Scrum

Praktyczny przewodnik dla początkujących

Mitch Lacey

Helion

Tytuł oryginału: The Scrum Field Guide: Practical Advice for Your First Year

Tłumaczenie: Arkadiusz Romanek

ISBN: 978-83-246-8250-8

Authorized translation from the English language edition, entitled: THE SCRUM FIELD GUIDE: PRACTICAL ADVICE FOR YOUR FIRST YEAR; ISBN 0321554159; by Mitch Lacey; published by Pearson Education, Inc, publishing as Addison Wesley.

Wydawnictwo HELION
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/scrump
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

Spis treści

Rozdział 1

SCRUM: PROSTE, ALE NIE ŁATWE

Scrum jest zwinnie zwodniczy. To jedna z tych metod, które wydają się najłatwiejsze do opanowania, a okazują się najtrudniejsze do właściwego wdrożenia. Używam tu zwrotu „właściwe wdrożenie", ponieważ wrodzona prostota Scruma może nas zwieść i sprawić, że zaczniemy uważać tę metodykę za rozwiązanie, które łatwo zastosujemy w naszym otoczeniu, podczas gdy w rzeczywistości, zanim wszystko zacznie działać tak, jak należy, mogą minąć lata. Reguły Scruma wydają się przeczyć wszystkiemu, czego nauczyliśmy się podczas wielu, wielu lat realizowania projektów w systemie kaskadowym. Bez wątpienia oduczenie się złych nawyków i przystosowanie się do nowej rzeczywistości musi zająć trochę czasu.

W dodatku umieszczonym na końcu tej książki wyjaśniam, jak działają mechanizmy napędzające Scrum. Jeśli nie znasz Scruma i nie masz zielonego pojęcia, jak wygląda praca w tej metodzie, proponuję zacząć od zapoznania się ze wspomnianym dodatkiem. Jeśli natomiast masz już podstawową wiedzę na ten temat, zapewne dobrze wiesz, że system jest stosunkowo prosty. Tak prosty, że wielu ludzi szybko stwierdza, iż niczego więcej się już nie nauczą, i od razu zaczyna modyfikować zasady systemu tak, aby lepiej dopasować go do swojego środowiska projektowego. Zbyt często okazuje się jednak, że w konsekwencji takich działań gubią właściwą drogę, zniechęcają się pierwszymi niepowodzeniami, a potem szukają pomocy. Ta książka ma być rzuconym im kołem ratunkowym. Historia opisana na następnych stronach pokazuje, jak szybko można doprowadzić do sytuacji, w której Scrum „rozłazi się w szwach", jeśli reguły metody zostaną wdrożone bezrefleksyjnie i przy braku solidnych podstaw — to znaczy bez pełnego zrozumienia podstawowych pojęć tak zwanych zwinnych metod zarządzania projektami odpowiadających za skuteczność metody takiej jak Scrum.

Historyjka

Jeff był trenerem metod zwinnych, którego zadaniem było wspieranie wdrożeń Scruma w dużej firmie programistycznej. Któregoś dnia otrzymał e-mail od Suzy, kierowniczki programu w podległym mu oddziale.

„Jeff, proszę cię o pomoc. Pracujemy ze Scrumem już przez mniej więcej sześć miesięcy, ale jakość naszego kodu nie poprawia się tak, jak bym chciała. Myślę, że potrzebujemy cię, żebyś porozmawiał z zespołem na temat programowania parami. W następny poniedziałek zaczynamy nasz tydzień planowania. Czy mógłbyś nas wtedy odwiedzić?".

Mężczyzna odchylił się na oparcie fotela. Omówienie programowania parami było stosunkowo prostym zadaniem. Mógłby zabrać na spotkanie Julie, przyjaciółkę, świetnego dewelopera

i doświadczonego praktyka zwinnych metod projektowania. Nie ma sprawy. A jednak dwa słowa z tego listu sprawiły, że usłyszał w głowie głośny dźwięk dzwonka alarmu. *Cały tydzień* planowania? Zgodnie ze wszystkimi regułami Scruma wymagane są dwa spotkania planowania sprintu, każde z nich nie dłuższe niż cztery godziny. Tymczasem w tym zespole planowanie trwa tydzień? Coś mówiło Jeffowi, że czeka go więcej pracy, a nie tylko wykład z programowania parami. Zapowiadał się całkiem interesujący poniedziałek.

Gdy Jeff i Julie pojawili się na spotkaniu, Suzy i jej zespół składający się z ośmiu ludzi czekali już na nich w sali konferencyjnej. Suzy przygotowała krótkie wprowadzenie. Potem Jeff i Julie przedstawili się, a następnie szybko przeszli do rozmowy na temat kwestii związanych z jakością kodu, które tak niepokoiły Suzy.

Na reakcję zespołu nie trzeba było długo czekać. Najpierw głos zabrał główny tester, Mike:

— Kod jest kiepski, bo *nie mamy* czasu, żeby go porządnie przetestować. Deweloperzy pracują nad nim aż do ostatniego dnia naszego czterotygodniowego sprintu. Sprint tworzenia kodu i jego testowania to ma być sprint tworzenia kodu *oraz* testowania. Tymczasem testy albo zostają wciśnięte na koniec sprintu albo przerzucamy je do sprintu integracyjnego.

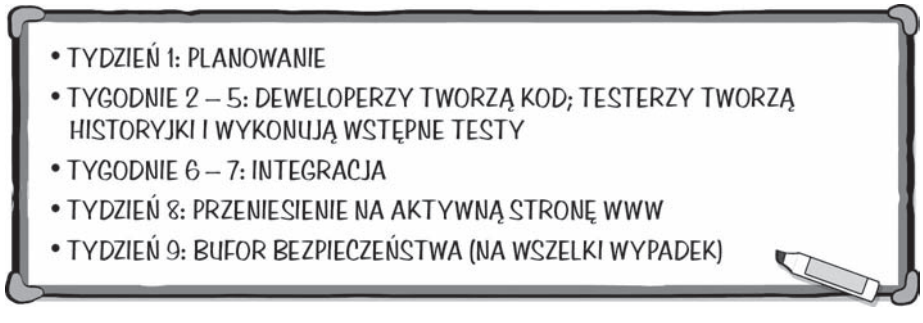
Julie przerwała Mike'owi:

— Przepraszam… powiedziałeś „sprintu integracyjnego"? — Julie spojrzała na Suzy, która kiwnęła głową.

— Och! Nie powiedziałam wam o naszych modyfikacjach, prawda? Otóż tak… wiemy, że Scrum wymaga przygotowania nowego wydania co cztery tygodnie. Jednak to nie jest możliwe w przypadku pracy, jaką wykonujemy. Chodzi o to, że przed przejściem na Scruma staraliśmy się wydawać nową wersję oprogramowania co kwartał i skończyło się kompletną katastrofą! Dlatego zmodyfikowaliśmy system tak, żeby lepiej dopasować go do naszego procesu i rzeczywistości — wyjaśniła Suzy, po czym podeszła do tablicy i zaczęła rysować.

— Zaczynamy od tygodnia planowania sprintu, po którym następują cztery tygodnie sprintu *rzeczywistego*, gdy deweloperzy piszą kod, a testerzy tworzą historyjki testowe. Po tym przeprowadzamy naszą integrację. Następnym etapem jest wdrożenie. Oczywiście zwykle dodaję tygodniowy bufor bezpieczeństwa… tak na wszelki wypadek — wyjaśniała.

Kiedy skończyła, niemal cała tablica została pokryta notatkami dotyczącymi harmonogramu prac zespołu.



Jeff i Julie spojrzeli po sobie, a potem skierowali oczy na Suzy. Reszta zespołu wyglądała na znudzoną. Jeff zadał oczywiste pytanie:

— Suzy, czy wasze sprinty naprawdę trwają osiem albo dziewięć tygodni?

— No tak… — odpowiedziała. — Wyglądasz na zaskoczonego. Wiem, że to nie przypomina klasycznego Scruma, ale w naszym przypadku się sprawdza. Myślę o jeszcze jednym tygodniu na pisanie specyfikacji i plany testów. Mamy problem z wyrobieniem się z tym na czas. Teraz robimy to w tygodniu buforowym, ale naprawdę nie lubię rezygnować z naszej poduszki bezpieczeństwa.

— OK, jeszcze do tego wrócimy — stwierdziła Julie. Spojrzała na Jeffa, który podniósł ręce do góry, jakby chciał spytać: „Co zamierzasz z tym zrobić?". Potem Julie zwróciła się do Mike'a: — Mówiłeś, że nie macie czasu na testowanie…

Zanim Julie usłyszała odpowiedź Mike'a, wtrącił się Wyatt:

— Nie słuchaj go. Mają mnóstwo czasu na testy. To my nigdy nie mamy wystarczająco dużo czasu. Robimy, co możemy, żeby w każdy sprint wcisnąć jak najwięcej kodu. A może my potrzebujemy wszystkich czterech tygodni? To czasochłonny proces. — Wyatt rzucił spojrzenie w kierunku Mike'a i kontynuował: — Od kiedy zaczęliśmy stosować Scrum, ty i reszta testerów przez cały czas tylko marudzicie, że nie macie czasu. Może problem leży w samym mechanizmie Scruma?

Jeff i Julie wymienili spojrzenia.

Suzy przerwała Wyattowi:

— Ludzie! Dajcie spokój. Nie jesteśmy tu po to, żeby narzekać na Scrum. Naszym celem jest poprawa jakości kodu. — Przerwała i wzięła głęboki oddech. — Powtarzam to od sześciu miesięcy — zwróciła się do Jeffa i Julie, przewracając oczami.

Jeff skinął głową:

— Widzę, że jesteś sfrustrowana. I widzę też, że Wyatt i Mike są równie mocno sfrustrowani. Czy mogę poznać więcej szczegółów i porozmawiać o tym z zespołem? Zobaczę, czy uda mi się zidentyfikować źródło problemu.

Suzy skinęła głową pełna entuzjazmu.

Jeff zaczął od pierwszego pytania standardowej oceny.

— W porządku… Zarówno Scrum, jak i reguły programowania ekstremalnego zalecają zastosowanie codziennych punktów kontrolnych. W tym przypadku mówię o Codziennym Scrumie. Co o nich myślicie? — Jeff zwrócił się do zespołu.

Mike wybuchnął śmiechem:

— *Codzienne* spotkania? Żartujesz sobie? Nie mamy na nie czasu. Spotykamy się dwa razy w tygodniu na godzinę. I nawet to nie wychodzi nam za dobrze.

— OK. Mike, opowiedz mi o tych spotkaniach — poprosił Jeff.

— No cóż… przede wszystkim za każdym razem odtwarzamy takie same dialogi. Deweloperzy twierdzą, że pracują nad zadaniami, a my mówimy, że budujemy historyjki testowe. Cóż za wspaniałe wieści! Potem spędzamy mniej więcej 20 minut, dyskutując o naszej liście błędów. Sprowadza się to do odczytania listy i stwierdzenia: „Ten błąd wynika z projektu" lub „Załatwimy to podczas kolejnego sprintu". Oczywiście nigdy nie eliminujemy tych błędów. To po prostu jeden wielki dysfunkcyjny bałagan.

Dało się zauważyć, że w Suzy wzbiera gniew. Dlatego Jeff dał jej możliwość wypowiedzenia się:

— Dzięki. Suzy, chciałabyś coś powiedzieć?

— Mike ma rację w jednej kwestii. W naszym przypadku codzienne spotkania by się nie sprawdziły. *Musimy* organizować je co drugi dzień. Mam zbyt wiele zajęć, żeby uczestniczyć

w nich co dzień, a niektórzy członkowie zespołu pracują także przy innych projektach. Wiem, że to nie jest idealne rozwiązanie, ale właśnie tak musimy robić. Wkurza mnie, że zespół się ze mną nie zgadza, uważając spotkania odbywające się dwa razy w tygodniu za zbyt wielkie obciążenie. Słyszałeś, co mówił Mike. Wszyscy narzekają na spotkania, harmonogram, brak czasu! A ja nic nie poradzę, że kierownictwo zmusza nas do przygotowywania częstszych wydań. Co więcej, powtarzam wszystkim, że to jest *mój* projekt. To *ja* ustalam plan i *ja* jestem od tworzenia struktury. Powiedz im, Jeff. Ja jestem Mistrzynią Młyna. Powinni robić to, co każę. Prawda? — Suzy domagała się wsparcia.

Jeff niezobowiązująco wzruszył ramionami i ugryzł się w język. Zaczynał zdawać sobie sprawę, że ten zespół tak naprawdę nie rozumie zasad Scruma. Popatrzył na Julie i rzucił jej jedno z tych spojrzeń, które znaczyły: „Oni zupełnie nie łapią, o co tu chodzi". Julie myślała podobnie, potwierdzając jego ocenę lekkim skinieniem głowy. Jeff przeszedł do dalszej analizy sytuacji:

— OK, rozumiem, co mówisz. Póki co nie zapędzajmy się za daleko. Wydaje mi się, że wasz problem wynika z tego, że codzienne spotkania wcale nie są codziennie, nic nie wnoszą i trwają za długo. Z tym możemy sobie jakoś poradzić. Ale zostawmy na chwilę ten temat i przejdźmy na wyższy poziom procesu myślowego. Powiedzcie mi, skąd wziął się pomysł ośmiotygodniowego sprintu?

Znów odezwał się Wyatt:

— Pracuję w tej firmie ponad dziesięć lat i wierzcie mi, widziałem już wszystkie nowe, chwilowe mody, jakie istnieją. Pojawiają się i znikają. Tym razem jednak dałem się przekonać i zacząłem wierzyć, że Scrum jest inny. Co za żart! Wszystko zaczęło się od tego, że szefostwo domagało się od nas coraz częstszych wydań. Przeszliśmy na system wersji pojawiających się kwartalnie. I powiem ci, że przejście z rocznych okresów na kwartalne było bolesne, ale operacja zakończyła się sukcesem. Tyle że szefom było za mało, prawda? — Wyatt przerwał na chwilę i rozejrzał się po pokoju, szukając wsparcia u innych członków zespołu. — Któregoś dnia podczas lunchu spotkaliśmy z Suzy mojego znajomego, który pracuje w innym dziale. Opowiedział nam, w jaki sposób jego zespół korzysta ze Scruma i jak udaje im się przygotowywać kolejne wersje oprogramowania co cztery tygodnie. Wszyscy byli zadowoleni. Jakość oprogramowania wystrzeliła w kosmos. Szefowie firmy od lat nie byli tak usatysfakcjonowani, a klienci podchodzili do całego zagadnienia wprost ekstatycznie. A tamten nasz znajomy był podobnym do mnie sceptykiem, wiesz? Pomyślałem więc, że jeśli on twierdzi, że to działa, to system naprawdę musi się sprawdzać. Spędziliśmy z Suzy i moim znajomym całe popołudnie, rozmawiając na temat Scruma. Metoda wydawała się dość prosta, ale widzieliśmy, że będziemy musieli rozwiązać kilka problemów. Po pierwsze: codzienne spotkania. Kto ma na nie czas? Prostym rozwiązaniem okazało się organizowanie ich dwa razy w tygodniu. Potem stwierdziliśmy, że cykl trwający cztery tygodnie w naszym przypadku się nie sprawdzi. W końcu ledwo udawało nam się zamykać projekty w cyklu kwartalnym, więc zdecydowaliśmy się podwoić te cztery tygodnie. I tak pojawił się pomysł sprintu ośmiotygodniowego. Na tej podstawie podzieliliśmy etapy procesu. Wystarczyło tylko przeprowadzić małe skalowanie. Przecież Scrum jest kolejnym procesem typu przyrostowego.

Jeff i Julie znów rzucili sobie wymowne spojrzenia.

Wyatt kontynuował:

— Widzę to spojrzenie. Ale mówię wam, że znamy nasz zespół i produkt. Nie ma mowy, żebyśmy mogli zastosować Scrum w wersji podstawowej. Zrobiliśmy to, co zrobiłby każdy zespół pracujący nad oprogramowaniem. Dostosowaliśmy system do naszych potrzeb. Nasza modyfikacja najlepiej pasuje do tego, co robiliśmy w przeszłości.

— Tak — potwierdziła Suzy. — Chodzi mi o to, że Scrum jest przecież metodą zarządzania projektem przez menedżerów, tyle że prowadzącą na skróty.

Jeff odchylił się na oparcie fotela, na którym siedział. W takiej sytuacji przygotowane przez niego pytania oceny do niczego się nie nadawały. Nie wiedział, co mógłby teraz powiedzieć. Z pomocą pospieszyła Julie:

— Powiedz mi, Wyatt, czy ustaliliście wspólną definicję ukończenia?

— Oczywiście, że tak. Po pierwszym tygodniu odbywa się spotkanie oceny projektu. Pod koniec piątego tygodnia docieramy do kamienia milowego ukończenia kodu. Na ostatnim etapie integracji wiemy, że zakończyliśmy etap testowania i pełnej integracji. Kiedy osiągamy te kamienie milowe, umieszczamy efekt naszej pracy na stronie internetowej. Naprawdę nietrudno zrozumieć nasze modyfikacje.

— Tak. Są proste. Rozumiem — powiedziała uspokajająco Julie. — Wyjaśniliście nam, jak wygląda wasz proces. Podsumujmy: odbywacie spotkania planowania dwa razy w tygodniu, zdecydowaliście się na ośmiotygodniowe (czasami dziewięciotygodniowe) sprinty i macie punkty oceny na niektórych kluczowych etapach, dzięki którym sprawdzacie poziom gotowości produktu. Jak to się sprawdza w praktyce? Dobrze się bawicie? Czy kod uległ znacznej poprawie?

— No cóż, kod nie jest taki zły… — powiedział Wyatt.

— Nie jest zły? — zaoponowała Suzy. — Jest okropny!

— To nie nasza wina, że jest okropny! Testujemy go, ale, jak powiedziałem, po prostu nie mamy dość czasu! — krzyknął Mike. Reszta ludzi pospuszczała głowy. Nie zamierzali brać udziału w tym starciu.

— Nie obwiniam cię, Mike, chociaż mógłbym. Nie. Prawdziwym problemem jest sam Scrum — powiedział Wyatt. — To głupia metoda. I nie działa.

— Tylko nie zaczynajcie od nowa — jęknęła Suzy. — Ile razy jeszcze będziemy o tym dyskutować? To powtarza się przy okazji każdej retrospektywy.

— Mówisz o retrospektywie? — przerwał jej Wyatt. — To tylko osobliwa nazwa dwudniowej sesji narzekań i marudzenia. Retrospektywy niczego nie zmieniają. Scrum niczego nie zmienia. A nie! Odwołuję to. Scrum zmienia jedną rzecz. Sprawia, że wszyscy jesteśmy nieszczęśliwi.

— To był też twój pomysł. Dlaczego zgodziłeś się na wdrożenie metody, jeśli teraz tylko ją sabotujesz? — pytała rozeźlona Suzy.

Jeff wstał. Nadszedł czas, aby przestać zadawać pytania i zrobić coś, żeby ten zespół zaakceptował prawdę.

— Słuchajcie, to nie niczyja wina i nie jest to też wina metody. Dla mnie, i jestem pewien, że Julie się ze mną zgodzi, wygląda na to, że wy w ogóle *nie robicie* Scruma. Robicie to, co zawsze, tyle że teraz wciskacie całą pracę w ośmiotygodniowy cykl. Takie zarządzanie projektem przypomina Scrum tylko z nazwy.

Wyatt i Suzy zaczęli się kłócić, ale Julie podniosła rękę, aby uspokoić zwaśnione strony.

— Pozwólcie mi zadać jedno pytanie. Adresuję je do całego zespołu. Wyatt, Suzy, Mike, proszę was o niezabieranie głosu. — Julie spojrzała po kolei na każdego członka zespołu. — Czy ten nowy sposób zarządzania sprawił, że wasze życie stało się nie do zniesienia, czy może zawsze tak było, tylko do tej pory nie wydawało się to takie oczywiste? — spytała.

Wszyscy pozwieszali głowy. Gdy ludzie zastanawiali się nad odpowiedziami, dotykali niemalże brodami piersi.

— Już wcześniej była kicha — zabrzmiał jakiś głos z tyłu sali.

— Tak jest — przyznał inny członek zespołu. — Wcześniej nie zdawaliśmy sobie tylko sprawy ze skali problemu.

Gdy wszyscy uświadomili sobie powagę sytuacji, w pomieszczeniu zapadła głucha cisza.

Wyatt westchnął i powiedział:

— Macie rację. Wcześniej wcale nie było lepiej. Po prostu nie było to takie oczywiste. Problem pojawiał się tylko raz na kwartał. A teraz cierpimy co osiem tygodni.

Głos zabrał Mike:

— Wiecie co? Jeśli spojrzeć wstecz na ostatnie sześć miesięcy, okazuje się, że odkryliśmy wiele rzeczy, które możemy poprawić i które trzeba naprawić, ale nic z tym nie zrobiliśmy. Naprawdę nic.

Całe spotkanie podsumowała Suzy:

— Chłopaki, jestem naprawdę zmęczona. Czy możemy to przełożyć i zebrać się ponownie w przyszłym tygodniu?

Ludzie z zespołu pokiwali głowami. Oni *naprawdę* byli zmęczeni.

Suzy opuściła salę konferencyjną ze świadomością, że sprawy nie idą w dobrym kierunku. Przez cały weekend i pierwsze dni następnego tygodnia zastanawiała się, jak może wpłynąć na poprawę sytuacji. Zaprosiła Jeffa i Julie na kolejne spotkanie, podczas którego — miała taką nadzieję — uda się tchnąć nowego ducha w członków zespołu i podnieść morale ludzi.

— Przykro mi — przyznała, otwierając spotkanie. — Wiedziałam, że nie wszystko nam się układa, ale nie zdawałam sobie sprawy, że jest aż tak źle. Najpierw miałam nadzieję, że Jeff i Julie pomogą nam w kwestii dopracowania programowania parami, bo wydawało mi się, że to rozwiąże nasze problemy z jakością. Okazuje się jednak, że nie dostrzegałam naszych rzeczywistych potrzeb. I za to przepraszam. Źle do tego podeszliśmy. Problemem nie jest Scrum, ale to, w jaki sposób próbujemy wdrożyć metodę. Chciałabym prosić wszystkich o zgodę, żeby zacząć wszystko od zera. Myślę, że Jeff i Julie mogą nam w tym pomóc — powiedziała Suzy.

Wyatt skinął głową i spojrzał na zespół.

— Wiem, że czasami zachowuję się jak palant. Jestem tu już tak długo, że chwilami wydaje mi się, że jestem właścicielem tej firmy. Nie jestem. Wiem, że stać mnie na więcej. Przestanę narzekać i naprawdę przyłożę się do roboty, ale zrobię to tylko pod jednym warunkiem.

— Jakim? — spytał Jeff.

— Że weźmiemy się do tego na poważnie. Nie chcę już nigdy słyszeć o dostosowywaniu. I potrzebujemy trenera. Kogoś, kto pokaże nam, co mamy robić. Metoda nie jest nawet w przybliżeniu tak prosta, na jaką wyglądała.

Podniósł się Mike.

— I ja mam warunek. Rozwiążemy problemy. Naprawdę je rozwiążemy, nikogo za nie nie obwiniając. — Wyciągnął rękę do Wyatta. — Myślisz, że damy radę?

Wyatt spojrzał na Mike'a, potem na jego rękę, którą po chwili uścisnął i potrząsnął nią.

— Myślę, że jesteśmy gotowi na to wyzwanie. Oczywiście jeśli Jeff może nam pomóc… — zażartował z lekkim szyderstwem w głosie, unosząc brwi i spoglądając na Jeffa.

Wszyscy wybuchli śmiechem. To był pierwszy prawdziwy śmiech, jakiego nie słyszano w zespole od dłuższego czasu. Nowy początek. Wszyscy zgromadzili się przy stole, wyrażając głośno swoją gotowość do podjęcia kolejnej próby ze Scrumem. Tym razem mieli zrobić to naprawdę. Jeff i Julie opuścili salę konferencyjną, czując, że udało im się wiele osiągnąć. Wiedzieli jednak, że nadal pozostało sporo do zrobienia.

— Co masz zamiar z tym zrobić, Jeff? — spytała Julie.

— Przede wszystkim zaczniemy od omówienia, czym jest Scrum. Najpierw wyjaśnię im, jakich zmian mentalnych będzie wymagać od nich nowy system. Opowiem im o wartościach i regułach Scruma — odpowiedział Jeff.

— Nie zapomnij pokazać im, jak wygląda zarządzanie ryzykiem i identyfikacja problemów.

— Nie zapomnę. Zacznę od podstaw, a potem pokonamy wszystkie przeszkody, w miarę jak będą się pojawiać. Oznacza to, że czasami czeka nas prawdziwa walka. Wierzę jednak, że to się uda. Bez pracy nie ma kołaczy, prawda?

Scrum

Scrum wydaje się prosty. Jednak ludzie często nie rozumieją, że aby właściwie wdrożyć cały system, trzeba całkowicie zmienić sposób myślenia o tworzeniu oprogramowania. A to nie jest łatwe. Trzeba będzie stanąć do walki. Radzić sobie z napotkanymi przeszkodami. Zespół z naszej historyjki przekonał się o tym na własnej skórze, a jeśli już chwyciłeś za tę książkę, to prawdopodobnie masz podobne doświadczenia. Żeby zrozumieć, dlaczego coś tak prostego może być tak trudne, przyjrzymy się teraz nieco regułom Scruma.

Czym jest Scrum?

Jednym z moich ulubionych teleturniejów telewizyjnych od zawsze był quiz *Jeopardy!*[1]. Wciąż marzę o specjalnej edycji przeznaczonej dla deweloperów oprogramowania, w której mogłyby pojawić się takie kategorie jak: *Metodologia i reguły metod*, *Typowe przyczyny błędów w oprogramowaniu*, *Słynni architekci oprogramowania* lub *Głupie cytaty mądrych ludzi*. Przychodzą mi do głowy dziesiątki pytań, które mogłyby trafić do tych kategorii. Na przykład: „Kogo uważa się za autora powiedzenia: *Bądź miły dla nerdów. Istnieje duże prawdopodobieństwo, że będziesz pracować dla jednego z nich*?". Wyobrażam sobie inne pytanie z kategorii *Metodologie i reguły metod*: „Nazwa tej metody zarządzania projektami programistycznymi wywodzi się z terminologii rugby, a jej podstawowa reguła zakłada wydanie nowej wersji oprogramowania co 2 – 4 tygodni". Jedna z akceptowalnych odpowiedzi miałaby oczywiście formę pytania: *Czym jest… Scrum?*

A zatem czym tak naprawdę jest Scrum? Scrum tak naprawdę *nie jest* metodyką ani zestawem praktyk programistycznych. Jest to natomiast lekki szkielet systemu stworzony z myślą o zarządzaniu projektem programistycznym i procesem rozwoju produktu. Oto jak definiują Scrum Ken Schwaber i Jeff Sutherland [SCHWABER 01]:

[1] Polska wersja teleturnieju, emitowana w latach 1996 – 2003, nosiła nazwę *Va banque — przyp. tłum.*

Scrum (rzecz.): metoda, przy użyciu której ludzie mogą z powodzeniem rozwiązywać złożone problemy adaptacyjne, by w sposób produktywny i kreatywny wytwarzać produkty o najwyższej możliwej wartości. Scrum jest:

- *lekki,*
- *łatwy do zrozumienia,*
- *bardzo trudny do opanowania.*

Scrum stanowi ramy wykorzystywane w zarządzaniu procesami wytwarzania złożonych produktów od początku lat dziewięćdziesiątych. Sam w sobie Scrum nie jest procesem czy techniką wytwórczą; opisuje jedynie ogólne sposoby postępowania, w obrębie których możliwe jest stosowanie różnego rodzaju procesów i technik. Scrum pomaga odkrywać nieefektywności praktyk zarządczych i technik inżynierskich tak, by można było je doskonalić[2].

Scrum bazuje na iteracyjnych cyklach zwanych *sprintami*. Każdy sprint rozpoczyna się spotkaniem nazywanym Planowaniem Sprintu, a kończy demonstracją produktu, który potencjalnie nadaje się do pracy. System charakteryzuje się wysokim poziomem interakcji uczestników i transparentnością działań — zarówno w ramach zespołu, jak i poza nim. Krótkie cykle oraz cecha systemu zakładająca ścisłą współpracę sprawiają, że Scrum idealnie nadaje się do zastosowania przy szybko zmieniających się projektach i wszędzie tam, gdzie często dochodzi do modyfikacji zgłoszonych wcześniej założeń lub (i) pojawiają się nowe potrzeby.

Scrum został zbudowany na bazie pięciu podstawowych wartości i przewiduje istnienie trzech ról, trzech artefaktów i trzech (lub czterech) typów spotkań. Więcej szczegółów na temat sposobu funkcjonowania Scruma znajdziesz w dodatku A, zatytułowanym „Przewodnik po Scrumie".

Zastosowanie Scruma

Chociaż Scrum może wydawać się systemem prostym do zastosowania w środowisku projektowym, tak naprawdę stanowi spore wyzwanie. Dlaczego? Ponieważ wymaga czegoś więcej niż tylko wprowadzenia odpowiednich mechanizmów i naciśnięcia przycisku start. Do poprawnego wdrożenia Scruma konieczne jest istnienie zespołów, które są gotowe na:

- Podjęcie wysiłku zrozumienia istoty podstawowych wartości Scruma.
- Zaakceptowanie (często ogromnej) zmiany sposobu myślenia i nastawienia.
- Przygotowanie się na zmiany oraz ciągłe adaptowanie się do nowej sytuacji.
- Reagowanie na pojawiające się problemy i działanie na rzecz ich rozwiązywania.
- Włączenie do swoich działań zasad zwinnego projektowania.

[2] Fragment pochodzi z oficjalnego polskiego tłumaczenia (wersja z lipca 2011 r.) przewodnika Scrum dostępnego pod adresem: *http://www.scrumguides.org* (dostęp w sierpniu 2013 r.). Autorami przekładu są: Tomasz Włodarek, Katarzyna Terlecka i Bogdan Baraszkiewicz — *przyp. tłum.*

Scrum bazuje na wartościach

Każda metoda, która zasługuje na zastosowanie, bazuje na zasadach i wartościach. Każda z oryginalnych praktyk określanych wspólnym mianem zwinnych — XP, Scrum, DSDM, Crystal i FDD — a także Kanban i Lean — wyróżnia się zestawem fundamentalnych wartości, które nas prowadzą, zapewniają jasną wizję wtedy, gdy opadają nas wątpliwości, i co najważniejsze, pomagają zrozumieć, *dlaczego robimy to, co robimy*. Historia przywołana w tym rozdziale opisuje sytuację, w której zespół programistów próbuje zastosować mechanizmy Scruma bez zastanowienia się nad tym, dlaczego tak postępuje. Bohaterom historyjki brakowało zrozumienia wartości, na których bazuje metoda: skupienia, szacunku, zaangażowania, odwagi i otwartości [SCHWABER 02].

Skupienie

„Skupić się" oznacza skoncentrować się, poświęcać czemuś uwagę. Jeśli efektem pracy zespołu ma być sprawny element przyrostu funkcjonalności, tworzący go ludzie muszą się skupić na tym, aby zrobić wszystko, co pozwoli im zrealizować ten cel. Skupienie oznacza skoncentrowanie się tylko na tym jednym, właśnie realizowanym projekcie. Może to oznaczać wydzielenie określonego czasu zespołu, gdy cała grupa nie zagląda do poczty e-mail, nie korzysta z komunikatorów, telefonów komórkowych i nie uczestniczy w innych spotkaniach. Skupienie na tym, co najważniejsze, oznacza robienie wszystkiego, co konieczne, aby umożliwić zespołowi skoncentrowanie się na pracy z myślą o jak największej efektywności na przestrzeni całego sprintu.

Szacunek

Wszyscy znamy powiedzenie, że na szacunek trzeba sobie zapracować i nikt nie otrzymuje go z urzędu. W przypadku Scruma to stwierdzenie jest wyjątkowo prawdziwe. Szacunek do kolegów, lub jego brak, może decydować o fiasku lub sukcesie projektu. Skuteczne w swoich działaniach zespoły scrumowe muszą składać się z ludzi darzących się wzajemnym zaufaniem, gotowych do wspólnego pokonywania wszystkich przeszkód. Jeśli jeden członek zespołu angażuje się w rozwiązanie problemu, to samo czynią inni. W naprawdę efektywnym zespole scrumowym nie ma konfliktów na linii *my* versus *oni*. W opisywanej na początku tego rozdziału historii wyraźnie rysuje się konflikt pomiędzy testerami oprogramowania i programistami. Przedstawiciele tych grup najwyraźniej zaczęli tracić do siebie szacunek. Na szczęście udało się nie dopuścić do pogorszenia się konfliktu, o czym świadczy fakt, że w końcu wszyscy podejmują wspólną decyzję o daniu sobie jeszcze jednej szansy.

Zaangażowanie

Zaangażowanie jest zapewnieniem lub obietnicą wykonania pewnej pracy. Te zobowiązania należy traktować poważnie, przy zapewnieniu jak największej ilości informacji. Zespół angażuje się i zobowiązuje wobec organizacji i wzajemnie — wobec innych członków grupy — podczas każdego spotkania planowania sprintu. Pod koniec spotkania wszyscy członkowie zespołu powinni znajdować się na tym samym poziomie zrozumienia tego, co zespół zobowiązuje się wykonać podczas sprintu.

Odwaga

Odwaga to gotowość stawienia czoła trudnościom na przekór swoim lękom. Jednym z najlepszych sposobów wspierania członków zespołów i organizacji, a także podejmowania aktywności promujących odważne decyzje, są aktywne działania podejmowane z myślą o pokonywaniu lęków. Zespół, który okazuje zrozumienie i nie lęka się szczerej dyskusji, albo organizacja, która udowadnia, że jest gotowa wysłuchać wszystkich (także tych negatywnych) opinii, podchodząc do przekazywanych informacji obiektywnie — takie postawy pozwalają promować odważne zachowania osób, które nie obawiają się mówić to, co myślą. Pamiętaj, że gdy zespołowi brakuje odwagi, aby robić to, co uważa za słuszne, mało prawdopodobne, iż zrobi to, co *powinno* być zrobione.

Otwartość

Otwartość pozwala nam przyjmować nowe idee. Poziom otwartości zespołu najwyraźniej widać podczas retrospektywy następującej po sprincie. Gotowość zaakceptowania nowych pomysłów, spostrzeżeń i sposobów podejścia do pewnych spraw pomaga nam rozwijać się w organizacji uczącej się, a także tworzyć bardzo efektywne zespoły.

Scrum wymaga zmiany sposobu myślenia

Albert Einstein powiedział kiedyś: „Nie możemy rozwiązywać problemów, odwołując się przy tym do schematów myślenia, z których te same problemy wyniknęły". Wśród największych przeszkód stojących na drodze do skutecznego wdrożenia systemu Scrum (lub każdej innej metody należącej do grupy zwinnych metod zarządzania projektami) wymienia się nieumiejętność zmiany nastawienia — to znaczy niezdolność do odwołania się do nowego sposobu myślenia podczas rozwiązywania problemów. W rezultacie zarówno Scrum, jak i wszystkie inne zwinne praktyki wymagają otwarcia się na nowe koncepcje — co najmniej przez pierwsze 3 – 6 miesięcy. Gdy pracowałem nad moim pierwszym projektem scrumowym, musiał minąć prawie rok, zanim naprawdę zrozumiałem, o co tak naprawdę chodzi w tej metodzie.

Przez ten rok zrozumiałem, że Scrum jest potężnym narzędziem, które jednak w niewłaściwych rękach może się okazać niebezpieczne. Czy widziałeś kiedyś, drogi Czytelniku, którykolwiek z odcinków serialu *Home Improvement*[3]? Główny bohater serialu, Tim „The Toolman" Taylor, zwykł wpadać w spore kłopoty, gdy zaczynał korzystać z jakiegoś lśniącego nowością narzędzia, ale nie stosował odpowiednich środków bezpieczeństwa i używał go niezgodnie z przeznaczeniem, lub po prostu zabierał się do pracy, która przerastała jego możliwości. Podobnie jest ze Scrumem. Jeśli nie stosuje się metodyki zgodnie z instrukcją, zwłaszcza na początku przygody ze Scrumem, można szybko pogrążyć każdy projekt. Przekonało się o tym (zbyt) wiele zespołów posiadających tylko podstawową wiedzę, ludzi, którzy uznali: „My wiemy lepiej. Tu jest inaczej. Wszystko przemyśleliśmy".

Weź sobie do serca moją radę: Zanim zdecydujesz o dostosowaniu Scruma do swoich potrzeb, poznaj wszystkie tajniki systemu. Jeśli zdecydujesz się na wdrożenie, użyj Scruma zgodnie z przeznaczeniem, w wersji podstawowej. Poświęć swój czas na dowiedzenie się wszystkiego, co musisz wiedzieć. Wygospodaruj nieco miejsca w mózgu, tak aby pozwolić tej wiedzy rozrastać

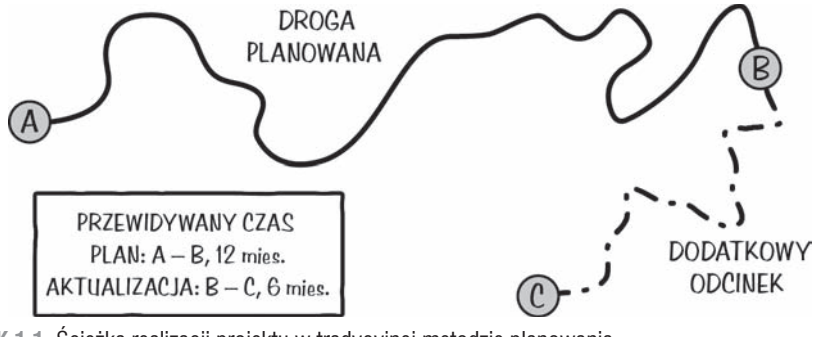
[3] W Polsce serial nadawany przez TVP1 pod tytułem *Pan złota rączka — przyp. tłum.*

się, i… pozwól, aby ta wiedza „naciągnęła", niczym woda w filiżance herbaty. Rozmawiając z programistami, używam innego języka. Mówię im: „Zarezerwujcie trochę przestrzeni adresowej w pamięci swojego mózgu". Nie próbuj jeszcze (podkreślam: nie rób tego!) łączyć Scruma z innymi narzędziami, które znasz. Jeszcze nie czas. Dopiero po osiągnięciu mistrzostwa w posługiwaniu się jednym narzędziem możesz nauczyć się używać go sprawnie w innym otoczeniu (z innymi narzędziami). Teraz skup się przede wszystkim na Scrumie i odwołaj się do zdyscyplinowania, dając metodzie szansę, nawet (i zwłaszcza) wtedy, gdy stwierdzisz, że masz przed sobą zadanie trudne i wymagające. Zdziwisz się, stwierdzając, jak niewiele musisz zmienić w systemie Scruma i jak poważne zmiany zajdą w sposobie Twojego patrzenia na pracę. Być może stwierdzisz, że moja rada jest sprzeczna z pierwszą zasadą wartości Manifestu Agile, która brzmi: *Ludzie i interakcje ponad procesami i narzędziami*. Wręcz przeciwnie! To osoby i interakcje pozwalają Ci nauczyć się korzystania ze Scruma (lub innej zwinnej praktyki) i trzymanie się tej zasady ułatwia podjęcie świadomej decyzji na bazie własnych praktycznych wniosków.

Scrum prowadzi najkrótszą ścieżką, a nie tą ustaloną

Najdłuższa ścieżka realizacji projektu jest określana mianem ścieżki krytycznej (lub tzw. *rate-limiting path*). Tworzymy plan zakładający podążanie po ścieżce krytycznej, a jego realizacja zakłada wypełnienie kolejnych zadań, to znaczy dotarcie z punktu A do punktu B. W tym czasie na naszej drodze pojawiają się kłopoty i problemy: na przykład okazuje się, że na etapie planowania zainteresowane strony nie zawsze wiedzą, czego dokładnie chcą, albo dochodzi do zmiany celów biznesowych w miarę rozwoju produktu, w reakcji na czynniki wewnętrzne i (lub) zewnętrzne, takie jak działania konkurencji, albo przychodzi nam radzić sobie z sytuacjami, które często negatywnie wpływają na proces rozwoju produktu. Tego rodzaju zdarzenia mają miejsce w niemal każdym projekcie, a nie tylko w projekcie programistycznym.

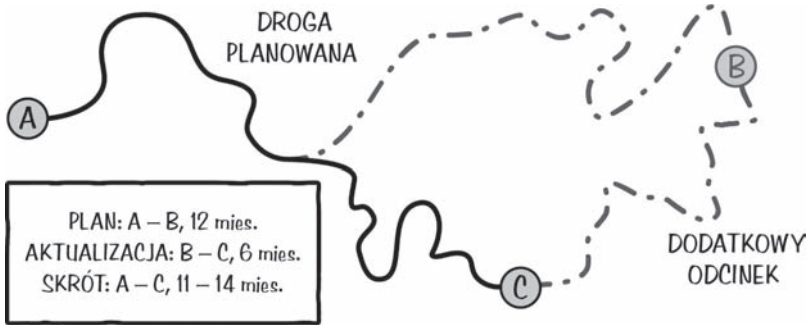
W przypadku tradycyjnej metody planowania mimo pojawiających się problemów nadal jesteśmy zmuszeni pokonać drogę z punktu A do punktu B, godząc się przy tym na poniesienie ofiary: poświęcając jakość, funkcjonalność i rezygnując z zadowolenia klienta. Gdy w końcu docieramy do punktu B, sprawdzamy, co udało nam się uratować, a potem zaczynamy tworzyć plan dotarcia do punktu C — do miejsca, które w międzyczasie okazało się prawdziwym celem, ale do którego nie mogliśmy udać się od razu, bo nie pozwalał na to nasz dotychczasowy plan (patrz rysunek 1.1).



RYSUNEK 1.1. Ścieżka realizacji projektu w tradycyjnej metodzie planowania

W ramach Scruma przygotowujemy się na wystąpienie zdarzeń takich jak to wspomniane wyżej. Wiemy, że coś takiego się stanie. Zamiast tworzyć plan „doskonały", który ma maksymalnie łagodzić konsekwencje zmian, przyjmujemy za pewnik, że przyjdzie nam zaakceptować zmianę. Dlatego tworzymy mechanizmy, które umożliwiają odpowiednią reakcję już na etapie realizacji projektu, tak aby nie trzeba było koniecznie dotrzeć do B, tylko ruszyć bezpośrednio w kierunku C.

Ten cel jest realizowany poprzez zaplanowanie codziennych, cotygodniowych i comiesięcznych spotkań z ludźmi na różnych poziomach hierarchii projektu (członkami zespołu, kadrą zarządzającą, klientami). To dzięki nim wiemy, że jesteśmy na dobrej drodze i budujemy właściwą funkcjonalność odpowiadającą oczekiwaniom. Zamiast liczyć na wszechogarniające definicje wymagań tworzone na etapie gromadzenia danych, informacje na temat oczekiwań zbieramy w trakcie realizacji projektu, korygując je i ulepszając na bieżąco. Dzięki temu projekt można dostosować do potrzeb biznesu i zewnętrznych warunków rynkowych, zapewniając sobie jak najkrótszą drogę do sukcesu (patrz rysunek 1.2).



RYSUNEK 1.2. Planowanie ścieżki w Scrumie

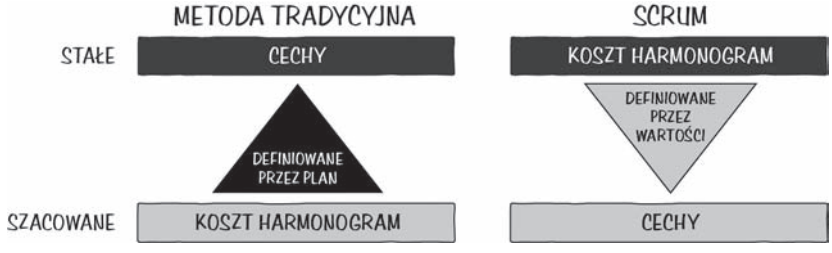
Sukces nie oznacza braku kosztów. W modelu planu idealnego często mamy wrażenie (często nieprawdziwe), że znamy dokładną datę zakończenia projektu. Jakże trudno zrezygnować z tego uspokajającego oszustwa. W rzeczywistości przecież wszyscy wiemy, że albo dochodzi do opóźnień, albo do zmiany oczekiwanej funkcjonalności (niezależnie od tego, czy drobnej, czy poważnej), albo do pogorszenia się jakości produktu — a wszystko to w imię osiągnięcia określonej funkcjonalności w określonymi dniu. W końcu wszystkie te ofiary kosztują. Koszty liczone są w wartościach takich jak stracony czas, spadek jakości lub niepotrzebnie wydane pieniądze.

W ramach Scruma nie obiecujemy dostarczenia określonego zestawu funkcji w określonym czasie. Zarówno data zakończenia projektu, jak i zestaw funkcji mogą się zmieniać. W przypadku niektórych projektów obiecujemy dostarczenie w określonym terminie przybliżonej funkcjonalności. Robimy to poprzez wstępne blokowanie kosztów (kwoty, jaką klient może wydać na każdy sprint) i trzymanie się ustalonego harmonogramu (wskazując, kiedy projekt zostanie zakończony, w oparciu o liczbę planowanych sprintów). Następnie szacujemy, ile funkcji da się przygotować w ramach tych ograniczeń. Ponieważ w projektach Scruma zawsze najpierw wykonuje się prace o najwyższym priorytecie, funkcje, których nie udaje się wprowadzić przed końcem projektu, zawsze mają niską wartość i są prawdopodobnie mniej istotne dla powodzenia całego projektu.

W przypadku projektów, w których zestaw funkcjonalności jest ważniejszy niż termin zakończenia prac, możemy zablokować zmienną funkcjonalności i pozwolić sobie na oscylację daty (a tym samym kosztów). Tym samym obiecujemy dostarczenie produktu o określonej liczbie funkcji w przybliżonej dacie lub zakresie dat. W przypadku wystąpienia zmian (dodawania nowych funkcji lub modyfikacji istniejących funkcji w konsekwencji lepszego rozumienia zadań produktu) klient wie, że musi dokonać wyboru między dostarczeniem uzgodnionej funkcjonalności w ustalonym czasie lub wprowadzeniem do harmonogramu zmian, które mają uwzględnić czas potrzebny na osiągnięcie pożądanej funkcjonalności.

Takie podejście oznacza duże odchylenie od tradycyjnego sposobu pracy, w którym funkcje definiowane są z góry, a następnie członków zespołów prosi się o oszacowanie czasu i ograniczenie kosztów niezbędnych do realizacji postawionych celów, często tylko po to, aby w toku prac jeszcze bardziej ograniczyć koszty i zmniejszyć czas przeznaczony na realizację projektu, bez zmian w zakresie zestawu funkcji.

Na rysunku 1.3 przedstawiamy podsumowanie różnic pomiędzy dwoma podejściami.



RYSUNEK 1.3. Różnice metod planowania

Warto zwrócić uwagę na konieczność ogromnej zmiany mentalności. Z tego powodu tak ważne jest zaufanie i zastosowanie wartości omówionych w tym rozdziale.

Scrum pozwala wykrywać punkty zapalne

Scrum unaocznia problemy, zarówno te, które dawno temu zostały zamiecione pod dywan i zapomniane, jak i te, o których istnieniu niektórzy ludzie nie wiedzieli. A zatem ukazuje także nowe punkty zapalne. Wykrywane problemy nie ograniczają się przy tym do programowania i pracy zespołowej. Na przykład jedno z najczęściej słyszanych przeze mnie pytań brzmi: „Jak mamy płacić naszym ludziom?" lub „*Jak długo* należy wykonywać zestaw testów akceptacyjnych?". W tradycyjnym modelu deweloper może otrzymywać wynagrodzenie na podstawie kodu przekazanego na czas i spełniającego określone warunki jakościowe. Jak przekłada się to na interdyscyplinarny zespół, który nie jest oceniany na podstawie indywidualnych wskaźników dotyczących obszarów funkcjonalnych? Przyjęcie Scruma oznacza konieczność poradzenia sobie z wyzwaniami z kategorii norm organizacyjnych, zmuszając kierownictwo do dokonania trudnych wyborów: rozwiązania problemów lub ich zignorowania. Zaletą Scruma jest jednak to, że gdy te zachowania i schematy już zostaną, trudno będzie je zignorować.

Scrum najlepiej sprawdza się w parze

Scrum to metoda zarządzania projektem. Definiuje zasady, ale nie zawiera konkretnych rozwiązań technicznych, które pozwalają dostarczyć potencjalnie gotowe do działania oprogramowanie co 2 – 4 tygodni. Do tego potrzebujesz prawdziwego księcia z bajki o Scrumie: programowania ekstremalnego, choćby pod postacią metodyki takiej jak XPrince. Podczas gdy Scrum i XP mają wiele wspólnego (np. XP ma grę planowania, w Scrumie prowadzi się spotkania planowania), XP zawiera kilka różnych metod, które pięknie uzupełniają Scrum — takich jak ciągła integracja, testy poprzedzające i programowanie w parach.

Wdrożenie Scruma pomaga zespołom, a świetne efekty przynosi połączenie Scruma i XP. To nie znaczy, że powinieneś przejść do praktycznych działań w XP przed przygotowaniem solidnej bazy wiedzy o Scrumie — należy ostrożnie podchodzić do wprowadzenia zbyt wielkich modyfikacji w tym samym czasie. Gdy zespół zbierze więcej doświadczeń z rolami, artefaktami i spotkaniami, będzie gotowy do rozpoczęcia integracji Scruma z praktykami XP. Uważaj jednak! Zwinne zarządzanie projektem podczas korzystania z kaskadowych praktyk inżynierii często tworzy niestabilne konstrukcje, powodując więcej problemów, niż przynosi korzyści z rozwiązań. Może to potrwać dzień, tydzień lub miesiąc, wszystko zależy od tolerancji zespołu na zmiany. Jednak gdy już zaczniesz stosować XP, będziesz mieć do czynienia z prawdziwą magią.

Oto cechy XP, które uważam za obowiązkowe w przypadku moich projektów:

- **Zrównoważone tempo**. Czterdziestogodzinny tydzień pracy przy stałej efektywności wynoszącej 85% (patrz rozdział 23., zatytułowany „Zrównoważone tempo", gdzie znajdziesz więcej szczegółów).
- **Współdzielenie kodu**. Cały zespół pracuje na jednym i tym samym kodzie. Nie ma on jednego właściciela (patrz rozdział 21., „Gdy dochodzi do zderzenia kultur").
- **Programowanie w parach i TDD** (ang. *test-driven development*). TDD nazywane jest: „Najpierw testy". Chodzi o to, aby przed stworzeniem kodu napisać test, sprawdzający funkcjonalność. System bazuje na standardowym modelu kolorów: czerwonym (jeśli test zostanie zakończony niepowodzeniem), zielonym (wynik pozytywny). Następnie należy dokonać poprawek, dzięki którym kod będzie lepszy. Twoim celem zawsze powinien być test zielony (więcej informacji w rozdziale 19., zatytułowanym „Programowanie w parach — utrzymanie zaangażowania członków zespołu").
- **Ciągła integracja** (ang. *continuous integration*, CI). Dzięki CI w każdym momencie wiemy, jak zachowuje się i w jakim stanie znajduje się baza kodu. Przeprowadzaj ewidencjonowanie kodu co najmniej raz dziennie. Dąż do tego, aby kończąc pracę każdego dnia, iść do domu ze świadomością, że kod ma sygnaturę „zielony", podobnie jak każdy test TDD.
- **Standardy kodowania**. Często zapomina się o tym, że nieutrzymanie dobrych standardów kodowania sieje prawdziwe spustoszenie w kodzie, który ma być współdzielony, jako że każdy członek zespołu charakteryzuje się swoim własnym stylem. Opisz standardy, jakich oczekujesz, i udostępnij je — najlepiej powieś na ścianie, żeby przypominać wszystkim o ich istnieniu.

- **Refaktoryzacja**. Ponieważ ograniczamy przygotowanie skomplikowanych planów projektu i architektury przed rozpoczęciem pracy (nie rezygnujemy z nich całkowicie!), a system przygotowywany jest tak, aby spełnić konkretne, dzisiejsze potrzeby, kod musi zostać poddany refaktoryzacji. Bez niej zmieniające się wymagania mogą prowadzić do powstawania planów projektów znacznych rozmiarów, które niezbyt dobrze dostosowuje się do potrzeb biznesowych.

„Potencjalnie gotowe" (do przekazania klientowi) — te dwa słowa często mogą wywoływać bóle głowy i, jak stwierdził na swoim blogu mój przyjaciel Chris Sterling, są one jednym z elementów Scruma, o których często się zapomina [STERLING]. Aby produkt naprawdę uzyskał funkcjonalność dającą firmie możliwość przekazania go klientom, zespoły muszą zmienić nie tylko proces zarządzania projektami, ale i sposób opracowywania oprogramowania. W przeciwnym razie wszystkie problemy, jakie zaakcentowałem w historii opisanej w tym rozdziale — słaba jakość kodu, brak czasu na testy oraz trudności z dostarczaniem produktu końcowego — pozostaną naszymi bolączkami i doprowadzą projekt do katastrofy. Zintegrowanie praktyk XP z metodami scrumowymi jest najlepszym znanym mi sposobem wprowadzenia prawdziwych i trwałych zmian.

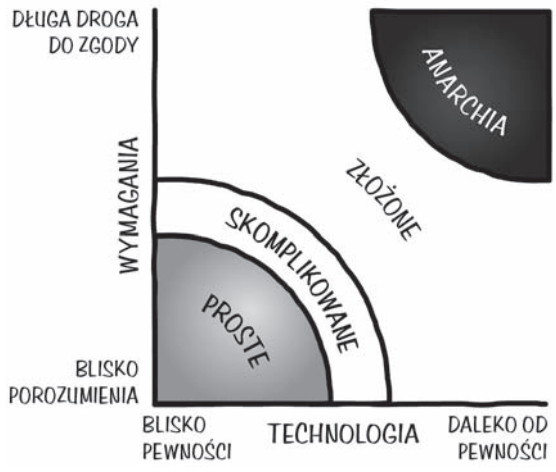
Kiedy Scrum jest dla mnie dobry?

Ustaliliśmy, że wdrożenie Scruma jest zadaniem znacznie bardziej skomplikowanym, niż mogłoby się wydawać na pierwszy rzut oka. A czy w ogóle warto próbować? To zależy.

Mam kilku przyjaciół, którzy pracują w branży budowlanej. Często proszę ich o pomoc w niewielkich domowych pracach. W ramach zadośćuczynienia funduję im kolację. Oni cieszą się, że mogą mnie zobowiązać do jakiejś przysługi. Zauważyłem, że gdy któryś z nich pojawia się w moim domu, zawsze ma ze sobą mnóstwo narzędzi. Spytałem kiedyś Joachima, czy nosi ze sobą standardowy zestaw, czy specjalnie przygotowuje narzędzia zależnie od zadania, nad jakim pracuje jego zespół. Odpowiedział, że zawsze ma ze sobą podstawowe sprzęty (metrówkę, ołówek i okulary ochronne), ale to, czy zabiera jakieś inne narzędzia, niemal zawsze zależy od konkretnego zlecenia. Jeśli pracuje nad przygotowaniem szkieletu domu, przygotowuje swój pas według potrzeb. Kiedy robi stolarkę lub układa płytki, używa innego zestawu. Przy czym narzędzia, dla których nie znalazło się miejsce na pasie, zawsze znajdują się niedaleko (w samochodzie). Jak ma się to do Scruma?

Dla mnie Scrum jest jednym z takich instrumentów wkładanych do pasa z narzędziami. Chcąc uchodzić za dobrego podwykonawcę, każdy powinien korzystać ze Scruma zawsze wtedy, gdy takie działanie jest właściwe. To wcale nie znaczy, że trzeba używać go non stop. Tak jak nie użyłbyś śrubokręta do wbicia gwoździa, tak nie używaj Scruma do realizacji projektu, który nie wymaga zastosowania tego systemu. A zatem pojawia się pytanie: Kiedy *powinniśmy* zastosować Scrum?

Projekty programistyczne można podzielić na cztery kategorie: proste, skomplikowane, złożone i będące kompletną anarchią (patrz rysunek 1.4). System ten przyjął Ralph Stacey w swojej książce zatytułowanej *Strategic Management and Organisational Dynamics* [STACEY, wydanie 2.]. Na proste projekty składają się głównie wiadome: Podobne projekty wykonuje się regularnie i często, a to znaczy, że panuje powszechna zgoda w kwestii tego, co jest potrzebne, a użyta technologia jest dobrze rozumiana. Proste projekty są łatwe do wdrożenia i zrozumienia.



RYSUNEK 1.4. Zależności pomiędzy wymaganiami, technologią i charakterem projektu

Gdy jednak odchodzimy od prostych projektów, sprawy się trochę komplikują. Nasze wymagania, nasza technologia — lub zarówno jedno, jak i drugie — mogą nie okazać się tak stabilne, jak się spodziewaliśmy. Być może ludzie nie są pewni, jak ma wyglądać produkt końcowy. Może technologia jest nowa i jeszcze nikt jej nie sprawdził. W tym obszarze — w kategorii projektów skomplikowanych i złożonych — mamy do czynienia z dużą liczbą zmian. O wiele trudniej jest przewidzieć przebieg pracy oraz doprowadzić projekt do końca. Następnie, gdy pozostawiamy za sobą kategorię projektów definiowanych jako złożone i zmierzamy w stronę anarchii, stajemy przed prawdziwym wyzwaniem nieznanych wcześniej niewiadomych. Na tym etapie projektów firmy powinny skupić się na radzeniu sobie z problemami, które pojawiają się na drodze, a nie na budowaniu systemu.

To wszystko oznacza, że jeśli Twój projekt znajduje się poza granicami obszaru „prosty projekt", zmiany — wymagań i (lub) technologii — są najzwyczajniej w świecie nieuniknione. Mam praktykę w stosowaniu Scruma i XP do realizacji prostych projektów i w takich przypadkach najczęściej stwierdzałem, że użycie tych metod jest przesadą. Jednak w miarę oddalania się od tego, co uzgodnione i pewne, użycie Scruma i XP staje się coraz bardziej uzasadnione, ponieważ metody te pozwalają łatwiej dostosować się do wymagań biznesowych, które z pewnością ulegną zmianie.

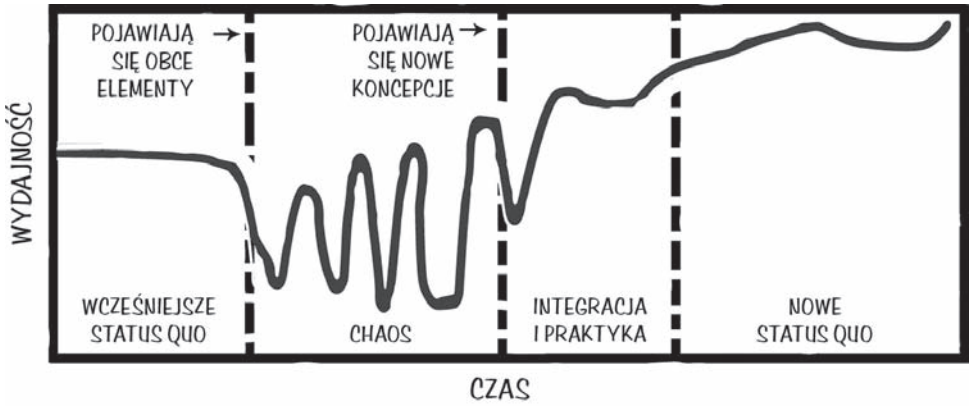
Zmiana jest trudna

W Scrumie wszystko sprowadza się do zmian. Są ludzie, którzy cieszą się na zmianę. Inni unikają jej jak ognia. Obie reakcje są zupełnie naturalne. Twórcy oprogramowania zwykle starają się unikać zmian. Nie postępujemy tak dlatego, że boimy się zmian, ale ponieważ przyzwyczailiśmy się do przekazywania klientowi właściwego produktu już za pierwszym razem. W naszej głowie zmiana jest utożsamiana z błędami lub właściwościami, które przeoczyliśmy na etapie rozwoju produktu. Dlatego dla nas żądanie zmiany jest sugestią, że coś się nam nie udało. Często nie rozumiemy, że zmiany są nieuniknione. Nie możemy przewidzieć przyszłości, nie możemy się spodziewać, że z góry przewidzimy nagłe zmiany zachodzące na rynku lub pojawienie się najnowszej wersji oprogramowania wprowadzonego przez konkurenta. Aby odnieść sukces,

musimy nauczyć się radzenia sobie ze zmianami. Scrum daje nam odpowiednie narzędzia, ale przeniesienie się do miejsca, w którym zmiana jest zintegrowana z naszym procesem, jest trudną podróżą.

Zespół z naszej historii chciał korzystać ze Scruma, ale bał się skoku w nieznane. Na próżno próbował trzymać się ustalonych wzorców i uniknąć błędów, modyfikując proces i dodając do niego sprint planowania i integracji. Jednak modyfikacje te, zamiast okazać się pomocne, w rzeczywistości *spowodowały* problemy i bardzo pogorszyły sytuację.

Zrozumienie typowych reakcji jest kluczem do udanego wdrożenia Scruma. Virginia Satir, terapeutka rodziny, poprzez badania kliniczne określiła wzór opisujący sposób doświadczania zmiany. Wśród etapów zmiany wymienia się: wcześniejsze status quo, wprowadzenie obcego elementu, chaos, integrację i praktykę, a także nowe *status quo*. Przedstawione wzorce zachowań są znane jako etapy zmian [SATIR] i zostały przedstawione na rysunku 1.5.



RYSUNEK 1.5. Etapy zmian wg Satir

Uważam, że etapy zmian Satir można zastosować do różnych środowisk. Gdy byłem dzieckiem, moja matka próbowała rzucić palenie i miałem okazję zaobserwować, jak wzorce Satir przekładają się na praktykę. Widać je wyraźnie na przykładzie dzieci próbujących opanować jazdę na rowerze. My przyjrzyjmy się, jak zespół scrumowy przechodzi przez etapy zmiany, takie jak te opisane przez Virginię Satir.

Wcześniejsze status quo

Kiedy panuje status quo, ludzie czują się pewnie. Często znajdują sobie usprawiedliwienie tego, że trzymają się status quo, ponieważ żyją w otoczeniu, które dobrze znają. Ludzie mają skłonność do ignorowania potrzeby zmiany i po prostu dalej robią to samo, tylko lepiej, mocniej się starają lub stają się bardziej skrupulatni. Aż nazbyt dobrze znamy takie status quo, w którym mamy do czynienia z marszem śmierci, gdy zespoły nieuchronnie zmierzają w kierunku katastrofy, pracując przy tym dłużej i ponosząc niepotrzebne ofiary w zakresie jakości, bo starają się za wszelką cenę dotrzymać terminów. Ostatecznie wszyscy zaczynają wątpić i mówić o swojej niewierze w to, że wszystko skończy się dobrze, jeśli tylko będziemy pracować dłużej i intensywniej. W tym momencie zaczynamy zbliżać się do masy krytycznej,

jako że pewna grupa ludzi uznaje się za zmęczonych kulturą pracy lub kadra zarządzająca wprowadza nowe zasady pracy. To jest moment, gdy pojawia się obcy element.

Obcy element

Obcy element jest wyzwaniem. To coś, co sprawia, że nie może już być mowy o zachowaniu status quo. Może się zdarzyć, że będzie to coś trywialnego, na przykład nowy szablon specyfikacji projektu. Jednak równie dobrze może chodzić o coś poważniejszego — na przykład zupełnie nowy sposób pracy, choćby takie wdrożenie Scruma. Czym by nie był obcy element, po jego pojawieniu się stwierdzamy, że nie można go zignorować. W nasze działania wkrada się chaos. A on z kolei sprawia, że pojawia się strach. Ludzie wiedzą, że zbliża się zmiana, ale opierają się jej, ponieważ nie wiedzą, jak mają się zmienić i czego się od nich oczekuje. Nie wiedzą, czy w tych nowych warunkach, tak różnych od tego, co już znają, uda im się osiągnąć sukces.

Chaos

Chaos pojawia się wtedy, gdy w konsekwencji pojawienia się obcego elementu zostaje zniesione status quo. Chociaż poziom zamieszania zależy od charakteru obcego elementu, w przypadku znacznych zmian można odnieść wrażenie, że wszystko, co znamy, po prostu zniknęło. Nawet ludzie najbardziej przekonani o potrzebie przyjęcia nowych rozwiązań są zmuszeni do pracy na warunkach, które wydają się im obce i niewygodne. Stają się wtedy nerwowi i niepewni. Na dodatek często czują się zagrożeni. Boją się podejmowania złych decyzji. W rezultacie rośnie strach i pojawiają się wahania wydajności. Kiedy grupa przechodzi przez ten etap, zanim ludzie z zespołu poczują się lepiej, sytuacja ulegnie dalszemu pogorszeniu. Jednak prędzej czy później poprawa stanie się faktem. Końcowi chaosu będzie towarzyszyć identyfikacja zmieniającej wszystko koncepcji.

Jaka koncepcja kończy etap chaosu, wywołanego początkowo przez Scrum? To zależy od zespołu, ale zazwyczaj jest to myśl z rodzaju: „Hej, nie jest tak źle. Damy radę". Ludzie zaczynają dostrzegać związek pomiędzy obcym elementem i wizją przyszłej rzeczywistości. Oto udało się znaleźć most łączący dwa brzegi.

Praktyka i integracja

Moje dzieci uprawiają różne sporty. Za każdym razem, gdy narzekają na trudne treningi, przypominam im, że w tej samej chwili ktoś ćwiczy po to, aby pokonać ich zespół podczas najbliższej konfrontacji. Ćwiczenia w zakresie rozwoju oprogramowania są równie istotne. Bez nich żaden obcy element nigdy nie stanie się znany, nigdy nie będzie drugą naturą ani zakorzenionym nawykiem.

Gdy zespół zdobywa powoli doświadczenie i coraz lepiej rozumie zasady Scruma, zaczyna też dostrzegać zalety systemu. Gdy pojawiają się pierwsze korzyści, członkowie zespołu stopniowo akceptują swoją nową rzeczywistość. Chaos staje się pieśnią przeszłości, a zespół zmierza w kierunku nowej równowagi. W zespole (i w grupie zaangażowanych osób) pojawia się wzajemne zaufanie, kształtują się nowe relacje i dochodzi do rozwoju wspólnej tożsamości. Gdy ludzie uczą się współpracować ze sobą na nowych zasadach, rośnie też wydajność, często wykraczając poza poziomy znane z przeszłości. Nie należy przyjmować tych wczesnych symptomów za znak, że zespół jest już wystarczająco doświadczony. Przeciwnie! Zespoły

znajdujące się na tym etapie potrzebują teraz jeszcze więcej opieki. Choć prawdziwe jest stwierdzenie, że ludzie czują się coraz pewniej w swojej pracy, nadal towarzyszy im poczucie zagrożenia, mimo że nie dają tego dowodów. Osiągnięty właśnie sukces jest konstrukcją chwiejną i delikatną. Dlatego powinno się go odpowiednio pielęgnować. A jednak z każdym takim sukcesem coraz bardziej jasna staje się świadomość tego, co można osiągnąć. Większości członków zespołu wydaje się, że stan, w jakim się znajdują, przypomina nirwanę. Pewna grupa wciąż jednak może czuć, że otacza ją chaos.

Nowe status quo

Gdy członkowie zespołu nabierają jeszcze więcej doświadczenia i czują się coraz pewniej z nową koncepcją przekształcenia, zaczynają osiągać nieporównywalnie lepsze wyniki. Czują się świeżo i mają poczucie spełnienia, które wynika ze wspólnego przeżycia transformacji. Etap ten pozwala ludziom odkrywać nowe sposoby samodoskonalenia, aby czuć, że mają prawo do robienia pewnych rzeczy lepiej niż dawniej i współpracy z myślą o stworzeniu samodoskonalącej się organizacji. Kluczowe znaczenie ma swoboda oznaczająca szansę wykonywania ćwiczeń w bezpiecznym środowisku. Tak jak w sporcie: Nigdy nie jesteś tak dobry, aby odpuścić sobie treningi i zrezygnować z uczenia się nowych rzeczy.

Klucze do sukcesu

Jak pokazała historia opisana w tym rozdziale, na wczesnych etapach wdrażania Scruma wiele zespołów musi radzić sobie ze sporymi problemami. Nie dzieje się tak dlatego, że członkowie zespołów podchodzą do systemu z niechęcią lub są to ludzie o ograniczonych poglądach, ale dlatego, że Scrum wymaga pracy w nieznanym środowisku, w którym ludzie czują się nieswojo. Bohaterowie naszej opowieści próbowali trzymać się tych reguł, które wydawały im się znajome, nie wiedząc, że to właśnie one spowalniają ich i utrudniają osiągnięcie sukcesu. Aby wdrożenie Scruma zakończyło się powodzeniem, zespoły muszą stosować się do podstawowych założeń systemu.

Po pierwsze, konieczne jest zrozumienie zasad. Scrum jest prostą strukturą, która ma prowadzić jak najprostszą drogą do ukończenia projektu. Ze względu na tę prostotę ludzie mają skłonność do wprowadzania zmian zasad Scruma, bez zrozumienia konsekwencji swoich działań. To niebezpieczne. Tak jak nie jeździłbyś samochodem bez znajomości zasad ruchu drogowego, tak nie należy próbować wdrażania Scruma, nie posiadając odpowiedniej wiedzy na temat systemu. Nauka zasad wymaga czasu. Wykorzystaj go do zrozumienia, w jaki sposób zasady Scruma odnoszą się do Ciebie, Twojego zespołu i Twojej firmy.

Po drugie, członkowie zespołu muszą najpierw opanować podstawowe techniki. Zespoły i ich organizacje muszą zrozumieć, jak to wszystko działa, i „poczuć" Scrum. Na to potrzeba czasu. Jak przekonaliśmy się, analizując etapy zmiany opisane przez Satir, przejście od znanego do nieznanego jest procesem trudnym, który wymaga cierpliwości i praktyki. Proces ten przypomina naukę jazdy na rowerze. Kiedy mamy pięć lat, nie możemy brać udziału w Tour de France. Najpierw uczymy się, jak utrzymać się na rowerze. Często lądujemy na ziemi. To dzięki praktyce nabywamy coraz większej pewności i zaufania do własnych umiejętności. Daj ludziom czas, aby opanowali podstawy, i pomagaj im podczas nauki. Upewnij się, że zwracasz

uwagę na ich odczucia, i zrozum, że podczas ćwiczeń, w trakcie których wykonuje się podstawowe działania, ludzie mogą czuć się niepewnie, być sceptycznie nastawieni i odczuwać niepokój. Wraz z upływem czasu ta niepewność zniknie, ale zdarzy się to tylko wtedy, gdy znajdziecie dość czasu na ćwiczenia.

Po trzecie, nie spiesz się. Na opanowanie nowych metod pracy potrzeba więcej czasu, niż masz do dyspozycji podczas weekendowych warsztatów. Z mojego doświadczenia wynika, że większość zespołów scrumowych i firm wdrażających system potrzebuje co najmniej trzech miesięcy, aby nauczyć się podstaw i zacząć sobie radzić w nowym otoczeniu. Zdarza się, że wszystko udaje się szybciej. Są też takie grupy, które potrzebują więcej czasu, ale średnio ten pierwszy etap trwa od trzech do sześciu miesięcy (w zależności od grupy, w której dochodzi do zmiany trybu zarządzania zmianą).

Po czwarte, nie zabieraj się za wdrożenie Scruma w połowie drogi. Widziałem niezliczone przypadki porażek, gdy zespoły wdrażały Scruma w samym środku cyklu rozwoju oprogramowania. Często była to konsekwencja błędnej oceny kierownictwa firmy, które widziało w systemie cudowne panaceum na wszelkie problemy. Zarząd, widząc potencjalne korzyści, które wynikają z posiadania zespołu o wysokiej efektywności, oczekuje, że ludzie z dnia na dzień staną się ekspertami od Scruma. Łatwo sobie wyobrazić, że takie podejście sprawia, iż zespół natychmiast wpada w korkociąg, pikując ku ziemi, ponieważ ludzie są zmuszani do pracy z wykorzystaniem nowych narzędzi przy niezmiennym ostatecznym terminie zakończenia projektu. Chaos będący konsekwencją takich decyzji powoduje opóźnienia i prowadzi do wielkich wyrzeczeń w zakresie jakości. Zalecam wdrażanie Scruma na początku nowego projektu. W międzyczasie można utworzyć Rejestr Przejścia, zawierający listy warunków koniecznych i elementów, dzięki którym będzie można zapewnić jak najsprawniejsze przyjęcie Scruma.

I wreszcie zadbaj o wygospodarowanie czasu na ustawiczny rozwój i poszerzanie wiedzy. Zbyt często zespoły ignorują retrospektywy, uznając je za luksus i jednocześnie stratę czasu. Zespoły ignorujące retrospektywy lub przechodzące przez kolejne etapy bez wyciągania jakichkolwiek wniosków na potrzeby przyszłych sprintów najczęściej wracają do swoich starych, złych nawyków. Dlaczego? Ponieważ nie potrafią wygospodarować czasu na naukę. Uczenie się jest jednym z kluczowych aspektów Scruma, z których nie można zrezygnować. Żaden zespół nie jest w stanie osiągnąć stanu wysokiej efektywności bez wyciągania wniosków z doświadczeń.

Mając na uwadze wszystkie te wyzwania techniczne i psychologiczne, pamiętaj o wytrwałości. Opanowanie Scruma nie różni się od opanowania innej wiedzy lub umiejętności. Powinieneś oczekiwać chwil wielkiej satysfakcji, ale i wielkiego zawodu. Ucz się i zadbaj o otwarty umysł. Pamiętaj też, że jeśli NASA mogła wysłać człowieka na Księżyc na pokładzie Apollo 11, w którym znajdował się komputer dysponujący pamięcią w granicach 36 – 72 kB, Ty dasz sobie radę z opanowaniem Scruma.

Bibliografia

[SCHWABER 01] Ken Schwaber, Jeff Sutherland. *Scrum Guide*. 2011. *https://www.scrum.org/portals/0/documents/scrum guides/scrum_guide.pdf*. Plik dostępny również pod adresem *http://www.scrum.org/scrumguides/*, a stąd można pobrać kopię: *http://www.mitchlacey.com*.

Polska oficjalna wersja Przewodnika jest dostępna pod adresem: *http://scrumguides.squarespace.com/s/Scrum-Guide-PL.pdf#zoom=100*.

[SCHWABER 02] Ken Schwaber. *Agile Software Development with Scrum*. Microsoft Press. Redmond 2004.

[STERLING] Chris Sterling. Strona Bartona Sterlinga: *http://www.gettingagile.com/2009/07/15/the-forgotten-scrum-elements/* (dostęp 3 lipca 2010 r.).

[STACEY] Ralph Stacey. *Strategic Management and Organisational Dynamics, 2nd Edition*. Financial Times Management. 1996.

[SATIR] Virginia Satir, John Banmen, Jane Berber, Maria Gomori. *The Satir Model: Family Therapy and Beyond*. Science and Behavior Books. Palo Alto 1991, s. 98 – 119.

Skorowidz

M

N

O

P

U

W

Z

Poznaj potencjał metodyki Scrum!

Tempo prac nad współczesnymi projektami wymusiło wypracowanie nowych metodyk pozwalających w zwinny sposób zarządzać projektami. Jedna z nich — Scrum — zdobyła szczególną popularność. Jasne zasady, dobrze dobrane role w projekcie oraz przemyślany sposób jego prowadzenia zdecydowały o tym sukcesie. Jeżeli chcesz w swoim projekcie wprowadzić ten sposób zarządzania, trafiłeś na doskonałą książkę.

Dzięki niej zrealizujesz swoje plany szybko i bez zbędnych kłopotów. W trakcie lektury dowiesz się, czym charakteryzuje się ta metodyka oraz jak ją wprowadzić w Twojej organizacji. Dzięki kolejnym rozdziałom nauczysz się określać prędkość zespołu, przydzielać role oraz ustalać długość sprintu. Dalsze strony zawierają cenne informacje praktyczne na temat godzin pracy, planowania wydań oraz prowadzenia retrospektyw. Ostatnia część książki została poświęcona zaawansowanym technikom. Dowiesz się, jak dokumentować w projekcie scrumowym, kosztorysować oraz formułować umowy. Książka ta jest wciągającym kompendium wiedzy, niezbędnym dla każdego korzystającego ze zwinnych technik zarządzania projektem opartych na metodyce Scrum.

Dzięki tej książce:

- poznasz założenia metodyki Scrum
- zdefiniujesz i przydzielisz odpowiednie role
- określisz długość i zakres sprintu
- doprowadzisz projekt do szczęśliwego końca

helion.pl
księgarnia internetowa

Nr katalogowy: 17260

Księgarnia internetowa:
http://helion.pl

Zamówienia telefoniczne:
0 801 339900
0 601 339900

Addison-Wesley
Pearson Education

Helion

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

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

Informatyka w najlepszym wydaniu

sięgnij po WIĘCEJ
KOD KORZYŚCI

ISBN 978-83-246-8250-8
cena: 69,00 zł
9 788324 682508