Jak pracować z programistami? 10 porad

1. Nie udawaj technicznego zrozumienia.

W momencie, gdy programista zaczyna wdawać się w zbyt techniczne szczegóły, grzecznie daj mu o tym znać. Nie przytakuj głową udając, że rozumiesz. Nie ma nic bardziej irytującego niż skończenie technicznego wywodu i zrozumienie, że nasz rozmówca zgubił się mniej więcej na początku.

2. Zadawaj odpowiednie pytania.

Zadawaj pytania, które nakierują rozmowę na odpowiedni poziom techniczności. Z reguły specjaliści techniczni pracują wśród wielu programistycznych detali i to właśnie non stop przewija się przez ich myśli. Kiedy zaczynasz z nimi rozmawiać, trudno wyrwać im się z tych torów myślenia. Mogą tutaj pomóc odpowiednie pytania:

- To jest dla mnie dosyć techniczny opis, mógłbyś powiedzieć, jaki będzie to miało wpływ na funkcjonalność?
- Poczekaj, to znaczy, że może to mieć wpływ na termin wykonania tego zadania? Jeśli tak, to jak duży?
- Powiedziałeś, że 3 dni, to znaczy, co musimy tam dokończyć? -> tutaj, gdy pojawi się monolog techniczny, dopytujemy: to znaczy, jaka będzie różnica w funkcjonalności? Jaki wpływ będzie to miało dla klienta? Jak mogę mu to prosto wytłumaczyć?

- Czy znaleziony błąd dotyczy tylko Twojego zadania czy ogólnie jest przekrojowy dla aplikacji? -> Przekrojowy? To może powinniśmy wystawić dla niego osobnego buga a Twoje zadanie zamknąć?

3. Oddzielaj od siebie zadania, bugi i change requesty

Staraj się jasno określać z zespołem: co jest zadaniem, co jest bugiem, co jest change request'em. **Zadanie** dotyczy dostarczenia funkcjonalności zgodnej ze specyfikacją i uzgodnionej z klientem. **Change request** (*inaczej CR) dotyczy zadania, które powstało w wyniku zmiany w specyfikacji już po pierwotnych ustaleniach. **Bug** to błąd w funkcjonowaniu aplikacji, to znaczy aplikacja nie działa zgodnie z ustaloną pierwotnie specyfikacją i trzeba to naprawić.

Przydatne pytania:

- Czy mieliśmy informację (*o tym fragmencie specyfikacji) już wcześniej? Czy wiedzieliśmy o tym w momencie szacowania zadania? Nie wiedzieliśmy? Czyli możemy to uznać za change request ze strony klienta? **Zmieniły się wymagania?**
- Czy obecnie **zachowanie jest inne** niż kryteria akceptacyjne zapisane w User Story? Ok, a więc rzeczywiście to jest bug. (sytuacja rodem z projektów Scrum'owych)

4. Ufaj swoim specjalistom i wierz w ich umiejętności

Nic tak nie dodaje skrzydeł jak wiara project managera w umiejętności techniczne swojego zespołu. Powinieneś wychodzić z założenia, że pracujesz z najlepszymi specjalistami i organizować pracę tak, aby mieli możliwość całkowitego skupienia się na aspektach technicznych.

Jednocześnie nawet jeśli wydaje Ci się, że już znasz się na inżynierii oprogramowania, nie wtrącaj się do aspektów technicznych. Tutaj decyzje należą do architekta i zespołu.

5. Używaj logicznych argumentów.

Specjaliści IT to osoby, dla których logiczne myślenie to codzienność. To, co zdecydowanie ułatwi Wam komunikację, to właśnie logiczne argumenty. Informując o danej decyzji, koniecznie dodaj **DLACZEGO** się na nią zdecydowałeś. Wówczas również Twojemu zespołowi będzie łatwiej ją zrozumieć. Jeśli Twój klient czasami zachowuje się nielogicznie również dla Ciebie, nie bój się tego przyznać przed zespołem.

6. Informuj, jaki jest plan.

Programiści uwielbiają wiedzieć na czym stoją i jaki jest plan. Są wtedy w stanie wszystko zaplanować i dopasować, czują się bezpiecznie. Chaos, niepewność i nieustannie zmieniajace się decyzje to czynniki niepożądane i szkodliwe.

7. Nie rozpraszaj.

Praca umysłowa wymaga skupienia i koncentracji. Trzeba wejść w tak zwany flow, skupić się na problemie i zacząć implementację. Organizacja zebrań w odstępach 2-godzinnych, podchodzenie co chwila z nowymi pytaniami nie jest najlepszym wyborem.

8. Rozmawiając o wymaganiach, bądź precyzyjny.

Programiści to osoby odpowiedzialne za działanie oprogramowania, w każdych okolicznościach. Nie dziwi więc fakt, że są to osoby precyzyjne. Pisząc kod muszą mieć one pełny obraz sytuacji. Jeśli rozmawiasz o wymaganiach, nie poprzestawaj na ogólnikach. Staraj się uprościć cały przekaz do konkretnych reguł funkcjonowania. Bądź precyzyjny. Nie zapominaj o przypadkach brzegowych. Na przykład, jeśli mówisz o polu tekstowym, określ, co powinno się stać, jeśli ktoś wpisze bardzo długi tekst, który nie będzie się w tym polu mieścił. Co ma się wydarzyć, jeśli nie wpisze nic. Wspomnij o niedozwolonych znakach, jeśli takie występują. Staraj się przewidzieć wszystkie sytuacje. Jednocześnie nie przejmuj się, jeśli Ci się nie uda. Doświadczony programista z pewnością dopyta o szczegóły,

które wydadzą mu się istotne. Z czasem identyfikowanie wszystkich przypadków będzie szło Ci coraz lepiej.

9. Pozwól programistom samemu dobierać sobie zadania.

Nie przypisuj programistom na siłę konkretnych zadań, chyba, że bardzo dobrze znasz ich słabe i mocne strony, a także dogłębnie rozumiesz techniczne i fachowe aspekty projektu. Pozostaw tę rolę liderowi technicznemu.

10. Bądź szczery.

Jeśli nie miałeś wcześniej zbyt dużej styczności z technicznymi rzeczami, po prostu powiedz o tym zespołowi. Docenią Cie jako specjalistę w zakresie zarządzania lub np. Scruma, a jednocześnie będą wiedzieć, że w rozmowach z Tobą należy dostosować język rozmowy. Powinien on być bardziej ogólny, skoncentrowany na funkcjonalnościach, nie aż tak bardzo techniczny.