty w tym momencie. Każda nowa technologia wprowadzana na produkcję to jest nasz koszt. Koszt nauki, koszt poprawiania potencjalnych błędów i wreszcie koszt utrzymania. Niektóre narzędzia po pewnym czasie są usuwane z produkcji, ale z doświadczenia wiem, że to nie jest prosty proces. Nie dość, że trzeba się upewnić, że nigdzie nie zostały pozostałości, to trzeba włożyć mnóstwo pracy w przeróbki. Rzadko zdajemy sobie z tego sprawę na początku. Nasłuchaliśmy się wielu pochwał i obietnic,

chcemy, żeby i dla nas się spełniły. Rzeczywistość często nas nieco rozczarowuje, mimo to wycofanie się z używania narzędzia nie przychodzi nam łatwo.

KROK W STRONĘ JAKOŚCI

Kiedy następnym razem napotkasz technologię, która wydaje się być właściwa w projekcie, zaproponuj hackathon albo inną formę prototypowania. Rozwiążcie przykładowy problem, który występuje w projekcie, żeby spotkać prawdziwe problemy.

2.4. Równowaga

Ciężko jest się odnaleźć w projektach zastanych, które mają stare technologie. Sama byłam nieraz uwięziona w technologiach, które bardziej kojarzyły mi się z latami 90. Mój wewnętrzny głos buntował się i podpowiadał, że przecież chcę się uczyć czegoś innego. Czegoś, co nie uwięzi mnie w tym projekcie na lata. Ale jest też druga strona medalu. Jeśli pracujemy w zespołach, które nieustannie wypatrują nowinek i do każdego modułu wykorzystują inną technologię, łatwo możemy się poczuć przytłoczeni. Ciężko jest ciągle gonić za czymś nowym i bazować wyłącznie na nowinkach.

Czy ktoś ma rację? W wytwarzaniu oprogramowania jest jak w życiu — najlepsza jest względna równowaga. Dla jednych projektów to będzie bardziej po stronie eksperymentów. Przykładem mogą być firmy, które zajmują się wytwarzaniem oprogramowania dla kogoś. Ich zespoły muszą być na bieżąco i pełnić rolę innowacyjnych doradców swoich klientów. Dla innych równowaga znajduje się dużo bliżej starszych, ale stabilnych narzędzi. Nie mówię o technologiach sprzed dwóch dekad, ale raczej sprzed dwóch lat. W IT to i tak przepaść.

Patrząc z perspektywy pracownika, nie możemy oczekiwać, że przychodząc do stabilnej korporacji, będziemy mieć możliwość wprowa-

dzania najnowszych narzędzi. W niektórych zespołach może tak być, ale przeważnie będziemy mieć do czynienia ze starszymi technologiami. Z kolei przychodząc do start-upu, trzeba się przygotować na szybkie tempo zmian. To, jaki jest cykl życia produktu i klimat organizacji, ma wpływ na używane narzędzia; nie odwrotnie. Ciekawym zjawiskiem jest chęć eksperymentów nie na swój koszt. Wiele razy spotkałam się z programistami, którzy narzekają na brak ryzyka ze strony pracodawców, równocześnie pracując od wielu lat na umowę o pracę w tej samej firmie. Jeśli chcemy większego ryzyka na produkcji, to sami też musimy trochę zaryzykować.

Noblista w dziedzinie ekonomii, Milton Friedman, wskazał cztery sposoby wydawania pieniędzy. Możemy wydawać swoje lub cudze pieniądze, na siebie lub na kogoś innego. Najrozsądniej dysponujemy swoimi pieniędzmi wydawanymi na własne potrzeby. Natomiast lekką ręką wypływają środki na własne potrzeby finansowane przez kogoś innego. Dlatego często, kiedy inżynierowie zaczynają pracować nad własnym oprogramowaniem *SaaS*, zdecydowanie zmieniają podejście do używanych narzędzi, ich kosztów i ryzyka.

To, czego potrzebujemy w naszej codziennej pracy, to wejście w buty osób decyzyjnych i zadanie sobie kilku kontrolnych pytań wymienionych w poprzednim podrozdziale („Nowinki"). Zamiast wychodzić z luźną propozycją niepopartą żadną pracą, odrabiamy zadanie domowe. Znając odpowiedzi na te pytania, możemy wykazać się dojrzałością i przechylić szalę na swoją korzyść. Pokazujemy, że jesteśmy nie tylko zainspirowani, ale i gotowi do sprawnego wprowadzenia tych technologii na produkcję.

KROK W STRONĘ JAKOŚCI

Kiedy następnym razem zaproponujesz nową technologię, zastanów się, co musiałoby się stać, żeby została ona wprowadzona w Twoim start-upie.

2.5. Po co nam maruderzy?

Kiedy uczestniczymy w konferencji lub spotkamy starego znajomego, często dowiadujemy się o jakiejś przełomowej technologii czy narzędziu. Z wielkim entuzjazmem chcemy wdrożyć to rozwiązanie w naszym projekcie. Najlepiej od razu. Wtedy zawsze znajdzie się ktoś, kto zacznie marudzić. Nie lubimy takich sytuacji. Jak ktoś może być na nie w obliczu tak pomocnego narzędzia? Jak ktoś może być tak odporny na innowacje? Czy nie powinniśmy ciągle usprawniać naszego projektu? Tymczasem taka osoba na naszej drodze jest najlepszym, co może się przydarzyć.

Ostatnie, czego chcemy dla naszych zespołów, to jednorodność. Grupa złożona z osób, które godzinami mogą dyskutować o tym, jak bardzo się ze sobą zgadzają, przestaje się rozwijać. W jednomyślnym środowisku trudniej o innowacje i dopracowane pomysły. Kiedy rodzi się w naszej głowie idea, jest jak małe dziecko, niedojrzała i wymagająca rozwoju. Jeśli nie zderzymy jej z osobami, które zaczną zadawać trudne pytania i podrzucać pomysły, o których nie pomyśleliśmy, zostanie na tym poziomie. Nas za to czeka sporo rozczarowań i błędów do naprawienia. Właśnie dlatego potrzebujemy osób, które myślą inaczej od nas. Wtedy wiele problemów uda się rozwiązać już na etapie planowania.

W zespole, który składa się z samych entuzjastów, może i będzie szybki rozwój, ale będą mu towarzyszyć ciągłe wpadki. Z kolei w zespole złożonym z samych maruderów bardzo rzadko będą wprowadzane jakiekolwiek zmiany. Większość rzeczy będzie dopięta na ostatni guzik, ale ciągle za późno. Jak zwykle najlepsza jest równowaga. Dobrze jest zwrócić na to uwagę w procesie rekrutacyjnym. Jeśli to zespół decyduje, kto ma do niego dołączyć, ludzie będą szukać osób podobnych do

siebie[4]. Tych, którzy myślą podobnie, uważamy za bardziej inteligentnych i zaradnych, więc łatwo wpaść w tę pułapkę[5]. To, czego najbardziej potrzebujemy, to ludzie inni od nas, ale tacy, z którymi ciągle możemy się dogadać. Jak pokazuje scenariusz *Dwunastu gniewnych ludzi*, jedno pytanie może zmienić wszystko. W tym filmie wszyscy bez mrugnięcia okiem uznaliby podejrzanego za winnego, gdyby nie znalazła się jedna osoba, która zaczęła zadawać trudne pytania.

KROK W STRONĘ JAKOŚCI

Na najbliższym spotkaniu zabaw się w adwokata diabła. Zacznij zadawać trudne pytania i zasiewać wątpliwości. Nie po to, żeby utrudnić pracę nad zadaniem, ale żeby ją udoskonalić.

2.6. Zabijanie entuzjazmu

Kiedy nowa osoba przychodzi do firmy, jest przepełniona entuzjazmem. Zakładam, że zmiana pracy to zazwyczaj dobrowolna decyzja i że nawet jeśli w poprzedniej firmie było nam dobrze, to liczymy na coś jeszcze lepszego. Przychodzimy pierwszego dnia gotowi wykorzystać wszystkie swoje umiejętności i talenty, żeby tworzyć wartość. Jeśli pozwolimy nowo zatrudnionym się wykazać, wszyscy na tym skorzystamy. Po pierwsze wykorzystamy tę energię. Po drugie dodamy skrzydeł nowej osobie, zanim zderzy się z codziennymi problemami. Po trzecie uwiarygodnimy ją w oczach klienta i całego zespołu. Niestety często te pierwsze dni wyglądają zupełnie inaczej.

[4] Mamy tendencje do tego, żeby zatrudniać ludzi, którzy są do nas podobni; *https://www.forbes.com/sites/forbescoachescouncil/2018/05/01/why-you-mistakenly-hire-people-just-like-you/*.

[5] Bardziej lubimy ludzi, którzy są do nas podobni lub z którymi mamy coś wspólnego; *https://www.psychologytoday.com/us/blog/close-encounters/201812/why-do-we-people-who-are-similar-us*.

Kilka razy zdarzyło mi się pracować w projektach prowadzonych przez duże organizacje, gdzie pierwszych kilka dni spędziłam na… czekaniu. Przynajmniej raz czekałam na własny komputer, mysz i klawiaturę. Czekałam na dostępy. Czekałam na to, aż ktoś zainstaluje mi oprogramowanie albo dostanę możliwość zainstalowania go samodzielnie. Do tego miałam dużo czasu na rozmowy w kuchni, gdzie inni pracownicy utwierdzali mnie w przekonaniu, że tutaj wszystko trwa, i dzielili się swymi doświadczeniami i problemami. Efekt był taki, że po tygodniu pracy zamiast być już rozpędzona w swoich codziennych zadaniach, zaczynałam czytać dokumentację albo uczestniczyłam w nieskończonej ilości spotkań. Tak niestety wygląda proces wdrażania pracownika w większości zespołów.

Przecież osoba, która przychodzi, może o większości spraw dowiedzieć się w trakcie pracy. To nie jest tak, że od razu będzie pracować na własną rękę, bez *code review* czy innej formy weryfikacji. Dużo lepsze od zarzucenia nowego pracownika ogromną ilością informacji i podarowania kilku wolniejszych dni na początek jest wprowadzenie do projektu przez mentora. Osoba bardziej doświadczona w projekcie może podpowiadać, odpowiadać na pytania i dbać o wszystkie niezbędne formalności, podczas gdy osoba mentorowana wchodzi od razu w swoją rolę i wykonuje pracę najlepiej, jak umie. Zatrudniamy świetnych specjalistów, nierzadko bardzo samodzielnych, a potem boimy się im zaufać.

Wiele firm wprowadza politykę *first day commit*[6], czyli wykonanie choćby najmniejszego zadania samodzielnie już pierwszego dnia pracy.

[6] Bardzo ciekawą rozmowę na temat polityki *first day commit* i optymalizacji całego procesu wdrażania pracownika przeprowadził z Andrzejem Krzywdą Grzegorz Kotfis w podcaście Devsession; *https://devsession.pl/developer-experience/*.

Oczywiście jest to łatwiejsze w małych organizacjach, gdzie wszystkie dostępy można przyznać wcześniej. Zdecydowanie lepiej zadziała to w Kanbanie niż w środku scrumowego sprintu[7]. To jednak nie znaczy, że w korporacji nie da się tego zrobić. Trzeba się po prostu o to zatroszczyć wcześniej. Usprawnić proces, żeby pozwolić nowej osobie działać od pierwszego dnia. Zyski mogą być znaczące, bo nie tylko nie marnujemy tych początkowych dni, ale też nie zabijamy entuzjazmu, który wpłynie na kolejne tygodnie pracy.

KROK W STRONĘ JAKOŚCI

Kiedy do Twojego zespołu dołączy kolejna osoba, dowiedz się, jak wygląda jej proces onboardingowy. Być może taki proces nie jest zdefiniowany, a może jest nieefektywny. Możesz też zostać przewodnikiem dla nowej osoby.

2.7. Feedback

Przekazywanie i odbieranie informacji zwrotnej to jedne z najtrudniejszych, ale i najważniejszych umiejętności miękkich. Rozmawiałam z wieloma osobami, które długie lata pracowały w pojedynkę jako freelancerzy. Wszyscy mówili o tym, jak bardzo brakowało im oceny ich pracy i powiązanej z tym możliwości rozwoju. Bo feedback to nie tylko krytyka, chociaż często takie formy może przybierać. Jednak przede wszystkim chodzi o to, żeby pomóc komuś się rozwijać. Kiedy

[7] Zarówno Scrum, jak i Kanban należą do metodyk zwinnych. W Scrumie mamy sprinty dające nam większą kontrolę nad zadaniami, które mają być realizowane w najbliższej przyszłości. Kanban pozwala na większą elastyczność, bo jest po prostu systemem wizualizacji pracy; *https://www.guru99.com/scrum-vs-kanban.html*.

powstaje kod, tekst albo muzyka, wizja w naszej głowie jest bardzo niedoskonała. Nawet pierwsza wersja jest taka sobie, chociaż może nam się wydawać inaczej. Kiedy wrócimy do owoców naszej twórczości po jakimś czasie, zobaczymy, że wiele można ulepszyć. Gdy skorzystamy z cudzej opinii, to może się wydarzyć od razu. Powinniśmy wykorzystywać informacje zwrotne, żeby poszerzać swoją wiedzę i zdobywać umiejętności w obszarach, o których często nawet nie wiemy.

Mówi się, że stajemy się uśrednieniem pięciu osób, z którymi przebywamy najczęściej. To właśnie przez informacje zwrotne. Zaczynamy myśleć w podobny sposób, kiedy ktoś zwraca nam uwagę na coś, co wcześniej nie było dla nas ważne. Dlatego warto otaczać się ludźmi, którzy nie tylko są proaktywni, ale przede wszystkim nieco inni od nas. Dyskutując w takim zespole, nie tylko sami się rozwijamy, ale również dzielimy się swoją perspektywą z innymi i mobilizujemy ich do rozwoju. Wszyscy korzystają.

Jest wiele porad, jak skutecznie przekazywać feedback, ale najważniejsze, żeby robić to często. Jeśli wystąpił jakiś problem albo wydarzyło się coś wyjątkowego, warto od razu o tym porozmawiać. Po pierwsze dlatego, że im dłużej zwlekamy, tym jest trudniej. Po drugie po pewnym czasie możemy już wszystkiego nie pamiętać, przekręcać fakty albo nabrać emocjonalnego stosunku do sprawy. Jeśli będziemy to robić często i bez zadęcia, będziemy to robić coraz lepiej. Wtedy nasza informacja zwrotna będzie użyteczna. Najgorsza rzecz w komunikacji to feedback, który jest z natury krytykancki i jest wyrażony tylko po to, żeby ktoś mógł się popisać swoją wiedzą.

Dzielimy się informacją zwrotną z troski. Troski o projekt, ale też troski o osobę, której go przekazujemy. Zależy nam na tym, żeby pewne błędy się nie powtórzyły, ale też żeby wzmocnić pozytywne zachowanie. Tylko w ten sposób jest szansa na to, że ta informacja zostanie przyjęta pozytywnie i wdrożona. Każdy, kto otrzymał kiedyś negatywny

feedback (choćby od rodziców czy rodzeństwa), który wydawał się być tylko wymądrzaniem się, wie, że ciężko do niego podejść z otwartą głową.

KROK W STRONĘ JAKOŚCI

Poproś o informację zwrotną z proaktywną postawą, zadając pytania w rodzaju „co mogę tu usprawnić?" albo „jak mogę zrobić to lepiej następnym razem?".


PROGRAM
PARTNERSKI
GRUPY HELION
1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ
Zmień swoją stronę WWW w działający bankomat!
Dowiedz się więcej i dołącz już dzisiaj!
http://program-partnerski.helion.pl
GRUPA
Helion

Zespół projektowy na drodze do ideału

Jakość została zdefiniowana już w starożytności. Platon określał ją jako pewien stopień doskonałości. I tu nic się nie zmieniło — im bliższy ideału produkt, tym wyższa jego jakość. Problem w tym, że każdy postrzega ów ideał nieco inaczej. Na gruncie programistycznym wysoka jakość może oznaczać co innego dla project managera, a co innego dla developera odpowiedzialnego za warstwę frontendu. Jeszcze inaczej do tematu podejdzie klient. W dużym projekcie, nad którym pracuje wiele osób, podejście do tej kwestii bywa kompletnie różne od tego w niewielkim projekcie, za który odpowiedzialny jest zespół trzyosobowy. Do tego dochodzą zmieniające się trendy. Wzorce projektowe, które były słuszne dwa, trzy lata temu i gwarantowały wówczas wysoką jakość uzyskanego oprogramowania, dziś mogą się okazać nieprzydatne.

Czy zatem nie istnieją uniwersalne wzorce ani normy jakości projektów w IT? Oczywiście, że istnieją. Aleksandra Kunysz, dla której propagowanie wiedzy na temat jakości stanowi życiową pasję, dzieli się nimi w tej książce. Przygląda się w niej wszystkim czynnikom, które mają wpływ na ostateczny produkt, czyli zamówione oprogramowanie. Zwraca oczywiście uwagę na sprawy techniczne, ale przede wszystkim podkreśla rolę czynnika ludzkiego: cechy osobowe klienta, przedstawicieli firm negocjujących warunki, członków zespołów projektowych itd. Ponieważ wyjściowo wygląda to zwykle tak, że ile osób, tyle pomysłów na jakość. A rzecz w tym, by uwspólnić oczekiwania i — co za tym idzie — uzyskany rezultat.

Aleksandra Kunysz — w branży IT działa od kilkunastu lat. Doświadczenie zdobywała w Polsce i Stanach Zjednoczonych. Pracowała w wielu międzynarodowych projektach — od telekomunikacji, przez ubezpieczenia, po e-commerce. Pomaga pisać lepsze testy. Tematyką jakości zajmuje się w stworzonej i prowadzonej przez siebie szkole testów (*https://szkolatestow.online/*), na Twitterze, YouTube i Instagramie. Prowadzi też bloga pod adresem: *https://olaqnysz.blogspot.com/*.

Helion

helion.pl

HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

ISBN 978-83-283-7202-3

INFORMATYKA W NAJLEPSZYM WYDANIU

Cena: 44,90 zł