Joel Spolsky to prawdziwy geniusz. Sprawnie porusza się w świecie technologii i marketingu (oraz na polach wspólnych dla obu tych obszarów), których my wciąż musimy się uczyć.
Seth Godin, autor książki *Dołek*

Programista poszukiwany

Znajdź i zatrudnij najlepszego!

Joel Spolsky

autor książki „Zarządzanie projektami informatycznymi"
oraz redaktor „Sztuki pisania oprogramowania"

Helion

Programista poszukiwany. Znajdź i zatrudnij najlepszego!

Autor: Joel Spolsky
Tłumaczenie: Beata Pawlak
ISBN: 978-83-246-3015-8
Tytuł oryginału: Smart and Gets Things Done: Joel Spolsky's Concise Guide to Finding the Best Technical Talent
Format:A5, stron: 168

Pracownicy Twoich marzeń

- Naucz się stosować praktyczne kryteria porządkowania zgłoszeń od kandydatów
- Poznaj podręczną instrukcję prowadzenia rozmów kwalifikacyjnych
- Dowiedz się, jak poprawiać niedoskonałe zespoły

Joel Spolsky to prawdziwy geniusz. Sprawnie porusza się w świecie technologii i marketingu (oraz na polach wspólnych dla obu tych obszarów), których my wciąż musimy się uczyć.
Seth Godin, autor książki Dołek

Joel Spolsky to jeden z najbardziej rozważnych autorów piszących na temat wytwarzania oprogramowania. Z jego przemyśleniami na temat zatrudniania pracowników powinien zapoznać się każdy, kto planuje rekrutację programistów.
Jessica Livingston, partner w firmie Y Combinator i autorka książki Founders at Work

Jak zaoszczędzić na programistach?

Menedżerowie, przedsiębiorcy i szefowie działów kadr wciąż toczą wewnętrzną walkę, czy zatrudnić świetnego, choć kosztownego programistę, czy przyoszczędzić i wybrać kogoś przeciętnego. Ci, którzy wybierają opcję numer 2, szybko przekonują się, że oszczędzanie na programistach skutkuje oprogramowaniem kiepskiej jakości, a co za tym idzie, korzyści są iluzoryczne.

Po co komu dobry programista?

Największym problemem związanym z zatrudnianiem wielu przeciętnych programistów w miejsce kilku naprawdę dobrych jest to, że niezależnie od czasu poświęconego na realizację zadań przeciętni programiści nigdy nie stworzą kodu, którego jakość będzie choćby zbliżona do jakości kodu najlepszych programistów. Pięciu Salierich nigdy nie stworzy dzieła na poziomie Requiem Mozarta, nawet gdyby pracowali nad nim sto lat.

Po co komu dobry produkt?

Nie możemy pozwolić sobie na to, by konkurencja nas prześcignęła, czy na tworzenie produktów zaledwie wystarczająco dobrych. Nasze dzieła muszą być niesamowite, by przyciągały uwagę potencjalnych użytkowników. Największym prezentem, który możemy otrzymać od naprawdę utalentowanych programistów, jest właśnie nadzieja na stworzenie czegoś zauważalnego.

Dlaczego trzeba przeczytać tę książkę?

Proces zatrudniania największych talentów technicznych jest jak zupełnie pozbawiony elementów zabawy tor przeszkód. Każdy, kto kiedykolwiek zastanawiał się, na co zwracać uwagę podczas przeglądania aplikacji, kto bił się z myślami po zakończeniu rozmowy kwalifikacyjnej lub kto nie może pojąć, dlaczego tak trudno znaleźć doskonałych programistów, powinien rzucić wszystko i przeczytać tę książkę.

Jak to się robi w profesjonalnych firmach?

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

SPIS TREŚCI

Rozdział 3

PODRĘCZNY PRZEWODNIK PO OCZEKIWANIACH PROGRAMISTÓW

Możemy reklamować swoje oferty pracy w doskonałych miejscach, organizować fantastyczny program praktyk i zapraszać na rozmowy wprost wymarzonych kandydatów, ale jeśli najlepsi programiści nie będą chcieli dla nas pracować, nigdy nie zbudujemy zespołu na miarę naszych oczekiwań. Ten rozdział można więc traktować jako swoisty przewodnik po oczekiwaniach programistów — wyjaśnię, czego szukają, co lubią, a czego nie lubią w miejscu pracy oraz co decyduje o wyborze pracodawcy przez najlepszych programistów.

Gabinety

W zeszłym roku uczestniczyłem w konferencji informatycznej w Yale. Jeden z prelegentów, weteran Doliny Krzemowej, jeden z najbardziej cenionych ekspertów zatrudnianych przez nowo powstające firmy, trzymał w dłoni książkę *Czynnik ludzki*[1].

[1] Tom DeMarco, Timothy Lister, *Czynnik ludzki — skuteczne przedsięwzięcia i wydajne zespoły*, Wydawnictwa Naukowo-Techniczne, Warszawa, 2002.

„Musicie przeczytać tę książkę", mówił. „To prawdziwa biblia wskazująca, jak prowadzić firmę wytwarzającą oprogramowanie. To najważniejsza publikacja na temat prowadzenia przedsiębiorstw zajmujących się oprogramowaniem".

Trudno się z tym nie zgodzić. *Czynnik ludzki* to rzeczywiście świetna książka. Jedną z najważniejszych (ale też najbardziej kontrowersyjnych) koncepcji proponowanych w tej książce jest idea udostępniania programistom mnóstwa wolnej przestrzeni, nawet w formie osobistych gabinetów — właśnie ta przestrzeń ma być warunkiem osiągnięcia należytej produktywności przez programistów. Autorzy, DeMarco i Lister, stale wracają do tego założenia.

Po odczycie podszedłem do prelegenta i powiedziałem: „Zgadzam się z twoją oceną książki *Czynnik ludzki*. „Powiedz tylko: czy we wszystkich firmach, w których zakładaniu uczestniczysz, rzeczywiście dajesz swoim programistom do dyspozycji osobiste gabinety?".

„Oczywiście, że nie, właściciele tych firm nigdy nie zgodziliby się na takie koszty".

Hm.

„Ale przecież właśnie ta koncepcja jest istotą książki, którą tak zachwalałeś", odpowiedziałem.

„To prawda, ale realizacja każdego naszego postulatu wymaga stoczenia prawdziwej batalii. Z perspektywy inwestora gabinety dla programistów to po prostu strata pieniędzy".

W Dolinie Krzemowej wprowadzono praktykę, która nakazuje wszystkim sadzać programistów w wielkich, otwartych przestrzeniach wbrew dowodom wskazującym na nieporównanie większą produktywność programistów pracujących we własnych gabinetach. Mam ogromne problemy z przekonywaniem ludzi do tych oczywistych racji — jak się wydaje, opór branży wynika z tego, że programiści są po prostu zbyt towarzyscy (nawet jeśli ta cecha przekłada się na niższą produktywność). W tej sytuacji walka jest bardzo nierówna.

Słyszałem nawet programistów mówiących: „To prawda, rzeczywiście wszyscy pracujemy w otwartych przestrzeniach biurowych, ale przecież wszyscy tak pracują, *włącznie* z naszym prezesem!".

„CEO? Naprawdę prezes waszej firmy pracuje w otwartej przestrzeni?".

„Cóż, *ma* swoje miejsce między naszymi biurkami, jednak — skoro już o to pytasz — mamy też salę konferencyjną, gdzie odbywa wszystkie ważne spotkania...".

Mmmm, hm. To w Dolinie Krzemowej dość powszechna praktyka polegająca na tym, że prezesi, którzy robią mnóstwo szumu wokół swojej rzekomej pracy wśród szeregowych pracowników, zawsze mają w zanadrzu jakąś salę konferencyjną, którą traktują jak własny gabinet (twierdzą przy tym, że wspomniana sala służy im tylko do poufnych rozmów, jednak w rzeczywistości spędzają tam większość czasu, często sami, rozmawiając przez telefon z partnerem od golfa, trzymając swoje drogie buty na stole konferencyjnym).

Tak czy inaczej nie chcę w tym miejscu ponownie uzasadniać, dlaczego wydzielone gabinety podnoszą produktywność twórców oprogramowania[2,3,4], dlaczego samo założenie na głowę słuchawek i słuchanie muzyki zagłuszającej dźwięki otoczenia utrudnia programistom dochodzenie do wartościowych wniosków[5] ani dlaczego

---

[2] Tom DeMarco, Tim Lister, *Programmer Performance and the Effects of the Workplace*, „Proceedings of the 8th International Conference on Software Engineering", IEEE Computer Society Press, London, 1985.

[3] Capers Jones, *How Office Space Affects Programming Productivity*, „IEEE Computer" 28, nr 1 (styczeń 1995), s. 76 – 77.

[4] Gerald M. McCue, *IBM's Santa Teresa Laboratory — Architectural design for program development*, „IBM Systems Journal" 17, nr 1 (1978).

[5] Tom DeMarco, Tim Lister, *Peopleware. Second Edition*, s. 78.

wydzielenie gabinetów dla programistów w praktyce nie zwiększa łącznych kosztów funkcjonowania firmy[6].

W tym rozdziale skoncentruję się na rekrutacji programistów i roli osobistych gabinetów w tym procesie.

Niezależnie od tego, co sądzimy na temat produktywności i jak sami postrzegamy egalitarystyczną koncepcję wspólnych przestrzeni, dwie kwestie nie budzą niczyich wątpliwości:

1. Osobiste gabinety sugerują wyższy status pracownika.
2. Boksy i inne wspólne przestrzenie są wyjątkowo niefortunne w kontekście relacji towarzyskich.

Na podstawie tych dwóch punktów można przyjąć, że programiści będą bardziej zainteresowani przyjęciem oferty pracy w miejscu, gdzie mogą liczyć na pracę w wydzielonych gabinetach. Atrakcyjność oferty będzie jeszcze większa, jeśli gabinety będą wyposażone w drzwi z możliwością zamknięcia oraz okna z ładnym widokiem.

Niewątpliwym utrudnieniem jest to, że na niektóre z tych czynników ułatwiających rekrutację po prostu nie mamy wpływu. Zdarza się, że nawet prezesi i założyciele firm nie mogą swobodnie wdrażać koncepcji gabinetów dla programistów, ponieważ ich decyzje wymagają aprobaty inwestorów. Większość ogranicza się więc do przenoszenia lub przebudowy swoich przestrzeni biurowych raz na pięć czy dziesięć lat. Mniejsze firmy, które dopiero zaczynają działalność, często w ogóle nie mogą sobie pozwolić na koszty związane z utrzymaniem gabinetów. Z własnego doświadczenia wiem, że ta całkiem spora liczba wymówek łącznie prowadzi do przekonania, iż udostępnienie programistom prywatnych gabinetów jest najzwyczajniej w świecie niemożliwe. Nawet

[6] Joel Spolsky, *Bionic Office*, artykuł opublikowany na witrynie *www.joelonsoftware.com* 23 września 2003 r. (należy wpisać w wyszukiwarce słowo *Bionic*).

w najbardziej oświeconych przedsiębiorstwach decyzje o zmianie siedziby i rozmieszczeniu pracowników są podejmowane raz na dziesięć lat przez komitet pracowniczy złożony z sekretarki, menedżera i pracownika wielkiego biura architektonicznego, któremu już na studiach wmówiono, że „otwarte przestrzenie oznaczają otwarte firmy". Oznacza to, że tego rodzaju decyzje w ogóle nie uwzględniają potrzeb programistów ani zespołów projektowych, których dotyczą.

Sytuacja jest więc skandaliczna, jednak sam nie ustaję w słusznej walce o wykazanie, że stworzenie warunków, w których programiści dysponują osobnymi gabinetami, jest *możliwe*. W mojej firmie udało się osiągnąć ten cel przynajmniej w przypadku większości programistów zatrudnionych na pełnym etacie, mimo że ceny najmu w Nowym Jorku są najwyższe w kraju. Zdecydowałem się na takie rozwiązanie, ponieważ nie mam wątpliwości, że w ten sposób znacznie poprawiam samopoczucie pracowników Fog Creek. Jeśli mimo wszystko ktoś upiera się, twierdząc, że korzyści wynikające z udostępniania gabinetów programistom nie pokrywają związanych z tym kosztów, *trudno* — niech to nadal będzie przewaga konkurencyjna mojej firmy.

Fizyczna przestrzeń biurowa

Problem fizycznej przestrzeni w biurze jest poważniejszy niż kwestia osobistych gabinetów dla programistów. Kiedy w dniu rozmów kwalifikacyjnych kandydat przychodzi do naszej firmy, w pierwszym odruchu rozgląda się, analizując miejsce, w którym pracują obecni zatrudnieni, i próbując wyobrazić sobie samego siebie w tym otoczeniu. Jeśli przestrzeń biurowa jest atrakcyjna, jeśli jest olśniewająca, jeśli znajduje się w ładnym sąsiedztwie oraz jeśli wszystko jest nowe i czyste, kandydat z pewnością odniesie pozytywne wrażenie. Jeśli jednak biuro jest zatłoczone, dywany

brudne i wytarte, a na latami niemalowanych ścianach wiszą plakaty z wioślarzami i wielkim napisem PRACA ZESPOŁOWA, nie możemy liczyć na pozytywne skojarzenia — kandydat przypomni sobie raczej perypetie Dilberta.

Wiele osób zajmujących się nowoczesnymi technologiami w ogóle nie zwraca uwagi na ogólne warunki pracy we własnym biurze. Co ciekawe, nawet osoby dostrzegające urok przestrzeni biurowych w innych firmach (odwiedzanych na przykład w trakcie rekrutacji) nierzadko pozostają ślepe na istotne braki w tym względzie we własnych firmach — wielu programistów jest po prostu przyzwyczajonych do tych niedociągnięć.

Warto wczuć się w rolę naszych kandydatów i uczciwie odpowiedzieć sobie na następujące pytania:

- Co kandydaci sądzą o miejscu, w którym znajduje się nasze biuro? Jak reagują na Buffalo, a jak na przykład na Austin? Czy programiści naprawdę są skłonni do przeprowadzki do Detroit? Czy firmy z siedzibą w Buffalo czy Detroit mogą sobie pozwolić na organizowanie rozmów kwalifikacyjnych we wrześniu?
- Kiedy wchodzą do naszego biura, jakie jest ich pierwsze wrażenie? Co widzą? Czy ich oczom ukazuje się czyste i ekscytujące miejsce? Czy przechodzą przez hall z żywymi palmami i fontanną, czy raczej czują się jak w poczekalni przed gabinetem dentystycznym w dzielnicy slumsów z więdnącymi kwiatami i starymi numerami „Newsweeka”?
- Jak wygląda właściwa przestrzeń biurowa? Czy wszystko lśni i pachnie nowością? A może wciąż trzymamy na ścianie wielki, pożółkły plakat Teamu Banana wydrukowany jeszcze na papierze harmonijkowym za pomocą drukarki igłowej (w czasach, kiedy korzystano z takiego papieru i drukarek)?

- Jak wyglądają biurka programistów? Czy każdy z nich dysponuje wieloma wielkimi, płaskimi ekranami, czy jednym starym monitorem kineskopowym? Czy siedzą na fotelach Aeron, czy zwykłych fotelach Staples Specials?

Chciałbym przy tej okazji poświęcić chwilę słynnemu fotelowi Aeron firmy Herman Miller. Taki fotel kosztuje około 900 dolarów, czyli o dobre 800 dolarów więcej niż zwykłe fotele biurowe w sieci Office Depot czy Staples.

Fotele Aeron są *nieporównanie* wygodniejsze od tanich foteli biurowych. Wybór właściwego rozmiaru i prawidłowe dostosowanie ustawień fotela sprawia, że większość ich właścicieli może w nich siedzieć cały dzień bez wrażenia dyskomfortu. Oparcie i siedzisko mają postać drobnej siatki, która eliminuje problem pocenia się. Ich ergonomia, szczególnie w przypadku nowszych modeli z dodatkowym wsparciem odcinka lędźwiowego, jest wprost doskonała.

Fotele Aeron są trwalsze od tanich foteli biurowych. Moja firma istnieje od sześciu lat i wszystkie nasze fotele wyglądają jak nowe. Zdarza mi się nawet prosić swoich gości o wskazanie różnic między fotelami kupionymi w 2000 roku i takimi, które mamy w firmie od zaledwie trzech miesięcy. Fotele bez trudu wytrzymują dziesięć lat intensywnej eksploatacji. Tanie fotele zaczynają szwankować już po około miesiącu. W czasie korzystania z jednego fotela Aeron zwykle trzeba kupić przynajmniej cztery tanie fotele w cenie 100 dolarów.

Oznacza to, że fotel Aeron kosztuje tylko o około 500 dolarów więcej niż jego najtańsi konkurenci, co przy trwałości na poziomie dziesięciu lat daje różnicę 50 dolarów rocznie. To tylko jeden dolar tygodniowo na każdego programistę.

To mniej więcej tyle, ile kosztuje rolka dobrego papieru toaletowego. Nasi programiści prawdopodobnie zużywają właśnie jedną taką rolkę tygodniowo.

Oznacza to, że podarowanie pracownikom foteli Aeron kosztuje dokładnie tyle, ile wydajemy na *papier toaletowy*. Zapewniam, że gdyby ktokolwiek próbował zakwestionować wydatki na papier toaletowy podczas dyskusji nad budżetem firmy, natychmiast zostałby przywołany do porządku za zbaczanie na nieistotne tematy — grupa odpowiedzialna za budżet ma przecież nieporównanie ważniejsze kwestie do omówienia.

Do fotela Aeron niesłusznie przylepiono łatkę mebla snobistycznego i ekstrawaganckiego (szczególnie w przypadku firm dopiero rozpoczynających działalność). Fotel Aeron jest wymieniany wśród symboli bumu i późniejszego krachu inwestycyjnego związanego z dot-comami, co jest o tyle niesprawiedliwe, że sam fotel nie jest szczególnie drogi, zważywszy na jego trwałość. W praktyce, jeśli przeanalizujemy czas spędzany na tym fotelu, uwzględnimy wsparcie lędźwiowego odcinka kręgosłupa i doskonałe *wykończenie*, fotel okaże się na tyle tani, że jego zakup będzie wprost doskonałą *inwestycją*.

Gadżety

Z podobnym zjawiskiem mamy do czynienia w przypadku pozostałych gadżetów cenionych przez programistów. Nie ma powodu, by rezygnować z zakupu najwydajniejszych komputerów i przynajmniej dwóch 21-calowych monitorów LCD (lub jednego 30-calowego) dla zatrudnionych programistów. Warto też umożliwić programistom swobodne kupowanie wszystkich tych książek technicznych w księgarni internetowej *Amazon.com*, które uznają za przydatne w swojej pracy. Wymienione aspekty nie tylko — co oczywiste — podnoszą produktywność zatrudnionych, ale też stanowią istotne narzędzia w procesie rekrutacji, szczególnie w czasach, w których większość przedsiębiorstw traktuje programistów jak wymienialne pionki, trybiki maszyny odpowiedzialne tylko za wpisywanie czegoś na klawiaturze. Wielu menedżerów dziwi się, co jest złego w 15-calowych monitorach CRT, i zaczyna snuć opowieści o swoich komputerach z dzieciństwa.

Życie towarzyskie programistów

Twórcy oprogramowania nie różnią się od zwykłych ludzi. Oczywiście zdaję sobie sprawę z tego, że w powszechnym odczuciu programiści są postrzegani jako osoby upośledzone społecznie, niezdolne do budowy relacji międzyludzkich. Z doświadczenia wiem, że to nieprawda — nawet osoby z zespołem Aspergera zwracają uwagę na społeczny wymiar swojego otoczenia w miejscu pracy, w tym następujące aspekty:

Jak traktuje się programistów w ramach organizacji?

Czy są traktowani na specjalnych warunkach, czy raczej postrzega się ich jako wyrobników przykutych do klawiatury? Czy

w kierownictwie firmy są inżynierowie lub osoby z doświadczeniem programistycznym? Czy programiści wysyłani na konferencje lecą pierwszą klasą? (Nie interesuje mnie, czy to sprawia wrażenie wyrzucania pieniędzy w błoto. Gwiazdy zawsze latają pierwszą klasą. Warto się do tego przyzwyczaić). Czy kandydaci zapraszani na rozmowy kwalifikacyjne są odbierani z lotniska i przywożeni do firmy limuzyną, czy muszą sami szukać sposobu dotarcia do biura potencjalnego pracodawcy? Jeśli oferty pracy w dwóch firmach będą zbliżone, możemy być pewni, że kandydat wybierze pracodawcę, który traktuje go jak gwiazdę. Jeśli prezes firmy jest zrzędą, który całe życie pracował w dziale sprzedaży i nigdy nie zrozumie, dlaczego najlepsi programiści mają prawo żądać takich luksusów jak podkładki pod nadgarstki, wielkie monitory czy wygodne fotele, taka firma najprawdopodobniej wymaga daleko idących zmian. Trudno przecież oczekiwać, by najlepsi programiści chcieli pracować w firmie, która ich nie szanuje.

Z kim pracują?

Jednym z najważniejszych aspektów, na które zwracają uwagę programiści podczas rozmów kwalifikacyjnych, są osoby, z którymi się spotykają. Czy są miłe? Czy — co ważniejsze — są inteligentne? Odbywałem kiedyś letnie praktyki w firmie Bellcore, spółce córce Bell Labs, w której każda napotkana osoba ciągle powtarzała: „Najlepszym aspektem pracy w Bellcore są zatrudnieni tutaj ludzie".

Jeśli więc firma zatrudnia programistów, którzy wiecznie mają zły humor i narzekają na otaczający ich świat, i jeśli z jakiegoś względu nie można się ich pozbyć, warto chociaż ukryć ich przed kandydatami w dniu rozmów kwalifikacyjnych. Jeśli w tej samej firmie pracują osoby urocze, pomocne, o miłym usposobieniu, należy za wszelką cenę włączyć je do procesu rekrutacji. Musimy stale

pamiętać, że kiedy nasz kandydat uda się do domu i będzie musiał podjąć decyzję o wyborze miejsca pracy, z pewnością nie będzie miał dobrych wspomnień z firmy, w której spotkał samych ponuraków.

Warto przy tej okazji wspomnieć o zasadzie rekrutacji pracowników początkowo obowiązującej w firmie Fog Creek (i zaczerpniętej z firmy Microsoft), czyli: „inteligentny, realizujący wyznaczone zadania". Już przed rozpoczęciem właściwej działalności zdaliśmy sobie sprawę z tego, że powinniśmy uzupełnić tę zasadę o trzecią regułę: „nie palant"[7]. W przeszłości bycie palantem nie było sprzeczne z wymaganiami stawianymi kandydatom do pracy w firmie Microsoft, choć jestem pewien, że choćby podświadomie zwracali uwagę na usposobienie kandydatów. Inna sprawa, że nigdy nie dyskwalifikowali kandydatów za samo bycie palantem, ponieważ nierzadko właśnie ta cecha bodaj najbardziej predestynuje do awansu na stanowiska dyrektorskie. Z biznesowego punktu widzenia zatrudnienie palanta nie stanowi wielkiego problemu, choć na pewno utrudnia rekrutację kolejnych pracowników. Kto chciałby pracować w firmie tolerującej palantów?

Niezależność i autonomia

Kiedy w roku 1999 odchodziłem z pracy w Juno (przed założeniem firmy Fog Creek Software), odbyłem standardową rozmowę z pracownikiem działu HR. Do dzisiaj nie wiem, jak to się stało, ale wpadłem w pułapkę zastawioną przez rozmówcę i opowiedziałem mu o wszystkich swoich zastrzeżeniach dotyczących sposobu zarządzania firmą. Doskonale wiedziałem, że moja wylewność nie przyniesie mi najmniejszych korzyści, a jedynie może mi istotnie zaszkodzić. Mimo to zdecydowałem się na opisanie irytującego stylu zarządzania firmą Juno, który można by opisać słowami „uderz

[7] Robert I. Sutton, *The No Asshole Rule: Building a Civilized Workplace and Surviving One That Isn't*, Warner Business Books, New York, 2007.

i uciekaj". Przez większość czasu szeregowi pracownicy samodzielnie realizowali swoje zadania, bez udziału menedżerów, ale też zdarzało się, że kierownictwo wykazywało niezdrowe zainteresowanie jakimś mikroskopijnym szczegółem i wymuszało na podwładnych realizację tego aspektu w określony sposób (bez żadnych wyjaśnień). Niedługo potem ten sam schemat powtarzał się w przypadku innego szczegółu, jednak nigdy nie starczało menedżerom czasu, by śledzić skutki swoich groteskowych decyzji. Pamiętam na przykład wyjątkowo stresujący okres, w którym przez dwa czy trzy dni wszyscy moi przełożeni, od bezpośredniego kierownika po samego prezesa, narzucali mi precyzyjny sposób wpisywania dat w kwestionariuszu rejestracji użytkowników na witrynie firmy Juno. Nie mieli pojęcia o projektowaniu interfejsu użytkownika i nie znaleźli dla mnie wystarczająco dużo czasu, abym mógł im wyjaśnić przyczyny, dlaczego zdecydowałem się na określone rozwiązanie. To wszystko było dla nich bez znaczenia — nie zawracali sobie głowy rzeczywistymi problemami i nie byli skłonni choćby wysłuchać moich argumentów. Decyzję o wyborze określonego rozwiązania podjęto na spotkaniu z prezesem, *na które nikt mnie nie zaprosił*.

W największym uproszczeniu, jeśli ktoś planuje zatrudnienie inteligentnych pracowników, powinien umożliwić im praktyczne wykorzystywanie swoich umiejętności podczas realizacji zlecanych zadań. Menedżerowie mogą oczywiście dawać rady — te są mile widziane — ale powinni za wszelką cenę unikać „rad" interpretowanych jako polecenia, ponieważ w zdecydowanej większości przypadków ich wiedza jest mniejsza niż wiedza podwładnych (szczególnie jeśli firma zatrudnia właściwych ludzi).

Programiści chcą być zatrudniani z uwagi na swoje umiejętności, chcą być traktowani jak eksperci i chcą mieć możliwość podejmowania decyzji na podstawie swojej najlepszej wiedzy.

Żadnej polityki

W praktyce problem polityki występuje zawsze wtedy, gdy w jednym miejscu gromadzą się przynajmniej trzy osoby. Co więcej, polityka może być zupełnie nieszkodliwa. Kiedy mówię „żadnej polityki", w rzeczywistości mam na myśli zakaz „żadnej polityki dysfunkcyjnej". Programiści są wyjątkowo wrażliwi na problem niesprawiedliwości. Kod albo działa, albo nie działa. Nie ma sensu odwoływanie się do jakiejkolwiek argumentacji na temat istniejącego lub wyimaginowanego błędu — wszystko można sprawdzić za pomocą stosownych testów. Świat programowania jest wyjątkowo precyzyjnie uporządkowany. Wiele osób decyduje się na programowanie właśnie dlatego, że woli spędzać czas w uporządkowanym środowisku z jasnymi regułami — w ustroju merytokracji, gdzie każdy spór wygrywa ten, kto po prostu ma *rację*.

Właśnie takie środowisko należy stworzyć w firmie, aby skutecznie przyciągać programistów. Kiedy programista skarży się na „politykę", w rzeczywistości ma na myśli dowolną sytuację, w której relacje osobiste biorą górę nad argumentami technicznymi. Nic tak nie denerwuje programisty jak nakaz użycia określonego języka programowania, który nie jest najlepszym wyborem w przypadku realizowanego zadania, a tylko ulubionym językiem przełożonego. Nic tak nie wkurza pracowników jak awanse wyłącznie za zdolność budowania relacji z przełożonymi zamiast za osiągnięcia stricte merytoryczne. Nic tak nie drażni programisty jak zmuszanie do stosowania technicznie niefortunnych rozwiązań tylko dlatego, że tak decyduje ktoś wyżej postawiony lub ktoś dysponujący lepszymi kontaktami w organizacji.

Nic nie daje satysfakcji podobnej do tej, którą odczuwa zwycięzca dyskusji na argumenty wynikające wyłącznie z wiedzy technicznej dyskutanta (zwłaszcza wtedy, gdy ten sam rozmówca przegrałby na płaszczyźnie politycznej). Kiedy zaczynałem pracę

w firmie Microsoft, realizowano tam poważny, choć chybiony projekt oznaczony nazwą kodową MacroMan. Projekt polegał na stworzeniu graficznego języka programowania opartego na makrach. Nowy język nie tylko byłby wyjątkowo niewygodny dla zwykłych programistów, ponieważ jego graficzny charakter tylko utrudniałby implementowanie pętli czy wyrażeń warunkowych, ale też nie stanowiłby żadnego ułatwienia dla osób niebędących programistami, które — jak sądzę — i tak nie myślą w kategoriach algorytmów i miałyby ogromne problemy ze zrozumieniem choćby podstaw języka MacroMan. Kiedy zasugerowałem to mojemu bezpośredniemu przełożonemu, odpowiedział: „Nic już nie może wykoleić *tego* pociągu. Poddaj się". Nie ustawałem jednak w swoich wysiłkach, przytaczając kolejne argumenty (co ważne, byłem wtedy absolwentem college'u, który nie dysponował praktycznie żadnymi znajomościami w firmie Microsoft), aż wreszcie moi rozmówcy przyznali mi rację i zamknęli cały projekt MacroMan. Nie miało znaczenia, kim jestem — liczyło się tylko to, że mam rację. Właśnie takie niepolityczne organizacje cieszą się największym uznaniem wśród programistów.

Ostatecznie dbałość o dynamikę relacji społecznych w ramach organizacji jest kluczem do stworzenia zdrowego, przyjemnego miejsca pracy, które nie tylko zniechęci już zatrudnionych programistów do poszukiwania alternatywy, ale też przyciągnie nowych pracowników.

Nad czym pracuję?

Jednym z najlepszych sposobów przyciągania dobrych programistów jest umożliwienie im pracy nad interesującymi ich zagadnieniami.

W pewnych przypadkach zmiana tego aspektu może być wyjątkowo trudna — jeśli na przykład nasza firma tworzy oprogramowanie dla przemysłu żwirowego i piaskowego, trudno udawać

przed kandydatami, że właśnie zaczynamy działalność w obszarze najatrakcyjniejszych technologii internetowych.

Programiści lubią też pracować nad rozwiązaniami na tyle prostymi lub na tyle popularnymi, że mogą bez trudu wytłumaczyć swoje zadania cioci Genowefie podczas świątecznego spotkania rodzinnego. Ciocia Genowefa, która na co dzień pracuje na stanowisku fizyka jądrowego, oczywiście nie zna się na programowaniu w języku Ruby na potrzeby branży żwirowej i piaskowej, za to ma bzika na punkcie symulatorów dynamiki cieczy.

I wreszcie, wielu programistów coraz częściej zwraca uwagę na społeczny wymiar działalności firmy, w której pracuje. Praca w firmach tworzących portale społecznościowe czy blogi pozwala zbliżać ludzi i nie prowadzi do żadnych szkód społecznych, zatem jest popularna; na drugim biegunie jest praca w przemyśle wojskowym lub wątpliwych etycznie firmach słynących z nadużyć księgowych.

Nie jestem niestety pewien, czy przeciętny menedżer odpowiedzialny za rekrutację ma jakikolwiek wpływ na ten aspekt oceny swojego przedsiębiorstwa. Zawsze można podjąć próbę zmiany wizerunku firmy na przykład poprzez wprowadzenie na rynek nowego, atrakcyjnego produktu, jednak w dłuższej perspektywie takie rozwiązanie nie zdaje egzaminu. Istnieją jednak dwa możliwe kroki w tym obszarze, które są z powodzeniem stosowane w rozmaitych firmach:

Niech najlepsi nowo przyjęci sami wybierają swoje projekty

Przez wiele lat w firmie Oracle Corporation stosowano program nazwany MAP (od ang. *Multiple Alternatives Program* — program wielu alternatyw). Program był corocznie oferowany najlepszym absolwentom college'ów. Koncepcja polegała na możliwości przyjścia do firmy Oracle, spędzenia tam tygodnia lub dwóch na obserwowaniu pracy w tej firmie, uczestnictwie w spotkaniach wszystkich grup

rozpoczynających nowe projekty i możliwości wyboru dowolnego spośród tych projektów.

Także firma Google stwarza programistom możliwość wyboru spośród wielu realizowanych projektów — przyjęto tam model, który każdemu programiście pozwala poświęcać 20 procent czasu pracy na wybrany przez siebie projekt (bez konieczności akceptacji tego wyboru ze strony przełożonych).

Stosowanie ciekawych, nowych technologii (nawet zbytecznych)

Wielkie banki inwestycyjne z Nowego Jorku uważa się za wyjątkowo kiepskie miejsce pracy dla programistów. Warunki, w których pracują, są wprost nieludzkie — siedzą oni wiele godzin w jednym miejscu, w którym nie mogą liczyć choćby na ciszę, a ich przełożeni są prawdziwymi tyranami. O ile programistów traktuje się w bankach jak obywateli trzeciej kategorii, o tyle ich koledzy (zwykle z wyjątkowo rozdętym ego) zajmujący się sprzedażą stanowią prawdziwą elitę korporacji, która może liczyć na wielomilionowe premie i tyle cheeseburgerów, ile tylko może zjeść (nierzadko przynoszonych przez znajdujących się w pobliżu programistów). To oczywiście tylko stereotyp, niemniej jednak banki inwestycyjne od pewnego czasu stosują dwie strategie mające na celu zatrzymanie eksodusu najlepszych programistów: albo wypłacają im niewyobrażalne pieniądze, albo dają im swobodę w kwestii wielokrotnego implementowania tych samych rozwiązań w nowych, popularnych językach programowania, których akurat ci programiści chcą się nauczyć. Chcecie przepisać cały system transakcyjny w Ruby? Niech będzie. Tylko przynieście mi tego cholernego cheeseburgera.

Niektórzy programiści przywiązują mniejszą wagę do używanego języka programowania, ale bardzo cenią sobie każdą okazję do pracy z nowymi, ekscytującymi technologiami. Obecnie tę funkcję

może pełnić Python bądź Ruby on Rails; trzy lata temu był to język C#, a jeszcze wcześniej wszyscy mówili tylko o Javie.

Chciałbym podkreślić, że nikogo nie namawiam do rezygnacji z narzędzi, które najlepiej nadają się do realizacji bieżących zadań, ani nie przekonuję do ponownego implementowania gotowych rozwiązań co dwa lata tylko dlatego, że pojawił się jakiś *nowy, popularny* język. Jeśli jednak możemy stworzyć programistom warunki do poznawania nowych języków, frameworków i technologii, z pewnością poprawimy w ten sposób komfort ich pracy. Nawet jeśli nie chcemy ryzykować tworzenia od zera całej podstawowej aplikacji, zwykle dysponujemy jakimiś wewnętrznymi aplikacjami lub mniej ważnymi nowymi aplikacjami, które można bez obaw zaimplementować od początku w nowym języku (w ramach poznawania nowych rozwiązań).

Czy identyfikuję się ze swoją firmą?

W większości programiści nie szukają tylko miejsca, w którym zarobią godziwe pieniądze. Nie oczekują „zwykłej pracy" — chcą raczej mieć świadomość, że pracują nad czymś naprawdę ważnym. Chcą identyfikować się ze swoją firmą. Dla młodych programistów szczególnie atrakcyjne są przedsiębiorstwa reprezentujące pewną ideologię. Wiele firm ma doskonałe kontakty ze środowiskiem twórców oprogramowania open source i ruchem wolnego oprogramowania, co może mieć spore znaczenie dla co bardziej idealistycznych programistów. Inne firmy aktywnie uczestniczą w programach społecznych lub tworzą produkty postrzegane (z tego czy innego powodu) jako prospołeczne.

Zadaniem osoby odpowiedzialnej za rekrutację jest identyfikacja tych idealistycznych aspektów funkcjonowania przedsiębiorstwa i dbanie o to, aby zapraszani kandydaci mieli świadomość ich istnienia.

Niektóre firmy decydują się nawet na inicjowanie własnych ruchów ideologicznych. Na przykład pewna początkująca firma z rejonu Chicago, nazwana 37signals, włożyła sporo wysiłku w to, by jej działalność kojarzyła się z prostotą — w ten sposób promowała proste, łatwe w użyciu aplikacje, na przykład Backpack, oraz prosty, łatwy w użyciu framework programistyczny Ruby on Rails.

Dla firmy 37signals prostota jest jak ustrój — swoisty międzynarodowy ruch polityczny. Prostota to nie tylko prostota, o nie! To słoneczne lato, kojąca muzyka, spokój, sprawiedliwość, szczęście i piękne dziewczęta z kwiatami we włosach. David Heinemeier Hansson, twórca frameworku Rails, twierdzi, że historia jego firmy to „Piękno, szczęście i motywacja. To zdolność odczuwania dumy i przyjemności z pracy i tworzonych narzędzi. Nasza historia to nie jakiś kaprys — to raczej trend. To historia wprowadzająca takie słowa jak pasja czy entuzjazm do codziennego słownika programistów bez konieczności usprawiedliwiania czegokolwiek czy odczuwania zakłopotania z powodu robienia tego, co się naprawdę lubi”[8]. Mówienie o frameworku do tworzenia aplikacji internetowych w kontekście „piękna, szczęścia i motywacji” może sprawiać wrażenie czczych przechwałek, jednak wydaje się wyjątkowo atrakcyjne i z pewnością wyróżnia tę firmę na tle konkurencji. Promowanie frameworku Ruby on Rails jako pochodnej szczęścia jego twórców ma tę praktyczną zaletę, że przynajmniej część programistów będzie żywo zainteresowana pracą nad jego rozwojem.

Okazuje się jednak, że firma 37signals stawia dopiero pierwsze kroki na gruncie kampanii zarządzania tożsamością. Pracownicy tej firmy *nie mogą się równać* ze specjalistami z koncernu Apple Computer, którzy za pomocą jednej reklamy w przerwie meczu Super Bowl w roku 1984 zbudowali trwały wizerunek (utrzymywany

[8] David Heinemeier Hansson, *Rails steps into year three*, *http://loudthinking.com/arc/2006_08.html*, 6 sierpnia 2006 r.

w świadomości konsumentów *do dzisiaj*) siły kontrkultury przeciwstawiającej wolność dyktatowi, swobodę uciskowi, kolory czerni i bieli oraz piękną kobietę w czerwonych szortach wymuskanym facetom w garniturach. Sprawne zarządzanie wizerunkiem prowadzi do zjawisk godnych dzieł Orwella — gigantyczne korporacje manipulują sposobem postrzegania swojej działalności w sposób, który z pozoru ma niewiele wspólnego ze zdrowym rozsądkiem (skoro produkują komputery, co, u licha, ma oznaczać ich walka z dyktatem?), skutecznie tworząc kulturę tożsamości, dzięki której nabywcy komputerów mają wrażenie *uczestnictwa w jakimś wielkim, ogólnoświatowym ruchu*. Kiedy kupujemy iPoda, możemy być pewni, że wspieramy w ten sposób walkę Gandhiego z kolonializmem brytyjskim. Każdy kupiony MacBook to dowód na sprzeciw wobec tyranii i głodu na świecie!

Weźmy teraz głęboki oddech... W tym podrozdziale chciałem zwrócić uwagę na konieczność analizy skojarzeń budzonych przez naszą firmę, sposobu jej postrzegania i potencjalnych działań na rzecz poprawy wizerunku. Właściwe zarządzanie marką korporacji jest równie istotne z perspektywy rekrutacji, jak z perspektywy kampanii marketingowych.

Jedna kwestia, która nie interesuje programistów

Programiści nie myślą tylko o pieniądzach, chyba że kompletnie zaniedbamy pozostałe aspekty. Jeśli więc od jakiegoś czasu słyszymy skargi (które nie pojawiały się wcześniej) na zbyt niskie wynagrodzenia, możemy uznać to za sygnał sugerujący, że nasi programiści nie kochają swojej pracy. Jeśli potencjalni kandydaci na programistów nie są skłonni do najmniejszych ustępstw podczas negocjowania wynagrodzenia, najprawdopodobniej mamy do czynienia

z następującą postawą: „Cóż, jeśli już muszę kompromitować się pracą w tej firmie, powinni mi przynajmniej odpowiednio zapłacić".

To oczywiście nie oznacza, że możemy sobie pozwolić na zbyt niskie wynagradzanie pracowników, ponieważ programiści są wrażliwi na niesprawiedliwość. Jeśli odkryją, że pracownicy innych firm otrzymują nieporównanie wyższe wynagrodzenie za tę samą pracę lub że pensje w naszej firmie są na przykład o 20 procent niższe niż w firmie za rogiem, kwestia pieniędzy z dnia na dzień urośnie do miana poważnego problemu. Nasze wynagrodzenia muszą być konkurencyjne, jednak same płace (o ile, o czym wspominałem, są uczciwe) zajmują zaskakująco niską pozycję na liście czynników decydujących o wyborze pracodawcy i jako takie stanowią zadziwiająco nieefektywne narzędzie do radzenia sobie z takimi problemami jak wysłużone monitory 15-calowe, marudzący pracownicy działu sprzedaży czy zaangażowanie w produkcję broni nuklearnej przenoszonej przez płetwonogi.

SKOROWIDZ

N

O

P

R

S

Ś

T

U

W

Z

Ź

Ż

Jak zaoszczędzić na programistach?

Menedżerowie, przedsiębiorcy i szefowie działów kadr wciąż toczą wewnętrzną walkę, czy zatrudnić świetnego, choć kosztownego programistę, czy przyoszczędzić i wybrać kogoś przeciętnego. Ci, którzy wybierają opcję numer 2, szybko przekonują się, że oszczędzanie na programistach skutkuje oprogramowaniem kiepskiej jakości, a co za tym idzie, korzyści są iluzoryczne.

Po co komu dobry programista?

Największym problemem związanym z zatrudnianiem wielu przeciętnych programistów w miejsce kilku naprawdę dobrych jest to, że niezależnie od czasu poświęconego na realizację zadań przeciętni programiści nigdy nie stworzą kodu, którego jakość będzie choćby zbliżona do jakości kodu najlepszych programistów. Pięciu Salierich nigdy nie stworzy dzieła na miarę *Requiem* Mozarta, nawet gdyby pracowali nad nim sto lat.

Po co komu dobry produkt?

Nie możemy pozwolić sobie na to, by konkurencja nas prześcignęła, czy na tworzenie produktów zaledwie wystarczająco dobrych. Nasze dzieła muszą być niesamowite, by przyciągały uwagę potencjalnych użytkowników. Największym prezentem, który możemy otrzymać od naprawdę utalentowanych programistów, jest właśnie nadzieja na stworzenie czegoś zauważalnego.

Dlaczego trzeba przeczytać tę książkę?

Proces zatrudniania największych talentów technicznych jest jak zupełnie pozbawiony elementów zabawy tor przeszkód. Każdy, kto kiedykolwiek zastanawiał się, na co zwracać uwagę podczas przeglądania aplikacji, kto bił się z myślami po zakończeniu rozmowy kwalifikacyjnej lub kto nie może pojąć, dlaczego tak trudno znaleźć doskonałych programistów, powinien rzucić wszystko i przeczytać tę książkę.

DODATEK SPECJALNY: Test Joela: 12 kroków ku lepszemu oprogramowaniu

Joel Spolsky jest absolwentem Uniwersytetu Yale i znanym na całym świecie ekspertem w dziedzinie procesów wytwarzania oprogramowania. Jego witryna internetowa — *www.joelonsoftware.com* — jest wyjątkowo popularnym miejscem, odwiedzanym przez programistów. Jako założyciel nowojorskiej firmy Fog Creek Software przyczynił się do powstania produktu FogBugz, czyli popularnego systemu zarządzania projektami, stworzonego z myślą o zespołach programistów. Pracował w firmie Microsoft, gdzie jako członek zespołu odpowiedzialnego za arkusz kalkulacyjny Excel projektował język VBA, oraz w firmie Juno Online Services, gdzie uczestniczył w pracach nad klientem internetowym, wykorzystywanym obecnie przez miliony użytkowników.

Nr katalogowy: 6017

Księgarnia internetowa
http://helion.pl

Zamówienia telefoniczne:
0 801 339900
0 601 339900

helion.pl
księgarnia internetowa

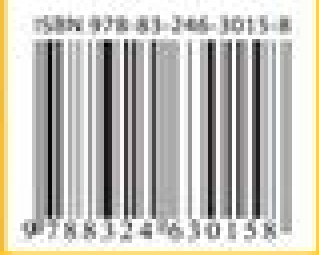
Sprawdź najnowsze promocje:
http://helion.pl/promocje
Książki najchętniej czytane:
http://helion.pl/bestsellery
Zamów informacje o nowościach:
http://helion.pl/nowosci

Helion

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
http://helion.pl

Cena 37,00 zł

ISBN 978-83-246-3015-8



Informatyka w najlepszym wydaniu