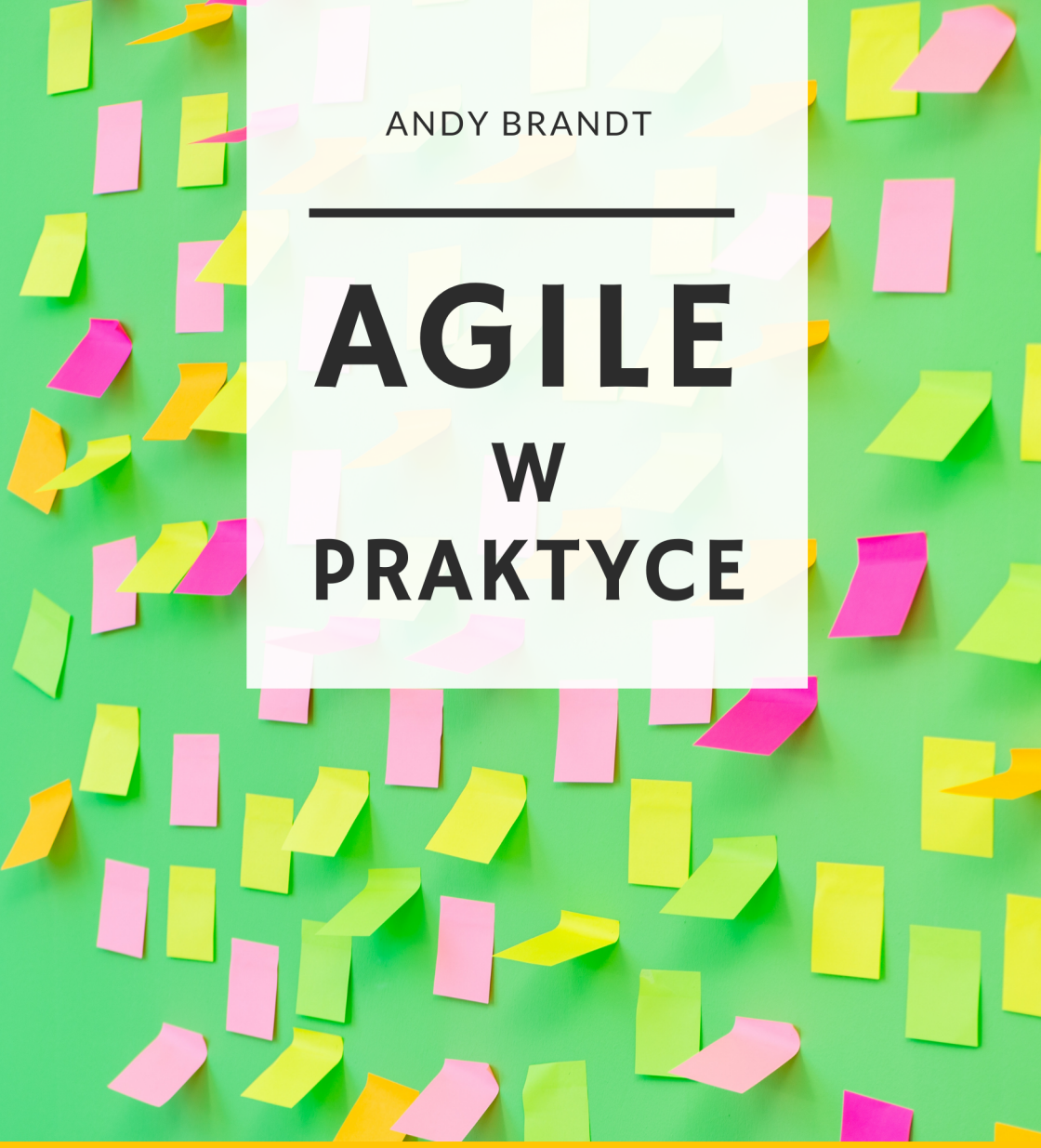


ANDY BRANDT

AGILE W PRAKTYCE

Podstawowy podręcznik Agile od 2013 roku.

Zawiera aktualne omówienie Scruma, Kanbana, prowadzenia backlogu, retrospekcji, zagadnień skalowania i problemów menedżerskich związanych z wprowadzaniem Agile.

DRUGIE WYDANIE 2020

Agile w praktyce

Podręcznik metod zwinnych

Andy Brandt

Ta książka jest do kupienia na
http://leanpub.com/Agile_w_Praktyce

Wersja opublikowana 2020-11-28

ISBN 978-83-945494-0-4

Tweet'nij tę książkę

Proszę pomóż Andy Brandt rozpowszechniać informację o tej książce na Twitter'ze!

Sugerowany tweet o tej książce to:

A teraz będę czytać "Agile w praktyce" od @agileroshi

Sugerowany tag do tej książki to #agilewpraktyce.

Sprawdź co inni ludzie mówią o tej książce i kliknij na ten link, żeby poszukać tego tagu na Twitter'ze:

#agilewpraktyce

Pozostałe pozycje autora

Andy Brandt

Environment for Agile Teams

Jak oswoić Agile

Praca Zdalna

Spis treści

1. Scrum

Siła Scruma polega na stworzeniu prostej, minimalnej „ramy postępowania", wewnątrz której możliwe jest łączenie różnych praktyk w sposób, który umożliwia tworzenie wartościowych produktów w środowisku niestabilnej złożoności.

Scrum został stworzony przez Kena Schwabera i Jeffa Sutherlanda na początku lat dziewięćdziesiątych ubiegłego wieku[1]. Już w pierwszej dekadzie XXI wieku zdobył sobie popularność i jest obecnie wiodącą[2], podstawową metodą z rodziny Agile. Metoda[3] ta była początkowo przekazywana ustnie przy okazji jej wprowadzania w zespołach czy na konferencjach, następnie w postaci zorganizowanych szkoleń (pierwsze takie szkolenie odbyło się w 1997 roku). W 2006 roku ukazała się książka „Software development with Scrum"[4] napisana przez Kena Schwabera i Mikea Beedle. Znana niegdyś wśród praktyków Scruma jako „mała czarna książeczka" była to pierwsza kompletna prezentacja metody w formie książkowej, która pozwoliła szerszemu gronu zainteresowanych na zapoznanie się z nią. Od 2010 roku Scrum jest zdefiniowany w Scrum Guide - niezwykle zwięzłym

[1] Sami twórcy podają, że Scrum powstał w 1995 roku. 18 listopada 2020 obchodzono oficjalnie 25-cio lecie Scruma (przy tej okazji wydając Scrum Guide 2020).

[2] W najróżniejszych badaniach, w których respondenci proszeni są o wskazanie metody czy też procesu stosowanego w ich firmie do budowy oprogramowania Scrum jest wskazywany najczęściej jeśli respondent twierdzi, że jego firma używa metod agile.

[3] Za Słownikiem Języka Polskiego PWN: „metodyka [gr.], *metodol.* zbiór zasad dotyczących sposobów postępowania, efektywnych ze względu na określony cel". Scrum wyczerpuje słownikową definicję metodyki, jednak ze względu na skojrzenia jakie niesie ze sobą to słowo (duże, ciężkie, zbiurokratyzowane metodyki takie jak Prince2) oraz na osobistą prośbę twórców Scruma stosuje się raczej na jego określenie słowo „metoda".

[4] http://goo.gl/ITW6a

bo zaledwie 13-o stronicowym dokumencie napisanym i utrzymywanym wspólnie przez obu jego twórców. Ostatnia wersja Scrum Guide ukazała się 18 listopada 2020 roku.

Poniżej przedstawiam samą metodę Scrum w sposób maksymalnie zwięzły, by następnie omówić niektóre praktyczne jej aspekty. Szczegółowe omówienie niektórych praktyk wykorzystywanych często wraz ze Scrumem lub będących jego częścią znajduje się w innych rozdziałach. Niektóre z tych praktyk są używane także w innych metodach Agile, a wiele z nich jest technikami samymi w sobie czyli można je stosować także „pojedyńczo". Przy tym rozdział ten nie zastępuje Scrum Guide, który i tak zalecam przeczytać.

Na końcu tego rozdziału znajduje się także krótkie omówienie ostatnich zmian w Scrumie (kolejne wersje Scrum Guide z lat 2011, 2013, 2016, 2017 i 2020).

1.1 Scrum w pigułce

Scrum to metoda organizacji pracy nad wytwarzaniem produktu (w kontekście oprogramowania: aplikacji, systemu itp.), w której nacisk kładzie się na dostarczanie maksymalnej wartości poprzez regularne i częste dostarczanie działającego oprogramowania, co umożliwia szybką empiryczną inspekcję stanu produktu i ułatwia wprowadzanie zmian. Łatwość zmiany umożliwia adaptacyjne reagowanie na zmianę, a więc radzenie sobie z problemami z domeny niestabilnej złożoności.

Scrum Guide opisuje w jaki sposób jeden niewielki zespół może pracując nad jednym produktem dostarczać często jego nowych, wartościowych wersji reagując na zmianę. Scrum jest celowo niekompletny, a więc nie jest szczegółową instrukcją

działania – przeciwnie, określa jedynie ramy postępowania dla zespołu Scrum. Konkretne praktyki zastosowane w ramach pracy w Scrum są wynikiem decyzji tego zespołu.

Oficjalna definicja metody jest następująca: *„Scrum: lekki framework, który pomaga ludziom, zespołom i organizacjom tworzyć wartość poprzez adaptacyjne (empiryczne) rozwiązywanie złożonych problemów”.*

W ostatnich latach coraz częściej podkreśla się, że metoda Scrum może być wykorzystywana przy każdej pracy polegającej na wytwarzaniu złożonego produktu, cechy/wymagania którego są niestabilne lub nie w pełni znane z góry, a który wymaga wspólnego, jednoczesnego wysiłku grupy ludzi o zróżnicowanych kompetencjach. Z tego powodu w aktualnym Scrum Guide usunięto odniesienia do oprogramowania. Jednak ja w moim omówieniu będę nadal koncentrował się na zastosowaniu Scruma właśnie w tej dziedzinie. Wynika to z dwóch powodów. Po pierwsze, to dziedzina, którą ja sam znam najlepiej. Po drugie mam pewnę wątpliwości, czy takie „uogólnianie” Scruma jest w ogóle dobrym trendem.

Elementy Scruma

Scrum składa się z następujących elementów:

- Trzech odpowiedzialności: Deweloperzy (*Developers*), Scrum Master oraz Product Owner tworzący razem Zespół Scrum (*Scrum Team*), w skrócie zespół (*Team*).
- Pięciu zdarzeń: iteracji zwanych Sprintami oraz następujących podczas Sprintów czterech spotkań: Planowania Sprintu (*Sprint Planning*), codziennego Daily Scrum, Przeglądu Sprintu (*Sprint Review*) oraz Retrospekcji Sprintu (w skrócie Retrospekcji, *Sprint Retrospective*).

- Trzech obiektów (*artifacts*), z każdym z których związane jest zobowiązanie (*commitment*):
  - Backlog Produktu (*Product Backlog*) i zobowiązanie Cel Produktu (*Product Goal*),
  - Backlog Sprintu (*Sprint Backlog*) i zobowiązanie Cel Sprintu (*Sprint Goal*),
  - Przyrost (*Increment*), czyli nowa wersja produktu oraz zobowiązanie Kryterium Ukończenia (*Definition of Done*).
- Innych opisanych w Scrum Guide elementów nie należących jednak do żadnej z powyższych trzech kategori.

Elementy te są spięte w całość zestawem reguł, a opierają się na fundamencie wartości Scruma oraz Manifestu Agile.

Backlog Produktu

Scrum Guide definiuje Backlog Produktu jako listę rzeczy, które są potrzebne aby ulepszyć produkt. Kolejność na tej liście to kolejność ważności i wartości, jest to zarazem kolejność realizacji owych rzeczy. Za listę tą, a więc jej zawartość i kolejność odpowiada Product Owner. *Product Backlog* jest jedynym źródłem pracy dla zespołu – nie może on pracować nad żadnymi innymi rzeczami (zleceniami, wymaganiami itp.), które nie przeszłyby przez Product Ownera.

Co ważne, kolejność na backlogu jest zmienna w czasie. Kolejność i zawartość Backlogu Produktu powinna zawsze odzwierciedlać relatywną ważność znajdujących się na nim rzeczy w danej chwili. Istota roli PO polega na takim zmienianiu Backlogu Produktu – dodawaniu, usuwaniu i zmianie kolejności rzeczy – by co Sprint praca zespołu dostarczała maksymalnej wartości.

Szczególnym elementem Backlogu Produktu jest Cel Produktu. Jest to pośredni cel, którego osiągnięcie będzie przybliżać do realizacji wizji produktu lub istotnie podniesie jego wartość. Może on być dobrym punktem wyjścia do stworzenia elementów backlogu, a więc wymyślenia jakie konkretne zmiany w produkcie najlepiej pozwolą nam ów cel osiągnąć. Pytanie „czy to pomoże osiągnąć Cel Produktu?" służy także jako przewodnik dla Product Ownera przy decydowaniu czy dana rzecz powinna znaleźć się na backlogu w danej chwili.

Cel Produktu jest zawsze jeden. Dopiero po jego osiągnięciu można określić nowy Cel Produktu. Możliwe jest też, oczywiście, porzucenie Celu Produktu jeśli osiągnięcie go nie ma już obecnie sensu lub jest niemożliwe.

Słowo „*rzeczy*" powyżej, które jest bardzo ogólne może nie być do końca jasne. Zostało ono użyte, ponieważ Scruma można używać na różnych poziomach abstrakcji. Można zatem ułożyć w listę wymagania – wstępnie przeanalizowane rzeczy do zaimplementowania - i wówczas Deweloperzy (zapewne programiści i testerzy) kolejno będą te wymagania w Sprintach implementować czyli zamieniać na kolejne wersje oprogramowania. Można jednak na Backlogu Produktu umieścić problemy biznesowe, czyli nie oczekiwaną implementację, ale oczekiwane wyniki działania produktu lub oczekiwane możliwości produktu. Wówczas Deweloperzy również będą w Sprintach tworzyć kolejne wersje oprogramowania, ale ich praca będzie polegać na rozwiązywaniu problemów biznesowych z pomocą oprogramowania. Jest to lepsze, bo wówczas Deweloperzy znając problem będą mogli lepiej dobrać do niego rozwiązanie technologiczne. Można podejrzewać, że w takiej sytuacji zespół będzie jeszcze bardziej wszechstronny kompetencyjnie. Zapewne też łatwiej będzie o dobre Cele Produktu, a następnie Cele Sprintów. Choć

pierwszy scenariusz to też Scrum, to jednak w tym drugim lepiej wykorzystujemy zarówno potencjał intelektualny zespołu – jak i potencjał Scruma.

Scrum nie definiuje formatu w jakim mają być opisane elementy Backlogu Produktu[5], stwierdza natomiast, że backlog powinien być przejrzysty – a więc widoczny i zrozumiały dla wszystkich zainteresowanych. Scrum zakłada, że to, co jest na backlogu ma sens i czyni Product Ownera odpowiedzialnym za to, by tak było w istocie. Scrum jako ramy postępowania (*framework*) nie dostarcza innych narzędzi do oceny sensowności i wartości wymagań jak i całego przedsięwzięcia poza *Product Ownerem*. Innymi słowy, sam Scrum nie określa w jaki sposób PO ustala lub skąd wie, co jest w danej chwili ważniejsze.

Więcej na temat Backlogu Produktu piszę w poświęconym mu rozdziale.

Zespół Scrum, odpowiedzialności

Zespół Scrum to samozarządzający się (*self-managing*), wszechstronny (ang. *cross-functional*) zespół, który dostarcza wartość poprzez częste dostarczanie Przyrostów (*Increment*) produktu. Scrum nie narzuca ani nie zaleca żadnej struktury wewnętrznej tego zespołu, poza tym, że wyróżnia trzy odpowiedzialności:

- **Deweloperzy** (ang. *Developers*) – odpowiadają za wytworzenie co Sprint kolejnego działającego Przyrostu (*Increment*), mają zatem niezbędne kompetencje. Z punktu widzenia Scrum Deweloperem jest określany każdy, kto wkłada aktywnie pracę w Przyrost - nie ma tu znaczenia spe-

[5]Na określenie jednej rzeczy na backlogu stosuje się skrót PBI od słów *Product Backlog Item*. W potocznym użyciu w zespołach spotykam się ze spolszczeniem tego na „ajtemy".

cjalizacja (np. tester, front-end itp.) czy konkretne kompetencje.

- **Product Owner** – odpowiada za maksymalizację wartości produktu poprzez dbanie aby Zespół Scrum zajmował się wyłącznie rzeczami wartościowymi – i to wartościowymi na daną chwilę wedlego jego najlepszej wiedzy. Procesowo PO realizuje to poprzez porządkowanie Backlogu Produktu i decydowanie o jego zawartości, a przez to decydowanie o kolejności realizacji czyli ostatecznie kształcie produktu. Praktycznie PO robi wiele innych rzeczy, żeby mieć na czym opierać swoje decyzje oraz umieć sprawdzić czy były one trafne. *Szerzej o roli Product Ownera piszę w poświęconym mu rozdziale.*
- **Scrum Master** – odpowiada za proces, ma dbać o to, by Scrum był prowadzony zgodnie ze Scrum Guide zarówno w samym zespole jak i w całej organizacji. Scrum Master odpowiada za efektywność zespołu Scrum - a więc produktywność i zaangażowanie. Dba o zespół pomagając mu w rozwiązywaniu problemów i usuwaniu przeszkód (w Scrum nazywanych *impediments*) oraz wspomagając jego samozarządzanie. Nie jest jednak kierownikiem zespołu ani liderem technicznym, nie może bezpośrednio polecać członkom zespołu wykonania takiej czy innej pracy, karać ich czy nagradzać lub określać jaką rolę będą pełnić w zespole.

  Sposób działania Scrum Mastera jest w metodzie definiowany jako lider, który służy zespołowi i organizacji. Chodzi tu o koncepcję przywództwa znaną jako *servant le-*

adership[6]. Często stosowaną analogią jest porównywanie Scrum Mastera do trenera drużyny sportowej - pomaga on graczom, formuje z nich zespół, ale nie wybiega na boisko by brać udział w grze. Więcej o tej roli piszę w osobnym rozdziale.

Ważną cechą Scrum, dotyczącą przede wszystkim Developerów, jest **wspólnota pracy**. Polega ona na tym, że Developerzy mający różne umiejętności, specjalności (kompetencje) ściśle współpracują ze sobą każdego dnia pracując nad Przyrostem. Ściśle współpracują, to znaczy nie „przerzucają" kolejno pracy między swoimi „mikrosilosami" kompetencyjnymi tracąc nią zainteresowanie z chwilą kiedy zajmuje się nią kolega o innej specjalności. Przeciwnie, bez względu na specjalność wspólnie pracują nad ukończeniem kolejnych wymagań (zmian), tworząc w ten sposób kolejne Przyrosty. Scrum w ogóle ma sens tam, gdzie natura produktu wymaga takiej ścisłej współpracy. Jeśli praca każdej ze specjalności dałaby się wykonywać w oderwaniu i bez szerokiej komunikacji z innymi specjalistami to wówczas budowanie wszechstronnych, ściśle współpracujących zespołów byłoby zbędne.

Zespół Scrum powinien liczyć nie więcej niż 10 osób, w tym dokładnie jedną osobę mającą odpowiedzialność Scrum Master i jedną osobę mającą odpowiedzialność Product Owner. Innymi słowy, powinno być zawsze wiadomo kto w Zespole Scrum jest Scrum Masterem, a kto Product Ownerem.

Scrum Guide nie określa w jaki sposób wyłania się Product Ownera i Scrum Mastera. W związku z tym nic nie stoi na przeszkodzie, aby Scrum Master był wyłoniony z zespołu. W

[6]Przyjęło się tłumaczenie tego terminu na polski jako „przywództwo służebne" , jednak nie oddaje ono moim zdaniem dobrze sensu tego termin. Dużo lepszym nazwaniem tego modelu byłoby „wspierające przywództwo".

przypadku Product Ownera byłoby to niepraktyczne, bo powinien on kierować się wartością produktu i zwykle mieć zupełnie inne kompetencje niż pozostali członkowie zespołu Scrum. Dodatkowo, Product Owner to jedyna odpowiedzialność w Scrum, z którą wiąże się konkretna władza - władza decydowania o tym, jaki będzie kierunek rozwoju produktu, co się w nim znajdzie. Oczywiście można sobie wyobrazić scenariusz, w którym również i Product Owner będzie wybrany przez zespół, na przykład niewielki startup. W większości organizacji wskazane jest jednak, żeby Product Ownerem był przedstawiciel biznesu o odpowiedniej randze i zakresie władzy.

Scrum Guide nie zakazuje wprost łączenia odpowiedzialności Scrum Mastera i Product Ownera przez jedną osobę, jednak powszechnie przyjmuje się, że nie powinno to mieć miejsca. Wynika to z tego, że pomiędzy tymi odpowiedzialnościami wstępuje naturalne napięcie – Scrum Master dba o proces (a więc np. o to, by produkt był zawsze takiej jakości jaką wyznacza *Definition of Done*) zaś Product Owner chce zrealizować jak najwięcej wartościowej funkcjonalności w każdym sprincie. Jest korzystne aby napięcie to było uzewnętrznione w dwóch osobach reprezentujących te nieco odmienne punkty widzenia i przez to mogło prowadzić do otwartych, konstruktywnych dyskusji.

Scrum Guide nie zakazuje również łączenia odpowiedzialności Scrum Mastera i Dewelopera. Początkowo wydawało się, że nie jest to dobre rozwiązanie. Można mieć obawy, że osoba mająca dwie odpowiedzialności może być mniej efektywna w którejś z nich - lub w obu. Drugi problem z takim połączeniem wynika z tego, że inni – zwłaszcza inni członkowie zespołu – mogą nie mieć jasności kiedy dana osoba wypowiada się jako Deweloper, a kiedy jako Scrum Master. Jest to na przykład widoczne w przypadku konfliktu między Deweloperami – Scrum Masterowi,

który jest jednym z nich trudniej może być mediatorem. Z czasem okazało się jednak, że w wielu przypadkach takie połączenie się sprawdza, także dlatego, że osadza Scrum Mastera w zespole i gwarantuje odpowiedni poziom wiedzy merytorycznej. Dziś więc trudno jest odradzać takie rozwiązanie. W praktyce wiele zależy tu od charakteru, umiejętności i zaangażowania konkretnego człowieka mającego taką podwójną odpowiedzialność oraz zespołu, w którym pracuje.

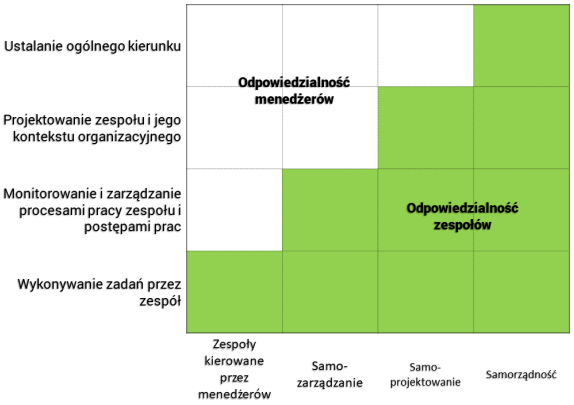
Podobnie, Scrum Guide nie zakazuje także łączenia odpowiedzialności Product Ownera i Dewelopera. W praktyce zdarza się to niezmiernie rzadko – znam na tyle mało takich przypadków, że ciężko mi ocenić czy jest to dobry pomysł. Skoro jednak zdarza się tak rzadko, to chyba nie jest to naturalne, sensowne połączenie. Szukając przyczyn warto zwrócić uwagę, że praca Product Ownera wymaga dużo czasu na działania nie związane z zespołem i bieżacym Sprintem – pracę z interesariuszami czy analizę rynku, tak, by modyfikując Backlog Produktu realizować właściwie, aktualne cele biznesowe. Łączenie tego z pracą nad Przyrostem w produkcie mogłoby być trudne. Dodatkowe zagrożenie w przypadku takiego połączenia jest takie, że wówczas PO może zacząć dyktować deweloperom nie tylko co ma być na backlogu, ale również jak ma być to zaimplementowane.

Scrum jako metoda opisana w Scrum Guide skupia się z założenia na jednym produkcie rozwijanym przez jeden Zespół Scrum. Gdyby jednak zespołów miało być kilka – co zasadniczo jest problemem skalowania – to byłoby potrzeba więcej Scrum Masterów, ale Product Owner mógłby pozostać jeden. Product Owner jest związany bardziej z produktem niż konkretnym zespołem. W przypadku jednoczesnej pracy kilku zespołów nad jednym produktem mogą one mieć wspólny Backlog Produktu, a co za tym idzie wspólnego Product Ownera.

Warto podkreślić tutaj pewną ewolucję jaką przeszedł Scrum – kiedyś mówiło się o „rolach", a w 2020 ze Scrum Guide zniknęło to określenie. Miało to na celu podkreślenie, że Product Owner, Scrum Master i Deweloperzy to nie są osobne stanowiska pracy, zawody nawet, ale członkowie jednego zaangażowanego, samozarządzającego się zespołu profesjonalistów. Ma to na celu przeciwdziałanie nie służącym budowaniu efektywnych zespołów podziałom.

Kolejną zmianą wprowadzoną w 2020 roku jest odejście od mówienia o zespole samoorganizującym się (*self-organizing*) na rzecz zespołu samozarządzającego się. (*self-managing*). Zmiana ta jest widziana przez autorów Scrum Guide jako zmiana głównie językowa, użycie terminu, który lepiej opisuje pożądany stan zespołu, a nie zmiana w samym Scrumie. I rzeczywiście, już wcześniej można było spotkać się z omówieniami, które traktowały *self-organizing* i *self-managing* jako określenie tego samego stanu zespołu[7]. Przy tym określenie „samozarządzający się" lepiej wpisuje się w istniejącą literaturę o zarządzaniu oraz modele takie jak model Richarda Hackmana.

[7]Np. Mike Cohn pisze o tym tak: „In Hackman's authority hierarchy, self-organizing agile teams would be classified as self-managing teams. Agile teams manage their own work (which team member should do this task?) and how they go about that work (should we start with Kanban or Scrum? How long should our sprints be?). Agile teams are not, however, self-designing or self-governing. In other words, a self-organizing agile team has authority over its work and the process it uses, but not over who is on the team or the team's purpose." Więcej: https://bit.ly/38CcZCU



Model Hackmana.

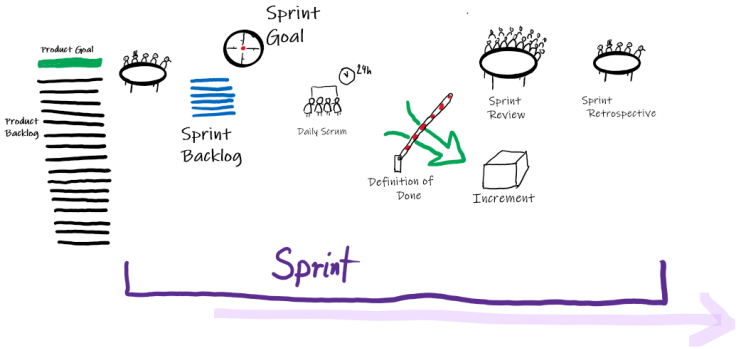
Jak widać z tego modelu samozarządzający się (*self-managing*) zespół to najniższy stopnień autonomii zespołu. Najwyższy to zespół samorządny (czyli *self-governing*), a więc taki który decyduje również o sensie i celach swojego istnienia.

Warto tu pamiętać, że Scrum jest minimalistyczny nie tylko jeśli idzie o objętość Scrum Guide czy liczbę praktyk, ale także o określenie warunków koniecznych do jego działania. Innymi słowy Scrum Guide specyfikuje minima, a nie stan idealny. W tym kontekście do działania Scruma wystarczy zespół, który jest samozarządzający się choć oczywiście Scrum będzie działał dobrze również w zespole samorządnym (*self-governing*).

Proces

Patrząc na Scrum jako na proces jest to metoda opisująca przetwarzanie rzeczy zebranych na **Backlogu Produktu** (*Product Backlog*) na kolejne wersje gotowego do użycia produktu (*potentially usable product increment*) zwane Przyrostami. Rzeczy

(jednostki pracy) z Backlogu Produktu przetwarzane są w partiach (*batches*), które są realizowane podczas krótkich iteracji zwanych **Sprintami**. Każdą iteracje (Sprint) otwiera Planowanie Sprintu, podczas którego ustala się cel oraz jakie konkretnie rzeczy znajdą się w partii przetwarzanej w danym Sprincie. Każdego dnia następuje weryfikacja postępu i planu prac. Na koniec każdej iteracji następuje weryfikacja czy stworzony Przyrost oraz stan Backlogu Produktu odpowiadają potrzebom, jeśli nie wówczas zmieniany jest Backlog Produktu. Podobnie, na kończącej iteracje Retrospekcji weryfikowany jest również sam proces i praktyki zastosowane wewnątrz niego, a zidentyfikowane usprawnienia są wprowadzane w życie natychmiast, czyli od kolejnego Sprintu.



Schemat procesu Scrum

Wszystkie zdarzenia opisywane w Scrumie odbywają się wewnątrz Sprintu. Każdy Sprint rozpoczyna się od spotkania

zwanego **Planowaniem Sprintu** (*Sprint Planning*) podczas którego Zespół Scrum tworzy prognozę co zaimplementuje w ciągu tego Sprintu. Następnie podczas trwania Sprintu Deweloperzy synchronizują się podczas 15-o minutowych spotkań zwanych **Daily Scrum**, w których nie bierze aktywnego udziału nikt poza nimi. Ponadto, w miarę potrzeb odbywają się zazwyczaj podczas Sprintu spotkania z udziałem Product Ownera poświęcone pielęgnacji backlogu czyli wspólnego pogłębiania zrozumienia Backlogu Produktu, a więc realizowany jest proces *backlog refinement*.

Co najmniej raz w Sprincie, na jego zakończenie zespół wytwarza nową wersję produktu zwaną Przyrostem (*Increment*) - kolejną, gotową (*done*) wersję rozwijanego systemu/aplikacji. W produkcie tym znajdują się tylko te implementacje elementów backlogu, które spełniają techniczne **Kryterium Ukończenia** (*Definition of Done - DoD*). Na zakończenie sprintu Zespół Scrum współpracuje z interesariuszami podczas **Przeglądu Sprintu** (*Sprint Review*), którego celem jest sprawdzenie stanu produktu i podjęcie decyzji co do dalszych jego losów. Bezpośrednio po przeglądzie Zespół Scrum odbywa **Retrospekcję Sprintu** (*Sprint Retrospective*) - spotkanie, podczas którego zespół analizuje swój sposób pracy, zastanawia się w jaki sposób mógłby go poprawić i planuje jakie konkretne usprawnienia wprowadzi do swej pracy w następnym Sprincie.

Cykl ten powtarzany jest w kolejnych Sprintach aż do przerwania prac nad produktem z tego czy innego powodu.

Poniżej omawiam w szczegółach każdy z tych elementów Scruma.

Sprint

Wszystkie zdarzenia w Scrumie odbywają się wewnątrz iteracji o stałej długości zwanych **Sprintami**. Początek Planowania Sprintu jest początkiem Sprintu. Koniec Retrospekcji jest końcem Sprintu. Sprint może mieć długość do 1 miesiąca i jest wskazane, aby w obrębie jednego przedsięwzięcia długość Sprintu pozostawała stała. Należy bardzo dokładnie i stanowczo przestrzegać ustalonej długości Sprintu. Sprintu nie wolno skracać ani wydłużać choćby o parę godzin.

Celem takiego podejścia jest umożliwienie działania empiryzmu poprzez wyznaczenie stałego, obiektywnego interwału, po którym produkt jest poddawany sprawdzeniu (*inspect*) i może nastąpić dostosowanie (*adapt*). Co Sprint interesariusze poznają rzeczywisty stan produktu. Wyznaczają go rzeczy, które zespół całkowicie ukończył podczas ostatniego Sprintu (które znajdują się w Przyroście). Ponieważ długość Sprintu jest sztywna i stała jest zarazem znana rzeczywista prędkość (*velocity*) zespołu - wydłużanie sprintu np. o 6 godzin by coś dokończyć prowadziłoby do zafałszowania tego wskaźnika rzeczywistych możliwości zespołu.

Ponadto celem Sprintu o niezmiennej długości jest wytworzenie stałego rytmu pracy. Rytm ten będzie istotny nie tylko dla zespołu, ale i dla całej organizacji. Wszyscy będą wiedzieć, że co najmniej raz na dwa albo trzy tygodnie pojawia się nowa wersja produktu oraz rozpoczyna praca nad następną.

Długość Sprintu ustala się podczas rozpoczynania pracy nad danym przedsięwzięciem, nie powinna być ona potem pochopnie zmieniana. Jeśli z jakichś powodów zmiana jest konieczna nie powinna ona dotyczyć jednego Sprintu, lecz po zmianie nowa długość Sprintu winna być następnie stosowana przez dłuższy czas.

Planowanie Sprintu

Każdy Sprint rozpoczyna się od **Planowania Sprintu** (*Sprint Planning*). Jest to spotkanie, w którym bierze udział cały Zespół Scrum. Jego celem jest zastanowienie się co można zrobić w nadchodzącym Sprincie, a także jak to zrobić. Produktem planowania są Backlog Sprintu (*Sprint Backlog*) oraz Cel Sprintu (*Sprint Goal*).

Maksymalny czas trwania planowania to 8h dla Sprintu miesięcznego, jest zalecane aby planowanie dla krótszych sprintów było proporcjonalnie krótsze niemniej pozostawione jest to do wyboru zespołów. W każdym jednak przypadku czas trwania tych spotkań musi być określony z góry (*timebox*) i stały w danym przedsięwzięciu. Dobry wskaźnik to max. 2h na tydzień Sprintu. Przy tym, jak przy wszystkich spotkaniach Scrumowych, przyjęty *timebox* należy traktować jako maksymalną długość spotkania. Innymi słowy, jeśli temat danego spotkania zostanie wyczerpany należy je zakończyć.

Podczas planowania roważane są trzy tematy:

1. **Dlaczego ten Sprint jest wartościowy** (Why?) - Product Owner proponuje co może być wartościowym Celem Sprintu (odnosząc się do Celu Produktu), następnie cały zespół współpracuje nad ustaleniem Celu Sprintu.
2. **Co możemy zrobić w tym Sprincie?** (What?) - poprzez dyskusję z Produkt Ownerem Deweloperzy wybierają z Backlogu Produktu rzeczy, które zrobią w tym Sprincie. Efektem tej dyskusji jest Sprint Backlog.
3. **Jak można to zrobić?** (How?) - to dyskusja Deweloperów jak rzeczy „wzięte do Sprintu" zrobić. Co istotne, nikt nie może narzucać Deweloperom w jaki sposób zamierzają pracować w Sprincie.

Scrum Guide nie narzuca żadnej kolejności dyskusji na te tematy. Może być tak, że dyskusja ta następuje po kolei – a może być tak, że się przeplata. Scrum Guide kładzie więc tu nacisk na *współpracę* pomiędzy członkami zespołu Scrum, przede wszystkim Product Ownerem i Deweloperami. Najważniejsze efekty Planowania – Cel Sprintu i rzeczy wzięte do Backlog Sprintu z Backlogu Produktu – powinny być wynikiem współpracy, a nie narzucania czegoś sobie nawzajem.

Kiedyś Scrum Guide proponował nawet konkretny przebieg tego spotkania, z podziałem na dwie fazy itp. Praktyka pokazała jednak, że ważne nie jest to jak spotkanie przebiega ale co się na nim omawia i co z niego wynika. Scrum Guide skupił się zatem na opisaniu efektów a nie przebiegu. Nie należy przez to rozumieć, że spotkanie to nie powinno mieć zorganizowanego przebiegu, jakiegoś planu czy formatu. Pozostawiono to jednak zespołom bo praktyka pokazała, że dobre rezultaty można osiągnąć bardzo różnym przebiegiem tych spotkań.

Podobnie, kiedyś podkreślano, że Cel Sprintu może być określony dopiero na samym końcu planowania, bo dopiero wtedy wiadomo co konkretnie weszło do Sprintu. Takie podejście wydawało się wielu osobom słusznie niepraktyczne a nawet nielogiczne, bo logicznie to cel powinen być punktem wyjścia do planu, a nie na odwrót. Obecnie więc Scrum Guide mówi jedynie, że Cel Sprintu musi być określony przed końcem Planowania.

Scrum Guide nie narzuca także konkretnej formy Sprint Backlogu, poza tym, że powinien on umożliwiać śledzenie postępów pracy. Niegdyś częstą praktyką było tworzenie dla każdej z rzeczy na Sprint Backlogu zadań (*tasks*), a więc opisów czynności jakie Deweloperzy zamierzają wykonać aby dane wymaganie zaimplementować. Były one nawet szacowane w godzinach, co służyło następnie do monitorowania postępu prac z pomocą

tzw. wykresów spalania (*burndown charts*). Obecnie takie podejście stosuje się coraz rzadziej, bo – zwłaszcza w połączeniu z elektronicznymi „narzędziami do Scruma" – prowadziło często do skupienia się Deweloperów na „swoich taskach" ze szkodą dla pracy wspólnej a także do sztywnego trzymania się planu wewnątrz Sprintu.

Obecnie coraz częściej zespoły Scrum stosują tablice kanbanowe i śledzenie na nich jedynie stanu zadań bez oszacowań lub stanu całych rzeczy (PBI) oraz stosowanie ograniczenia liczby rzeczy, które mogą być jednocześnie „otwarte" (WIP). Praktyka ta jest na tyle popularna i zalecana, że Scrum.org opublikował specjalny podręcznik do stosowania praktyk Kanban w Scrum, oferowane jest także szkolenie Professional Scrum with Kanban (PSK) jak i certyfikat o tej samej nazwie.

Podkreślić też należy, że Scrum Guide nie wymaga się, aby w trakcie planowania powstał na cały Sprint plan działań o równym poziomie szczegółowości. Plan na najbliższe dni powinien być w miarę szczegółowy – tak, by zaraz po zakończeniu planowania można było przystąpić do pracy nad Przyrostem. Plan na dalsze dni Sprintu może być ogólny. Zostanie on doprecyzowany w miarę postępów prac.

Daily Scrum

Jest to codzienne, krótkie spotkanie mające na celu synchronizację Deweloperów i dla nich przede wszystkiem przeznaczone. Celem jest uzyskanie stanu, w którym wszyscy Deweloperzy wiedzą jak wygląda sytuacja w kontekście postępu prac w Sprincie oraz zaplanowanie dalszych działań na kolejny dzień (kolejne 24h).

Spotkanie to ma następujące cechy:

- trwa maksymalnie 15 minut (limit ten nie zależy ani od długości Sprintu ani liczebności zespołu),
- odbywane jest zawsze w tym samym miejscu i o tej samej godzinie by ograniczyć złożoność[8],
- jest to spotkanie Deweloperów - jeśli osoby mające odpowiedzialność Scrum Mastera czy Product Ownera są jednocześnie Deweloperami to biorą w tym spotkaniu udział jako Deweloperzy.

Celem tego spotkania jest wykonanie sprawdzenia postępów pracy do momentu, w któym się ono odbywa w kontekście Celu Sprintu oraz zaplanowanie dalszej pracy nad jego osiągnięciem. Innymi słowy jest to codzienna „dawka empiryzmu" dla Deweloperów, codzienne *inspect&adapt* - podczas tego spotkania Deweloperzy dokonują sprawdzenia (*inspect*) jak wygląda postęp prac nad osiągnięciem celu Sprintu oraz dostosowania (*adapt*) poprzez odpowiednie, wspólne zaplanowanie kolejnego dnia. Konkretny sposób w jaki Deweloperzy osiągną ten cel podczas Daily Scrum nie jest określony w Scrum Guide. Przeciwnie, pozostawiono to zespołom podkreślając, że sam format spotkania jest mniej ważny od tego czy służy ono swojemu celowi.

Przez długie lata w Scrum Guide, a wcześniej na szkoleniach i w literaturze przekazywano format Daily Scrum oparty o odpowiedź na trzy pytania:

- co wczoraj zrobiłem/am, co pomogło w osiągnięciu celu sprintu?
- co zamierzam zrobić dziś by pomóc w osiągnięciu celu sprintu?

[8] W Guide „reduce complexity". Chodzi o to, że dzięki temu uczestnicy tego spotkania nie muszą się zastanawiać i sprawdzać gdzie i o której się ono danego dnia odbywa, co redukuje złożoność i wysiłek z ich strony potrzebny by w nim uczestniczyć.

- czy dostrzegam jakieś problemy, bariery, które przeszkadzają mi lub całemu Zespołowi Deweloperskiemu w osiągnięciu celu sprintu?

Częstą praktyka jest, że członkowie zespołu po koleji odpowiadają na wszystkie trzy pytania (a więc wpierw jedna osoba mówi co zrobiła, co planuje i jakie ma problemy, a potem następna). Wydaje się, że lepszym sposobem jest aby najpierw wszyscy odpowiedzieli na pierwsze dwa pytania, które dotyczą przeszłości, a dopiero potem na ostatnie, które tyczy się przyszłości. Mogą wtedy w swoich planach uwzględnić otrzymaną właśnie informację na temat tego co udało się osiągnąć innym, co jest zgodne z założeniem Daily Scrum jako codziennej empirycznej inspekcji i adaptacji pracy w sprincie. Takie postępowanie zalecam często początkującym zespołom.

W toku dalszego rozwoju zespołu kwestia tego jak najlepiej wykorzystać Daily Scrum jest dobrym tematem na Retrospekcje. Jest wskazane aby odejść od sztywnego formatu trzech pytań nawet, jeśli był on przyjęty w zespole wcześniej. Zastanowienie się przez Zespół Scrum nad tym jak najlepiej wykorzystać te 15 minut w ciągu dnia jest dobrym krokiem w rozwoju samozarządzania w zespole.

Warto podkreślić, że to Deweloperzy są odpowiedzialni za przeprowadzenie Daily Scrum. Nie jest to spotkanie dla Scrum Mastera. Scrum Master ma za zadanie zadbać o to, by spotkanie to się odbywało i służyło swojemu celowi – oraz zmieściło się w 15 minutach. Nie powinien – poza być może początkowymi Sprintami nowego zespołu, dla którego Scrum jest zupełną nowością - „prowadzić" tego spotkania, nie musi być nawet na nim obecnym. Całkowitym nieporozumieniem jest jeśli Daily Scrum staje się spotkaniem „statusowym", podczas którego Deweloperzy składają Scrum Masterowi lub - co jeszcze gorzej - komuś

innemu raporty co do stanu przydzielonych im zadań. Takie spotkanie nie ma nic wspólnego z sensem jaki ma Daily Scrum.

Powszechnie przyjętą praktyką jest odbywanie tego spotkania na stojąco (nazywane bywa „daily standup"). Takiego wymogu jednak nie było nigdy w Scrum Guide, po prostu powszechnie się to przyjęło bo w postawie stojącej łatwiej jest o utrzymanie koncentracji uwagi na tym, co się dzieje - oraz zwięzłości wypowiedzi.

Przyjmuje się zarazem, że przynajmniej raz dziennie, właśnie w okolicach tego spotkania zespół powinien uaktualnić sprint backlog i narzędzie, z pomocą którego monitoruje postępy swojej pracy. A więc jeśli używane są zadania i wykres burndown, to powinny one być aktualne przed kolejnym Daily Scrum, jeśli używana jest tablica kanban to ona powinna być zaktulizowaną. W praktyce nie nastręcza to problemu, bo większość zespołów uaktualnia tego rodzaju narzędzia na bieżąco, w miarę postępów prac.

I na koniec ważna uwaga (która od pewnego czasu jest nawet w Scrum Guide): Daily Scrum zdecydowanie **nie jest** jedynym momentem w ciągu dnia kiedy Deweloperzy czy w ogóle Zespół Scrum rozmawia ze sobą. Zespół Scrum może zatem i powinien komunikować się dużo częściej i szerzej każdego dnia Sprintu. Może także zmienić swoje plany co do dalszej pracy nad osiągnięciem Celu Sprintu w dowolnym momencie, niekoniecznie na Daily Scrum. Podobnie jak w innych miejscach Scrum określa tu pewne absolutnie minimum synchronizacji i współpracy, które musi nastąpić aby praca w ogóle mogła się posuwać do przodu w zorganizowany sposób.

Pielęgnacja Backlogu Produktu

Aby zachować płynność pracy jest wskazane aby Backlog Produktu był wypielęgnowany[9], a więc aby rzeczy, które w najbliższych 2-3 sprintach będą przedmiotem pracy były przedyskutowane zawczasu w Zespole Scrumowym, a także oszacowane i zdekomponowane odpowiednio do swojej pozycji na backlogu. Rzeczy, które są na początku backlogu powinny być mniejsze, dokładniejsze i lepiej zrozumiane niż te poniżej.

Rzeczy, które są tak zdekomponowane, że mogą być przez zespół całkowicie zrealizowane w czasie jednego Sprintu mogą być brane pod uwagę do wzięcia do Sprintu podczas Planowania Sprintu. Rzeczy większe wymagają wcześniejszej dekompozycji, czyli zamiany na mniejsze kawałki funkcjonalności (wartości). Każda taka dekompozycja jest jednocześnie krokiem analizy, bo wymaga najczęściej lepszego zrozumienia tematu. Wiąże się także zazwyczaj z oszacowaniem powstałych elementów backlogu. Całość tych działań nazywamy właśnie pielęgnacją Backlogu Produktu (*Product Backlog refinement*).

Obecnie Scrum Guide nie określa już zupełnie jak i w jakim czasie działalność ta ma się odbywać. Poprzednio sugerowano, aby przeznaczać na pielęgnację do 10% czasu trwania Sprintu. Tak więc przykładowo dla dwutygodniowego Sprintu byłby to maksymalnie jeden cały dzień pracy. W Scrum nigdy nie określano na poziomie metody czy czas ten ma zostać spożytkowany jako jedno długie spotkanie czy kilka mniejszych - ani czy w ogóle mają to być spotkania - pozostawiając to decyzji zespołów. Dlatego właśnie pielęgnacja (*refinement*) nie jest jednym ze zdarzeń Scruma, gdyż jest procesem i niekoniecznie musi on być realizowany poprzez spotkania.

[9] Więcej na temat pielęgnacji backlogu w rozdziale o Backlogu Produktu.

W praktyce większość zespołów Scrum realizuje ten proces poprzez kilka specjalnych spoktań z udziałem Deweloperów i Product Ownera (zwanych przez zespoły w Polsce potocznie *grumingami*). Podczas tych spotkań wspólnie pracują oni nie nad tematami bieżącego Sprintu, lecz nad Backlogiem Produktu. Jest to rozsądniejsze rozwiązanie niż gdyby miało to być jedno spotkanie, trwające cały dzień.

Jest dobrą praktyką planowanie takich spotkań z góry na cały Sprint, licząc się z tym, że nie wszystkie mogą zostać wykorzystane. Można przyjąć ustalenie, że jeśli do 24h przed planowaną pielęgnacją backlogu ani Deweloperzy ani Product Owner nie zgłoszą jego potrzeby, to wówczas dane spotkanie się nie odbywa. Jest to lepsze rozwiązanie niż tzw. „refinement ad hoc", z którym można się czasem spotkać, a który polega na tym, że spotkanie poświęcone pielęgnacji backlogu zwołuje się na życzenie Product Ownera lub Deweloperów kiedy jest taka potrzeba. Wadą tego rozwiązania jest potencjalna dezorganizacja pracy zespołu w bieżącym Sprincie przez nagle pojawiające się spotkanie.

Powtórzę tu jednak, że proces pielęgnacji może być realizowany w ogóle w inny sposób niż poprzez spotkania. To, co opisałem w powyższych akapitach to jedynie podsumowanie dość powszechnego sposobu, w jaki jest to realizowane wielu zespołach, które miałem okazję widzieć.

Produkt i Kryterium Ukończenia

Na zakończenie każdego Sprintu zespół dostarcza nową wersję rozwijanego produktu, czasem zwaną „Przyrostem" albo „inkrementem" (od *potentially usable product increment*). Produkt ów musi być tak dobry (takiej jakości), by mógł być zastosowany produkcyjnie - by użytkownicy mogli z niego korzystać (*„usa-*

ble"). Innymi słowy produkt nie może wymagać wykonywania nad nim żadnej dodatkowej pracy poza ewentualną instalacją lub wystawieniem na środowisko produkcyjne. W praktyce spotyka się wiele zespołów, które co Sprint rzeczywiście wydają nową wersję produktu użytkownikom (poprzez wydanie na serwery produkcyjne lub w inny sposób). Są także zespoły, które robią to jeszcze częściej, a więc gdy tylko jakaś funkcjonalność jest gotowa, co ma miejsce wiele razy w trakcie Sprintu.

Nacisk na to, by co Sprint był dostępny w pełni zintegrowany, działający produkt, który może być natychmiast wdrożony i używany ma następujące przyczyny:

- tylko produkt, który jest zdatny do użytku[10] przedstawia jakąkolwiek wartość dla użytkownika/klienta, ponieważ Scrum kładzie nacisk na dostarczanie wartości konieczne jest aby co Sprint dostarczany był zdatny do użytku produkt,
- jedynie informacja o tym jakie funkcje są już w produkcie zaimplementowane i dostępne jest informacją pewną, wszelkie inne informacje o postępie prac to z punktu widzenia interesariuszy jedynie prognozy - aby więc na poziomie produktu mógł działać empiryzm konieczne jest przedstawianie im podczas przeglądu tylko tego, co jest naprawdę ukończone, a więc gotowe do użytku,
- konieczność tworzenia w pełni zintegrowanego, działającego produktu conajmniej co Sprint wymusza podnoszenie poziomu praktyk technicznych zespołu zarazem wykluczając możliwość pojawienia się „okresu stabilizacji", kiedy to

[10] W sensie kompletności technologicznej a nie funkcjonalnej. Innymi słowy zdatny do użytku, bo dobrze zbudowany i przetestowany, niekoniecznie zaś mający już wszystkie funkcje jakie może są planowane.

okazuje się że integracja zbudowanych na przestrzeni długiego czasu fragmentów nie daje działającego produktu,
- podnosi to jakość produktu, ponieważ kod napisany na początku przedsięwzięcia jest cały czas testowany i sprawdzany w każdym kolejnym Sprincie - jako że zaczynamy zazwyczaj od elementów (funkcji) najważniejszych dla produktu (a więc zwykle funkcji stanowiących jego istotę) dzięki temu podejściu są one najlepiej sprawdzone.

Aby mieć pewność, że produkt co Sprint jest w istocie zdatny do użytku definiuje się listę kryteriów, które musi on spełniać. Lista ta jest znana jako **Kryterium Ukończenia** (*Definition of Done*, w skrócie *DoD*) czasem nazywana też „Definicją Ukończenia". Określa ona co to znaczy, że produkt jest technicznie ukończony. Powstaje ona w wyniku współpracy Product Ownera i Deweloperów. Znajdują się tam zarówno kryteria wydajnościowe czy inne wymagania niefunkcjonalne wynikające z potrzeb (te zwykle dostarcza Product Owner) jak i kryteria pozwalające utrzymać jakość strukturalną produktu (te zwykle określają Deweloperzy). Zazwyczaj jest to lista sprawdzeń (np. jakości kodu) i testów (np. regresji, wydajności itp.).

Jeśli w danej organizacji obowiązują jakieś standardy techniczne mające wpływ na to co to oznacza, że produkt jest ukończony, to wówczas są one minimalnym - podstawowym - DoD, którym Deweloperzy musi się kierować. Zespół może je jednak rozszerzać dodając do swojego Kryterium Ukończenia dodatkowe, potrzebne w jego sytuacji testy i sprawdzenia.

Tylko rzeczy, których implementacja spełnia DoD mogą wejść do produktu (Przyrostu, inkrementu) na końcu Sprintu. Jeśli implementacja jakiejś funkcjonalności nie spełnia DoD jest uznawana za nie zrobioną i nie wchodzi do Przyrostu (inkrementu)

danego Sprintu. W takiej sytuacji procesowo element backlogu, który tą zmianę opisywał powraca do Backlogu Produktu (potocznie mówimy, że „zrzuca się go ze Sprintu") do decyzji Product Ownera co do jego dalszych losów.

Warto podkreślić, że Kryterium Ukończenia musi być tak określone, by miało charakter obiektywnego, binarnego testu (tak - spełnia, nie - nie spełnia). Nie ma tu miejsca na dowolność, subiektywność a już szczególnie niewłaściwe jest stosowanie tu „odbioru" przez Product Ownera czy interesariuszy, którzy wedle uznania „przyjmują" albo nie daną implementację. Powoduje to rozmycie tego co właściwie jest zrobione, a co nie jak również brak pewności co do tego czy ten sam poziom jakości był utrzymany w całym produkcie. Taka niepewność utrudnia również planowanie pracy - nie wiedząc co to znaczy, że praca będzie uznana za ukończoną trudno planować jej wykonywanie.

Kryterium Ukończenia (DoD) może ewoluować podczas pracy nad danym systemem/produktem, przy czym wraz ze wzrostem oczekiwań użytkowników/klientów jak i możliwości zespołu naturalne jest raczej podnoszenie wyrażonych tak wymagań jakościowych niż ich obniżanie.

Przegląd Sprintu

Gdy Przyrost produktu (*increment*) jest już gotowy odbywa się Przegląd Sprintu (*Sprint Review*). Jest to spotkanie, w którym uczestniczy Zespół Scrum oraz wszyscy zainteresowani interesariusze i użytkownicy. Jego maksymalny czas trwania to 4 godziny dla miesięcznego sprintu, przy sprintach krótszych zazwyczaj jest ono również proporcjonalnie krótsze (dobry wskaźnik to 1h na tydzień Sprintu).

Celem tego spotkania jest empiryczne sprawdzenie stanu przedsięwzięcia i podjęcie decyzji co dalej. Sprawdzenie stanu

obejmuje zarówno informację o tym, co już w produkcie jest (a więc co jest w Przyroście, najlepiej poprzez pokazanie go w działaniu) jak i co jest planowane, czyli Backlog Produktu wraz z Celem Produktu. Współraca na spotkaniu oznacza dyskusję głównie nad kształtem Backlogu Produktu i Celu Produktu, tylko bowiem poprzez następne Sprinty można coś w produkcie zmienić.

Istotą Przeglądu Sprintu jest zatem następująca na kanwie zapoznania się ze stanem produktu (a więc najczęściej pokazem Przyrostu, *increment*-u oraz aktualnego stanu Backlogu Produktu) dyskusja co do dalszych losów przedsięwzięcia. Jej celem jest zastanowienie się jakie dalsze postępowanie dostarczy największej wartości. Najczęściej efektem jest dodanie do Backlogu Produktu nowych pomysłów i rzeczy do dodania w produkcie oraz zmiana kolejności elementów backlogu. Mogą jednak być podjęte inne decyzje, na przykład takie jak o wydaniu produktu (jeśli nie dzieje się to automatycznie dla każdej nowej wersji - wówczas wydanie może następować jeszcze przed Przeglądem Sprintu), o zwiększeniu lub zmniejszeniu zespołu albo liczby zespołów czy wreszcie o przerwaniu przedsięwzięcia. W dyskusji tej bierze się pod uwagę nie tylko stan produktu i projektu (Backlog Produktu), ale także posiadane informacje o czynnikach zewnętrznych wpływających na produkt (np. reakcja klientów/użytkowników na wprowadzone zmiany, zmiana budżetu itp.).

Także i tutaj widzimy empiryzm w działaniu - interesariusze dokonują przeglądu (*inspect*) produktu i całego przedsięwzięcia, by następnie dokonać dostosowania (*adapt*) dalszych prac nad nim. Jest to „dawka empiryzmu" dla interesariuszy nad produktem i przedsięwzięciem. Jest to bardzo istotne, bo należy pamiętać, że sensem istnienia Zespołu Scrum jest do-

starczanie poprzez wartościowe Przyrosty więcej wartości niż wynoszą koszty istnienia tego zespołu. Regularne empiryczne inspekcje stanu przedsięwzięca przez interesariuszy pomagają Product Ownerowi w zapewnieniu, że tak jest w istocie.

Scrum Guide oczywiście nie opisuje jak miałoby to spotkanie przebiegać. Jednak dotychczasowa praktyka pozwala mi na sformułowanie kilku wskazówek.

Najlepiej jest gdy gospodarzem tego spotkana jest Product Owner. Wówczas to PO zaprasza interesariuszy oraz wyjaśnia, które rzeczy z Backlogu Produktu zostały w Sprincie zrobione a które nie. Deweloperzy pokazują aktualną wersję produktu (Przyrost, *increment*) i odpowiadają na pytania na jej temat, ale to PO pokazuje i objaśnia aktualny stan Backlogu Produktu i - w razie potrzeby - przedstawia prognozy co do możliwych dat ukończenia funkcjonalności lub prawdopodobnej zawartości zaplanowanych na przyszłość wydań.

Chociaż Deweloperzy mogą opowiedzieć jak przebiegała praca w Sprincie - co poszło dobrze, jakie problemy napotkano i jak je rozwiązano - to niejako wbrew swej nazwie spotkanie to nie koncentruje się na analizowaniu przebiegu Sprintu, lecz skupia się na aktualnym stanie produktu (przyrost, *increment*) i przedsięwzięcia (Backlog Produktu).

Podczas Przeglądu Sprintu zespół przedstawia ukończony właśnie przyrost, często pokazując (demonstrując) jego działanie. Przegląd Sprintu nie ma jednak charakteru demo - celem ewentualnego demo (pokazu) podczas Przeglądu jest zapoznanie uczestników (przede wszystkim interesariuszy) z aktualnym stanem produktu poprzez pokazanie jakie funkcje są w nim obecnie dostępne. W wielu firmach ten fragment Przeglądu ma obecnie taką formę, że Deweloperzy „klikają" przez nową wersję produktu co interesariusze oglądają dzięki rzutnikowi (lub pokazaniu

ekranu przy sesji zdalnej). Możliwe są jednak - i stosowane - inne formy pokazania Przyrostu interesariuszom na przykład poprzez udostępnienie im przed przeglądem nowej wersji w taki sposób, który umożliwia im samodzielne „pobawienie się" nią, zapoznanie się zarówno z nowymi czy zmienionymi funkcjami jak i upewnienie się, że inne znane im funkcje nadal działają tak jak wcześniej.

Retrospekcja Sprintu

Bezpośrednio po zakończeniu Przeglądu zespół odbywa już we własnym gronie Retrospekcję Sprintu (*Sprint Retrospective*). Jest to spotkanie, w którym udział bierze cały Zespół Scrum, i podczas którego analizuje się sposób pracy zespołu podczas dobiegającego końca Sprintu. Celem jest zastanowienie się w jaki sposób pracę tą można by było usprawnić w przyszłości, w kolejnych Sprintach. Wypracowane konkretne usprawienia – zmiany w spobie działania – należy natychmiast wprowadzić. Mogą one nawet zostać dodane do Backlogu Sprintu w kolejnym Sprincie.

Retrospekcja Sprintu trwa maksymalnie 3 godziny dla miesięcznych sprintów, zazwyczaj jest odpowiednio krótsza przy krótszych sprintach (dobry wskaźnik to 45 min na tydzień sprintu).

Wymaganie, aby zespół co Sprint analizował swoją pracę i szukał usprawnień poprzez wprowadzenie retrospekcji jako obowiązkowej części Scrum ma na celu zastosowanie empiryzmu do samego zespołu i sposobu jego pracy. Co Sprint następuje sprawdzenie (*inspect*) stanu zespołu i metod pracy a następnie oparte o jego wyniki dostosowanie (*adapt*). Dzięki temu można powiedzieć, że w Scrumie coraz lepszy Zespół Scrum, jest drugim (po Przyroście, *increment*) produktem każdego Sprintu.

Scrum Guide nie określa niczego więcej na temat retrospek-

cji. W szczególności nie wypowiada się jak miałyby przebiegać. Prowadzenie retrospekcji jest praktyką (czy nawet sztuką) samą w sobie. Więcej o retrospekcji piszę w rozdziale poświęconym w całości temu tematowi.*

Niezależnie od formy podkreślić trzeba, że w Scrum retrospekcje mają prowadzić do wypracowania konkretnych usprawnień. Są to rzeczy, które da się realnie zrobić lub reguły, które można wprowadzić by poprawić jakość procesu, produktu i ogólnie pracy zespołu. Oczekujemy się więc od zespołu przygotowania nie ogólnych życzeń ale konkretnych działań - i to takich, których przeprowadzenie jest realne w czasie następnego Sprintu. Tak więc zamiast np. „chcemy pisać więcej testów" (życzenie) powinno to być raczej „w kolejnym Sprincie dla jednego z wymagań zaimplementujemy komplet testów" (konkretne działanie, realistycznie realizowalne w jednym Sprincie). Zatem choć dyskusje o wizji tego jak chcielibyśmy pracować są wartościową częścią Retrospekcji, to jednak powinny być kończone wypracowywaniem konkretnych, realizowalnych działań, które będą zespół do owej wizji przybliżać.

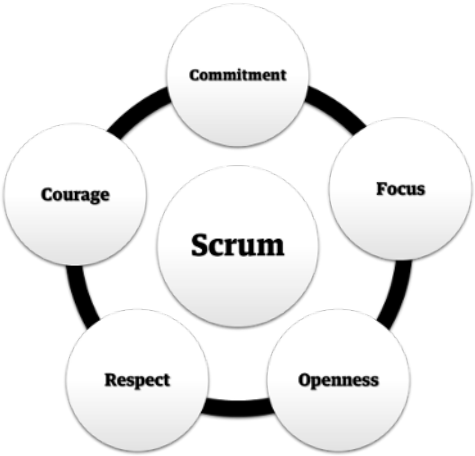
Choć Scrum Guide już o tym nie wspomina, to jednak zadbanie o to, by retrospekcje się odbywały, a uczestnicy rozumieli ich cel i sens jest przede wszystkim obowiązkiem Scrum Mastera. Zarazem Scrum Master powinien zadbać by retrospekcje były pozytywne (unikać bezpłodnego narzekania) - i produktywne, a więc owocowały wypracowaniem wspomnianych usprawnień.

Warto podkreślić, że wszyscy - w tym Scrum Master - biorą udział w retrospekcji jako równorzędni członkowie Scrum Team. Podczas tej sesji Product Owner nie zajmuje się produktem a Deweloperzy nie programują i nie testują, ale wszyscy współpracują nad wypracowaniem działań i zasad, które zaowocują poprawą sposoby pracy, podniesieniem jakości produktu itp.

Podejście to wynika ze słusznej obserwacji, że najczęściej usprawnienia są osiągane nie metodą przełomów i „wielkich postanowień" lecz systematycznych, małych kroków ku lepszemu. Konieczność realizowania retrospekcji co sprint ma zapewnić, że zespół będzie owe kroki podejmował.

Wartości Scruma

Scrum opiera się na pięciu wartościach (ang. *values*):



Wartości Scrum

- **zaangażowanie** (ang. *commitment*) - zaangażowanie w produkt, zespół, jakość, profesjonalizm i ciągłe usprawnianie zespołu, produktu i procesu,
- **skupienie** (ang. *focus*) - skupienie zespołu na Sprincie, na jego Celu, na produkcie, który ma być dostarczony - co oznacza również odrzucanie wszystkiego, co mogłoby to

skupienie naruszyć (np. żądania ad hoc, próby „wywrócenia" Sprintu itp.) - elementem skupienia jest także konstruktywny optymizm[11], a więc koncentracja na tym co jest możliwe, na poszukiwaniu najprostszych rozwiązań, które będą działać, na wiedzy, która jest potrzebna w tej chwili, na tym, co jest robione teraz i co będzie robione w najbliższym czasie, na tym co „się da" a nie na tym co „się nie da",

- **otwartość** (ang. *openness*) - przede wszystkim otwartość oznacza przejrzystość - wolę stałego ujawniania, pokazywania rzeczywistego stanu prac nad produktem, stanu zespołu, stanu praktyk technicznych, oznacza to jednak także otwartość na współpracę pomiędzy osobami o różnych specjalizacjach i wywodzących się różnych organizacji,
- **szacunek** (ang. *respect*) - szacunek dla innych, niezależnie od ich doświadczenia i historii zarówno zawodowej jak i osobistej jest warunkiem dobrej współpracy, szacunek który powinien rozciągać się także na klientów, interesariuszy biznesowych itp.
- **odwaga** (ang. *courage*) - jest niezbędna by utrzymać otwartość, by dostarczać prawdziwych wiadomości, które są niezbędne do podejmowania dobrych decyzji a czasem są nieprzyjemne (odbiegają od oczekiwań i wcześniejszych planów), by wreszcie trzymać się ustalonych zasad oraz postępować w sposób profesjonalny.

Te wartości – zwane wartościami Scruma – muszą być obecne w organizacji, aby Scrum mógł działać dobrze i przynosić

[11] Oczywiście, określenie „konstruktywny optymizm" nie jest dokładnym, słownikowym oddaniem angielskiego słowa *focus*, jednak proste tłumaczenie - „skupienie" - nie oddawałoby dobrze sensu, jaki twórcy Scruma przypisują wartości określonej angielskim *focus*.

oczekiwane korzyści z jego stosowania. Innymi słowy nie są to wartości, których się poza organizacjami stosującymi Scruma nie spotyka, lecz raczej wartości które organizacja powinna mieć lub starać się mieć, by Scruma dobrze zastosować.

Scrum zatem działa najlepiej tam, gdzie pracują zaangażowani ludzie, szukający raczej możliwości niż wymówek, otwarcie mówiący jak jest, odnoszący się z szacunkiem do innych i mający odwagę mówić prawdę. Można powiedzieć, że Scrum jest metodą dla takich właśnie ludzi, dzięki której mogą lepiej zorganizować swoją pracę - zdecydowanie nie jest narzędziem do poganiania wyalienowanych pracowników, którzy są w naszym zespole i firmie bo muszą spłacać raty kredytu.

W tym kontekście jest bardzo ważne, aby podkreślić, że **Scrum jest czymś co**, **zaangażowani**, **profesjonalni Deweloperzy robią**, **żeby lepiej pracować** a nie czymś co inni – Scrum Masterzy, Agile Coache, kierownicy – robią Deweloperom. Jeśli tak jest Scrum przynosi świetne efekty, bo jest niemal naturalnym sposobem działania. Jeśli jednak Scrum jest czymś z pogranicza religii („ceremonie", dziwna mowa) narzucanym Deweloperom, to nic dobrego z tego nie wynika.

Podsumowanie i uwagi końcowe

Scrum jest metodą, która opisuje sposób organizacji pracy jednego niewielkiego zespołu profesjonalistów, który przetwarza pewne rzeczy (życzenia, wymagania, problemy) uszeregowane w postaci listy (backlogu) na gotowy produkt. Produkt ów powstaje stopniowo, iteracyjnie - w kolejnych Sprintach powstają jego kolejne wersje, zawsze technicznie ukończone i działające (spełniające DoD).

Warto zauważyć, że wbrew dość rozpowszechnionemu prze-

konaniu Scrum zdecydowanie nie jest metodą zarządzania projektami. Projekt - z definicji - jest przedsięwzięciem, które ma określony z góry zakres i cel. Scrum niczego takiego nie oczekuje ani nie wymaga - jest procesem ciągłym, o nie zakreślonych granicach zakresu i czasu. Tworzy potok przetwarzania strumienia wymagań na produkt, który może działać tak długo jak długo są nowe wymagania czy potrzeby i zespół mogący je przetworzyć na oprogramowanie. Dobrze odpowiada to zresztą współczesnej rzeczywistości pracy większości zespołów, które nie realizują *de facto* projektów, lecz nie kończący się ciąg zmian w jakimś systemie czy aplikacji (produkcie) będących wynikiem napływających stale potrzeb, żądań i pomysłów.

Scrum nie jest także metodą przetwarzania dobrze zrozumianych, szczegółowo przeanalizowanych i wyspecyfikowanych wymagań na działające oprogramowanie - choć bywa stosowany także w taki sposób. W istocie jednak Scrum jest metodą szukania zrozwiązań na problemy biznesowe poprzez budowę oprogramowania - metodą poszukiwania optymalnej, maksymalnej wartości w warunkach dużej złożoności produktu i domeny oraz niestabilności wymagań, a co za tym idzie ograniczonej przewidywalności.

Scrum jest także metodą, która ma wąski zakres. Scrum nie zajmuje się tym skąd wzięły się elementy Backlogu Produktu albo pracownicy tworzący Zespół Scrum - czasem podkreślam to mówiąc, że „Scrum jest metodą organizacji wykonania pracy, która już jest przez ludzi, którzy już są". W Scrum nie przewidziano również fazy czy etapu poświęconego analizowaniu sensowności przedsięwzięcia jako całości. Zakłada się, że taka analiza jest prowadzona cały czas, co jest możliwe dzięki dostarczaniu działającego oprogramowania co Sprint. Umożliwia to wcześniejsze sprawdzenie przydatności produktu czy to jako

oferty rynkowej czy narzędzia na własne potrzeby i podjęcie decyzji o kontynuowaniu lub przerwaniu jego rozwoju.

Wszystkie wydarzenia w Scrumie - począwszy od Sprintu po spotkania odbywane podczas Sprintu - mają wyznaczone sztywne ramy czasowe. W oryginale Scrum Guide stosowane jest tutaj określenie „timebox". Sugeruje ono sztywność owego czasu - i tak rzeczywiście jest. W prawidłowym stosowaniu Scruma bardzo ważne jest ścisłe i stanowcze przestrzeganie maksymalnych limitów czasu na wszystkie spotkania i inne zdarzenia. Oznacza to, że zdarzenie kończy się wraz z końcem czasu na nie przeznaczonym niezależnie od tego czy znajdujemy się właśnie w trakcie jakiejś dyskusji czy pracy.

Poniżej podsumowanie maksymalnych czasów trwania poszczególnych spotkań w Scrumie:

Spotkanie	**Maksymalny czas trwania**
Planowanie Sprintu	8h
Przegląd Sprintu	4h
Retrospekcja Sprintu	3h
Daily Scrum	15 minut

Jest zalecane proporcjonalne skracanie spotkań dla Sprintów krótszych niż miesięczne. W każdym jednak przypadku długość wszystkich spotkań powinna być ustalona z góry i stała, to znaczy nie zmieniana co Sprint i bez istotnych powodów. Retrospekcja Sprintu to dobre miejsce na dyskusję o dłuości spotkań i ewentualną ich zmianę.

W Scrum wszystkie czasy trwania spotkań (*timeboxes*) to czasy maksymalne. Oznacza to, że każde spotkanie może trwać krócej i zostać naturalnie zakończone kiedy omawiane tematy zostanie wyczerpane. Jedyne zdarzenie Scruma, które nie może trwać ani dłużej ani krócej to Sprint.

1.2 Zmiany w Scrumie

Scrum jest metodą żywą, nadal rozwijaną przez swoich twórców. Dlatego praktycy Scruma powinni zawsze znać jej aktualną wersję. Dotyczy to szczególnie Scrum Masterów.

Oficjalną definicją metody jest od 2010 roku dokument zwany Scrum Guide, który jest przygotowywany i rozwijany przez dwóch twórców metody - Kena Schwabera i Jeffa Sutherlanda. Dokument ten w zwięzły sposób (w oryginale angielskim[12] liczy tylko 13 stron) prezentuje metodę. Jest on także podstawą szkoleń i egzaminów certyfikacyjnych Scrum.org, powinien być także podstawą wszystkich innych dobrych szkoleń ze Scruma[13]. Aktualna wersja tego dokumentu jest zawsze do pobrania ze strony www.scrumguides.org[14].

Ostatnie zmiany - listopad 2020

W tej edycji Scrum Guide został skrócony do 13 stron poprzez uproszczenie języka, usunięcie powtórzeń i skomplikowanych sformułowań. Usunięto także pozostałe jeszcze w Scrum Guide odniesienia do IT i oprogramowania w zgodzie z linią ewolucji Scruma jako frameworku do każdej skomplikowanej pracy.

Wprowadzono ponadto następujące zmiany:

- **usunięcie ról oraz Development Team** – obecnie Deweloper, Product Owner, Scrum Master to odpowiedzialności

[12]Chociaż dostępne jest polskie tłumaczenie przygotowane przez Tomka Włodarka zdecydowanie zalecam osobom znającym język angielski korzystanie z oryginalnej, angielskiej wersji dokumentu. Ułatwia to rozumienie literatury (papierowej i webowej), która w większości jest po angielsku. Również egzaminy na certyfikaty wydawane przez Scrum.org zdaje się w tym języku.

[13]Jeśli idzie o szkolenia, to oczywiście polecam Code Sprinters: https://codesprinters.pl/

[14]http://www.scrumguides.org

(ang. *accountabilities*) zespołu Scrum. Zmiana ta ma na celu podkreślenie jedności zespołu Scrum oraz przeciwstawienie się powstającym w praktyce sztywnym podziałom między Product Ownerami czy Scrum Masterami a Deweloperami.

- **wprowadzenie Celu Produktu** (ang. *Product Goal*) – jest to pośredni cel produktowy, do którego zespół Scrum może dążyć przez wiele sprintów.
- **wprowadzenie pojęcia zobowiązanie** (ang. *commitment*) na określenie Celu Sprintu, Celu Produktu i Kryterium Ukończenia. Każde z tych zobowiązań jest związane z jednym z artefaktów Scruma.
- **wprowadzenie trzeciego tematu na Planowanie Sprintu** - poza „co" i „jak" pojawia się także „dlaczego", co – w połączeniu z Celem Produktu – ma ułatwić wypracowywanie Celów Sprintu i pomóc zespołom w koncentracji na zwiększaniu wartości produktu. Ponadto, Cel Sprintu jest obecnie ustalany przez cały zespół Scrum i nie musi być ustalony na sam koniec Planowania Sprintu, jedynie musi być już wtedy znany.
- zmiana pojęcia **samoorganizacja na samozarządzanie** - zmiana wynikająca z tego, że zespół określa nie tylko kto i jak pracuje, ale także nad czym (co jest logiczne, skoro mowa obecnie o całym zepsole Scrum, a więc łącznie z Product Ownerem),
- usunięcie zalecenia, by na pielęgnację Backlogu Produktu (*refinement*) poświęcać nie więcej niż 10% czasu Sprintu. Obecnie jest mowa o tym, że elementy Backlogu Produktu zazwyczaj osiągają stan gotowości (są nie większe niż „pojemność" Sprintu) w wyniku takich właśnie działań pielęgnacyjnych.

Poprzednie zmiany

listopad 2017

W edycji Scrum Guide z 2017 roku pojawiły się następujące zmiany:

- Dodanie rozdziału podkreślajacego iż Scrum jest używany także w innych dziedziniach niż tylko rozwój oprogramowania, głównie chodzi tu o rozwój i utrzymanie różnego rodzaju produktów.
- Rezygnacja z „trzech pytań" jako wymaganego metodą formatu spotkania Daily Scrum - zmiana ta wynikła z obserwacji, że w wielu miejscach spotkanie to zaczęło być traktowane jako rytuał, który należy wypełniać by „robić Scrum" przy braku zrozumienia sensu Daily Scrum jako codziennej empirycznej inspekcji i adaptacji zespołu w jego pracy nad osiągnięciem Celu Sprintu. Twórcy Scrum mają nadzieję, że określenie „trzech pytań" jako dobrej ale nie jedynej praktyki pomoże w upowszechnieniu lepszego zrozumienia sensu tego spotkania. Dodatkowo wyjaśniono, że Daily Scrum to wewnętrzne spotkanie Development Team, jeśli są na nim obecne inne osoby Scrum Master ma dopilnować, że nie zaburza to (ang. *disrupt*) Daily.
- Podkreślenie, że fundamnetem Scrum jest nieduży zespół, elastyczny i dostosowujący się.
- Wyraźne zapisanie, że działająca wersja prodktu (*„Done" increment*) musi być gotowa przed rozpoczęciem Przeglądu Sprintu.
- Podkreślenie, że na Przeglądzie mamy inspekcję przyrostu a nie jego „akceptację".

- Zmiana opisania roli Scrum Mastera - podkreślenie, że Scrum Masterzy mają promować i wspierać Scrum takim, jakim jest on zapisany w Scrum Guide oraz wyjaśnienie, że mają to robić przede wszystkim poprzez pomoc innym w zrozumieniu Scruma. Dodatkowo roszerzono opis co Scrum Master robi dla PO, zespołu i reszty organizacji - ważne dodatki to podkreślenie, że Scrum Master powinien umieć zaplanować wprowadzanie Scruma w organizacji oraz współpracować z innymi obecnymi w niej Scrum Masterami nad ulepszaniem efektywności i jakości Scruma w niej.
- Podkreślenie, że Retrospekcja powinna być pozytywna i produktywna - skutkować podejmowaniem przez zespół działań usprawniających. Temu ostatniemu celowi służy również zapisanie, że co najmniej jedno konkretne działanie usprawniające z retrospekcji ma bezpośrodnio trafić do Backlogu Sprintu w kolejnym sprincie. Podkreślono także, że Scrum Master uczestniczy w retrospekcji jako członek zespołu - nie jest zewnętrznym, obojętnym facylitatorem lecz uczestniczy w dyskusji.
- Backlog Produktu zawiera tylko takie funkcje, co do których na tą chwilę wiemy, że będą potrzebne w produkcie (dotychczas było iż wiadomo, że mogą być potrzebne).

lipiec 2016

Zmiana w edycji z 2016 r. była co do treści niewielka - dodano rodział o wartościach Scrum składający się z dwóch akapitów. Bardziej znaczący od samej treści tego rozdziału jest sam fakt jego dodania. Wynikło to nie tylko z wyrażonego przez praktyków Scruma żądania w głosowaniu na portalu „User Voice" (spośród różnych propozycji zmian w Scrum Guide propozycja

dopisania tam 5 wartości Scrum zyskała najwięcej głosów), ale także z poczynionej przez twórców Scrum obserwacji, że w wielu organizacjach stosuje się Scrum ale nie przynosi on takich wyników, jakich się po nim spodziewano. Przyczyny tego stanu rzeczy twórcy Scruma upartują przede wszystkim w braku w tych organizacjach owych pięciu wartości.

Warto podkreślić, że było to dodanie warości Scrum jedynie do Scrum Guide, nie zaś do Scruma. Wynika to z tego, że wartości te były w Scrumie obecne. Co więcej, były na długoprzed 2016 nauczane jako część oficjalnych, certyfikowanych szkoleń ze Scrum.

lipiec 2013

Pojawiły się wówczas następujące zmiany:

- Dodano rozdział poświęcony znaczeniu przejrzystości (ang. transparency). Przejrzystość oznacza, że rzeczywisty, prawdziwy stan procesu - a więc przede wszystkim zawartość backlogu i stan produktu - są znane uczestnikom procesu. Jest to istotne by w Scrumie mógł działać empiryzm. Nie jest to zmiana w Scrumie jako takim, celem jest przede wszystkim podkreślenie rangi przejrzystości dla prawidłowego działania Scruma.
- Zniesiono zalecane rozdzielnie Planowania Sprintu na część pierwszą i drugą. Uprzednio w części pierwszej zespół deweloperski stawiał prognozę co do tego które wymagania z Backlogu Produktu ukończy w nadchodzącym w sprincie. Następnie w części drugiej szczegółowo planował jak to zrobi. Każda z części miała trwać połowę ogólnego limitu czasu (timebox) na Planowanie Sprintu. Okazało się w praktyce, że wiele zespołów praktykuje z powodzeniem

Scruma nie dzieląc planowania na ściśle limitowane czasowo sesje.

- Przy okazji doprecyzowano, że Cel Sprintu (ang. Sprint Goal) tworzy na Planowaniu Sprintu Zespół Scrum (Scrum Team) po przygotowaniu już przez Zespół Deweloperski prognozy jakie zadania zostaną wykonane.
- Ze względów językowych[15] zmieniono angielska nazwę „Wypielęgnowanego Backlogu" z „Groomed" na „Refined".
- Dodano nowy stan wymagania na Backlogu Produktu - „Gotowe" (ang. Ready). Uprzednio takiego oddzielnego, nazwanego stanu wymagania w metodzie oficjalnie nie było, choć (czasem wraz z formalną definicją - Defintion of Ready) było praktykowane przez wiele zespołów już co najmniej od 2008 roku.
- Usunięte zostały sztywne przeliczniki limitu czasu na spotkania Scrumowe proporcjonalnie do długości sprintu. Uprzednio limit czasu (timebox) na Planowanie, Przegląd i Retrospekcję był sztywno przeliczany w propocji do długości trwania sprintu. Obecnie niezależnie od długości sprintu obowiązuje ten sam, maksymalny limit czasu, który odpowiada limitowi uprzednio dopuszczanemu jedynie dla najdłuższych, miesięcznych sprintów[16].
- Zmieniono formułę Daily Scrum by podkreślić, iż nie jest to spotkanie raportowe (zespół składający raport Scrum Masterowi to w istocie jedna z najczęściej spotykanych dysfunkcji), lecz moment synchronizacji samo-organizu-

[15] Okazało się, że termin „grooming" w języku angielskim używanym w Wielkiej Brytanii zaczął mieć negatywne konotacje kulturowe.

[16] Można się spodziewać, że ta właśnie zmiana będzie najbardziej kontrowersyjna i wielu Scrum Masterów pozostanie przy starej praktyce. Wydaje mi się to wskazane, ponieważ krótkie limity czasu wpływają bardzo dobrze na zespół a przy tygodniowym sprincie nie ma sensu poświęcać więcej niż dwie godziny na planowanie.

jącego się zespołu próbującego osiągnąć wspólny cel. Poprzednio pytania brzmiały „Co zrobiłem wczoraj? Co zrobię dziś? Jakie mam problemy?".
- Zmieniono omówienie Przeglądu Sprintu (Sprint Review) aby podkreślić koncentrację na inspekcji inkrementu produktu i zastanowieniu się jaką wartość on przyniósł i jaką wartość można jeszcze do produktu wprowadzić.

październik 2011

Pojawiło się w niej kilka istotnych zmian w stosunku do starszych wersji Scrum (zarówno tej z pierwszego wydania Scrum Guide jak i przede wszystkim Scruma znanego z wydanych wcześniej książek i szkoleń). Potocznie mówimy o Scrumie sprzed tej aktualizacji „stary Scrum". Choć zmiany te nie są duże, to jednak są dość istotne:

- Backlog produktu nie jest już spriorytetyzowany (ang. *prioritized*), lecz **uporządkowany** (ang. *ordered*). Zmiany tej dokonano, ponieważ okazało się, że istnieją inne dobre metody porządkowania backlogu niż tylko priorytet (*temat omawiam szerzej w rozdziale o Backlogu Produktu*).
- Wykluczono możliwość udziału innych osób niż Scrum Master i Zespół Deweloperski w spotkaniu Daily Scrum (uprzednio mieli oni możliwość uczestnictwa pod warunkiem nie zabierania głosu). Zmiana ta wynikła z praktyki - okazało się, że obecność inych osób (zwłaszcza klientów, kierowników liniowych, interesariuszy) zaburza przebieg Daily Scrum uniemożliwiając temu spotkaniu spełnienie jego funkcji.
- Wynik Planowania Sprintu - a więc lista wymagań, które zespół wziął „na Sprint" - jest obecnie definiowana jako

prognoza (ang. *forecast*). Uprzednio mówiło się o zobowiązaniu (**commitment**), co implikowało zobowiązanie do dostarczenia wszystkich wymagań wziętych do sprintu. Praktyka pokazała, że umożliwia to powrót dyfunkcji klasy „*musi być na wtorek, nie ważne jak*" w skali sprintu. Zobowiązanie przesunięto na Cel Sprintu (ang. *Sprint Goal*), który uprzednio nie miał dużego praktycznego znaczenia.

Ta zamiana była chyba najtrudniejsza do zaakceptowania i wdrożenia, zwłaszcza przez zespoły pracujące w dość sztywnych organizacjach, które zdążyły się już przyzwyczaić do traktowania Backlogu Sprintu jako kontraktu. Zresztą, w wielu miejscach taki sposób patrzenia na Backlog Sprintu pokutuje nadal, prawie dziesięć lat później.

- Doprecyzowano, że określenie developer odnosi się do wszystkich członków zespołu niezależnie od ich specjalności i sposobu w jaki partycypują w pracy.
- Planowanie Wydań (**Release Planning**) przestało być częścią Scruma. Twórcy metody uznali, że praktyka ta - choć wartościowa - nie jest niezbędna dla prawidłowego działania zespołu w Scrumie.
- Backlog Sprintu przedefinowano jako „plan ułożony w taki sposób, aby możliwe było śledzenie postępów pracy podczas Sprintu i określanie jej trendów". Zatem Backlog Sprintu nie musi zawierać zadań (ang. *tasks*), choć zespół Scrum może tą praktykę stosować. Zmiana wynikła z obserwacji, że istnieją zespoły, które w ogóle nie praktykują rozbijania wymagań na zadania lecz śledzą postęp prac właśnie na poziomie wymagań.

  Wraz z usunięciem wymogu stosowania zadań usunięto także wymóg prowadzenia wykresu burndown. W to miejsce określono, że zespół ma śledzić postępy swojej pracy

podczas sprintu - może to robić poprzez stosowanie zadań i wykresu burndown albo w jakiś inny sposób.
- Doprecyzowano kto ma prawo podjąć decyzję o nieplanowanym przerwaniu sprintu (*abnormal sprint termination*). Uprzednio nie było to określone, obecnie osobą tą jest Product Owner. Zmieniono też sposób traktowania już wykonanej pracy w chwili przerywania sprintu - uprzednio cała praca była wyrzucana, obecnie możliwe jest wydanie inkrementu zawierającego te wymagania, których implementacja spełniała w chwili przerwania kryteria ukończenia (*Definition of Done*).

2010 rok

Pierwsza edycja Scrum Guide. Choć nie ma w niej - z konieczności - zmian w stosunku do poprzednich edycji warto podkreślić pewne zmiany wprowadzone w stosunku do wcześniej znanego (i szeroko praktykowanego) Scruma znanego ze wspomnianej książki Kena Schwabera i szkoleń:

- wprowadzenie **Retrospekcji** wraz z listą konkretnych działań (**actionable improvements**) jako produktem jako obowiązkowego zdarzenia na koniec każdego Sprintu,
- wprowadzenie **pielęgnacji backlogu** (wówczas jeszcze zwanej **Backlog Grooming**, później *refinement*) jako obowiązkowego elementu Scruma, określenie maksymalnego czasu jaki można na to poświęcić w Sprincie na 10%.