DOSKONALENIE SCRUMA

PRZEWODNIK DLA PRAKTYKÓW O WYZWANIACH, KORZYŚCIACH I ZWINNYCH ZESPOŁACH

STEPHANIE OCKERMAN
SIMON REINDL

Przedmowa KEN SCHWABER, DAVE WEST

Helion

Tytuł oryginału: Mastering Professional Scrum A Practitioners Guide to Overcoming Challenges and Maximizing the Benefits of Agility (The Professional Scrum Series)

Tłumaczenie: Anna Zawiła, Tadeusz Zawiła

ISBN: 978-83-283-6912-2

Authorized translation from the English language edition, entitled MASTERING PROFESSIONAL SCRUM: A PRACTITIONER'S GUIDE TO OVERCOMING CHALLENGES AND MAXIMIZING THE BENEFITS OF AGILITY, 1st Edition by OCKERMAN, STEPHANIE; REINDL, SIMON, published by Pearson Education, Inc, publishing as Addison-Wesley Professional,

Helion SA
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/doscru
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

SPIS TREŚCI

3 Dostarczanie ukończonych przyrostów produktu

Koncepcja ukończenia leży u podstaw Scruma. Scrum to ramy postępowania umożliwiające biznesową zwinność, ale bez ukończonych przyrostów produktu biznesowa zwinność nie istnieje, nie ma bowiem miejsca na coś takiego jak „tak jakby ukończone" czy „prawie ukończone" przyrosty. Dotarcie do celu i ukończenie zaplanowanych zadań wymaga od zespołów scrumowych otwarcia się na pracę zespołową, empiryzm oraz zwinne myślenie, a także dojrzewania procesu zespołu w celu dostarczenia produktu, który umożliwi zespołowi ocenę celu sprintu w sposób empiryczny.

Po przejściu przez etap *formowania* nowe zespoły scrumowe często mają problemy z uzyskaniem ukończonego przyrostu produktu w każdym sprincie. Kiedy prowadzimy kurs Professional Scrum Master (PSM)[1], jednym z najczęściej zadawanych pytań jest: „Rozumiem, dlaczego kwestia ukończenia ma znaczenie, ale jak mamy zrealizować ten cel, skoro nie możemy tego zrobić dzisiaj?".

W miarę jak tożsamość zespołu będzie się rozwijać i tworzyć silniejsze podstawy, jego członkowie będą jednocześnie pracować nad wyjaśnianiem i udoskonalaniem swojego procesu. Ramy postępowania Scruma celowo pozostawiają w gestii zespołu scrumowgo określenie własnego procesu. Pamiętaj, że w Scrumie nie ma żadnych „najlepszych praktyk"

[1] *https://www.scrum.org/courses/professional-scrum-master-training.*

— to znaczy, że najlepszymi praktykami lub narzędziami są te, które działają w przypadku Twojego produktu i zespołu w danej chwili. Ponadto proces zespołu musi ewoluować w miarę upływu czasu, aby dopasować się do zmieniających się potrzeb i nowych wyzwań.

Ta dynamiczna jakość wyjaśnia, dlaczego zespoły scrumowe muszą bazować na pracy zespołowej, empiryzmie i zwinnym sposobie myślenia, aby mogły dostarczać ukończone przyrosty produktu. Lepsza współpraca i efektywniejsza samoorganizacja wskażą drogę prowadzącą do uzyskania wyższej jakości oraz szybszej realizacji. Większa transparentność w odniesieniu do przepływu pracy, postępów i jakości pozwoli na świeże spojrzenie i lepszą adaptację, co z kolei poprawi szybkość oraz skuteczność tworzenia przyrostów. Jak widać wartości Scruma, takie jak skupienie i zaangażowanie, odgrywają ważną rolę w ukończeniu przez zespół tego, co zostało zaplanowane[2].

W niniejszym rozdziale zwrócimy uwagę na dobre praktyki, których zastosowanie można rozważyć. Należy jednak pamiętać, że kluczowym punktem jest zrozumienie, jaki jest cel zastosowania danej praktyki. Czy zwiększa ona transparentność w zakresie postępu lub jakości? Czy wpływa na powstawanie produktywnych konfliktów i zwiększa zaangażowanie w realizację podjętych zespołowo decyzji? Czy też wyznacza granice, aby poprawić koncentrację i zmniejszyć ryzyko? Poszukaj przyczyn leżących u źródeł Twoich wyzwań i wybierz praktyki — zarówno te przedstawione w niniejszym rozdziale, jak i poznane poza kartami niniejszej książki — które pomogą Ci w niezawodnym i konsekwentnym dostarczaniu ukończonych przyrostów produktu.

Chociaż to zespół deweloperski jest odpowiedzialny za wyprodukowanie ukończonych przyrostów, właściciel produktu i Scrum master również przyczyniają się do sukcesu zespołu; istnieje bowiem inherentne napięcie pomiędzy zakresami odpowiedzialności ról, które uzupełnia współpracę[3]. Właśnie z tego powodu ten rozdział jest przeznaczony dla wszystkich członków zespołu scrumowego oraz liderów, którzy ich wspierają i umożliwiają im pracę.

Jaka jest definicja ukończenia?

Definicja ukończenia (ang. *definition of done*, DoD) opisuje wspólne rozumienie przez zespół deweloperski pracy potrzebnej do stworzenia użytecznego, nadającego się do wydania przyrostu. Definicja ta — często opisywana za pomocą konwencji, standardów i wytycznych — musi spełniać (lub przewyższać) wszelkie istniejące definicje organizacyjne.

---

[2] Aby uzyskać więcej szczegółów na temat tego, jak wartości Scruma pomagają zespołom scrumowym w efektywnym wykorzystaniu Scruma, przeczytaj ten post na blogu: *https://guntherverheyen.com/2013/05/03/theres-value-in-the-scrum-values/*.

[3] Dla lepszego zrozumienia odpowiedzialności ról w Scrumie — i tego, jak współdziałają — przeczytaj ten post zamieszczony na blogu: *https://www.scrum.org/resources/blog/accountability--quality-agile*.

Definicja ukończenia musi być osiągalna i odpowiednia dla produktu oraz zespołu. Kiedy przyrost spełnia jej wymagania, to właściciel produktu decyduje o tym, czy zostanie on wydany (czy też nie). Przyrost może być wydawany wielokrotnie w trakcie sprintu (np. w trybie ciągłej dostawy lub modelu DevOps) albo dopiero po przeprowadzeniu wielu sprintów.

Definicja ukończenia musi zagwarantować, że cała praca włożona w przyrost jest zintegrowana z wcześniejszymi przyrostami. Kiedy nad danym produktem pracuje wiele zespołów, połączona praca wszystkich zespołów musi tworzyć ukończony przyrost. Definicje ukończenia poszczególnych zespołów mogą się różnić, ale muszą one mieć wspólne podstawowe minimum co do jakości i kompletności, co oczywiście wymaga integracji, aby stworzyć produkt, który można wypuścić na rynek.

Skoro ta koncepcja jest tak ważna, to dlaczego Scrum nie mówi wprost, co powinno stanowić część definicji ukończenia? Nie owijając w bawełnę: definicja ukończenia tak bardzo zależy od kontekstu, że stworzenie jej uniwersalnej wersji jest niemożliwe. Oznacza to, że definicja ukończenia będzie zupełnie inna dla zespołów pracujących nad grą na urządzenia mobilne, sprzętem medycznym, systemem kontroli lotów czy też międzynarodowym systemem bankowym. Użycie produktu, oddziaływanie na biznes, bezpieczeństwo i wpływ problemów na użytkowników będą kształtować sposób definiowania tego, co jest ukończone.

Korzyści wynikające z definicji ukończenia

Posiadanie solidnej definicji ukończenia oraz tworzenie przyrostu, który ją spełni, ma następujące korzyści:

- *Transparentność procesu.* Ukończony przyrost zapewnia zasadniczą transparentność co do poczynionych postępów i dostarczania wartości. Jeśli zespół nie dostarczy ukończonego przyrostu produktu w danym sprincie, jest to znak, że brakuje mu istotnej wiedzy lub umiejętności. Taka porażka stanowi dla zespołu scrumowego okazję do zbadania i udoskonalenia swoich procesów, narzędzi oraz sposobu współpracy.
- *Transparentność przyrostu.* Gdy istnieje jasność co do definicji ukończenia, członkowie zespołu scrumowego ani interesariusze nie powinni być zaskoczeni jakością i kompletnością tego, co zostało zaprezentowane w ramach przeglądu sprintu. Gdy przyrost spełnia wymogi definicji ukończenia, właściciel produktu może wydać produkt klientom i zrealizować pochodzącą z niego wartość (czyli zwrot z inwestycji). Założenia dotyczące wartości produktu i tego, jak zostanie odebrany na rynku, mogą zostać zweryfikowane.
- *Transparentność co do tego, gdzie jesteśmy.* Gdy istnieje jasność co do definicji ukończenia, właściciel produktu może lepiej komunikować się z interesariuszami na temat zakończonych prac. Pozwala mu to na aktualizację prognoz wskazujących na prawdopodobieństwo osiągnięcia celów danego przyrostu w zakresie dostarczenia produktu.

- *Transparentność co do tego, dokąd zmierzamy.* Po zakończeniu prac nad elementami rejestru produktu, zgodnie z definicją ukończenia, ich wielkość oraz złożoność można wykorzystać do prognozowania przyszłych prac o zbliżonej wielkości i złożoności.
- *Transparentność co do planowania sprintu.* Dzięki wyraźnemu zrozumieniu tego, co jest niezbędne do wykonania danej pracy — aby ukończyć zaplanowane zadania — zespół deweloperski zyska lepsze wyobrażenie o tym, co jest w stanie dostarczyć w ramach sprintu.

Jak stworzyć definicję ukończenia?

Chociaż to zespół deweloperski jest właścicielem definicji ukończenia, właściciel produktu może współuczestniczyć w jej tworzeniu, aby lepiej zrozumieć, co jest potrzebne do stworzenia nadającego się do wydania przyrostu wysokiej jakości. Właściciel produktu może również pomóc w określeniu standardów jakościowych produktu, takich jak liczba korzystających z niego w tym samym czasie użytkowników, których system musi być w stanie obsłużyć, czy też maksymalny dopuszczalny czas na przeprowadzenie transakcji.

Poniższe pytania mogą się okazać przydatne do stymulowania wspólnej dyskusji na temat definicji ukończenia:

- Co musimy zrobić, aby pomóc osobom, które będą obsługiwać produkt (np. stworzyć czytelny kod, konwencje nazewnictwa zmiennych)?
- Jak zminimalizujemy dług technologiczny (np. refaktoring)?
- W jaki sposób będziemy testować produkt (np. poprzez testy jednostkowe, funkcjonalne, regresyjne)?
- Które testy zostaną zautomatyzowane?
- Jakie defekty muszą zostać naprawione (np. ich znaczenie, rodzaj)?
- Jak spełnimy wymagania dotyczące wydajności i skalowalności (np. czas przetwarzania transakcji, liczba równoczesnych użytkowników)?
- Jakie standardy deweloperskie poprowadzą nas ku technicznej doskonałości?
- W jaki sposób będziemy weryfikować zgodność ze standardami deweloperskimi naszego zespołu (np. przez współpracowników w roli recenzentów)?
- Jak będziemy sprawdzać i gwarantować jakość danych?
- W jaki sposób zapewnimy, że nasz produkt jest bezpieczny?
- Jak zapewnimy, że nasz produkt będzie spełniał normy administracyjne, prawne albo inne wymogi?
- Co musimy zrobić, aby spełnić wymagania w zakresie brandingu?
- Jakie kroki należy podjąć, aby nasz produkt mógł być używany przez osoby niepełnosprawne (np. był zgodny ze standardami Konwencji o prawach osób niepełnosprawnych)?

- Jakie dokumenty są potrzebne do wdrożenia do produkcji (np. pomoc online, aktualizacje systemu zarządzania aktywami)?

W praktyce: Użycie techniki „teraz, potem, w przyszłości" do tworzenia i ulepszania definicji ukończenia

Technika ta może pomóc zespołowi we wspólnym stworzeniu wstępnej definicji ukończenia i ulepszaniu wraz z upływem czasu. Może ona również pomóc zespołom, zmagającym się z niezbyt udaną definicją ukończenia, skupić się na tym, co jest najważniejsze w tej chwili, i podjąć celowe działania, aby móc posunąć się naprzód.

1. Zespół deweloperski jest proszony o przeprowadzenie burzy mózgów na temat wszystkiego, co chciałby ująć w ramach definicji ukończenia, aby uzyskać najwyższą jakość i możliwie najpełniejszy przyrost. Członkowie zespołu zakładają, że mogą zrobić wszystko — bez ograniczeń.
2. Członkowie zespołu wspólnie identyfikują elementy rejestru, które są teraz w stanie wykonać, i przenoszą je na środek zestawu zagnieżdżonych prostokątów, jak pokazano na rysunku 3.1.
3. Członkowie zespołu określają, jakie będą kolejne ulepszenia, które chcą wprowadzić; przenoszą te elementy do prostokąta oznaczonego jako „potem".
4. Członkowie zespołu określają pożądane przyszłe ulepszenia, których wdrożenie prawdopodobnie będzie wymagało znacznej ilości czasu lub pieniędzy, i przesuwają je do prostokąta oznaczonego jako „w przyszłości".

Rysunek 3.1. *Użycie techniki „teraz, potem, w przyszłości" może pomóc zespołom deweloperskim w stworzeniu ich definicji ukończenia (zdjęcie Simona Reindla)*

Na końcu tego ćwiczenia zespół powinien mieć zaktualizowaną definicję ukończenia, jak również uporządkowaną listę ulepszeń, które należy wprowadzić w kolejnych kilku sprintach. Elementy te mogą wymagać podziału na mniejsze części lub przeformułowania dla większej przejrzystości, więc zespół może zdecydować się na taką czynność po zakończeniu sesji[4].

Używanie celów sprintu, żeby dotrzeć do etapu ukończenia

Zespół scrumowy zobowiązuje się do osiągnięcia celu sprintu, który stanowi powód przeprowadzenia sprintu. Kiedy zastanawiasz się nad sukcesem sprintu, najpierw zapytaj „Czy mamy ukończony przyrost?", a następnie „Czy zrealizowaliśmy cel sprintu?".

Dobry cel sprintu charakteryzują trzy cechy: skupienie, elastyczność i celowość.

- Czy cel sprintu zapewnia wystarczające *skupienie*, aby uzyskać działający przyrost produktu, który dostarczy wartość na koniec sprintu?
- Czy cel sprintu zapewnia wystarczającą *elastyczność*, żebyśmy mogli dostosować nasz plan (np. rejestr sprintu), gdy uczymy się nowych rzeczy i odkrywamy kompleksowość zagadnień?
- Czy daje nam poczucie *celowości*, abyśmy mogli dostrzec wartość i znaczenie naszej pracy? Czy ekscytuje interesariuszy albo przynajmniej wzbudza ich zainteresowanie udziałem w przeglądzie sprintu?

Tworzenie dobrych celów sprintu

Nie ma idealnej recepty na stworzenie dobrego celu sprintu. Każdy cel sprintu bardzo mocno zależy od kontekstu, a zespoły scrumowe muszą eksperymentować, aby dowiedzieć się, co w ich przypadku najlepiej działa, oraz zaadaptować się do zmieniającego się kontekstu (zob. rysunek 3.2)[5].

[4] Więcej informacji na temat techniki „teraz, potem, w przyszłości" oraz przykłady jej wykorzystania w celu poprawy definicji ulepszenia można znaleźć na stronie *https://www.scrum.org/resources/blog/improving-your-definition-done*.

[5] Więcej informacji na temat tworzenia dobrych celów sprintu można znaleźć na stronie *https://www.agilesocks.com/creating-good-sprint-goals/*.

Cel sprintu	Kontekst
Skrócić czas ładowania się raportu do mniej niż 2 sekund	Zespół pracujący nad raportami opartymi na bazie danych musiał odnieść się do zaistniałych problemów architektonicznych, żeby poprawić wydajność
Na zakończenie sprintu zaprezentujemy stronę docelową z wykorzystaniem infrastruktury podobnej do tej, która jest używana w środowisku produkcyjnym	Zespół pracujący nad nowym produktem musiał sprawdzić model infrastruktury i procesu wdrażania
Rozszerzenie metod płatności, żeby zawrzeć więcej opcji zapłaty	Produkt wspierał jednego dostawcę oraz jeden typ karty i musiał zostać rozszerzony o dodatkowe karty, PayPal, Amazon Pay i inne opcje. Istniało wiele niepewności, więc cel zostałby osiągnięty, gdyby można było przedstawić jeszcze jedną opcję

Rysunek 3.2. *Przykładowe cele sprintu*

Poniższe wskazówki pomogą Ci poprawić Twoje cele sprintu:

- *Unikaj złożonych celów sprintu.* Złożone cele sprintu, takie jak np. „zbudowanie X i Y oraz Z", rozpraszają uwagę zespołu scrumowego i nie pozwalają mu na bycie elastycznym. Czasami złożone cele sprintu występują wtedy, gdy zespoły pracują nad wieloma niepowiązanymi ze sobą inicjatywami lub gdy próbują wykonać zbyt wiele pracy w ramach sprintu. Złożone cele sprintu pozostawiają niewiele miejsca na pracę pojawiającą się w miarę tego, jak zespół scrumowy się uczy.
- *Nie próbuj uprawiać mikrozarządzania przy użyciu celów sprintu.* Nie ma potrzeby zawierania każdego elementu rejestru produktu w odrębnym celu sprintu. W rzeczywistości cel sprintu, który mówi o ukończeniu wszystkich elementów rejestru produktu, jest równoznaczny z nieposiadaniem żadnego celu sprintu. Jeżeli chcesz, żeby samoorganizujące się, mające realną władzę zespoły były efektywne, musisz wierzyć, że ludzie są zaangażowani i robią wszystko, co jest w ich mocy. Jeśli zespół scrumowy zrealizuje cel sprintu przed jego zakończeniem, członkowie zespołu zorientują się, co jeszcze mogą zrobić, żeby wnieść znaczący wkład w pracę nad produktem.
- *Ustanów mierzalny cel sprintu.* Kiedy dotrzecie do końca sprintu, cały zespół powinien być zgodny co do tego, czy cel sprintu został osiągnięty. Aby upewnić się, że cel sprintu jest jasny i zrozumiały, należy zapytać „Skąd będziemy wiedzieć, czy osiągnęliśmy cel sprintu?" podczas planowania sprintu i zastanowić się nad tym, czy możliwe jest przeprowadzenie obiektywnego pomiaru.
- *Zadbaj o to, żeby zespół osiągnął konsensus w sprawie celu sprintu.* Podczas planowania sprintu należy stosować technikę dochodzenia do konsensusu, aby potwierdzić zrozumienie i zaangażowanie wszystkich w osiągnięcie celu sprintu.
- *Stwórz cele sprintu, które pozwalają osiągnąć efekt biznesowy.* Ludzie chcą wykonywać sensowną pracę, która ma na coś wpływ. Aby to zapewnić, wypróbuj następujące sposoby:

- *Skoncentruj się na biznesie lub na użytkowniku.* Co użytkownik będzie w stanie zrobić po wdrożeniu tej funkcji? Co dany obszar biznesowy będzie w stanie osiągnąć dzięki temu udoskonaleniu?
- *Skoncentruj się na testowaniu założeń biznesowych i uzyskiwaniu informacji zwrotnej.* Trudno jest pozyskać wiedzę o tym, czego użytkownicy faktycznie potrzebują lub co chcą zrobić (ponieważ oni sami tego nie wiedzą). Właściciel produktu potrzebuje wczesnej informacji zwrotnej, aby przetestować założenia dotyczące wartości.
- *Skoncentruj się na ograniczaniu ryzyka.* Chodzi tu o udowodnienie, że wykorzystanie nowej technologii czy też architektury stanowi ważny element strategii zmniejszania ryzyka. Jeśli dowiesz się, że dana technologia nie spełnia Twoich potrzeb w zakresie wydajności, bezpieczeństwa czy skalowalności, możesz zmienić obrany kierunek. Im wcześniej wkroczysz na inną ścieżkę, tym tańszy będzie koszt takiej zmiany.

Użycie celów sprintu do przeprowadzenia efektywnego codziennego Scruma

Codzienny Scrum ma na celu umożliwienie zespołowi deweloperskiemu przeprowadzanie inspekcji w zakresie postępów w realizacji celu sprintu, adaptacji planu na podstawie nowo pozyskanej wiedzy, żeby osiągnąć cel sprintu, a także identyfikacji wszelkich przeszkód stojących mu na drodze do osiągnięciu tego celu. Codzienny Scrum oferuje możliwość ponownego zaangażowania się zespołu deweloperskiego i wspólne wzięcie odpowiedzialności przez jego członków.

Efektywny codzienny Scrum charakteryzują następujące cechy:

- *Promuje samoorganizację i współpracę.* Zespół deweloperski jest facylitatorem codziennego Scruma i aktualizuje rejestr sprintu. Kiedy wszyscy są skoncentrowani na swoim zaangażowaniu w realizację celu sprintu oraz widzą postępy w jego osiąganiu, pojawia się więcej możliwości dorzucenia swojej cegiełki i współpracy.
- *Sprzyja skupieniu i zmniejsza marnotrawstwo.* Codzienny Scrum powinien sprawiać wrażenie szybkiej sesji planowania współpracy. Jasny cel sprintu pomaga każdemu skupić się na celowości przeprowadzania codziennego Scruma i utrzymania go w 15-minutowych ramach czasowych.
- *Promuje transparentność i wspólne rozumienie postępów co do wyników w zakresie wartości.* Codzienny Scrum to nie tylko informowanie o statusie zadań, nad którymi pracuje każda jednostka[6]. Skupiając się na uzyskaniu ukończonego przyrostu, co z kolei pozwoli na osiągnięcie celu sprintu, zespół deweloperski przypomina sobie o swoim celu i zaangażowaniu w jego realizację. Członkowie zespołu mogą oceniać postępy w kontekście całego sprintu.

[6] Codzienny Scrum nie jest spotkaniem, na którym omawia się status prac; więcej informacji na ten temat można znaleźć na stronie *https://www.scrum.org/resources/daily-scrum-not-status-meeting*.

W przypadku zidentyfikowania nowych prac, które zagrażają celowi sprintu, mogą je przedyskutować i dostosować plan. Jeśli istnieją problemy spowalniające postępy prac, członkowie zespołu mogą je wcześnie zauważyć i również zmodyfikować plan.

Pod koniec codziennego Scruma każdy członek zespołu deweloperskiego powinien zrozumieć postępy w realizacji celu sprintu, aktualny plan jego osiągnięcia oraz prawdopodobieństwo dotarcia do tego celu. Ponadto członkowie zespołu powinni wiedzieć, co planują zrobić w ciągu najbliższych 24 godzin i jak zamierzają współpracować, aby osiągnąć ten cel. Dla przypomnienia: zastosowanie techniki konsensusu na końcu codziennego Scruma może być pomocne w kwestii zapewnienia każdemu możliwości wypowiedzenia się oraz zagwarantowania, że wszyscy tak samo postrzegają dane zagadnienie.

Wcześniejsze ukończenie prac nad elementami rejestru produktu

Oto kilka technik, które mogą pomóc Ci w szybszym ukończeniu prac nad elementami rejestru produktu:

- *Skorzystaj z definicji ukończenia w kontekście każdego elementu rejestru produktu.* Zamiast czekać aż do końca sprintu, aby przyjrzeć się definicji ukończenia i wykonać odpowiednie kroki, należy zastosować definicję ukończenia do każdego elementu rejestru produktu w ramach pracy nad nim i naprawdę ukończyć jedno zadanie, zanim przejdziesz do kolejnego. Takie podejście przynosi wymierne, dodatkowe korzyści: Może umożliwić rzeczywiste wydanie przyrostu produktu przed zakończeniem sprintu.
- *Rozbijaj elementy rejestru produktu na mniejsze części.* Mniejsze elementy rejestru produktu będą prawdopodobnie mniej złożone — wiązać się będzie z nimi mniej niewiadomych, a także będą wymagały mniejszego wysiłku.
- *Spróbuj jednodniowego sprintu.* Kiedy zespół scrumowy ma trudności z rozbiciem elementów rejestru produktu na mniejsze części i ze współpracą nad tymi samymi zadaniami, jednodniowy sprint może zmusić jego członków do zakwestionowania poczynionych założeń. Ten rodzaj sprintu raczej nie jest czymś, co chciałbyś robić na okrągło, ale może stanowić pouczające ćwiczenie w celu doskonalenia zespołu, pomocne w przełamywaniu ograniczających przekonań i zachęcające do kreatywnego podejścia do dostarczania wartości.

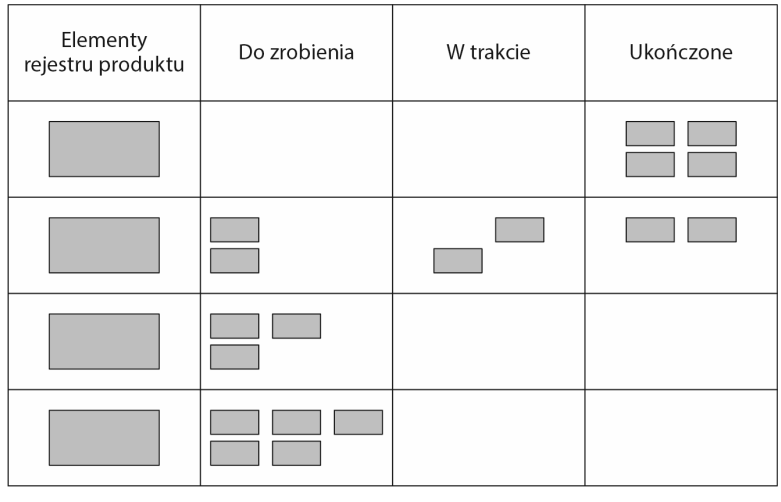
  Aby zaktywizować zespół, możesz zaproponować mu podjęcie następującego wyzwania: „Czy istnieje jakiś mały, mający wartość fragment pracy, który możemy dostarczyć użytkownikom w ciągu jednego dnia?” Zmniejszając przedział czasowy, wzmagasz koncentrację i poczucie pilności, zmuszasz członków zespołu do próbowania nowych rzeczy. Będą mniej martwić się o swoje specjalizacje i doświadczenie, a zamiast tego zaczną więcej myśleć o przyczynianiu się do wyników zespołu. Z próbą przeprowadzeniu jednodniowego sprintu wiąże się znikome ryzyko: jeśli bowiem to nie zadziała, na straty zostanie spisany tylko jeden dzień. Potencjalne korzyści jednak mogą być znaczące,

powstałe pod wpływem takiego sprintu spostrzeżenia są bowiem często istotne i skutkują podjęciem przez zespół wielu zobowiązań na rzecz następnego sprintu o normalnym czasie trwania.

- *Włącz zespół deweloperski w doskonalenie elementów rejestru produktu.* Zespół deweloperski musi rozumieć, co buduje, aby móc sprostać potrzebom biznesu. Rozmowy na ten temat mogą zacząć się toczyć przed rozpoczęciem sprintu, w trakcie doskonalenia elementów rejestru produktu. Co więcej, członkowie zespołu deweloperskiego będą mieli wiedzę, która pomoże im rozbić elementy rejestru na mniejsze części — i to w sposób umożliwiający większą elastyczność oraz kreatywność w osiąganiu wartościowych wyników. Zależności można zidentyfikować, a nawet da się potencjalnie rozwiązać związane z nimi problemy jeszcze przed zaplanowaniem, które elementy rejestru produktu wejdą do sprintu.

W praktyce: Używanie tablicy scrumowej do wizualizacji postępów

Tablica scrumowa stanowi wizualną reprezentację elementów rejestru produktu opracowywanych w sprincie oraz przebiegu samego sprintu (zob. rysunek 3.3). Zespoły deweloperskie mogą przechowywać w swojej przestrzeni fizyczną tablicę scrumową, która znajdując się w zasięgu wzroku, będzie dla nich łatwo dostępna. Dla pracujących zdalnie członków zespołu istnieje wiele cyfrowych narzędzi (zarówno darmowych, jak i płatnych), które zaspokajają tę potrzebę. Zespół deweloperski aktualizuje dane na tablicy scrumowej za każdym razem, gdy otrzyma nowe informacje na temat elementów rejestru produktu, nad którymi pracuje.



Rysunek 3.3. *Przykładowa tablica scrumowa*

Na rysunku 3.3, elementy rejestru produktu są przedstawione na większych kartach, zaś związany z nimi podział czynności w celu uzupełnienia tych elementów pojawia się w tym samym rzędzie. Kolumny na tablicy przedstawiają aktualny stan. Należy zachować czujność

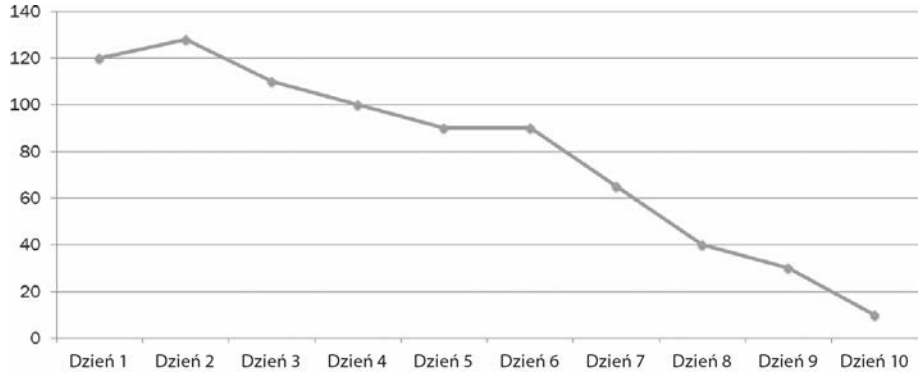
w przypadku tablic scrumowych, które zostały zaprojektowane w taki sposób, jaki sprzyja pracy w modelu kaskadowym lub powstawaniu silosów podczas sprintu, przez co poszczególne jednostki koncentrują się tylko na zawartości „swojej" kolumny. Przykładem może być kolumna „testowanie", która jest uważana za „własność" tych osób z zespołu deweloperskiego, które specjalizują się w testowaniu.

Kiedy zespół scrumowy efektywnie używa tablicy scrumowej, prawdopodobnie usłyszysz następujące rodzaje rozmów:

- Członkowie zespołu dyskutują o tym, jak wspólnie pracować nad elementami rejestru produktu, aby szybciej ukończyć pracę.
- Członkowie zespołu pytają, czy mogą pomóc przy zadaniach, które są już w toku, zamiast wybrać nowe elementy rejestru produktu i zacząć nad nimi pracować.
- Jeden członek zespołu doradza innemu na podstawie swojego doświadczenia, które zdobył w pracy nad danym obszarem produktu.
- Zespół, jako całość, dyskutuje nad tym, kto chciałby pracować nad poszczególnymi elementami rejestru produktu, biorąc pod uwagę obecne umiejętności oraz to, jak poszczególne osoby chciałyby się rozwijać.
- Członkowie zespołu wyrażają obawy, że nie uda się osiągnąć celu sprintu i renegocjują jego zakres — czyli to, co można usunąć z rejestru sprintu, co da się podzielić na mniejsze elementy, żeby mocniej skoncentrować się na celu sprintu.
- Członkowie zespołu potwierdzają, że nowe prace zagrażają osiągnięciu celu sprintu i dlatego nie zostaną one dodane do wykazu rejestru sprintu, ale przekierowane do właściciela produktu.
- Cały zespół zastanawia się nad tym, czy praca nie jest utrudniona, i dyskutuje, co zrobić, aby poprawić jej przepływ.
- Zespół, jako całość, omawia wpływ możliwych do zrealizowania zobowiązań, wynikających z retrospektywy sprintu, na to, jak zespół pracuje.
- Zespół, jako całość, odkrywa nową zależność i dyskutuje o tym, jak rozwiązać ten problem.
- Zespół, jako całość, potwierdza, że kryteria z definicji ukończenia zostały spełnione, a element rejestru produktu jest przenoszony na tablicy scrumowej do kolumny „ukończone".

W praktyce: Używanie wykresu spalania do śledzenia postępów w sprincie

Wykres spalania w sprincie śledzi elementy rejestru produktu w miarę zamykania prac nad nimi, co oznacza, że spełniły one wymogi zawarte w definicji ukończenia (zob. rysunek 3.4). Zauważ, że wykres ten różni się od wykresu śledzącego zadania lub godziny w sprincie, ponieważ skupia się na doprowadzeniu elementów rejestru produktu do ich ukończenia. Taki wykres pomaga zespołowi scrumowemu wizualizować jego pracę podczas sprintu.



Rysunek 3.4. *Przykładowy wykres spalania sprintu*

Wykres spalania sprintu może być wykorzystany podczas retrospektywy sprintu w celu omówienia tempa kończenia prac nad elementami rejestru produktu oraz określenia potencjalnych problemów i obszarów do poprawy. Pomaga to zespołowi scrumowemu zadawać pytania, które pozwolą mu lepiej zgłębić zagadnienia omawiane podczas retrospektywy sprintu:

- Co działo się w określonych dniach, gdy linia się spłaszczała i niemożliwe było ukończenie prac nad elementami rejestru produktu? Jakie zmiany mogłyby zmniejszyć marnotrawstwo w naszym procesie i umożliwić lepszy przepływ prac?
- Gdzie dostrzegliśmy konieczność wykonania nowej pracy? Jak zareagowaliśmy na to odkrycie? Co zrobimy inaczej w przyszłości?
- Co wydarzyło się w te konkretne dni, kiedy linia była bardziej stroma? Co pomogło nam ukończyć pracę nad większą liczbą elementów rejestru produktu? Jaką naukę możemy z tego wyciągnąć, żeby posunąć się do przodu?

Członkowie zespołu mogą również zapoznać się z wykresami spalania z poprzednich sprintów i zobaczyć, jak zmieniał się przepływ prac na przestrzeni czasu. Czy wprowadzone przez nich ulepszenia przyniosły pożądany efekt? Czy pojawiły się nowe wyzwania? Wizualizacja ta pomaga przypomnieć sobie szczegóły z przeszłości i odpowiednio się do nich dostosować.

Ograniczanie pracy w toku

Kiedy zespół deweloperski stosuje definicję ukończenia do każdego elementu rejestru produktu, może również chcieć ograniczyć liczbę elementów rejestru produktu, nad którymi trwają prace. Technika ta, która stosuje zasady Lean w celu poprawy przepływu, pomaga poprawić współpracę i proces uczenia się. Zespół deweloperski może na przykład ustalić limit dwóch elementów dla kolumny „w toku", która znajduje się na jego tablicy scrumowej. Kiedy dwa elementy rejestru produktu znajdują się w kolumnie „w toku", członek zespołu deweloperskiego, który ma wolne moce przerobowe, skoncentruje się na udzieleniu pomocy zespołowi w osiągnięciu postępów w pracach nad tymi dwoma elementami, a nie na rozpoczynaniu czegoś nowego.

Podejście to może stanowić wyzwanie dla zespołów deweloperskich, których członkowie mają trudności z ustaleniem, jak najlepiej wykorzystać swoje umiejętności i wiedzę do wspólnej pracy. Nie będą jednak w stanie sprostać temu wyzwaniu, jeśli nie spróbują się z nim zmierzyć. Ograniczanie pracy w toku wyznacza granicę, która zmusza ich do kwestionowania swoich starych sposobów pracy, do kreatywnego organizowania się wokół pracy i do chęci próbowania nowych rzeczy.

Na przykład załóżmy, że podczas codziennego Scruma Bob stwierdzi, iż chce rozpocząć pracę nad nowym elementem rejestru produktu, ponieważ aktualnie opracowywane elementy rejestru produktu nie mają żadnych potrzeb w zakresie projektowania ani rozwoju front-endu (który stanowi jego specjalizację). Zespół deweloperski może wskazać na limit pracy w toku i zachęcić Boba do podjęcia działań, które przyczynią się do ukończenia aktualnie opracowywanego zadania, takich jak np. przeprowadzenie niektórych testów, aktualizacja dokumentacji lub programowanie z kimś w parze.

Dlaczego mielibyśmy chcieć to zrobić? No cóż, krótka odpowiedź brzmi: z powodu prawa Little'a[7]. Nie będziemy tu wchodzić w szczegóły, ale zasadniczo im większa jest (przeciętnie) liczba rzeczy, którymi się zajmujesz w danym momencie, tym (przeciętnie) więcej czasu zajmie Ci ukończenie prac nad każdą z nich. Kiedy praca trwa długo, to wydaje się, że ludzie mają naturalną tendencję do jak najszybszego zaczynania nowych rzeczy, żeby mogli skończyć na czas, niezależnie od tego, co jest aktualnie w toku. Jednak w rezultacie — ze względu na wiele form marnotrawstwa powstałych w wyniku tej sytuacji — ukończenie prac trwa dłużej.

Żeby móc w sumie zrobić więcej, trzeba skupić się na robieniu mniejszej liczby rzeczy w tym samym czasie. Usuń przeszkody, które Cię spowalniają, zamiast wprowadzać więcej zadań do przepływu pracy.

Mierzenie i analizowanie przepływu

Kluczem do usprawnienia procesu zespołu jest zapewnienie transparentności co do tego, co się tak właściwie dzieje w ramach tego procesu. Obiektywne dane pomagają w identyfikacji potencjalnych wzorców, które mogą zostać pominięte, gdy po prostu polegamy na obserwacji lub staramy się zapamiętać, jak praca przebiegała w trakcie procesu.

Te środki i techniki mogą pomóc w zapewnieniu transparentności w procesie pracy zespołu:

- *Śledź czas swoich cykli.* Możesz mieć wiele cykli, które zdefiniujesz i zmierzysz na potrzeby swojego procesu. Aby nie komplikować niepotrzebnie spraw, zdefiniujemy tutaj czas trwania cyklu jako czas od momentu rozpoczęcia prac nad elementem rejestru produktu do chwili, gdy będzie on gotowy do wydania. (Uwaga: Niektóre zespoły definiują punkt końcowy jako rzeczywiste wydanie produktu, zwłaszcza jeśli stanowi ono część ich definicji ukończenia. Rób to, co jest sensowne dla Twojego zespołu).

[7] Przeczytaj tę białą księgę, aby uzyskać więcej informacji o tym, jak prawo Little'a stosuje się do zespołów scrumowych: *https://www.scrum.org/resources/littles-law-professional-scrum-kanban*.

- *Użyj wykresu spalania sprintu lub wykresu wydajności.* Monitoruj liczbę gotowych elementów rejestru produktu przypadających na daną jednostkę czasu. Niektóre zespoły mogą mierzyć spalanie w oparciu o wielkość elementów rejestru produktu (np. jeśli używają punktów historyjek do klasyfikacji wielkości). Klucz stanowi w tym wypadku zrozumienie przez zespół tego, kiedy poszczególne elementy rejestru produktu (a nie zadania) zostają ukończone.
- *Monitoruj swoją pracę w toku.* Monitoruj całkowitą liczbę elementów rejestru produktu, nad którymi prace już się rozpoczęły, ale jeszcze nie zakończyły w danej chwili. Elementy te są opracowywane, ale jeszcze nie dostarczają wartości.
- *Monitoruj zablokowane elementy rejestru produktu.* Kiedy prace nad elementem rejestru produktu zostały rozpoczęte, ale nie posuwają się naprzód, uważa się, że dany element jest zablokowany. Ponadto zespół może monitorować, jak długo konkretny element pozostaje zablokowany i jaka jest tego przyczyna.
- *Zwracaj uwagę na trendy!* Celem uchwycenia tych danych jest poszukiwanie trendów. W którym miejscu trendy wskazują na możliwości poprawy tego procesu?

W praktyce: Poprawa transparentności procesu i przepływu dzięki Kanbanowi

Kanban — strategia optymalizacji przepływu wartości dla interesariuszy poprzez proces, który wykorzystuje wizualny system ograniczonego przeciągania elementów do pracy w toku.

Przepływ to ruch wartości dla klienta w całym systemie rozwoju produktu. Kanban optymalizuje przepływ poprzez poprawę ogólnej wydajności, skuteczności oraz przewidywalności procesu[8].

Oparta na przepływie perspektywa Kanbana i skoncentrowanie się na transparentności oraz wizualizacji, w połączeniu z ramami postępowania Scrum, mogą pomóc zespołowi w zaprojektowaniu procesu optymalizacji środków dostarczania wartości dla klienta. Zespoły scrumowe osiągają optymalizację przepływu poprzez dodanie czterech poniższych praktyk do swojego procesu scrumowego:

- wizualizacja przepływu pracy,
- ograniczenie pracy w toku,
- aktywne zarządzanie elementami roboczymi należącymi do kategorii „praca w toku”,
- inspekcja oraz adaptacja zespołowej definicji przepływu pracy.

Kanban kładzie duży nacisk na większą transparentność przepływu pracy i określa minimalny zestaw metryk do jego pomiaru, które są wykorzystywane do aktywnego zarządzania elementami roboczymi, należącymi do pracy w toku, a także umożliwiają inspekcję i adaptację zorientowaną na przepływ. Wykorzystuje teorię kolejkowania

[8] Więcej informacji na temat wykorzystania Scruma z Kanbanem można znaleźć na stronie *https://www.scrum.org/resources/kanban-guide-scrum-teams*.

i prawo Little'a do pokazania, że aby ukończyć więcej zadań, należy pracować równolegle nad ich mniejszą liczbą. Tak, brzmi to jak wariactwo, ale jest to wręcz oszałamiająco prawdziwe stwierdzenie.

Ostatecznie Kanban umożliwia większy empiryzm poprzez dostarczenie kilku dodatkowych praktyk i metryk, które pomogą Ci w określeniu najlepszego procesu w ramach Scruma[9].

Budowanie z zachowaniem jakości od samego początku

Jakość obejmuje doświadczenie użytkownika w zakresie korzystania z produktu, w tym również to, jak dobrze wyważona jest jego elastyczność, łatwość konserwacji, wydajność oraz szybkość reakcji. Zespoły scrumowe potrzebują wczesnego i nieprzerwanego wglądu w jakość produktu, aby móc go dostosować do potrzeb klienta. W praktyce ta siła napędowa prowadzi zespoły do podejścia polegającego na testowaniu od samego początku, aby już na wstępie skupić się na jakości.

Przyjmując takie podejście, zespół scrumowy ustala swoje testy jeszcze przed budową rozwiązania, aby móc uzyskać wczesne informacje zwrotne, że realizowana solucja działa zgodnie z założeniami. Skutkuje to większym zasięgiem testów, jak również wyższą jakością, ponieważ wszystko to, co zostało zbudowane, będzie także zweryfikowane. Szereg praktyk wykorzystuje podejście polegające na testowaniu od samego początku, a niektóre z nich są połączone z automatyzacją:

- *Programowanie oparte na testowaniu (ang.* test-driven development*, TDD)*, w którym przed napisaniem kodu tworzone są zautomatyzowane testy jednostkowe, mające na celu napędzanie projektowania oprogramowania i wymuszenie odchodzenia od zależności. Praca składa się z:
  - napisania pojedynczego testu jednostkowego opisującego pewien aspekt funkcji,
  - przeprowadzenia testu, który powinien się nie udać, ponieważ program nie posiada tej funkcji,
  - napisania „dostatecznej ilości" kodu w najprostszy możliwy sposób, aby zaliczyć test,
  - refaktoryzacji kodu do czasu, aż będzie on zgodny z oczekiwanymi standardami programistycznymi (określonymi w definicji ukończenia).
    Proces ten jest powtarzany, tworząc zestaw zautomatyzowanych testów jednostkowych, który powiększa się w miarę upływu czasu. Ten zestaw zautomatyzowanych testów może być następnie w każdej chwili uruchomiony w celu potwierdzenia, że produkt nadal działa prawidłowo[10].

[9] Więcej informacji na temat Kanbana i Scruma można znaleźć na stronie *https://www.scrum.org/resources/blog/understanding-kanban-guide-scrum-teams*.

[10] *https://www.agilealliance.org/glossary/tdd/*.

- *Programowanie oparte na zachowaniu (ang.* behavior-driven development*, BDD)*, które bazuje na ogólnych zasadach programowania opartego na testowaniu i dodaje pomysły z projektowania opartego na domenie, aby stworzyć zautomatyzowane testy sprawdzające konkretne zachowanie, jakie chcesz, by przejawiała Twoja aplikacja. Elementy funkcjonalności są budowane przyrostowo, zgodnie z oczekiwanym zachowaniem. Deweloperzy wykorzystujący programowanie oparte na zachowaniu zazwyczaj używają swojego języka natywnego w połączeniu z językiem projektowania opartego na domenie, aby opisać cel i korzyści płynące z ich kodu[11].
- *Programowanie oparte na testach akceptacyjnych (ang.* acceptance test-driven development*, ATDD)* to praktyka w zakresie współpracy, w której użytkownicy i zespół deweloperski definiują kryteria akceptacji jeszcze przed zbudowaniem jakiejkolwiek funkcjonalności. Testy te reprezentują punkt widzenia użytkownika i opisują sposób działania systemu. W niektórych przypadkach zespół automatyzuje testy akceptacyjne. Takie podejście stawia walidację funkcjonalności biznesowej na pierwszym miejscu[12].

Automatyzacja i ukończenie

Automatyzacja staje się dziś coraz bardziej istotna w rozwoju produktów ze względu na szybkość, z jaką wiele firm musi dostarczać klientom wartość, z powodu złożoności i skali produktów oraz systemów, a także z uwagi na znaczenie jakości w świecie, który w coraz większym stopniu zależy od technologii. Automatyzacja pracy może być zniechęcająca dla zespołów, które nie mają z nią doświadczenia, wymaga bowiem znacznego wysiłku, ale płynące z niej korzyści są tego warte:

- zmniejszenie manualnego nakładu pracy poniesionego przez członków zespołu,
- zmniejszenie wąskich gardeł związanych z określonymi zestawami umiejętności,
- stwierdzenie wad wcześniej w cyklu dostawy,
- zmniejszenie szansy wystąpienia błędu człowieka,
- ukierunkowanie umiejętności oraz energii członków zespołu na stawianie czoła nowym wyzwaniom i uczenie się nowych rzeczy.

Ogólnie rzecz biorąc, korzyści te przyczyniają się do wykształcenia umiejętności szybszego dostarczania wartości o wyższej jakości, a także pozwalają ludziom szybciej się uczyć. Jeśli budujesz nowy produkt, nie oszczędzaj na automatyzacji i nie odkładaj jej na później. Łatwiej jest zacząć od małego produktu i rozwijać automatyzację w miarę tego, jak produkt rośnie. Poza tym wcześniej odniesiesz korzyści z automatyzacji. A jeśli znajdujesz się w sytuacji, w której produkt jest już całkiem spory, to będziesz musiał zjeść tę żabę, kawałek po kawałku.

[11] *https://www.agilealliance.org/glossary/bdd/*.

[12] *https://www.agilealliance.org/glossary/atdd/*

Niektóre zespoły wiedzą, że powinny automatyzować, ale ciągle są „zbyt zajęte", aby poczynić jakikolwiek krok w tym kierunku. Rozwiązanie tego dylematu stanowi wykonanie mniejszej ilości pracy w sprincie, aby zainwestować czas i zasoby w automatyzację — w celu wytworzenia dodatkowej zdolności do wykonywania większej ilości pracy w przyszłości. Zespół deweloperski powinien współpracować z właścicielem produktu, aby pójść na odpowiednie kompromisy pomiędzy zmniejszeniem postępów na chwilę obecną, a zrobieniem większych postępów w przyszłości.

Automatyzacja jest często omawiana w następującej kolejności:

1. kontrola wersji,

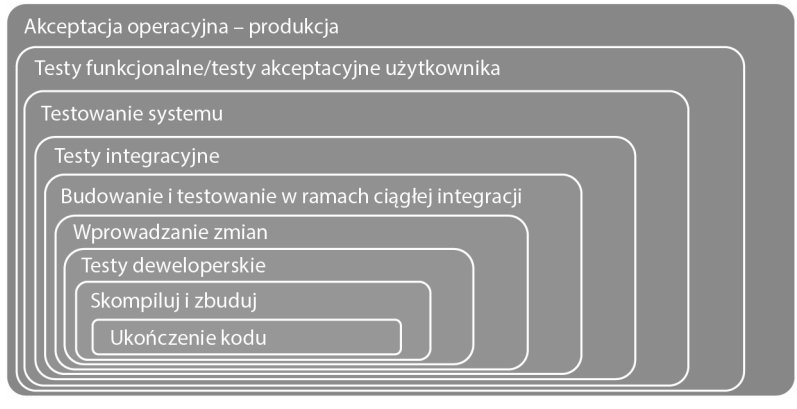
2. zautomatyzowana budowa danej wersji oprogramowania,

3. zautomatyzowane testy,

4. zautomatyzowane pakiety,

5. zautomatyzowane wdrożenie.

Działania związane z automatyzacją i kontrolą jakości mogą być częścią definicji ukończenia, którą stworzył zespół deweloperski. Da się także rozważyć zastosowanie wielu różnych poziomów testowania, co pokazano na rysunku 3.5. Ostatecznie im więcej poziomów poddanych automatyzacji, tym większą masz pewność co do jakości w budowanym produkcie.



Rysunek 3.5. *Poziomy testowania*

Manifest zwinnego wytwarzania oprogramowania stwierdza, że zespoły agile'owe bardziej sobie cenią działające oprogramowanie od obszernej dokumentacji. Oznacza to, że zespół musi budować i testować oprogramowanie. I to często. A tak właściwie to przez cały czas. W idealnym świecie zespół wydaje nową wersję oprogramowania za każdym razem, gdy dokonuje zmiany w repozytorium kodu źródłowego. Ale nowe wersje oprogramowania są tylko punktem wyjścia — zespół musi również przeprowadzać testy, i to za każdym razem,

gdy wprowadza zmiany. W dodatku nie tylko testy jednostkowe, ale również funkcjonalne, regresyjne, wydajności oraz skalowalności — wszystkie napędzane przez interfejs programistyczny aplikacji. Testy te powinny być przeprowadzane nie w prostych środowiskach, ale takich, które są możliwie najbardziej zbliżone do produkcji. Kiedy zespoły tak robią, dostarczają lepszy jakościowo kod[13].

Doskonałość techniczna stanowi niezbędną zdolność zespołów deweloperskich do ich dalszego wzrostu i rozwoju, zaś praktyki automatyzacyjne są głównym elementem tych wysiłków. Ciągła dostawa (CD) i ruch DevOps to obszary do dalszej eksploracji[14].

DevOps

Istnieją spore niejasności co do tego, czym DevOps (ang. *development and operations*, czyli połączenie rozwoju oprogramowania z jego eksploatacją) jest, a czym nie jest. Wiele źródeł skupia się na praktykach technicznych, niektóre z nich zostały już wyjaśnione wcześniej (np. ciągła integracja, testy automatyczne, ciągła dostawa, infrastruktura jako kod). Ważne jest jednak to, aby nie stracić z oczu nadrzędnego celu DevOps: DevOps przełamuje bariery pomiędzy eksploatacją a rozwojem — po to, żeby zwiększyć zwinność[15].

Przegląd kodu

W trakcie przeglądu kodu członek zespołu deweloperskiego (lub wielu członków) dokonuje recenzji pracy wykonanej przez innego członka zespołu w celu zagwarantowania kontroli jakości. Zespoły stosujące tę praktykę często dysponują listą kontrolną rzeczy, na które zwracają szczególną uwagę podczas przeglądu kodu, a które prawdopodobnie zostałyby ujęte w ramach definicji ukończenia. Oprócz dbania o jakość od samego początku przegląd kodu stanowi dla ludzi doskonałą okazję do nauki, do ciągłego rozwijania ich wiedzy o systemie, a także do pogłębiania ich wiedzy oraz umiejętności programistycznych[16].

[13] *https://www.scrum.org/resources/blog/what-devops-taught-me-about-agile*.

[14] Książka *Continuous Delivery* (Boston: Addison-Wesley, 2010) autorstwa Jeza Humble'a i Davida Farleya stanowi doskonałą pomoc dydaktyczną w zakresie szerszych tematów technicznych, w tym na temat zarządzania konfiguracją, ciągłej integracji, wdrażania potoków, zarządzania infrastrukturą i środowiskami, testowania oraz zarządzania danymi.

[15] Więcej informacji o tym, jak zwinność i DevOps uzupełniają się wzajemnie, można znaleźć na *https://www.scrum.org/resources/convergence-scrum-and-devops*.

[16] Aby zyskać perspektywę, dlaczego przeglądy kodu rzeczywiście oszczędzają czas, zajrzyj na *https://www.atlassian.com/agile/software-development/code-reviews*.

Metryki jakości

Mierzenie aspektów produktu, które są wskaźnikami jakości, pomaga stworzyć transparentność, żeby zespół scrumowy mógł kontrolować i dostosowywać sposób budowania produktu oraz to, jak dokonywane przez niego wybory wpływają na jakość. Niektóre z tych metryk są generowane przez zautomatyzowane narzędzia; inne mogą być monitorowane manualnie. Należy pamiętać, że w przypadku metryk trendy są bardziej informatywne niż konkretne punkty danych. Co więcej, zespoły scrumowe będą chciały dokonywać inspekcji wielu metryk, biorąc pod uwagę dużą liczbę zmiennych (zarówno tych znanych, jak i nieznanych), które mogą mieć wpływ na daną metrykę. Należy pamiętać, że te metryki są przeznaczone do użytku zespołu scrumowego. Aby uszanować i utrzymać transparentność, nie powinny być wykorzystywane przez osoby spoza zespołu do pomiaru wyników zespołu ani nie powinny być wykorzystywane w celach motywacyjnych.

Niektóre z powszechnie stosowanych metryk zostały podsumowane poniżej:

- *Metryki pokrycia kodu* wskazują, jak duża część bazy kodowej produktu została objęta przez zautomatyzowane testy. Jest to powszechnie spotykany punkt danych dla zespołów stosujących programowanie oparte na testach. Chociaż wysoki procent pokrycia nie gwarantuje, iż kod został dobrze napisany i że testy są przeprowadzane, to stanowi jednak pomoc dla zespołów.
- *Metryki złożoności* pomagają zespołowi deweloperskiemu zrozumieć stopień trudności utrzymania kodu. Ten rodzaj współczynnika jakości odzwierciedla długoterminową skuteczność i wydajność zespołu scrumowego. Przykłady sposobów pomiaru złożoności obejmują złożoność cyklomatyczną, głębokość dziedziczenia, sprzężenia klasowe, zagnieżdżanie oraz metryki Halsteada.
- *Metryki stabilności wersji* są wiodącym wskaźnikiem ogólnej stabilności produktu. Wskazują one, czy proces tworzenia wersji jest stabilny, i ujawniają problemy związane z jakością kodu. Stabilność tworzenia wersji można zmierzyć na podstawie liczby dni od ostatniego czerwonego statusu (niepowodzenia przy tworzeniu wersji) oraz kolejnych dni utrzymywania się czerwonego statusu.
- *Metryki defektów* dają wgląd w jakość produktu. Błędy są kosztowne w przypadku ich naprawiania w produkcji i często przeszkadzają zespołowi w dostarczaniu nowych cech oraz funkcji. Błędy mogą mieć znaczące oddziaływanie na użytkowników, wpływając ostatecznie na markę firmy, jej reputację oraz wyniki finansowe.

Oprócz samych pomiarów, ważne są również trendy:

- Ile defektów odkryto już w produkcji, a ile naprawiono przed wdrożeniem do produkcji i jak ich stosunek do siebie zmienia się w czasie?
- Ile średnio trwa naprawianie usterek produkcyjnych i jak zmienia się to w czasie?
- Jaka jest średnia krytyczność usterek (tzn. wysokość związanych z nimi kosztów oraz ich wpływ na firmę i klientów), a także jak zmienia się ona w czasie?

- Ile defektów się powtarza i jak się to zmienia w czasie?
- Jak długo defekty te pozostają otwarte i jak się to zmienia w czasie?

Walka z długiem technologicznym

Dług technologiczny to nic innego jak odroczenie prac nad produktem — które często jest wynikiem decyzji zespołu deweloperskiego o tym, aby zwiększyć tempo kosztem jakości.

Dług technologiczny może być postrzegany jako kruchy lub trudny do zmodyfikowania kod. Należy jednak zauważyć, że zaciągnięcie długu technologicznego nie musi zawsze być czymś złym — rzecz jasna, o ile istnieje realny plan jego spłaty. Tak jak w przypadku finansowego zadłużenia, czasami może się okazać czymś dobrym, jeżeli zwrot jest większy niż zapłacone odsetki. Na przykład rozsądną decyzją będzie stworzenie prototypu w celu przetestowania określonych założeń w warunkach rynkowych przed zainwestowaniem w stworzenie solidnego back-endu. Dług technologiczny musi być transparentny — tak samo jak wpływ jego zaciągnięcia na przyszłą zdolność do dostarczania wartości.

Przykłady długu technologicznego obejmują:

- brak zautomatyzowanych testów, wdrożeń lub zautomatyzowanego tworzenia wersji,
- brak testów jednostkowych,
- wysoką złożoność kodu,
- silnie sprzężony kod,
- logikę biznesową w niewłaściwych miejscach,
- zbyt małą liczbę testów akceptacyjnych,
- zduplikowany kod lub moduły,
- nieczytelne nazwy lub algorytmy.

Jeśli dług technologiczny produktu narasta i osiąga taki punkt, że trudno jest uzyskać nadający się do wydania przyrost (o mającej znaczenie wartości) na koniec każdego sprintu, trzeba zacząć spłacać to zadłużenie — i to szybko. Rzecz jasna chcesz uniknąć dotarcia do tego punktu, a sposobem na to będzie rozwiązanie problemu długu technologicznego — i zrobienie tego raczej prędzej niż później[17].

[17] Aby zgłębić temat długu technologicznego, obejrzyj na stronie *Scrum.org* webinarium autorstwa Marka Nonemana pt. *Dealing with Technical Debt: Avoiding Technical Bankruptcy*: *https://www.scrum.org/resources/dealing-technical-debt*.

W praktyce: Zmodyfikuj definicje ukończenia tak, aby zapewnić niską tolerancję dla długu technologicznego

Pomocne w tym względzie może się okazać uwzględnienie w definicji ukończenia konkretnych oczekiwań dotyczących długu technologicznego. Co zrobi zespół deweloperski, żeby uniknąć dalszego zaciągania długu technologicznego? A co zrobi w celu rozwiązania problemu już istniejącego długu technologicznego?

Oto kilka przykładów rzeczy, które zespół deweloperski może chcieć odnieść się do zaistniałych problemów w definicji ukończenia w związku z długiem technologicznym:

- Co należy poddać refaktoryzacji?
- W jaki sposób oddamy moduł w lepszym stanie od tego, w jakim go zastaliśmy (zasada harcerza)?
- Co zrobimy, gdy napotkamy dług technologiczny w trakcie prac nad przyrostem? Jakiego rodzaju dług technologiczny chcemy znaleźć i spłacić w trakcie prac?
- Co zrobimy z długiem technologicznym, który pozostaje niespłacony?

Definicja ukończenia autorstwa zespołu deweloperskiego stanowi formę porozumienia roboczego, dlatego ważne jest to, aby wyraźnie określić, w jaki sposób zamierzamy zapobiegać powstawaniu długu technologicznego i jak chcemy nim zarządzać. Porozumienie to zarówno pomaga członkom zespołu wzajemnie rozliczać się z wysokich standardów jakościowych, jak i sprawia, że jakość staje się bardziej transparentna dla zainteresowanych stron.

Zapewnienie transparentności długu technologicznego

Dodanie długu technologicznego do rejestru produktu sprawi, że będzie on przejrzysty dla zespołu scrumowego oraz interesariuszy. Krok ten umożliwi również właścicielowi produktu podjęcie lepszych decyzji odnośnie do zlecenia pracy. Jeśli to zrobisz, upewnij się, że wyraźnie uchwycisz wartość spłaty długu technologicznego w kategoriach biznesowych. Co zyska biznes dzięki spłaceniu tego długu technologicznego? Ile będzie to kosztować firmę, jeśli ta nie spłaci długu technologicznego? Jaki wpływ niespłacony dług technologiczny wywiera na użytkowników?

Oto kilka przykładów ujmowania wartości w kategoriach biznesowych. Rozważ dodanie danych odzwierciedlających istniejące problemy z jakością lub oczekiwany wzrost wydajności na podstawie historii danego produktu.

- Przeprowadź refaktoryzację, żeby zminimalizować liczbę ścieżek prowadzących przez kod, skracając w ten sposób czas potrzebny na testy o 30 procent.
- Stosuj spójne nazewnictwo i konwencje strukturalne, dzięki czemu członkowie zespołu mogą być skuteczniejsi i wydajniejsi przy tworzeniu nowych funkcji i rozwiązywaniu problemów.

- Dokonaj centralizacji logiki biznesowej dla funkcji X, ułatwiając tym samym aktualizację logiki biznesowej w przyszłości i zmniejszając prawdopodobieństwo wystąpienia błędów. W ciągu ostatnich czterech miesięcy zarejestrowaliśmy 24 błędy, które miały bezpośredni wpływ na klientów i spowodowały stratę zysku w wysokości 100 000 USD.
- Przeprowadź refaktoryzację cechy Y, zwiększając wydajność systemu o 35 procent i powodując tym samym poprawę szybkości transakcji o 2,5 sekundy.
- Dokonaj refaktoryzacji cechy Z, dzięki czemu teraz — gdy opłacalność w kontekście klienta została udowodniona — będziemy w stanie skalować rozwiązanie dla szerszej bazy użytkowników, co doprowadzi do zwiększenia przychodów.

W praktyce: Traktuj dług technologiczny tak, jak zadłużenie na karcie kredytowej

Przestań się dalej zadłużać. I zacznij spłacać ten dług na raty — po kawałku w każdym sprincie. Jeśli dług technologiczny wzrósł tak bardzo, że zespół stoi przed nie lada wyzwaniem za każdym razem, gdy ma dostarczyć ukończony przyrost, mający przynieść znaczącą wartość dla firmy, to zespół scrumowy powinien przeprowadzić dyskusję na ten temat — i ustalić, czy już nadszedł czas, żeby przeciąć tę symboliczną kartę kredytową i zacząć spłacać w każdym sprincie nie tylko odsetki, ale też część samego długu.

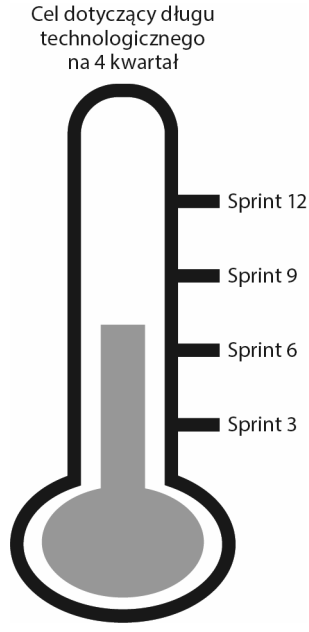
Zespół deweloperski może pomóc właścicielowi produktu zrozumieć krytyczność odniesienia się do zaistniałego problemu długu technologicznego i wynegocjować z nim odpowiednią ilość czasu, jaką trzeba będzie zarezerwować w nadchodzących sprintach na pracę nad tymi elementami rejestru produktu, które są związane z długiem technologicznym. Właściciel produktu może pomóc interesariuszom zrozumieć, jaką wartość zyskają dzięki rozwiązaniu tych problemów.

Podejście to należy wciąż poddawać inspekcji oraz adaptacji, aby zrozumieć uzyskane korzyści i odpowiednio dostosować ilość czasu poświęcanego na rozwiązywanie kwestii długu technologicznego.

Uwidocznienie „spłaty” długu technologicznego

Jeśli produkt ma znaczny dług technologiczny do spłacenia, należy rozważyć stworzenie dla zespołu widocznego wskaźnika postępu tej spłaty. Krok ten rozpoczyna się od wyznaczenia przez zespół scrumowy wstępnego celu początkowego, a może nawet kilku celów. Członkowie zespołu mogą następnie stworzyć np. kartę informacyjną (lub inną formę wizualizacji), która będzie prezentować ich postępy w spłacaniu długu technologicznego podczas każdego sprintu.

Rysunek 3.6 przedstawia przykład użycia termometru jako metafory i wskaźnika postępu. Być może zespół scrumowy stawia sobie za cel spłatę określonej części długu technologicznego w trakcie kilku sprintów, a ów termometr pomaga im celebrować drobne postępy poczynione w każdym sprincie. Innym podejściem do tego tematu jest wykres spalania pokazujący



Rysunek 3.6. *Wizualizacja długu technologicznego może dostarczyć motywacji do zrobienia czegoś w tej sprawie*

całkowitą kwotę znanego długu technologicznego (np. ujętą w transparentny sposób w rejestrze produktu), a także śledzenie w czasie występującego trendu, w miarę jak dług technologiczny jest spłacany, odkrywany czy też nawet celowo zaciągany.

Podsumowanie

Dotarcie do punktu ukończenia prac jest niezbędne do dostarczenia wartości przy jednoczesnym kontrolowaniu ryzyka i zarządzaniu kompleksowymi tematami. W miarę tworzenia przez zespół scrumowy coraz mocniejszych fundamentów znajdzie się on naturalnie w pozycji umożliwiającej mu rozwój procesu zespołu w celu efektywniejszego dostarczania wysokiej jakości, wartościowego produktu. W Scrumie nie ma najlepszych praktyk, więc zespoły muszą nieustannie sprawdzać, co robią, dlaczego to robią i jakie korzyści z tego tytułu odnoszą (lub jakich już nie odnoszą). Technologia i biznes zmieniają się cały czas, więc zespoły powinny patrzeć w przyszłość, zastanawiać się, co będzie dalej i co powinny robić inaczej, żeby sprostać nowym potrzebom, wyprzedzić konkurencję, a także zachwycić klientów.

Elastyczność wynikająca z minimalnych granic, jakie stawiają scrumowe ramy postępowania, otwiera wiele możliwości uwolnienia kreatywności współpracujących zespołów. Mogą one poruszać się po tych możliwościach, szukając wciąż sposobów na polepszenie transparentności swojego procesu oraz jego wpływu na wyniki, a także często przeprowadzając inspekcję

oraz adaptację procesu zespołu na podstawie swoich spostrzeżeń. Zespoły powinny przynajmniej zastanowić się nad tym, jak dobrze stosują definicję ukończenia i czy odpowiednio definiują cele sprintu, a także jak wygląda przepływ pracy w ich procesie oraz jakie praktyki i środki służące zapewnieniu jakości powinny być wdrożone.

Wezwanie do działania

Wspólnie ze swoim zespołem zastanów się nad poniższymi pytaniami:

- Czy niezawodnie docierasz do etapu ukończenia pracy przynajmniej na koniec każdego sprintu?
- Jeśli nie, to co zrobił zespół, żeby temu zaradzić? W przeciwnym razie — jakie macie możliwości, żeby działać jeszcze efektywniej, z większą radością, zapewniając jeszcze wyższą jakość?
- W jaki sposób wartości Scruma są realizowane w Twoim zespole?
- Jak bardzo solidna jest definicja ukończenia? Jak ewoluowała ona w czasie i jakie są możliwości jej poprawy?
- W jaki sposób wykorzystywane są cele sprintu w całym okresie jego trwania?
- Jaka jest transparentność przepływu pracy w ramach sprintu? W jakich obszarach należy zwiększyć transparentność?
- Jaki jest trend jakościowy dla danego produktu?
- Ile wynosi długu technologiczny produktu? Zwiększa się on czy też zmniejsza?
- W jakich obszarach zespół musi rozwijać wiedzę i umiejętności, aby móc wcześniej wytworzyć nadający się do wydania przyrost, który będzie mieć odpowiednią jakość?
- Które wyzwania stanową teraz największe bolączki zespołu? Zidentyfikuj jeden lub dwa możliwe do przeprowadzenia eksperymenty, aby pomóc rozwinąć proces zespołu, a tym samym efektywniej ukończyć dane zadania. Dla każdego eksperymentu należy określić jego pożądane skutki i sposób ich pomiaru.

SCRUM: PEWNA DROGA DO SUKCESU DLA PROFESJONALISTÓW!

Wszędzie można się natknąć na brak profesjonalizmu. W projektach, które wymykają się spod kontroli, kosztują krocie, a nie przynoszą żadnych korzyści. W blokowaniu cennych pracowników, którzy latami nie dostają szansy rozwoju, co podważa ich zaufanie do firmy i szkodzi relacjom w środowisku pracy. Większość tych problemów miał rozwiązać Scrum. Scrum jednak, mimo że prosty do zrozumienia, jest trudny do praktycznego opanowania. Mnóstwo zespołów go zaimplementowało, ale jedynie pozoruje jego stosowanie: używa terminologii scrumowej, nie rozumiejąc jej intencji ani nie wykazując się dyscypliną, której wymaga Scrum.

Ta książka przyda się każdemu, kto chce efektywniej stosować praktyki Scruma i w pełni wykorzystać jego potencjał transformacyjny. Skoncentrowano się tu na sprawdzonych podejściach do Scruma, aby sukcesywnie osiągać coraz wyższą jakość oraz uzyskiwać i natychmiast wykorzystywać informacje zwrotne. Przedstawiono również praktyczne sposoby zwiększania elastyczności zespołu i wprowadzania adekwatnych usprawnień pracy całej organizacji. Ten praktyczny przewodnik docenią zarówno Scrum masterzy, członkowie zespołów, jak i właściciele produktu. Zawarte w nim wskazówki pozwolą osiągnąć wysoki poziom transparentności i odwagi w podejmowaniu typowych wyzwań w nieustannie zmieniającym się i nieprzewidywalnym środowisku.

Dzięki tej książce:

- ocenisz, jakim jesteś Scrum masterem
- dowiesz się, jak prowadzić świetny zespół scrumowy
- nauczysz się mierzyć i optymalizować wartości każdego przyrostu produktu
- udoskonalisz sposób planowania, tworzenia i rozwijania
- usuniesz bariery organizacyjne blokujące zwinność i profesjonalizm

STEPHANIE OCKERMAN od 15 lat zajmuje się szkoleniami zespołów Agile, jest Scrum masterem, trenerką, coachem. Pomaga firmom w przeprowadzaniu zmian organizacyjnych w nieprzewidywalnym i złożonym środowisku. Lubi pisać, przemawiać, fotografować i podróżować.

SIMON REINDL od ponad 25 lat zajmuje się opracowywaniem i rozwojem przeróżnych produktów. Pracował we wszystkich rolach przewidzianych w Scrumie i pomagał zarówno start-upom, jak i międzynarodowym korporacjom czy instytucjom rządowym. Wyszkolił tysiące ludzi na całym świecie.

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

Sprawdź nasze szkolenia!
SZKOLENIA
AKADEMIA IT & BUSINESS
HELIONSZKOLENIA.PL

KOD KORZYŚCI
Sięgnij po więcej!

ISBN 978-83-283-6912-2

INFORMATYKA W NAJLEPSZYM WYDANIU

Cena: 39,90 zł

Pearson
Addison-Wesley