SZKOLENIA SII

Zostań Scrum Masterem

Str. 2
Przystępny słownik pojęć Agile

Str. 21
Scrum – nowe reguły gry 2020

Str. 28
Siedem grzechów głównych Scrum Mastera

Str. 28
Szkolenia „Zostań Specjalistą IT” – zdobądź niezbędną wiedzę, aby rozpocząć pracę w IT

SZKOLENIA SII

Przystępny słownik pojęć Agile

Co robi w pracy Scrum Master? Jak wytłumaczyć tę i wiele innych kwestii komuś, kto nigdy nie zetknął się z ideą Agile? Pełna definicja ze wszystkimi odniesieniami będzie zupełnie niezrozumiała, więc zdecydowanie nie trzeba startować od zera i objaśniać wszystkiego. Wszystko da się ładnie uprościć.

Mistrz młyna to ktoś, kto zwinnie przeskakuje wodospady. To mocno niepełna, ale formalnie poprawna definicja opisująca na czym polega praca Scrum Mastera. Jest kompletnie niezrozumiała dla osób postronnych i dobrze sprawdza się w trakcie rodzinnych spędów, powstrzymując kolejne pytania. W innych okolicznościach zdecydowanie lepiej spróbować rozwiać wątpliwości pytającego w zrozumiały dla niego sposób.

Niniejszy słowniczek powstał jako pomoc w łatwym zorientowaniu się o co chodzi w Agile i Scrum dla tych, którzy do tej pory nie mieli wcale lub prawie wcale do czynienia z tą tematyką. Z tego powodu niektóre sformułowania i używane słownictwo mogą wzbudzać kontrowersje wśród zaawansowanych praktyków lub/i ortodoksów tematyki. Uproszczone i niewyczerpujące tematu definicje zastosowano celowo, aby nie stwarzać dodatkowej bariery przez opisywanie niezrozumiałych pojęć za pomocą innych niezrozumiałych pojęć.

Do pełnego zrozumienia danego terminu niekiedy trzeba będzie przejrzeć dwie, trzy dodatkowe definicje lub wręcz przeczytać wszystko, zastanowić się, a potem przeczytać jeszcze raz zwracając uwagę na fragmenty, które umknęły lub były niezrozumiałe przy pierwszym podejściu.

Dokładnie tak jak z instrukcją obsługi dowolnego urządzenia. Nie jest to trudne, a do tego można to robić na raty (czyli jak na Agile przystało – w modelu iteracyjnym).

Agile

Sposób myślenia i pracy zgodny z założeniami Manifestu Agile. W ujęciu projektowym, wspólne określenie zwinnych metod wytwarzania oprogramowania opartych o model iteracyjno-przyrostowy. Skupia się na dostarczaniu produktu w sposób ciągły przy silnym zaangażowaniu klienta w proces. W założeniu zespoły pracują zwinnie, czyli szybko i elastycznie dopasowują się do zmieniających się wymagań klienta i warunków zewnętrznych. Polecany przy projektach posiadających niestabilne wymagania i brak jasnej, konkretnej i niezmiennej wizji produktu końcowego. Umożliwia rozpoczęcie działania bez kompletu szczegółowych wymagań oraz dopuszcza możliwość ich zmiany bez konieczności zaczynania pracy od nowa. Jego głównym celem jest podniesienie jakości produktu (rozumianej jako zadowolenie klienta oraz stopień dopracowania produktu) tak, by odpowiadał realnym potrzebom klienta, a nie tylko początkowym wymaganiom.

Agile Coach

Osoba edukująca innych czym jest Agile. Jego główną rolą jest zazwyczaj wspieranie organizacji w transformacji zwinnej oraz praca z jej częścią biznesową na poziomie kadry zarządzającej wyższego szczebla. Między Scrum Masterem a Agile Coachem granica jest płynna, ale podstawowym wymogiem aby Scrum Master został Agile Coachem, jest doświadczenie nie tylko w pracy z zespołem, ale też z całą organizacją. Nie decyduje o sposobie i metodach wprowadzania filozofii Agile w organizacji, a skupia się na edukacji, celem wspierania organizacji we wprowadzaniu oraz utrzymywaniu zwinności, rozumieniu i stosowaniu obranej metodyki.

Agile Manifesto (PL: Manifest Agile)

Dokument opracowany w 2001 roku przez 17 praktyków tzw. lekkich metod (m. in. Scrum, XP, DSDM, FDD, CC) opisujący wspólne podstawy i założenia tych metod oraz podkreślający wartości na jakich się opierają. Przy okazji opracowywania manifestu zmieniono nazewnictwo i zamiast “metod lekkich” pojawiły się “zwinne” (agile). Treść Manifestu Agile: https://agilemanifesto.org/iso/pl/manifesto.html

Burn-down chart (PL: Wykres Spalania)

Sposób wizualizacji, w formie wykresu, czasu i pracy pozostałych do wykonania w danym Sprincie/projekcie. Pionowa oś wykresu to mierzalny zakres pracy (np. Story Pointy), pozioma oś to czas (np. w dniach). Na wykresie zazwyczaj znajduje się linia wartości uśrednionych przedstawiająca idealny, liniowy spadek pozostałej do wykonania pracy. Wykres obrazuje tempo realizacji zadań w danym Sprincie, a linia idealna pokazuje w jakim tempie Zespół powinien pracować aby ukończyć założone zadania.

Burn-up chart

Sposób wizualizacji pracy i czasu pozostałych do wykonania w danym Sprincie/projekcie, w formie wykresu. Częściej stosowany w przypadku projektu niż Sprintu. Pionowa oś wykresu to mierzalny zakres pracy (np. Story Pointy), pozioma oś to czas (np. iteracje). Na wykresie znajduje się linia przedstawiająca zmiany w zakresie pracy (zmiana ilości Story Point), druga linia wskazuje ile pracy zostało wykonane w poszczególnych iteracjach. Wykres obrazuje w jakim estymowanym czasie przy bieżącym tempie, prace dla wycenionych SP zostaną ukończone. Jest to możliwe poprzez uśrednienie i przedłużenie linii wykonanej pracy.

Capacity (PL: Pojemność)

Estymowana praca możliwa do wykonania przez dany Zespół w trakcie Sprintu. Zawsze odnosi się do konkretnego Zespołu. Całkowicie zależna od nieobecności deweloperów oraz ich zaangażowania w inne zadania. Możliwa do określenia na podstawie Velocity, Load oraz dostępności poszczególnych deweloperów.

Daily

Jedno z wydarzeń (ceremonii) scrum'owych. Codzienne spotkanie Zespołu Deweloperskiego, odbywa się każdego dnia Sprintu, zazwyczaj w tym samym miejscu i o tej samej porze; zwane też stand up'em z racji tego, że często odbywa się na stojąco. Czasowo ograniczony jest do 15 minut. Służy koordynacji prac na okres do następnego Daily, określeniu czy istnieją problemy zagrażające realizacji Celu Sprintu oraz sprecyzowaniu co Zespół Deweloperski zamierza zrobić aby zrealizować Celu Sprintu.

Definition of Done (DoD; pl. Definicja Ukończenia)

Lista kryteriów jakie musi spełniać każde pojedyncze zadanie, nad którym Zespół Deweloperski pracuje w trakcie Sprintu, po spełnieniu których wszystkie strony procesu uznają zadanie za zakończone. Jest tworzona w oparciu o standardy obowiązujące w organizacji lub w przypadku ich braku – przez Zespół Deweloperski. Może być dyskutowana i zmieniana przez Zespół w trakcie Retrospektywy. Zazwyczaj im dojrzalszy Zespół, tym bardziej rozbudowana jest DoD. Istnienie listy z jasno sprecyzowanymi wymaganiami zapewnia, że wszyscy uczestnicy procesu wiedzą co oznacza stwierdzenie, że dane zadanie jest skończone i nie ma miejsca sytuacja, w której następuje rozbieżność pomiędzy tym, co uważa za skończone Zespół Deweloperski, a tym, co za skończone uważa Product Owner i Interesariusze. Stan ukończenia zadania jest zero-jedynkowy. Nie istnieją zadania częściowo skończone. W wypadku, gdy nad projektem pracuje kilka zespołów, wszystkie powinny mieć tą samą Definicję Ukończenia.

Definition of Ready (DoR; PL: Definicja Gotowości)

Lista kryteriów jakie musi spełniać każde pojedyncze zadanie w Backlogu, aby Zespół Deweloperski mógł je realizować podczas Sprintu. DoR jest tworzona przez Product Ownera i ulepszana z Zespołem Deweloperskim w trakcie Retrospektywy. W przypadku Definicji Gotowości „gotowy" nie oznacza, że zadania w Backlogu muszą być w 100% zdefiniowane, muszą być „wystarczająco gotowe", aby Zespół Deweloperski był przekonany, że może z powodzeniem przystąpić do realizacji zadania, rozumie ryzyko biznesowe i cel klienta.

Deployment

Fizyczne umiejscowienie produktu w wyznaczonym przez klienta środowisku (np. produkcyjnym).

Design Thinking (PL: myślenie projektowe)

Metoda twórczego rozwiązywania problemów, zdefiniowana po raz pierwszy na Uniwersytecie Stanforda w USA w latach 60. Zakłada, że celem jest dostarczenie innowacyjnych, dopasowanych do oczekiwań końcowego odbiorcy, rozwiązań poprzez wykorzystywanie metod pracy pobudzających kreatywność. Metoda ta skupia się na użytkowniku i zrozumieniu jego potrzeb. To właśnie te realne potrzeby, nie zaś założenia tworzone na podstawie wyobrażeń o użytkowniku są punktem wyjścia do pracy nad produktem. Proces twórczy podzielony jest na kilka etapów:

- Empatia – poznanie potrzeb końcowego klienta/użytkownika, jego perspektywy, obserwacja zachowań,
- Zdefiniowanie problemu – kluczowy etap polegający na zdefiniowaniu właściwego problemu, wymaga przełamania ram myślowych i i szerokiego spojrzenia,
- Generowanie pomysłów – tworzenie wielu, w tym nieszablonowych, sposobów rozwiązania zdefiniowanego problemu,
- Budowanie prototypów – wizualizacja rozwiązania problemu i zebranie opinii na jego temat,
- Testowanie – sprawdzanie efektywności rozwiązania w środowisku użytkownika.

Development Team (Dev Team; PL: Zespół Deweloperski)

Jedna z trzech ról w Scrumie, grupa profesjonalistów – od 3 do 9 osób, która wspólnie posiada wystarczające kompetencje do wytworzenia Przyrostu produktu. Liczebność zespołu jest ograniczona ze względu na stopień komplikacji komunikacji w większych grupach i zbyt małą produktywność w mniejszych grupach. W Zespole nie ma podziału na role (tester, frontend dev, backend dev itp.) – wszyscy są nazywani deweloperami (związane jest to z kwestią wspólnej odpowiedzialności zespołu), ale to nie znaczy, że wszyscy muszą mieć kompetencje programistyczne. Zespół Deweloperski ma bardzo dużą autonomię i sam określa co robi i w jaki sposób. Nikt nie narzuca Zespołowi sposobu w jaki praca zostanie wykonana (wyjątkiem są ogólne reguły organizacji). Zespół w całości ponosi odpowiedzialność za wykonaną pracę.

DevOps

Skrót od Development and Operations; Osoba łącząca w sobie kompetencje dewelopera i wdrożeniowca lub osoby odpowiedzialnej za utrzymanie programu/systemu. Niekiedy dodaje się także do tego kompetencje związane z testowaniem lub bezpieczeństwem (ang. DevSecOps). W klasycznym układzie te kompetencje należą do różnych osób lub zespołów. Może też oznaczać rodzaj kultury w skali całej organizacji. Wymaga to połączenia obszarów, które w firmach zazwyczaj funkcjonują oddzielnie: zespołu rozwijającego oprogramowanie (Dev) oraz zespołu operacji (Ops).

Elements of Scrum (PL: Elementy Scrum)

Wszystkie składowe frameworku Scrum. Brak jednego elementu z poniższej listy wyklucza możliwość pracy w Scrum:

- 3 role: Product Owner (Właściciel Produktu), Development Team (Zespół Deweloperski) i Scrum Master (razem tworzą Zespół Scrumowy)
- 3 artefakty: Product Backlog, Sprint Backlog, Przyrost (Increment)

- 4 zdarzenia: Planning (Planowanie), Daily, Sprint Review (Przegląd Sprintu), Sprint Retrospective (Retrospektywa Sprintu)
- 3 filary: Przejrzystość, Inspekcja, Adaptacja

Empiricism (PL: Empiryzm)

Doktryna filozoficzna o nazwie pochodzącej od starogreckiego słowa oznaczającego doświadczenie. Głosi, że źródłem poznania są bodźce zmysłowe docierające do człowieka, a wiedza wynika z doświadczania i podejmowania decyzji w oparciu o to, co zostało poznane. W zarządzaniu projektami oznacza to, że wszelkie decyzje powinny być oparte na obserwacji i doświadczeniu.

Increment (PL: Przyrost)

Wszystkie zadania i elementy z Backlogu ukończone w trakcie Sprintu oraz wszystkie zadania z Backlogu ukończone w trakcie poprzednich Sprintów. Każdy Sprint powinien zakończyć się stworzeniem działającego fragmentu oprogramowania/produktu możliwego do pokazania Interesariuszom. W pierwszym Sprincie powstaje pierwszy fragment, w każdym kolejnym zostaje dobudowany do niego kolejny, a suma tych fragmentów tworzy Przyrost produktu.

INVEST

Metoda określania poprawności konstrukcji User Story. W myśl tej metody US powinny być:

- Independent – niezależne od siebie nawzajem
- Negotiable – negocjowalne (zakres prac do wykonania jest ustalany na drodze rozmów/negocjacji między Zespołem Deweloperskim a Product Ownerem)
- Valuable – wartościowe (posiadają określoną wartość biznesową i są wartościowe z punktu widzenia użytkownika końcowego)
- Estimable – szacowalne (User Story powinno być na tyle precyzyjnie określone, aby było możliwe oszacowanie czasu ich implementacji)

- Small – małe (możliwe do zrobienia w trakcie jednego Sprintu)
- Testable – testowalne (implementacja jest możliwa do potwierdzenia).

Kanban

Metoda stosowana w procesach wytwórczych, w których produkcja jest uzależniona od zamówień odbiorców (nie zaś od planu produkcji), oparta o rzeczywiste zużycie materiałów. Opracowana została w latach ’40. XX w. na potrzeby systemów produkcyjnych Toyoty. Zapewnia ciągłość produkcyjną przy założeniu nieustannej optymalizacji procesu. Najbardziej efektywna jest w przypadku procesów ciągłych, których nie da się wygodnie i naturalnie podzielić na iteracje, np. praca przy obsłudze help-desku. Jednym z charakterystycznych elementów jest wskaźnik Work In Progress Limit, czyli ograniczenie zadań, które mogą być otwarte na jednym stanowisku.
W procesach wytwarzania oprogramowania Kanban można sprowadzić do trzech zasad:

- Wizualizacja przepływu
- Ograniczanie pracy cząstkowej (Work In Progress Limit)
- Zarządzanie przepływem

Manifestacją metody Kanban w IT jest tablica kanbanowa, na której zaznaczany jest przepływ zadań. Istotą tablicy Kanban jest oznaczenie limitów prac w toku na każdym etapie wytwórczym.

Lean Management (Lean; PL: Szczupłe Zarządzanie)

Strategia dostarczania produktów zgodnych z oczekiwaniami w najprostszy z możliwych sposobów, przy założeniu, że nie może się to odbywać kosztem załogi, oraz przy nastawieniu na maksymalną likwidację wszelkich marnotrawstw. Poprzez optymalizację dąży się do zużywania jak najmniejszej ilości zasobów – czasu, wysiłku, pieniędzy, przestrzeni produkcyjnej, narzędzi, etc. Usuwane są wszystkie elementy, które nie są niezbędne do wykonania produktu o założonej jakości.
Jednocześnie minimalizuje się zapasy preferując dostawy dokładnie na czas i takie projektowanie systemów, które umożliwia szybkie wykrywanie błędów.

Load (PL: Obciążenie)

Ilość pracy jaką Zespół Deweloperski zamierza dostarczyć w trakcie sprintu. Jest określane przez Zespół Deweloperski na podstawie Capacity oraz zakresu Backlogu Sprintu.

Pillars of Scrum (PL: Filary Scrum)

Trzy fundamentalne zasady, na których opiera się Scrum:

- Transparentność – dostępność dla wszystkich osób zaangażowanych w proces do wszystkich niezbędnych im informacji oraz jasna i wspólna dla wszystkich terminologia używa w trakcie procesu. Jeżeli dane są dostępne, ale nie są zrozumiałe nie można mówić o pełnej Transparentności.
- Inspekcja – nieustanne skupienie na procesie w celu szybkiego wyłapywania wszelkich zachodzących w nim zmian. Aby inspekcja była możliwa niezbędna jest Transparentność. Każde Spotkanie Scrumowe jest okazją do Inspekcji. Nie należy jednak mylić tego pojęcia z raportowaniem.
- Adaptacja – efekt wyciągania praktycznych wniosków z Inspekcji, czyli szybkie reagowanie na zmiany i wprowadzanie niezbędnych korekt. Aby adaptacja była możliwa niezbędna jest Inspekcja i Transparentność.

Sens pracy zwinnej polega m.in. na wykonywaniu niekończącej się pętli Inspekcji i Adaptacji.

Planning (PL: Planowanie)

Jedno z wydarzeń (ceremonii) Scrumowych, pierwsze w Sprincie. Powinno przynieść odpowiedź na pytanie co Zespół będzie robił w nadchodzącym Sprincie i jak będzie to robił. W tym celu Zespół Deweloperski określa, które zadania z Backlogu zrealizuje w nadchodzącym sprincie. Zadania powinny być powiązane z Celem Sprintu. Ponadto, Zespół Deweloperski ustala jak będzie realizował Cel Sprintu i zadania z Backlogu Sprintu. W razie potrzeby w trakcie tego spotkania większe zadania powinny zostać rozbite na mniejsze części oraz powinien pojawić się przynajmniej częściowy plan zrealizowania Celu Sprintu. Na koniec planowania Zespół Deweloperski powinien być w stanie wytłumaczyć jak zamierza zrealizować swoją pracę.

Planning Poker

Jedna z metod szacowania pracochłonności i stopnia złożoności zadań z Backlogu, używana m.in. w Scrum. Zazwyczaj zadania wyceniane są w Story Points. Jest to gra karciana wykorzystująca specjalne karty z oznaczeniami punktowymi (choć można użyć także aplikacji webowych czy mobilnych). Najczęściej wykorzystuje ciąg Fibonacciego lub zmodyfikowany ciąg Fibonacciego: 1, 2, 3, 5, 8, 13, 20, 40, 100. Podczas szacowania zadania, każdy deweloper określa indywidualnie wartość punktową jaką chce przypisać do zadania, po czym jednocześnie wszyscy ujawniają swoje oszacowanie. W wypadku rozbieżności, osoby które zadeklarowały skrajne wartości uzasadniają swoją ocenę zespołowi i rozpoczyna się dyskusja na ten temat. Następnie przeprowadzana jest kolejna runda szacowania. Jeżeli wyniki nadal nie są jednakowe, odbywa się kolejna dyskusja. Jeżeli po trzeciej sesji głosowania nadal nie ma zgodności, najczęściej przyjmuje się inną metodę przyjęcia ostatecznej wartości ustaloną przez Zespół, np. średnią, odrzucenie skrajnych ocen itp.

Product Backlog (PL: Backlog Produktu, Rejestr Produktu)

Uporządkowany spis (rejestr) zadań do wykonania podczas pracy nad produktem, gdzie priorytetowe i najpełniej opisane zadania znajdują się na górze listy. Zadania powinny zostać poukładane w Backlogu zgodnie z wartością biznesową jaką dostarczają. Zadaniami są nie tylko wymagania funkcjonalne, ale też cała praca jaką trzeba wykonać przy produkcie – usuwanie błędów, poprawki, oraz usuwanie długu technologicznego. Zadania umieszczane w Backlogu bardzo często mają postać User Stories, ale nie ma takiego formalnego wymagania. Za zarządzanie i porządkowanie Backlogu jest odpowiedzialny Product Owner.

Backlogu nie należy mylić z dokumentacją projektową, ta ostatnia jest zamkniętym zestawem wymagań, a Backlog podlega stałym zmianom (żyje). Zmienia się tak długo, jak długo żyje produkt, którego dotyczy. Aby uniknąć chaosu jedyną osobą uprawnioną do jego uzupełniania jest Product Owner.

Product Owner (PO; pl. Właściciel Produktu)

Jedna z trzech ról w Scrum, odpowiedzialna za wartość biznesową produktu. Zawsze jest to pojedyncza osoba, nigdy komitet czy grupa osób (choć może korzystać z pomocy analityka i innych specjalistów). PO jest jedyną osobą odpowiedzialną za Backlog Produktu, priorytetyzację zadań i przełożenie wymagań biznesowych na formę zrozumiałą dla Zespołu Deweloperskiego. Kontaktuje się z biznesem i wszelkimi Interesariuszami. Powinien być dostępny dla Zespołu Deweloperskiego w takim wymiarze, w jakim Zespół go potrzebuje.

Refinement (PL: Doskonalenie)

Działanie polegające na ulepszaniu Backlogu Produktu. Backlog stale się zmienia, zatem wymaga nieustannej pracy nad zawartymi w nim zadaniami. Działania w zakresie Refinementu polegają na wspólnej pracy PO oraz Zespołu Deweloperskiego nad uszczegóławianiem elementów Backlogu. Polega to na doprecyzowaniu i korygowaniu zadań, a także na zmianie ich priorytetyzacji. Odbywa się w trakcie trwania Sprintu w momencie dowolnie wybranym przez Zespół Scrumowy. Nie jest osobnym, wydzielonym wydarzeniem, odbywa się w zależności od potrzeb, jednak przyjmuje się, że powinien zająć nie więcej niż 10% długości Sprintu. Głównym celem Refinementu jest przygotowanie zadań z Backlogu do realizacji w nadchodzących 2-3 sprintach.

Release (PL: Wydanie)

Oficjalna dystrybucja Przyrostu produktu udostępniona dla użytkownika docelowego, na finalnym środowisku.

Scrum

Ramy postępowania (ang. framework) z grupy technik zwinnych (ang. Agile), wykorzystywane przy zarządzaniu wytwarzaniem złożonych produktów empiryczno-adaptacyjnie i w sposób przyrostowy. Założenia opisano w Scrum Guide autorstwa Ken'a Schwaber'a i Jeff'a Sutherland'a.
https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Polish.pdf
Ideą pracy przyrostowej jest możliwość szybkiego przygotowania działających produktów o ograniczonej funkcjonalności w krótkim czasie i częstego zbierania informacji zwrotnej od Interesariuszy i użytkowników końcowych. Często otrzymywana informacja zwrotna umożliwia natychmiastową zmianę kierunku pracy i dostarczenie dokładnie tej funkcjonalności, która jest niezbędna z punktu widzenia Interesariuszy. Aby framework działał właściwie należy wdrożyć wszystkie jego elementy. Scrum jest oparty na następujących wartościach:

- odwaga,
- poszanowanie,
- zaangażowanie,
- skupienie,
- otwartość,
- zaufanie.

Scrum Master

Jedna z trzech ról w Scrum, odpowiedzialna za wdrażanie frameworka Scrum oraz wspierająca Zespół Deweloperski. Dba o proces, pilnuje by praca przebiegała sprawnie, usuwa przeszkody, chroni Zespół Deweloperski przed niepożądanym wpływem z zewnątrz i wspomaga go w samodoskonaleniu. W początkującym zespole ma rolę coacha, nauczyciela. Później powinien się wycofać, aby Zespół miał możliwość samemu zadbać o siebie. Scrum Master dba także o zrozumienie i promowanie idei Scrum w organizacji. Współpracuje z Product Ownerem i wspiera go w procesie zarządzania Backlogiem od strony technicznej. Pomaga w kontaktach z Interesariuszami. Z pojęciem Scrum Mastera wiąże się pojęcie Servant Leader (pl. Lider Służebny).

Self-organization (PL: Samoorganizacja)

W zarządzaniu: zasada zakładająca, że zespoły samodzielnie organizują swoją pracę. Samoorganizacja nie jest tożsama z samowolą i anarchią, odbywa się w ramach i zgodnie z określonymi w organizacji celami. W wypadku Scrum, Zespoły same wybierają najlepszą metodę swojej pracy oraz metodę realizacji powierzonych zadań. W razie potrzeby Zespoły mogą zasięgać porad u zewnętrznych ekspertów, ale nie są kierowane przez osoby spoza Zespołu.
Cechy charakteryzujące samoorganizującego się Zespołu:

- Wspólne podejmowanie zobowiązania odnośnie zakresu realizowanej pracy,
- Wybór najlepszego sposobu realizacji pracy,
- Skupienie się na celu,
- Współpraca przy realizowaniu zadań,
- Wzajemna, jak również samodzielna kontrola (Inspekcja),
- Wzajemne wspieranie się i motywowanie.

Servant Leader (PL: Lider Służebny)

Lider służący swojemu zespołowi i organizacji, odróżnia on kwestię przewodzenia i wydawania poleceń. Tych ostatnich Lider Służebny w zasadzie nie wydaje. Jego rolą jest przede wszystkim budowanie Zespołu i pomoc jego członkom w rozwoju, motywowanie ich, udzielanie wsparcia i usuwanie przeszkód uniemożliwiających osiągnięcie założonego celu.

SMART

Metoda wspomagająca właściwe skonstruowanie wykonalnego Celu Sprintu. W myśl tej metody cel powinien być:

- Specifc – specyficzny – co chcemy osiągnąć? (cel jest jasno sprecyzowany)
- Measurable – mierzalny – jakie warunki muszą być spełnione, aby cel został osiągnięty? (możliwe jest jednoznaczne określenie czy cel został osiągnięty)

- Achievable – osiągalny – czy posiadamy wszystkie niezbędne zasoby? (dostępne są wszystkie zasoby niezbędne do realizacji celu)
- Relevant – istotny – czy to co chcemy osiągnąć jest wartościowe i potrzebne? (powinien nieść za sobą konkretną i wymaganą wartość biznesową)
- Time-bound – określony w czasie – kiedy chcemy osiągnąć cel? (określony jest ostateczny termin osiągnięcia celu)

Sprint

Określonej długości okres pracy w trakcie którego powstaje ukończony i gotowy do wydania Przyrost produktu. W trakcie pracy nad produktem długość Sprintu pozostaje zazwyczaj niezmienna. Pojedynczy Sprint może trwać od tygodnia (krótsze są nieefektywne) do miesiąca (dłuższe powodują utratę podstawowej wartości Scrum jaką jest częsta informacja zwrotna od Interesariuszy). Na początku Sprintu (podczas Planowania) Zespół Deweloperski wybiera spośród zadań umieszczonych w Backlog'u Produktu te, które jest w stanie realizować i przenosi je do Backlog'u Sprintu. W trakcie Sprintu mają miejsce określone spotkania: na początku – Planning, codziennie – Daily, zależnie od potrzeb – Refinement i na końcu Review oraz Retrospective. Po zakończeniu Sprintu od razu rozpoczyna się kolejny Sprint. Efektem Sprintu jest Przyrost.

Sprint Backlog (PL: Backlog Sprintu, Rejestr Sprintu)

Jeden z trzech artefaktów Scrum'a. Lista zadań wybranych z Backlog'u Produktu przez Zespół Deweloperski w trakcie planowania oraz opracowany przez Zespół Deweloperski plan ich dostarczenia czyli zamiany w Przyrost. Obejmuje całą pracą jaką Zespół Deweloperski zamierza wykonać w czasie Sprintu. Backlog Sprintu jest własnością Zespołu Deweloperskiego i tylko jego członkowie mogą dokonywać w nim zmian. Nie jest sztywnym i zamkniętym tworem i podlega zmianom tak często jak jest to konieczne.

Sprint Goal (PL: Cel Sprintu)

Jeden z elementów tworzonych w czasie Planowania. Założenia jakie Zespół Scrum'owy planuje osiągnąć jako efekt pracy w danym Sprincie. Cel sprintu pomaga Zespołowi Deweloperskiemu zrozumieć, w jakim celu tworzy Przyrost i jaką wartość biznesową w nim zawrze. Ułatwia pracę zespołową, daje wskazówki co do możliwości innego sposobu realizacji zadania i priorytetów w Sprincie. Nie wszystkie zadania realizowane w Sprincie muszą mieć związek z Celem Sprintu.

Sprint Retrospective (PL: Retrospektywa Sprintu, Retro)

Jedno z 4 wydarzeń w Scrum odbywające się na koniec Sprintu, w którym uczestniczy cały Zespół Scrum'owy. Spotkanie prowadzące do usprawnienia i udoskonalenia procesu wytwarzania Przyrostu produktu. Ma ono na celu podsumować zakończony Sprint w odniesieniu do procesów, ludzi, relacji i narzędzi. Pozwala na uzyskanie zwrotnej informacji na temat stanu wdrażanych elementów Scrum'a i na ich podstawie wyciągnięcie wniosków pozwalających je ulepszyć. Zespół ma za zadanie przedyskutować to, co się wydarzyło i znaleźć dobre i słabe punkty, oraz zaplanować co i jak postara się poprawić w kolejnym Sprincie. Innymi słowy Zespół Scrum'owy przeprowadza Inspekcję swoich działań i opracowuje plan usprawnienia. Wydarzenie to jest ograniczone do maksymalnie 3 godzin.

Sprint Review (PL: Przegląd Sprintu)

Jedno z 4 wydarzeń w Scrum (przedostatnie spotkanie w Sprincie), w którym uczestniczy cały Zespół Scrum'owy oraz Interesariusze. W trakcie spotkania następuje Inspekcja Przyrostu oraz w razie potrzeby dostosowanie Backlog'u. Głównym celem jest zebranie informacji zwrotnych na temat przedstawionego rozwiązania dla produktu oraz uzyskanie wskazówek co do kolejnych pożądanych funkcjonalności. Na podstawie tych danych zespół określa kierunek dalszych prac nad produktem a Product Owner aktualizuje Backlog. Wydarzenie to jest ograniczone do maksymalnie 4 godzin.

Stakeholder (PL: Interesariusz)

Wszystkie osoby i podmioty będące odbiorcą końcowego produktu, najczęściej mianem tym określani są klienci, użytkownicy końcowi, management czyli wszyscy mający wpływ na to jak będzie wyglądał produkt, ci którzy go zamawiają i za niego płacą jak również ci, którzy będą go używać.

Story Point (SP; PL: Punkty Historyjek)

Abstrakcyjne jednostki używane do szacowania stopnia skomplikowania zadań, które można łatwo zsumować w celu oszacowania wielkość wszystkich zadań. Wartości przyjmowane przez Story Pointy to zmodyfikowany ciąg Fibonacciego (1, 2, 3, 5, 8, 13, 20, 40). Kolejne wartości są celowo niemal dwukrotnie większe, gdyż w szacowaniu chodzi o wartość przybliżoną. Nie mają przełożenia na czas czy inne wartości, określają jedynie rozmiar pracy. Zespół Deweloperski potrzebuje kilku sprintów, aby nauczyć się poprawnie szacować stopień trudności zadań i zrozumieć co oznacza dla niego trudność na poziomie np. 5. Ta wiedza jest nieprzekładalna na inny Zespół, w tym także ten sam Zespół po wymianie jednego czy dwóch deweloperów.

Technical debt (PL: Dług Technologiczny)

Przeszkody na drodze do osiągnięcia celu powstałe w wyniku szybkiego i nieoptymalnego zaspokajania potrzeb biznesowych. Stanowią ukryty koszt przyszłych prac czyli tego, co jest konieczne do zrobienia zanim możliwe będzie wykonanie pracy bieżącej. Przykładowo, jeżeli puszki po napojach zamiast do śmietnika wyrzucamy pod drzwi serwerowni, to gdy zajdzie potrzeba dostania się do tego pomieszczenia niezbędne będzie usunięcie wszystkich puszek. Wyrzucanie puszek pod drzwi jest znacznie prostsze niż do np. daleko położonego śmietnika, ale w ten sposób zaciągamy dług technologiczny, który kiedyś w przyszłości trzeba będzie spłacić.

Timebox (PL: Ramy Czasowe)

Ograniczenie czasu trwania wydarzenia, mające na celu motywację do sprawnego i efektywnego przeprowadzenia zaplanowanych działań i wyeliminowanie efektu opisywanego przez prawo Parkinsona – „praca rozszerza się tak, aby wypełnić czas dostępny na jej ukończenie". Wszystkie wydarzenia w Scrum'ie mają określone timebox'y :

- Planning – 8 godzin dla miesięcznego Sprintu, dla krótszych proporcjonalnie mniej
- Daily – 15 minut
- Review – 4 godziny dla miesięcznego Sprintu, dla krótszych proporcjonalnie mniej
- Retrospektywa – 3 godziny dla miesięcznego Sprintu, dla krótszych proporcjonalnie mniej
- Refinement – nie więcej niż 10% czasu dostępnego w Sprincie, nie wlicza się w to indywidualnej pracy PO nad Backlog'iem

User Story – (US; PL: Historie Użytkownika)

Popularna w Agile metoda zapisywania wymagań biznesowych, polegająca na opisaniu zamawianych funkcjonalności z perspektywy użytkownika końcowego.

Typowa postać US to:

- Jako… (nazwa określająca użytkownika, z naciskiem na jego rolę),
- Chcę… (wymaganie lub funkcja produktu),
- Aby… (określenie powodu lub celu wymagania, albo końcowego efektu jaki ma zostać osiągnięty w jego wyniku).

Taka forma zapisu wymagań zawierająca potrzebę biznesową ułatwia Zespołowi Deweloperskiego zrozumienie realnej potrzeby użytkownika i opracowanie sposobu jego realizacji.

Velocity (PL: Prędkość)

Średnia ilość pracy jaką Zespół Deweloperski wykonuje w trakcie jednego Sprintu. Liczone jest na koniec Sprintu przez zsumowanie jednostek w jakich wyceniane były zadania wzięte do realizacji w Sprincie.

Do średniej służącej wyliczeniu Prędkości Zespołu wliczane są wyłącznie zadania zakończone, czyli takie które spełniają Definition of Done.

Waterfall Model (PL: Model Kaskadowy)

Jedna z najpopularniejszych klasycznych metod wytwarzania oprogramowania. Opisana pierwszy raz przez Winstona W. Royce'a w 1970 r. Zakłada występowanie kolejno po sobie poszczególnych faz projektu:

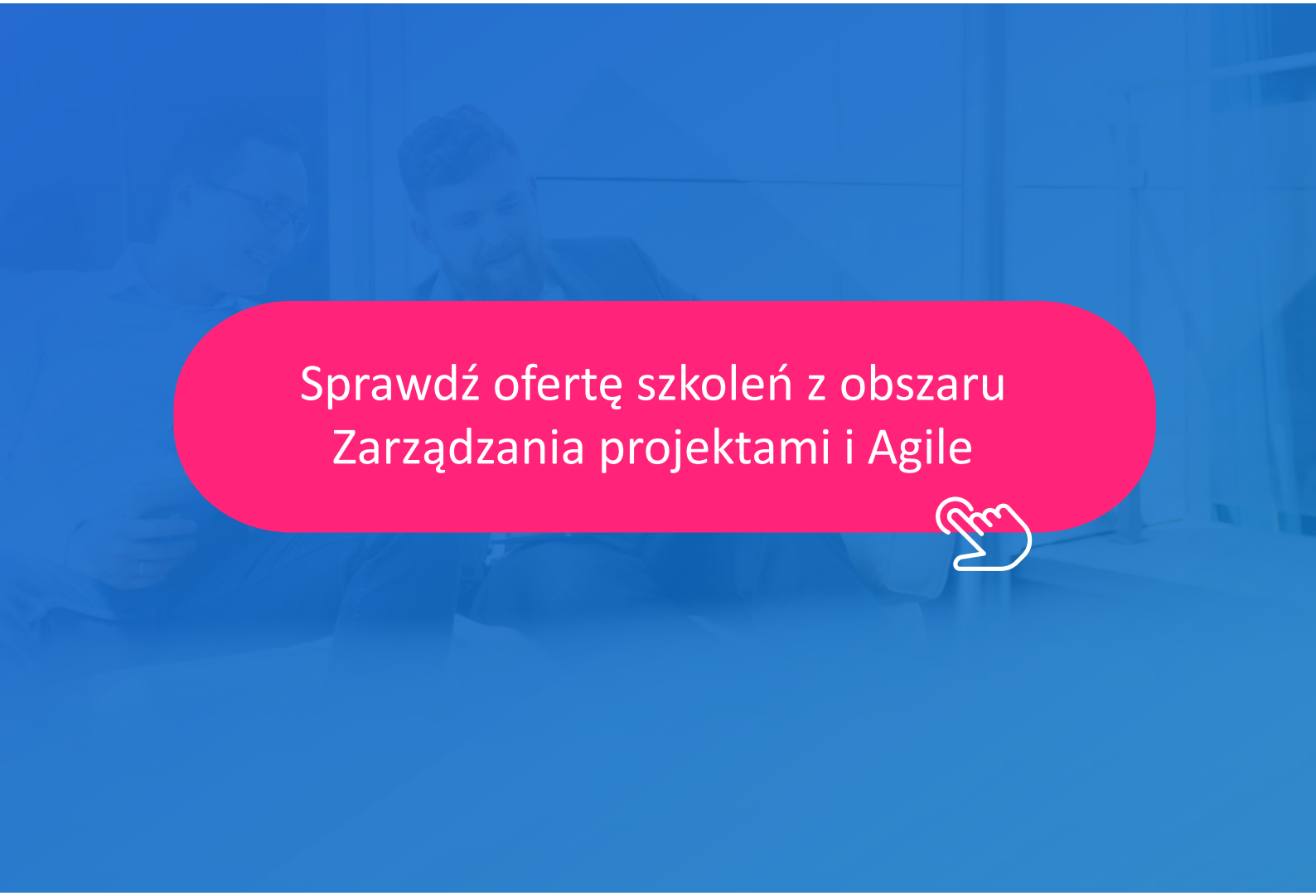
- planowanie,
- analiza,
- projekt,
- implementacja,
- testowanie,
- wdrożenie.

Przejście do następnej fazy następuje po zakończeniu prac w poprzedniej. Do kolejnego etapu pracy trafia komplet danych wytworzonych w poprzednim etapie na zasadzie przekazywania pałeczki w sztafecie. Minusem Waterfall'a jest całkowita niepodatność na zmiany wymagań i założeń początkowych na kolejnych etapach produkcji. W przypadku zmiany wymagań co do produktu końcowego projekt powinien wrócić do etapu, którego dotyczą zmiany i cały proces powinien być powtórzony od tego momentu. Przy często pojawiających się zmianach i konieczności każdorazowego wracania do wcześniejszego etapu, koszt projektu znacząco wzrasta i praktycznie w nieskończoność wydłuża się czas prac. Waterfall jest nieelastycznym modelem zarządzania, wykluczającym zaspokojenie zmiennych wymagań. Sprawdza się jednak dobrze w wypadku projektów, w których cały zakres prac jest znany od samego początku i nie podlega zmianom, oraz takich, które ze względu na swoją specyfikę nie mogą być budowane przyrostowo.
Agile charakteryzuje się różnorodnością metodyk, technik i możliwości, które wciąż są rozwijane i dodawane. Jeśli w powyższym słowniku nie udało ci się znaleźć odpowiedzi na nurtujące pytania, daj nam znać w komentarzu, chętnie pomożemy i wyjaśnimy wszelkie zawiłości Zwinnego Świata!

Autor:

Marek Konderski – Scrum Master, który po ponad 20 latach niemal zwinnej, iteracyjnej pracy w działach pokrewnych do IT, zdał sobie sprawę z tego co robi i oficjalnie przeszedł na agile'ową stronę mocy zaczynając pracę z developerami i próbując nawracać ich z Waterfalla na Scrum.

Justyna Michalak – Scrum Master z 6-letnim doświadczeniem w prowadzeniu projektów w tym 3,5 roku głównie we framework'u Scrum. Wykorzystuje zwinne techniki i metodyki zarówno przy pracy projektowej jak również na etapach negocjacji, ofertowania i późniejszego utrzymania.



BLOGERSII

Scrum – nowe reguły gry 2020

Autor: Rafał Kwatek

18 listopada 2020 opublikowana została nowa wersja Przewodnika po Scrumie (ang. The Scrum Guide): Przewodnik po Scrumie: opis reguł (ang. The Definitive Guide to Scrum: The Rules of the Game). Zmiana ta jest tyleż istotna, że od 10 stycznia 2021 wszystkie egzaminy prowadzone przez scrum.org (w tym Professional Scrum Master – PSM) będą bazowały wyłącznie na nowej wersji Przewodnika. Każda osoba przystępująca do egzaminu powinna zapoznać się z tymi zmianami. Warto również, aby osoby działające w zespołach pracujących z wykorzystaniem Scrum, przeczytały nową wersję Przewodnika po Scrumie i zastanowiły się, jakie korzyści może odnosić ich zespół przy implementacji tych zmian.

Scrum – co to jest?

Scrum to ramy postępowania (ang. framework) pozwalające dostarczać wartość poprzez adaptacyjne rozwiązywanie złożonych problemów, dla których nie ma prostych przepisów na znalezienie rozwiązania. Fundamentem Scruma jest empiryzm zakładający, że wiedza powstaje na podstawie doświadczeń (czasem nieudanych), a wszelkie decyzje powinny być podejmowane na podstawie dotychczasowych obserwacji. U podstaw Scrum leży też koncepcja lean koncentrująca się na optymalizacji pracy poprzez eliminacje marnotrawstw (ang. waste) oraz nakierowaniu działań i wysiłków na te obszary, które przyniosą największą wartość.

Scrum wykorzystuje iteracyjne i przyrostowe podejście do wytarzania produktów oraz filary empiryzmu, którymi są przejrzystość (ang. transparency), inspekcja (ang. inspection) i adaptacja (ang. adaptation). Efektywne wykorzystanie Scrum zakłada, że zespół realizujący prace będzie działa w oparciu o pięć wartości: Zaangażowanie, Skupienie, Otwartość, Szacunek i Odwaga (ang. Commitment, Focus, Openness, Respect, and Courage). Ramy postępowania opisane w Scrum składają się z 3 elementów: Scrum Teamu, Artefaktów (ang. Artefacts) tworzonych przez Team i Wydarzeń (ang. Events) określających komunikację i zasady współpracy.

Scrum został zaprezentowany po raz pierwszy przez jego autorów (Ken Schwaber i Jeff Sutherland) w roku 1995, jako podejście do wytwarzanie oprogramowania. Tworzenie nowych produktów IT jest złożonym problemem, gdzie zarówno technologia jak i zmieniające się otoczenie oraz wymagania biznesowe powodują, że zastosowanie prostych, dobrze opisanych sposobów postępowania często prowadzi do porażki. Wytwarzanie oprogramowania w oparciu o integracyjny i przyrostowy proces opisany w Scrum okazuje się przynosić dużo większą wartość niż działanie w tradycyjnych modelach. Dlatego też Scrum stał się tak popularny w środowisku IT. Warto jednak zauważyć, że Scrum jest coraz częściej stosowanych także w innych obszarach niż IT. Wiele organizacji może w różnych sposób stosować Scrum w zależność od kontekstu w jakim działają. Pewne podstawowe reguły opisane w Scrum są niezmienne, ale niektóre obszaru Scrum ewoluowały, aby opis zawarty w Przewodniku był jak najbardziej zrozumiały i precyzyjny oraz zgodny z doświadczeniami jakie zostały zidentyfikowane przez ostatnie lata.

Zmiany w wersji 2020 dotyczą głównie kilku obszarów:

- Ograniczenie Przewodnika do minimalnie wystarczających ram oraz jego uproszczenie
- Koncentracja na samozarządzającym się Scrum Team i usunięcie pojęcia Development Team
- Wprowadzenie pojęcia Cel Produktu (ang. Product Goal) oraz uporządkowanie i usystematyzowanie opisów pojęć Definicja Ukończenia (ang. Definition of Done) i Cel Sprintu (ang. Spring Goal)
- Zmiany w polskim tłumaczeniu

Scrum KISS (Keep It Simple Stupid)

Pierwsza rzecz, którą możemy zauważyć czytając nową wersję Scruma to fakt, że jest ona o kilka stron krótsza od wersji z 2017 roku. Wynika to z pewnych korekt stylistycznych i uproszczeń językowych, ale przede wszystkim ze zmian merytorycznych. Autorzy doszli do wniosku, że należy usunąć niektóre elementy opisane w poprzednich wersjach, gdyż są one zbyt nakazowe. Możemy je stosować, ale nie są one konieczne dla każdego zespołu. Takie rozwiązania dalej pozostają w środowisku Scrum, ale w publikacjach uzupełniających, a nie w samym Przewodniku. Przykładem takich zmian może być usunięcie trzech pytań z Daily Scrum (Co zrobiłem? Co będę robił? Jakie widzę problemy?). W nowej wersji Przewodnika usunięte zostały wszelkie pojęcia i odniesienia do sektora IT, które stanowiły pewną pozostałość po tym z jakiego obszaru Scrum się wywodził. Dzięki tym zmianom opis Scrum stał się jeszcze prostszy i bardziej uniwersalny.

Jeden zespół – wspólny cel

Podstawowym elementem Scrum jest Scrum Team. Tworzą go Scrum Master, Product Owner oraz Developerzy. Cały Scrum Team odpowiada za dostarczanie wartości poprzez rozwijanie produktu. Nowa wersja Przewodnika kładzie jeszcze większy nacisk na odpowiedzialność całego Zespołu Scrum. Dlatego też zdecydowano się usunąć nazwę Development Team, która sugerowała, że Developerzy działają jako odrębny podzespół niezależny od Scrum Mastera i Product Ownera. Nie jest to tylko zmiana teoretyczna. Jest to zaadresowanie problemu, który pojawia się w praktyce. Często zdarza się, że Product Owner, czy też Scrum Master nie jest postrzegany przez Developerów jako osoba z zespołu. Efektywne działanie wymaga synergii w pomiędzy całym Scrum Teamem. Tworzenie niepotrzebnych barier może powodować wyłącznie zmniejszenie jakości działania całego Scrum Teamu, a co za tym idzie dostarczanie gorszych, mniej wartościowych rozwiązań. W Scrum Teamie są różne role wynikające z kompetencji oraz zakresu odpowiedzialności, ale tylko wspólne działanie osób pełniących te role może sprawić, że wytwarzanie produktów będzie realizowane w sposób skuteczny i efektywny.

Z tego powodu w nowym Przewodniku położono nacisk na samozarządzanie Scrum Team w miejsce samoorganizacji Development Teamu. Samoorganizacja oznaczała, że Development Team sam określa kto i jak będzie wykonywał poszczególne zadania. Samozarządzanie jest szerszym pojęciem. Po pierwsze nie dotyczy one wyłącznie Development Teamu, ale całego Scrum Teamu. Po drugie do decyzji kto i jak wykona daną pracę dodany jest element określający kierunek działania Scrum Teamu, czyli zdefiniowanie nad czym będziemy jako zespół pracować.

Po co my to w ogóle robimy?

– Czemu tak biegacie z pustymi taczkami?
– Panie, tu tyle jest roboty, że nie ma czasu taczek załadować!

Ten stary i sucharowy dowcip oddaje coś z czym często boryka się wiele organizacji. Wszyscy tak mocno koncentrują się na efektywności wytwarzania produktu, że już nikt nie pamiętam w jakim celu go robimy i czy jest on w ogóle komuś potrzebny. W nowej wersji Przewodnika po Scrum ta kwestie została odpowiednio zaadresowana. Po pierwsze na Spring Planningu oprócz kwestii „co" i „jak" ma być robione w kolejnym Sprincie dodano trzeci temat odpowiadający na pytanie „po co".
Wprowadzono również pojęcie Celu Produktu, który opisuje długoterminowe zamierzenie, dla którego realizowany jest produkt. Cel Produktu jest powiązany z Product Backlogiem i staje się kompasem przy podejmowaniu decyzji odnośnie wszelkich zmian (np. piorytetyzacji) oraz wybieraniu elementów z Backlogu do realizacji. Cel Produktu jest zobowiązaniem Scrum Teamu powiązanym z Product Backlogiem, co powoduje, że każda praca zespołu jest nakierowana na realizację celu jaki jest stawiany przed produktem.

Dla pozostałych Artefaktów (ang. Artifacts) opisanych Scrum również określono zobowiązania. Dla Sprint Backlogu jest to to Cel Sprintu (ang. Sprint Goal), a dla Incrementu Definicja Ukończenia (ang. Definition of Done). Elementy te były opisane w poprzedniej wersji Przewodnika, ale dopiero w nowej wersji zostało to uporządkowane, a opisy precyzyjnie powiązane z poszczególnymi Artefaktami.

Do you Scrum in Polish?

Przewodnik po Scrum podobnie jak w poprzednich wersjach został również wydany w polskiej wersji językowej. Również tym razem autorzy tłumaczenia stanęli przed dylematem na ile tłumaczenia ma być wierne językowo, a na ile pozostawić pewne pojęcia w wersji angielskiej z uwagi na to, że takie ich stosowanie przyjęło się w praktyce. Tym razem zdecydowano się w polskiej wersji pozostawić wszystkie nazwy własne elementów Scruma po angielsku. Decyzja ta była założeniem narzuconym tłumaczom przez autorów Scrum. W efekcie nie ma już np. Właściciela Produktu (jest Product Owner), czy Rejestru Sprintu (jest Spring Backlog), czy Codziennego Scruma (jest Daily Scrum). Niemniej pojęcia opisujące filary empiryzm (przejrzystość, inspekcja, adaptacja), czy też wartości Scruma (Zaangażowanie, Skupienie, Otwartość, Szacunek, Odwaga) oraz zobowiązania Artefaktów (Cel Produktu, Cel Sprintu, Definicja Ukończenia) zostały pozostawione po polsku.

Scrum 2020 – plusy ujemne, czy dodatnie?

Na koniec chciałbym się pokusić o moją subiektywną ocenę nowej wersji Przewodnika po Scrum. Bardzo podoba mi się idea dodania Celu Produktu. Uważam, że bardzo porządkuje to sposób myślenia o produkcie i celu jego rozwoju. Jest to coś czego ewidentnie brakowało w poprzedniej wersji. Wyeliminowanie roli Development Teamu oraz położenie większego nacisku na samozarządzanie i pełną odpowiedzialność całego Scrum Team również jest dobrym pomysłem. Warto też docenić zmiany porządkujące. Poczynając na stylistyce, a kończąc na elementach merytorycznych (np. jasne umiejscowienia Celu Sprintu i Definicji Ukończenia).

W kwestii uproszczenia i ograniczenia Przewodnika mam trochę mieszane uczucia. Z jednej strony widzę dostrzegam korzyści wynikające z większej prostoty i czytelności. Z drugiej jednak strony brakuje mi paru elementów (np. 3 pytań zadawanych na codziennych spotkaniach), które w mojej ocenie można było spokojnie pozostawić, bo są one mocno zakorzenione w kulturze zespołów Scurm.

Nie do końca przekonuje mnie polskie tłumaczenie. W mojej ocenie niektóre pojęcia np. Właściciel Produktu, Rejestr Sprintu, czy Retrospektywa nie brzmią źle i często są używane w praktyce. Najbardziej razi mnie stosowanie nazwy Scrum Team zamiast Zespół Scrum, a już w szczególności w odmianie (np. Scrum Teamu, czy Scrum Teamowi). Dodatkowo pozostawienie polskich tłumaczeń pojęć, które nie są elementami Scruma np. Definicja Ukończenia, Cel Produktu, czy też samej nazwy Przewodnik po Scrumie, powoduje u mnie pewien dysonans, jeśli te pojęcia występują tuż obok nieprzetłumaczonych elementów (np. „Cel Produktu jest odzwierciedlony w Product Backlogu"). Nie jest to jednak krytyczna uwaga do tłumaczy, a raczej do założeń jakie zostały im z góry narzucone.

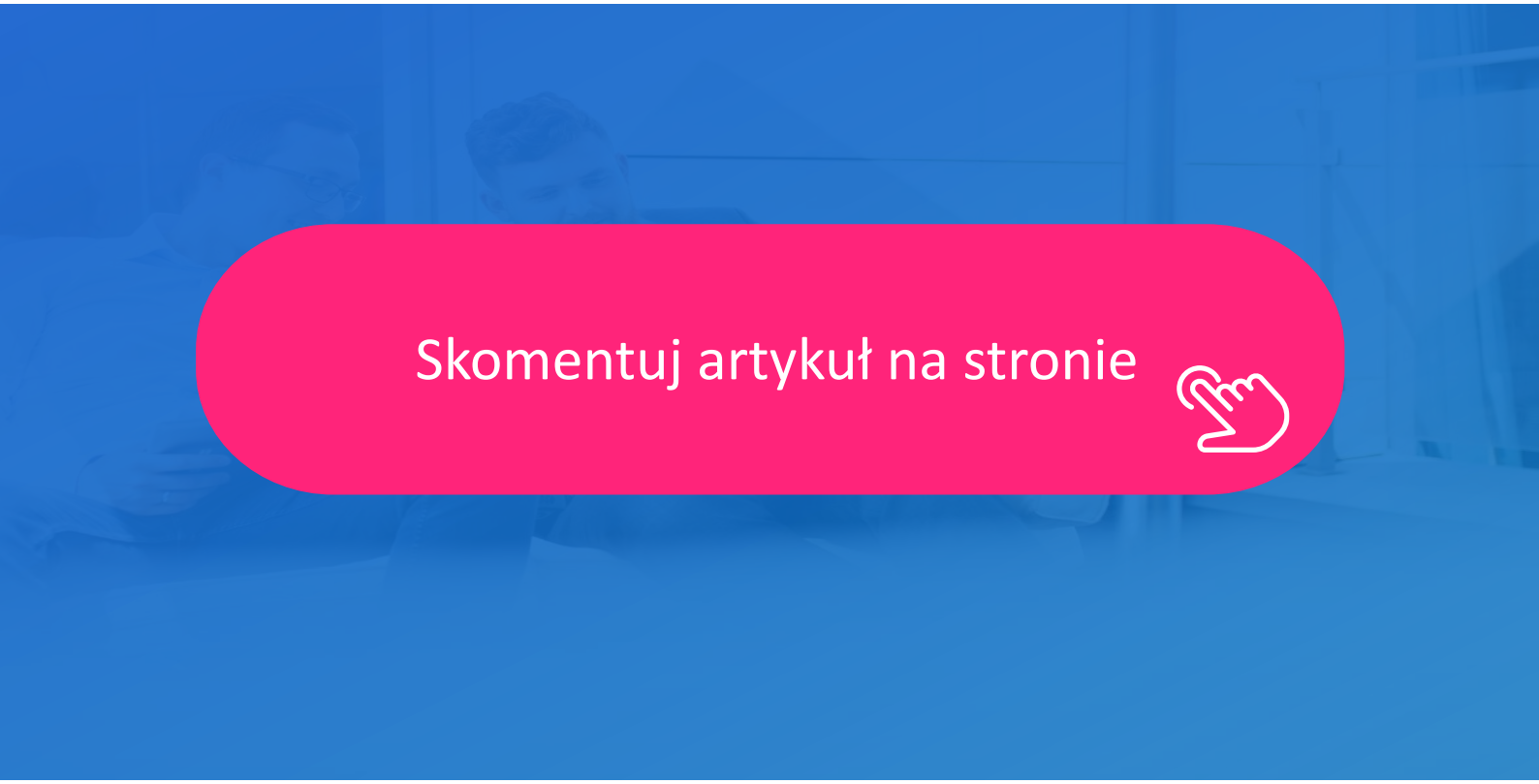
Reasumując bardzo pozytywnie oceniam kierunek i charakter zmian merytorycznych przedstawione w nowej wersji Przewodnika po Scrumie. Do kilku mniej istotnych zmian pozostaje jednak krytyczny (w szczególności do polskiego tłumaczenia). W mojej ocenie Przewodnik po Scrumie w wersji 2020 pozwoli każdemu jeszcze lepiej zrozumieć i stosować Scrum, co przełoży się na większą efektywność działania zespołów przy tworzeniu produktów dostarczających coraz większą wartość biznesową.

Źródła:

- https://www.scrumguides.org/index.html
- https://www.scrumguides.org/scrum-guide.html
- https://www.scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Polish.pdf

Autor:

Rafał Kwatek – Trener, konsultant i kierownik projektów. Posiada ponad 10-letnie doświadczenie praktyczne. Realizował projekty tradycyjne (waterfall), jak i zwinne (agile), w wielu różnych firmach, gdzie zajmował się budową usługowych organizacji IT, wdrażał metodyczne podejście do zarządzania projektami i architekturą korporacyjną, a także definiował, wdrażał, oceniał i optymalizował procesy biznesowe i IT. Ponadto dokonywał przeglądów i audytów organizacji, opracowywał struktury organizacyjne, budował katalogi usług, definiował umowy SLA/OLA. Uczestniczył także w projektach, obejmujących wytwarzanie oprogramowania oraz wdrożenia systemów i narzędzi IT. Posiada certyfikaty ITIL Expert, PRINCE2 Practitioner, PRINCE2 Agile Practitioner, Lean IT Foundation oraz Six Sigma Yellow Belt. Jest akredytowanym trenerem uprawnionym do prowadzenia szkoleń ITIL, PRINCE2 i PRINCE2 Agile. Posiada wieloletnie doświadczenie trenerskie – w prowadzonych przez niego szkoleniach i warsztatach wzięło udział kilka tysięcy osób.



BLOGERSII

Siedem grzechów głównych Scrum Mastera

Autor: Mateusz Szymula

W świecie IT, szczególnie wśród deweloperów, coraz częściej widzę pewnego rodzaju ,,szyderę" i niechęć wobec Scruma i Scrum Masterów. Mnie, jako Scrum Mastera, nieszczególnie to cieszy, ale staram się samemu z tym nie walczyć, tylko próbować zrozumieć drugą stronę i dopiero wtedy reagować. Scrum zaczyna być masowy, a masowość nie kojarzy mi się z jakością, może zatem warto wziąć na klatę krytykę i coś w sobie zmienić?

Niedawno stworzyłem prostą ankietę i poprosiłem trzy strony scrumowego tortu, tj. Deweloperów (po staremu zespoły deweloperskie), Product Ownerów oraz Scrum Masterów o odpowiedź na jedno, otwarte pytanie: Jakie błędy – zaniedbania – grzechy – popełnia/popełniał Scrum Master, z którym pracujesz/pracowałeś(aś)?

Moja ,,grupa badawcza" była dość skromna, bo liczyła 33 osoby (7 Product Ownerów, 7 Scrum Masterów 19 Developerów), badanie nie pretenduje więc do miana naukowego, ale wyciągnąłem bardzo ciekawy feedback.

Pycha i nieufność

Deweloperzy, Product Ownerzy i Scrum Masterzy spotkali się w swojej przeszłości z pychą tytułowych bohaterów. Pojawiły się zarzuty traktowania zespołu z góry, a także wynikające z braku zaufania narzucanie własnych rozwiązań, niepozwalanie na inicjatywę i samoorganizację. Jest to poważny zarzut i dość zasadnicze pogwałcenie scrumowych wartości. Scrum Master może oczywiście czasem proponować rozwiązania, które wynikać powinny z jego doświadczenia, ale pamiętajmy, że zespół powinien się samoorganizować, ba, nawet samozarządzać i powinniśmy mu nawet czasem umożliwić popełnić błąd. Dlaczego? Bo człowiek, tak i jak zespoły, uczy się właśnie na błędach.

Brak umiejętności miękkich

W opinii badanych zdarzają się Scrum Masterzy, którzy mają spore trudności w budowaniu relacji z ludźmi. Trudny charakter, słaba komunikatywność, brak empatii Scrum Mastera mogą doprowadzić czasem do kryzysu w zespole. Jeden z odpowiadających na zadane pytanie wskazał wprost przykład, kiedy Scrum Master nieświadomie (przez brak miękkiego skill setu) doprowadził do dużego napięcia i kryzysu pomiędzy Deweloperami a Product Ownerem. Nie czarujmy się, Scrum Masterzy, tak jak wszyscy ludzie, mają mniej lub więcej umiejętności społecznych, ale nie można nie zauważyć, że powinna to być rola, która jednoczy ludzi, a nie ich dzieli. Nie mam tu na myśli sztucznego eliminowania konfliktów, bo konflikty czasem są dobre i mogą być konstruktywne.

Uległość i brak asertywności

Grzech, przeciwstawny do tego, który został wymieniony jako pierwszy (czyli nie wszyscy SMowie popełniają te same grzechy 😉). W odpowiedziach pojawiały się postacie takich Scrum Masterów, którzy byli tak bardzo usłużni, tak bardzo dla zespołu, że dali sobie… wejść na głowę. I stali się popychadłem, a wraz z nim ich autorytet ,,lidera, który służy” zmieniał się w prawdziwego sługę, który nie jest już liderem. ,,SM staje się proxy i robi za sekretarkę” to kwintesencja tego grzechu. Są także zarzuty braku asertywności, pozwalanie by sprinty planowane były przez PO, a nie zespół. Jak taki Scrum Master ma potem bronić teamu przed nierealnym deadlinem, jak taki Scrum Master ma dać negatywny feedback leniom? Jak taki Scrum Master ma iść na urlop, skoro nikt nie poprowadzi daily czy planningu?

Błędy merytoryczne

Jest to grzech ciężki. Nie miejmy wątpliwości – takie grzechy psują cały rynek Scrum Masterów. Na szczęście nie było takich odpowiedzi wiele, ale pojawiły się i weźmy sobie to do serca. Nieznajomość Scruma, ustępowanie na każdym kroku, by było miło (scrumbuts), albo udawanie podczas DEMO (zakłamywanie rzeczywistości) absolutnie nie powinny się nam zdarzyć.

Brak zwinności

Dużo odpowiedzi pojawiło się pod tym zarzutem. ,,Za duży nacisk położony na proces scrumowy/ agile" był jednym z nich – robienie ze Scruma bożka, zamiast dążenia do autonomicznego zespołu. Niestety ludzie, z którymi jako Scrum Masterzy pracujemy, czasem widzą nas jako osoby, które wynoszą sam proces ponad korzyści, które mają z niego wynikać. Zdarza się, że sztywno trzymamy się procesu, skupiamy na mechanizmach, takich jak organizowanie kalendarza spotkań i głaskanie Jiry – to wszystko bez spojrzenia na wartość dla zespołu. Pomijamy przy tym najważniejsze – wsparcie zespołu np. w rozumieniu i realizacji celów sprintów. Dodatkowo, zespół wybija z rytmu nagminnie przeciągane spotkania, które do niczego nie prowadzą, nie mają agendy.

Apodyktyczność

Dość długa lista uwag pojawia się i tu. Nie brakowało przykładów, że Scrum Master za dużo zarządza, dyryguje zespołem, uprawia swego rodzaju micromanagement – jest koordynatorem, kontrolerem, przydziela zadania i ogranicza samoorganizację zespołu. Co więcej pełni de facto rolę Managera. Nie bądźmy jak Manager (nie mam nic do Managerów!) – nie sterujmy, pozwólmy zespołowi się rozwijać i zaufajmy mu.

Forsowanie własnych pomysłów, manipulowanie zespołem i podważanie kompetencji nie jest ok. Pamiętajmy, że Scrum zakłada płaską strukturę. Są momenty (szczególnie przy tworzeniu się zespołów), kiedy Scrum Master jest aktywniejszy i czasem więcej proponuje, ale trzeba rozróżnić zarządzanie od wspierania. Scrum Master wspiera.

Niedbalstwo

Dobry Scrum Master to zaangażowany Scrum Master. Jeśli np. brakuje follow-upu po retrospekcji, nie dziwmy się, że żaden Deweloper nie jest ,,fanem" tych spotkań. Jeśli jedyne, czym się zajmujemy to administrowanie Jirą i kalendarzem oraz spotkaniami, to rzućmy tę robotę. Niestety sporo uwag dotyczyło zaniedbywania zespołów przez Scrum Masterów: ,,było go za mało", brak dbałości o np. refinement, brak interwencji w trudnych momentach, brak wsparcia teamu i Product Ownera, jeśli tego potrzebują.

Podsumowanie

Po wymienieniu powyższych grzechów wyłania się dość negatywny obraz, ale trudno o optymistyczne przesłanie, jeśli mowa o grzechach. Dorzucę zatem do pieca i powiem, że tych grzechów jest jeszcze więcej! Zamiast jednak zwieszać głowę w zadumie proponuję potraktować te opinie i przykłady jako punkt wyjścia do swoistej auto-retrospektywy na temat błędów, które z pewnością czasem popełniamy wszyscy.

Co zrobić z tą wiedzą? Pomarudzimy sobie i tyle? Zostawiam to każdemu pod rozwagę. Jeśli mam coś sugerować Scrum Masterom (także sobie) to radzę, by mieć stale włączony czujnik słuchania i nakłaniania teamów do transparentności. Z pewnością pomoże to wyłapać i przeprocesować negatywny feedback, zanim trafi do nas poniewczasie. Zaufanie zespołu jest jak las – wolno rośnie, szybko płonie.

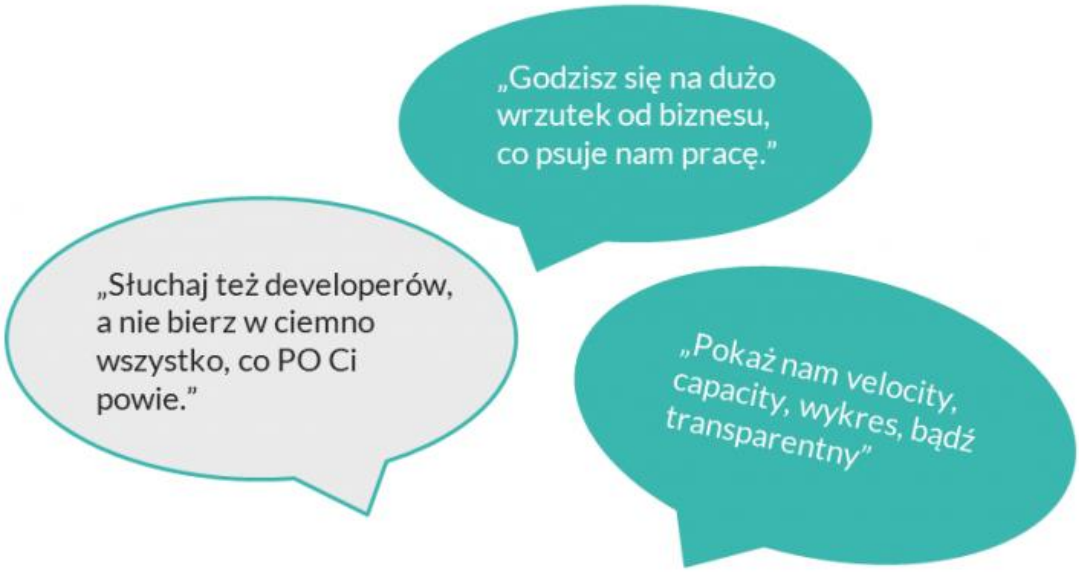
Co radzę osobom pracującym ze Scrum Masterami, tj. Deweloperom, Product Ownerom i innym osobom? Dbajcie o to, aby Scrum Master miał świadomość waszych potrzeb i bolączek. Lider, który służy, nie będzie w stanie robić tego efektywnie bez tej wiedzy.

Wszystkim teamom życzę wspaniałych Scrum Masterów! PS. Jakiś czas temu mówiłem o tym na pewnym meetupie. Zrobiliśmy krótkie ,,retro" po tym i to były najważniejsze błędy:

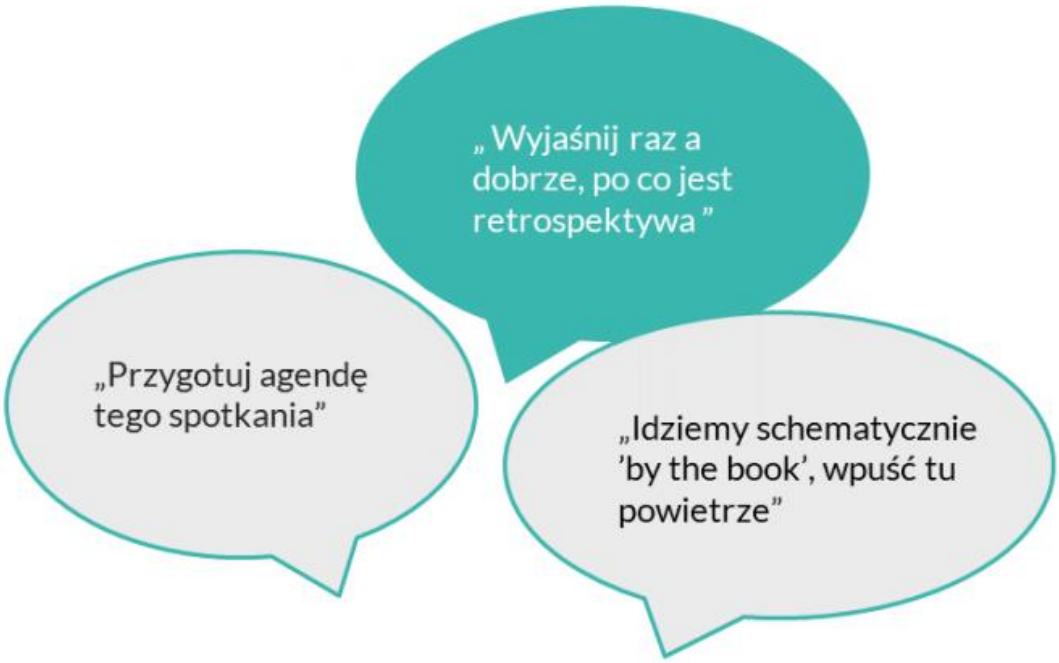
- Brak asertywności SM
- Proces ważniejszy dla SM niż pragmatyczne podejście
- SM nie rozumie ducha/idei/logiki Scruma i powtarza oraz wdraża bezmyślne schematy

I wyszliśmy z tego spotkania z wnioskami: ,,co powinniśmy mówić Scrum Masterom otwarcie”.

- Zarzut: brak asertywności:

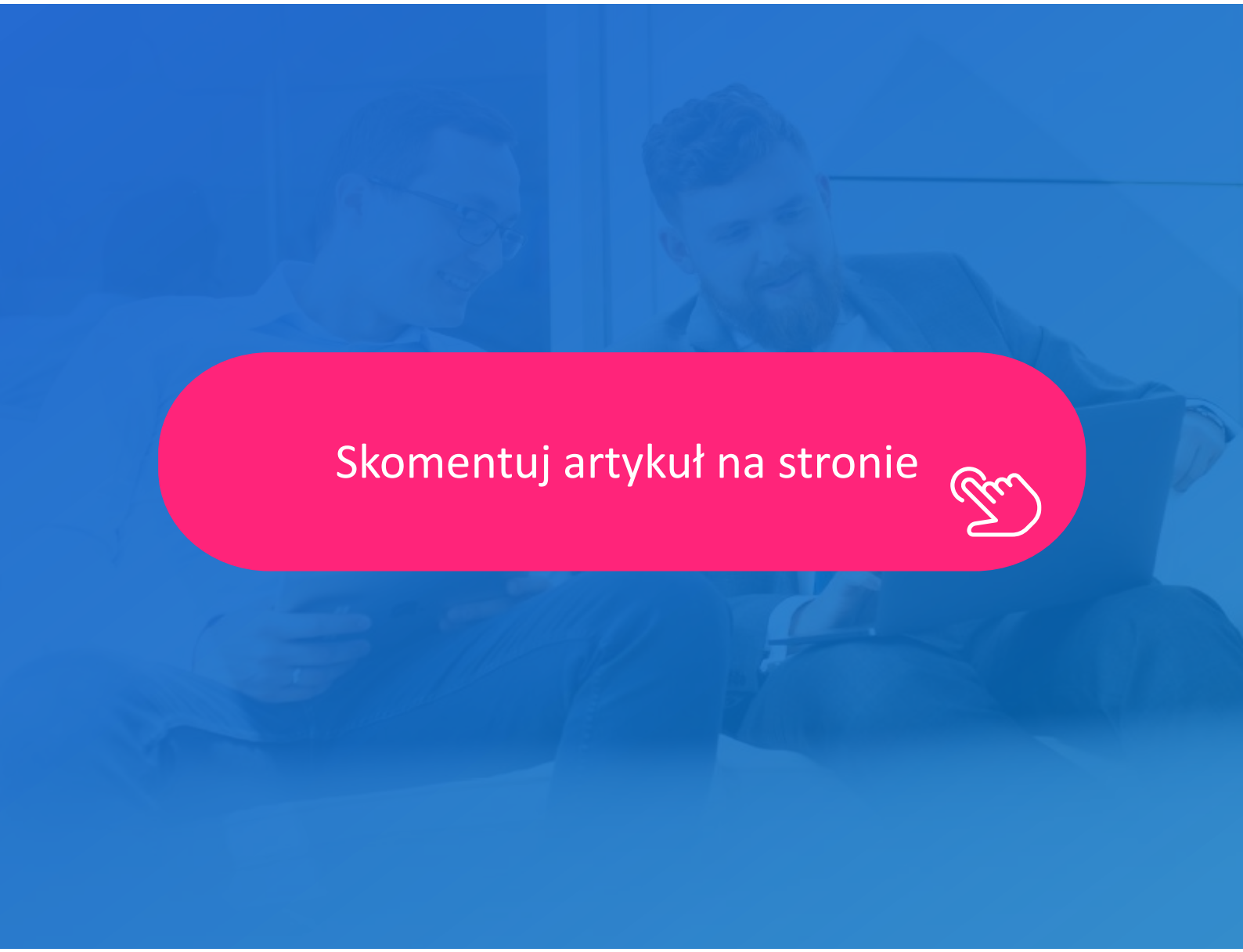


- Zarzut: proces ważniejszy niż pragmatyczne podejście & brak zrozumienia ducha/idei/logiki Scruma i powtarza oraz wdraża bezmyślne schematy:



Autor:

Mateusz Szymula – Scrum Master z ponad 4 letnim doświadczeniem i deweloperskimi epizodami oraz trener. W Sii i CC Agile & Atlassian od ponad roku pracuje z zespołami dla międzynarodowej grupy mediowej. Stara się bratać świat biznesu i inżynierów, by zbliżały się do jedności. Zwolennik podejścia Devops. Prywatnie mąż i tata, meloman i amator wina.



SZKOLENIA SII

Szkolenia „Zostań Specjalistą IT” – zdobądź niezbędną wiedzę, aby rozpocząć pracę w IT

Zastanawiasz się nad pracą w branży IT? Chciałbyś się przekwalifikować, ale nie wiesz od czego zacząć? W Sii Polska już od kilku lat udowadniamy, że wiedzę niezbędną do rozpoczęcia kariery w IT może zdobyć każdy, niezależnie od aktualnego doświadczenia czy wykształcenia. Z ponad 50 zrealizowanymi szkoleniami i 500 przeszkolonymi osobami, z sukcesem uruchamiamy kolejne edycje szkoleń z serii „Zostań Specjalistą IT”. Sprawdź, jak może rozpocząć się Twoja kariera w tej branży.

Które stanowisko będzie dla mnie najlepsze?

Czy jesteś skrupulatny i spostrzegawczy?

To dwie cechy dobrego testera, który podczas swej pracy musi wykazać się dokładnością i systematycznością, ponieważ jego celem jest przetestowanie aplikacji, aby wskazać jej wszystkie mocne i słabe strony. Osoba na tym stanowisku współpracuje zarówno z programistami, jak i z klientem, tworząc dla niego scenariusze testowe aplikacji.

A może myślisz nieszablonowo i lubisz zagadnienia techniczne?

Developer nie zajmuję się tylko programowaniem. Osoba na stanowisku developera wdraża nowe funkcjonalności i tworzy koncepcje ich działania. Podczas swojej pracy musi wykazać się nie tylko zdolnościami technicznymi, ale także samodzielnym planowaniem, by zaprezentowane funkcjonalności jak najlepiej spełniły wymagania klienta.

Czy łatwo rozwiązujesz problemy i wyróżniasz komunikatywnością?

Każdy dobry analityk biznesowy charakteryzuje się wysokimi umiejętnościami personalnymi, z uwagi na pośredniczenie w komunikacji pomiędzy biznesem a działem IT.
Analityk „tłumaczy" z języka biznesowego na techniczny opisując rzeczywistość z wykorzystaniem odpowiednich modeli oraz notacji, do czego potrzebna jest zarówno odpowiednia wiedza, jak i umiejętności miękkie.

Lubisz uczyć oraz pomagać innym usprawnić swoją pracę?

To właśnie Scrum Master pomaga zespołowi uporać się z problemami oraz uczy i rozwija go, aby ten działał jak najbardziej wydajnie. Rolą Scrum Mastera jest nieustanne bycie wsparciem dla zespołu, zatem najważniejsze są umiejętności miękkie, a także znajomość metodyk zwinnych oraz cyklu wytwarzania oprogramowania.

Jak zdobyć kompetencje?

Jak widać, pracę w IT znajdą nie tylko osoby posiadające techniczną wiedzę. Istnieje wiele ścieżek rozwoju, w których odnajdą się zarówno osoby o predyspozycjach technicznych, jak i interpersonalnych. Wzięliśmy to pod uwagę przygotowując nasz cykl szkoleń „Zostań Specjalistą IT", który pozwala zdobyć wiedzę oraz umiejętności niezbędne do rozpoczęcia kariery w IT.

Szkolenia adresowane są zarówno do osób początkujących:

- które nie posiadają doświadczenia ani wykształcenia IT,
- studentów, absolwentów różnych kierunków studiów,
- które swoją przyszłość chcą związać z zawodem testera, developera, analityka biznesowego, czy Scrum Mastera.

jak również do osób, które już pracują w zawodzie i chcą:

- ugruntować swoją wiedzę,
- zdobyć certyfikat potwierdzający kompetencje,
- nabrać nowych umiejętności praktycznych oraz poszerzyć swoją wiedzę.
- Odpowiednią dla siebie ścieżkę rozwoju znajdą tu zarówno osoby techniczne (tester, developer), jak i te, które w swojej pracy chcą skupić się na relacjach i pracy z ludźmi (analityk biznesowy, Scrum Master).

Dlaczego warto szkolić się z Sii?

Oferujemy kursy przygotowujące do pracy na stanowisku młodszego specjalisty IT w różnych obszarach. Szkolimy przyszłych testerów (manualnych i automatyzujących), developerów (Java, Python i Front-end), a od niedawna także analityków biznesowych oraz Scrum Masterów.

Naszymi trenerami są doświadczeni specjaliści – nasi pracownicy, którzy na co dzień realizują projekty dla polskich i zagranicznych klientów.

Realizujemy kursy zarówno w formule stacjonarnej, jak i online, która staje się coraz popularniejsza. Dzięki temu również osoby mieszkające poza lokalizacjami, w których organizujemy kursy stacjonarne, mogą uczestniczyć w naszych szkoleniach. Kursy online zwiększają też nasze bezpieczeństwo w dobie pandemii. Dodatkowo pozwalają przygotować się do pracy w online'owym i rozproszonym zespole, co w dzisiejszych czasach staje się coraz częstszą praktyką.

Czasem wśród naszych kursantów poszukujemy nowych pracowników. Aktualnie już ponad 50 przeszkolonych przez nas osób znalazło pracę w Sii.

Do tej pory zrealizowaliśmy już ponad 50 kursów z cyklu „Zostań Specjalistą IT”. Najlepszym dowodem na to, że robimy to dobrze, są opinie uczestników naszych szkoleń:

„Super podejście do każdej osoby, spokojnie wytłumaczone każde zadanie czasami nawet kilka razy. Naprawdę super szkolenie.”
Bartosz Wójcik, Zostań Testerem Online

„Cudowna atmosfera. Po samodzielnym przeczytaniu sylabusie miałam mętlik w głowie a po 2 dniach z trenerką wszystko było jasne. Bardzo dobrze przygotowany trener, pomagał i podchodził indywidualnie do każdego.”
Anna Królak, Zostań Testerem – Wrocław

„Żadnych zastrzeżeń, a wręcz przeciwnie! Szkolenie bardzo mi się podobało i żałuję trochę, że tak krótko trwało. Prowadzący bardzo dobry, widać, że podchodzi z pasją do tego co robi. Mam nadzieję, że po ukończeniu szkolenia będę miał szansę na znalezienie pracy w zawodzie programisty.”
Paweł Dąbrowski, Zostań Developerem Java