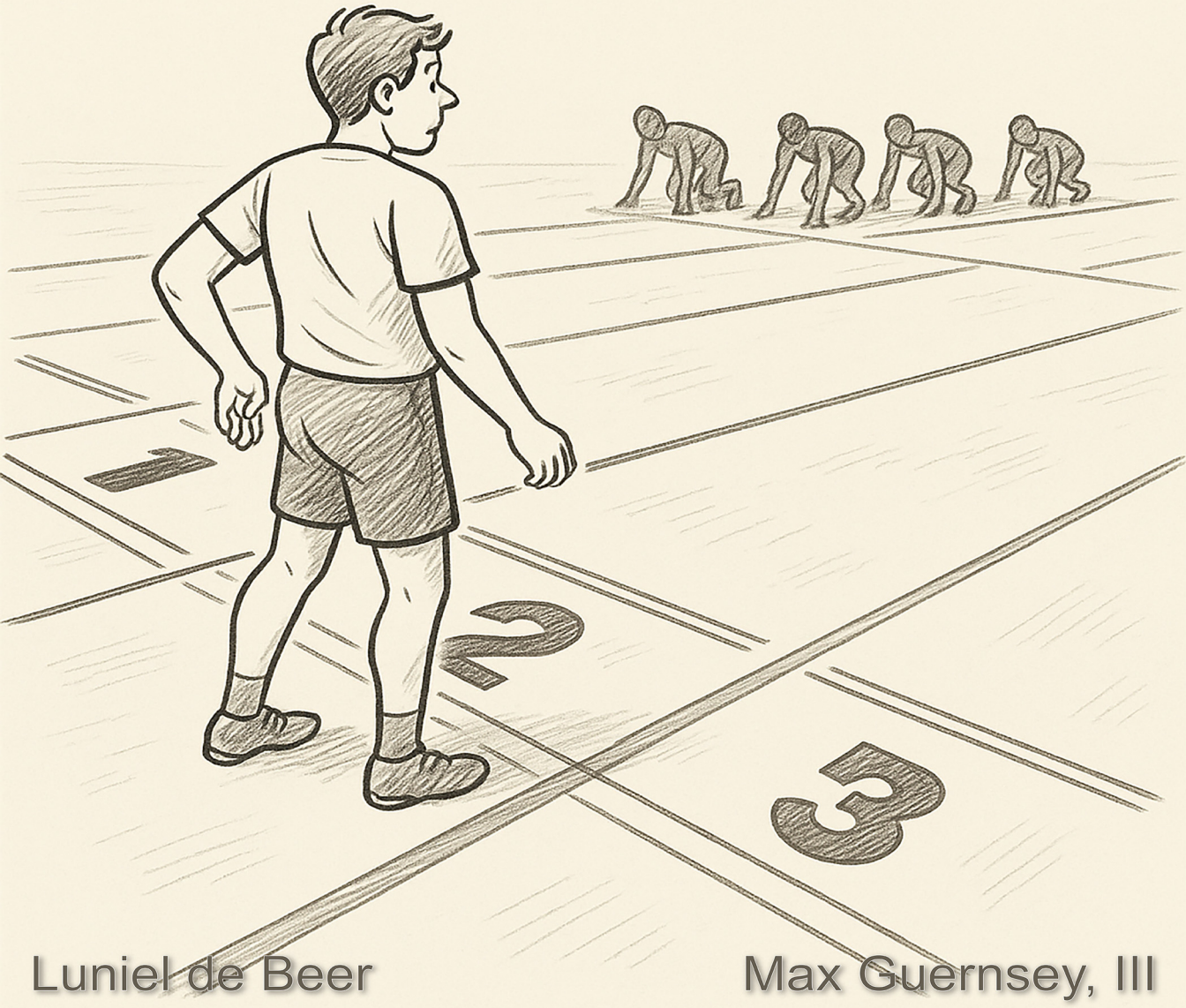
READY

WHY MOST SOFTWARE PROJECTS FAIL AND HOW TO FIX IT



Luniel de Beer

Max Guernsey, III

Polskie Wydanie

Tweetnij o tej książce!

Pomóż Luniel de Beer i Max Guernsey, III, rozpowszechniając informację o tej książce na Twitterze!

Proponowany tweet dla tej książki:

Właśnie kupiłem Ready — książkę dla liderów ds. oprogramowania i zespołów, którzy chcą wyeliminować przeróbki, przeniesienia i brak synchronizacji w dostarczaniu oprogramowania. #CodeReady

Proponowany hashtag dla tej książki to #CodeReady.

Sprawdź, co inni mówią o książce, klikając ten link, aby wyszukać hashtag na Twitterze:

#CodeReady

Ready (Polskie Wydanie)

Dlaczego większość projektów programistycznych kończy się niepowodzeniem i jak temu zaradzić

Luniel de Beer i Max Guernsey, III

Ta książka jest dostępna pod adresem https://leanpub.com/ready-pl

Ta wersja została opublikowana 2025-10-22

To jest książka Leanpub. Leanpub umożliwia autorom i wydawcom korzystanie z procesu Lean Publishing. Lean Publishing to proces publikowania książki w trakcie jej powstawania, wykorzystujący proste narzędzia i wiele iteracji, aby otrzymać informacje zwrotne od czytelników, wprowadzać zmiany, aż do uzyskania właściwej książki i zbudowania zainteresowania po jej ukończeniu.

Pamięci Johanna van Aardta, który dostrzegł moją pasję, wprowadził mnie w świat prawdziwego programowania i pomógł mi znaleźć nową drogę do domu. Oraz moim rodzicom, których niezachwiane wsparcie—od mojego pierwszego klienta po mój pierwszy dom w USA—umożliwiło to wszystko.

—Luniel

Dla mojej rodziny, z którą wschodzi i zachodzi słońce.

—Max

Spis treści

O tej książce

Ready to książka dla wszystkich zaangażowanych w rozwój oprogramowania, którzy mają dość **niedostarczania**, **chronicznego przerabiania** i **niejasnych wymagań**.

Być może już próbowałeś inwestować w umiejętności wykonawcze zespołu, ulepszać wdrożenie swojego frameworka procesowego lub odnawiać kod, a mimo to nadal potrzebujesz dalszych ulepszeń.

Dzieje się tak, ponieważ głównym ograniczeniem dla większości zespołów programistycznych nie są umiejętności zespołu, lecz dojrzałość wymagań. **Nawet dojrzałe zespoły z odpowiednimi umiejętnościami nadal mają trudności**, gdy pracują z niedojrzałymi wymaganiami.

Ready wprowadza RMF (Requirements Maturation Flow), praktyczne i głęboko ustrukturyzowane podejście do zharmonizowania Produktu i Inżynierii bez konieczności zastępowania istniejącego procesu.

Niezależnie od tego, czy korzystasz ze Scruma, Kanbana czy czegoś własnego, RMF pomoże ci **ustabilizować zakres**, **wyeliminować przenoszenie zadań** i **dostarczać to, co naprawdę ma znaczenie**.

Jeśli twoje zespoły czują się zatrzymane na krawędzi stanu "prawie gotowe", ta książka pokaże ci, jak **przerwać ten cykl** i **na dobre** odblokować twój zespół (lub zespoły).

Dla kogo jest ta książka

Ta książka jest dosłownie dla każdego zaangażowanego w rozwój oprogramowania. Od inżynierów po menedżerów produktu i od pojedynczych współpracowników po kadrę kierowniczą.

Ta książka jest dla ciebie, jeśli jesteś zaangażowany w rozwój oprogramowania i zauważyłeś, że zespół, z którym lub w którym pracujesz, ma jeden lub więcej z następujących problemów:

- Praca często przenosi się z jednej iteracji na następną
- Zespoły implementacyjne czują, że próbują trafić w ruchome cele
- Zadania pozostają otwarte zbyt długo
- Zadania są oznaczane jako zakończone, ale tak naprawdę nie są gotowe
- Wykonana praca nie jest zgodna z oczekiwaniami
- Praca regularnie generuje dużą liczbę defektów
- Duże ilości pracy muszą być regularnie przerabiane

Jeśli którykolwiek z tych problemów brzmi znajomo, *Ready* może pomóc.

Jak korzystać z tej książki

Ta książka została zaprojektowana tak, by być praktyczna. To nie jest teoretyczny traktat ani prezentacja strategiczna - to praktyczny podręcznik do wdrażania RMF (Requirements Maturation Flow), oparty na rzeczywistej pracy z klientami i przetestowany w warunkach rzeczywistej presji na dostarczanie rezultatów.

Rozdziały są napisane w kolejności, ale sam RMF jest modułowy. Składa się z trzech podstawowych praktyk:

- RMF 1: Współpraca dla wspólnego zrozumienia
- RMF 2: Kontrolowanie ukończenia pracy przy użyciu Dedykowanych Definicji Ukończenia
- RMF 3: Kontrolowanie wdrożenia przy użyciu Dedykowanych Definicji Gotowości

Każda część, czyli tak zwany "Nawyk", funkcjonuje samodzielnie, ale wszystkie wzajemnie się uzupełniają. Książka została zaprojektowana tak, aby pomóc w ich przyswajaniu pojedynczo, we właściwej kolejności. Ta struktura odzwierciedla zalecany przez nas sposób adoptowania RMF w praktyce - każdy Nawyk jest wprowadzany dopiero po tym, jak poprzedni działa prawidłowo.

Pozwala to uniknąć przytłoczenia zespołów i daje każdej zmianie największą szansę na utrwalenie. Dowiesz się więcej o tym, jak to zrobić, zaczynając od Rozdział 9.

Jeśli szukasz pomocy - czy to porady, coachingu, czy kogoś do rozmowy z zespołem kierowniczym - śmiało skontaktuj się z nami bezpośrednio.

A jeśli szukasz formalnego wsparcia we wdrażaniu RMF, Producore oferuje pełną serię Programów zaprojektowanych tak, aby przeprowadzić przez proces adopcji krok po kroku. Więcej informacji znajdziesz na stronie https://ready-book.link/rmf.

O Autorach

Luniel de Beer jest twórcą Requirements Maturation Flow (RMF), praktycznego systemu naprawiającego luki między intencjami produktowymi a realizacją inżynieryjną. Ma ponad 15 lat doświadczenia w prowadzeniu transformacji Agile, łączeniu produktu z inżynierią i pomaganiu zespołom w dostarczaniu rozwiązań z jasnością i pewnością.

Luniel jest również twórcą systemu Zarządzania Możliwościami Producore, skalowalnego podejścia do modelowania możliwości produktowych z możliwością śledzenia. Stworzył koncepcję PKB-Driven Development (PKBDD), systemu kontroli wersji do zarządzania trwałymi wymaganiami produktowymi. Narzędzia te stanowią część większego frameworka dostarczania rozwiniętego w Producore.

Max Guernsey, III jest architektem oprogramowania, edukatorem i współzałożycielem Producore, firmy konsultingowej zajmującej się naprawianiem niepowodzeń w dostarczaniu poprzez rygor strukturalny i techniczny. Z ponad dwudziestoletnim doświadczeniem w projektowaniu obiektowym, refaktoryzacji, programowaniu sterowanym testami i wzorcach projektowych, zarówno dostarczał systemy o krytycznym znaczeniu, jak i trenował zespoły inżynieryjne na dużą skalę. Jego praca łączy głębokie praktyki techniczne z transformacją behawioralną i procesową, aby pomóc organizacjom osiągnąć trwałą doskonałość w dostarczaniu.

Max wniósł znaczący wkład w PKBDD i kierował rozwojem podejścia Producore do Rozwoju Sterowanego Zachowaniem (BDD) dzięki swojej głębokiej wiedzy w zakresie specyfikacji behawioralnej.

Razem ich praca łączy przejrzystość, możliwość śledzenia i bramkowanie w spójny system dostarczania oprogramowania, który skaluje się od praktyki zespołowej do zdolności organizacyjnej.

Przedmowa

Nota do liderów inżynierii

Jeśli jesteś starszym liderem w organizacji inżynieryjnej, prawdopodobnie nie brakuje Ci wysiłku, dyscypliny ani mądrych ludzi. A jednak, z jakiegoś powodu, projekty wciąż się zatrzymują. Cele się wymykają. Oczekiwania nie są spełniane. Nie dlatego, że Twoje zespoły są leniwe–ale dlatego, że coś fundamentalnego jest nie tak w sposobie, w jaki praca jest definiowana, kształtowana i dostarczana.

Ta książka nie jest przewodnikiem po przywództwie. To narzędzie do diagnostyki strukturalnej. Pokazuje, co naprawdę dzieje się wewnątrz Twoich zespołów–dlaczego "prawie gotowe" ciągle zamienia się w "niegotowe" i dlaczego lokalny postęp tak rzadko przekłada się na strategiczne rezultaty.

Możliwe, że nie zobaczysz *siebie* na tych stronach. Ale jeśli Twoje zespoły nie mogą dostarczyć tego, czego potrzebujesz, zobaczysz *je*. A kiedy to się stanie, wreszcie będziesz mieć język–i system–żeby to naprawić.

Od Luniela

Przede wszystkim, ta książka nie byłaby możliwa bez Maxa, którego zdolność do przebijania się przez mgłę i plewy oraz destylowania idei do jej esencji absolutnie mnie przerasta.

Jak tu dotarliśmy?

Gdy patrzę wstecz, myślę, że to dlatego, że zawsze chciałem zrozumieć, jak rzeczy naprawdę działają. Czy chodziło o religię, odżywianie czy rozwój oprogramowania, ciągle napotykałem ten sam problem: powierzchowne odpowiedzi, które nie sprawdzały się pod presją. Więc kopałem głębiej–pytając nie tylko o to, co robimy, ale dlaczego i czego brakuje, gdy to nie działa.

Jedna z najwcześniejszych rys w systemie pojawiła się w roli, w której nosiłem trzy kapelusze: Scrum Mastera, Product Ownera i Development Managera (!!) dla zespołu dostarczającego usługi danych w znanej firmie technologicznej. Robiliśmy to, co Scrum

nakazywał–krótkie Sprinty, historyjki w backlogu, planowanie w pół dnia–ale za każdym razem, gdy zaczynaliśmy nowy Sprint, napotykaliśmy na tarcia. Zespół nie do końca rozumiał problem, musieliśmy wracać do wymagań i rewidować je w trakcie Sprintu, pojawiały się możliwe do uniknięcia zależności, które nas opóźniały, a kluczowe kroki były pomijane.

Więc zacząłem robić coś innego. Zbierałem zespół i interesariuszy w jednym pokoju dla każdej historyjki, szczegółowo omawialiśmy problem, wspólnie burzyliśmy mózgi nad rozwiązaniem i dopiero wtedy pisałem historyjkę. Planowanie Sprintu skurczyło się do godziny, a nasze sukcesy w dostarczaniu wystrzeliły w górę.

Powoli zacząłem zdawać sobie sprawę, że sukces nie bierze się z cięższej pracy w trakcie Sprintu. Bierze się ze struktury, którą tworzysz *zanim* on się rozpocznie.

Później, po wysłuchaniu wystąpienia Jeffa Sutherlanda na temat „Definicji Gotowości", wiedziałem, że jest w tym coś wartościowego – ale to nie wystarczało. Moje doświadczenie z wymaganiami, UX, UI, badaniami, a później z BDD pokazało mi, że różne elementy pracy wymagają różnych rodzajów gotowości. Niektóre potrzebują specyfikacji zachowania. Niektóre wymagają dostępu do systemu. Niektóre potrzebują pełnego śledzenia możliwości.

A wszystkie one wymagają wspólnego zrozumienia, które jest rzeczywiście potwierdzone – nie zakładane.

Pracując z kolejnymi zespołami, wszędzie dostrzegałem ten sam wzorzec: brakujące kroki, niespełnione zależności, zespoły dające z siebie wszystko, ale ciągle walczące z problemami, których można było uniknąć. Nawet świetne zespoły miały trudności – nie dlatego, że były słabe, ale dlatego, że brakowało im struktury, która czyniłaby gotowość jednoznaczną.

Rezultatem całego tego procesu uczenia się, iteracji i frustracji jest ustrukturyzowany system zarządzania gotowością.

O tym właśnie jest ta książka.

Mam nadzieję, że da ci jasność co do tego, gdzie leżą prawdziwe problemy i jak je naprawić. Mam nadzieję, że da ci język do obrony praktyk, które mogą wydawać się „dodatkowe", ale są w rzeczywistości niezbędne. A przede wszystkim mam nadzieję, że pomoże zespołom dostarczać produkty z mniejszym stresem, mniejszą liczbą niespodzianek i o wiele lepszymi rezultatami.

Jeśli zrobimy to dobrze, zaoszczędzimy branży miliardy dolarów.

Ale co ważniejsze, przywrócimy ludziom ich zdrowy rozsądek.

Od Maxa

Pracowałem nad tym problemem z różnych perspektyw przez dziesięciolecia, ale mój postęp był ograniczony, dopóki nie poznałem Luniela.

Działo się tak dlatego, że zanim go poznałem, podchodziłem do problemu zasadniczo jako do problemu technicznego. Skupiałem się na pomocy zespołom w przyjmowaniu takich praktyk jak Programowanie sterowane testami (TDD), refaktoryzacja, zaawansowane projektowanie oprogramowania, a później Programowanie sterowane testami akceptacyjnymi (ATDD) czy Programowanie sterowane zachowaniem (BDD).

W większości tych przypadków problem poruszany w tej książce był traktowany jako szczegół implementacyjny przy wprowadzaniu tych praktyk technicznych.

Nie oznacza to, że nie cenię już praktyk technicznych. Nadal uważam, że są one głęboko istotne, ale nie odnoszą się bezpośrednio do problemu gotowości w tworzeniu oprogramowania. Zamiast tego *ujawniają* ten problem, a ludzie nakładają łatkę na swój proces, aby rozwiązać go „w stopniu wystarczającym“ do wsparcia praktyk technicznych, które próbują wdrożyć.

Chcę też odnieść się do kwestii tego, komu ta książka może pomóc. Krótka odpowiedź brzmi „prawdopodobnie prawie każdemu w branży tworzenia oprogramowania“, ale prawdziwa odpowiedź zawiera niuanse, które pomagają dopasować ją do różnych środowisk bez zmiany podstawowego znaczenia.

Istnieją zespoły, które potrzebują rozwiązania przedstawionego w tej książce. Z jednym z nich, w ocenzurowanej wersji, spotkasz się w Rozdziale 1.

Są też zespoły, które nie **potrzebują** ściśle systemu takiego jak ten, który proponujemy, ale mimo to mogłyby na nim skorzystać.

Najlepszy zespół, z jakim kiedykolwiek pracowałem – z łatwością całą jednostkę odchylenia standardowego powyżej **następnego najlepszego zespołu**, jeśli nie dwie – znajdował się na uboczu Oregonu Środkowego. Byli tak wysokowydajni, że mogli przezwyciężyć brak takiego systemu samą siłą kompetencji. Jednak mój ówczesny menedżer, Tom Barreras, powiedział mi kiedyś coś w rodzaju “Zauważyłem, że nasze historyjki idą lepiej, gdy poświęcimy trochę czasu na rozmowę o testach z wyprzedzeniem.”

To znowu było coś, na co w tamtym czasie patrzyłem przez pryzmat rozwoju testów i technicznego wykonania, ale teraz wiem, że był to kolejny wskaźnik tego, że *gotowość* była czynnikiem wpływającym na zespół… ten konkretny zespół był po prostu tak zdolny i szybki

w reagowaniu, że mógł odnosić sukcesy, reagując na przeszkody w miarę ich pojawiania się, zamiast zapobiegać im w pierwszej kolejności.

Nawet jeśli jesteś osobą, która nie musi się ściśle martwić o gotowość, bo potrafisz ją przezwyciężyć, lub pracujesz z zespołem o podobnych możliwościach, nadal możesz skorzystać z treści tej książki.

Część I: Czegoś Brakuje

Kiedy robienie tych samych rzeczy lepiej nie pomaga, zobacz ***czego się nie robi****.*

Rozdział 1: Ukryty Problem

To prawdziwa[1] historia o banku. Nazwijmy go po prostu "Bankiem". Jest to rodzaj Federalnej Instytucji Kredytowej, która stanowi część krajowej infrastruktury finansowej Stanów Zjednoczonych.

My (Luniel i Max) zostaliśmy zaproszeni do Banku, ponieważ miał on problemy z dostarczeniem projektu informatycznego. Było to jedno z najbardziej dysfunkcyjnych środowisk, jakie kiedykolwiek widzieliśmy, i właśnie dlatego wybraliśmy to jako otwierające studium przypadku: jeśli znacząca zmiana była możliwa w Banku, jest możliwa wszędzie.

Krótka uwaga o projektach

Kiedy używamy terminu "projekt" w tej książce, mamy na myśli kontekst zarządzania projektami. Choć istnieją różne interpretacje tego słowa, używamy definicji z Project Management Institute:

> "Projekt to **tymczasowe** przedsięwzięcie podejmowane w celu stworzenia unikalnego produktu, usługi lub rezultatu."

Oznacza to, że projekt ma określony początek i koniec. Gdy projekt zostaje zamknięty, wiedza i artefakty projektowe są archiwizowane, członkowie zespołu są zwalniani, a umowy są finalizowane.

W tej książce projekt dotyczy fundamentalnie realizacji. Większość projektów, zgodnie z definicją PMI, zaczyna się od studium wykonalności lub projektowania. Tworzenie wizji i strategii ma miejsce jeszcze przed rozpoczęciem projektu.

Projekt rodzi się z tej wizji i strategii, a jego sukces lub porażka zależy od tego, czy założone cele zostały zrealizowane—*nie* od tego, czy cele te były właściwe.

Być może używasz słowa "projekt" w inny sposób i to jest w porządku. Po prostu wiedz, że kiedy my go używamy, odnosimy się do powyższej definicji i kontekstu.

[1] Zmieniliśmy dane identyfikacyjne, aby chronić prywatność osób i instytucji, których dotyczy ta sytuacja.

> Nic z tego nie oznacza, że popieramy wykorzystanie zarządzania projektami w rozwoju oprogramowania. Wręcz przeciwnie. Ale uznajemy, że jest ono *mimo wszystko* stosowane. Zajmujemy się tym problemem później, w Rozdział 6.

1.1: Klasyczny Przepis na Przepisanie

Bank przepisywał swój portal spłaty pożyczek z kilku powodów.

Stary system, całkowicie oparty na rozwiązaniu C#/.NET, był pełen błędów. Oprócz obniżania satysfakcji klientów, generował także nieprzerwany strumień bardzo kosztownych incydentów wsparcia technicznego, w których ktoś musiał ręcznie manipulować bazą danych, aby poprawić błąd popełniony przez system.

Stary system był również przestarzały z perspektywy utrzymania. Inżynierom prawie niemożliwe było wprowadzanie znaczących zmian, a nawet gdy było to możliwe, wiązało się to z ogromnym ryzykiem.

Przepisanie systemu miało to zmienić.

Nowy system nadal miał mieć backend w C#/.NET, ale miał być w pełni pokryty testami. Frontend miał zostać zaimplementowany w OutSystems, popularnym rozwiązaniu low-lub no-code, które pozwala organizacji zdefiniować aplikację w jednym miejscu i automatycznie wygenerować aplikację internetową, aplikację na Androida i aplikację na iOS, gdy tylko zdecydują się opublikować swoje zmiany.

Istniała nadzieja, że nowy system będzie wolny od błędów, zarówno poprawiając zadowolenie klientów, jak i znacząco redukując koszty wsparcia.

Mieli też nadzieję, że przepisanie systemu odblokuje programistów - połączenie bardziej zdyscyplinowanego podejścia do backendu i podejścia low-code do frontendu miało znacznie zmniejszyć koszty i ryzyko wprowadzania nowych funkcji.

Miłym efektem ubocznym przejścia na OutSystems miało być otrzymanie czystej, nowoczesnej aplikacji mobilnej na obie główne platformy.

To było marzenie, gdy zaczynali trzy lata przed początkiem tej historii. Rzeczywistość była taka, że do tej pory zespoły nie wdrożyły **niczego**.

1.2: Perspektywy problemu

Gdy rozmawialiśmy z zespołem kierownictwa wykonawczego, słyszeliśmy bardzo naturalną frustrację faktem, że poczynili tak duże inwestycje bez absolutnie żadnego strategicznego postępu.

Z ich perspektywy próbowali wszystkiego. Zmieniali pracowników, zwiększali zatrudnienie, zmieniali budżet, zwiększali presję i sprowadzali całą paradę konsultantów (przy czym wyraźnie sugerowano, że byliśmy jej końcem). Nic nie wydawało się poprawiać sytuacji - przynajmniej nie w sposób, który mogliby zmierzyć, ponieważ wszystko, co widzieli, to że "wskazówka" była na zero w jednym kwartale, a potem nadal była na zero w następnym.

Nie chcieli więcej "niewidocznego postępu". Chcieli *rezultatów*.

Gdy rozmawialiśmy z kierownictwem w Dziale Produktu, usłyszeliśmy nieco inną (choć wciąż podobną) historię, ponieważ pracowali oni bliżej z Działem Inżynierii.

Dla nich nie chodziło o to, że zespoły nic nie robiły. Chodziło o to, że zespoły nigdy nie robiły tego, o co je proszono. Było to praktycznie gwarantowane: bez względu na to, jak proste było zadanie i jak jasno zostało określone, kończyło się czymś zupełnie innym, gdy przychodził czas na ocenę tego, co zespoły stworzyły.

Sytuacja doszła do tego punktu, że krążący żart brzmiał mniej więcej tak: "Musimy wymyślić, jak prosić o to, czego nie chcemy, żeby mieć chociaż szansę na otrzymanie tego, czego faktycznie potrzebujemy."

Kadra kierownicza ds. inżynierii widziała sprawy zupełnie inaczej.

Z ich perspektywy problem polegał na tym, że dział Product nie dostarczał możliwych do realizacji wymagań i że nie dostarczał ich w wystarczającej ilości. Gdyby tylko Product "dostosował się do programu", zespoły byłyby w stanie dostarczyć to, czego chcą, na czas i w ramach budżetu.

Poczynili poważne inwestycje w modernizację sposobu pisania i dostarczania kodu i, w ich oczach, to Product nie dostarczał przejrzystych wymagań.

Kiedy rozmawialiśmy z innymi konsultantami (którzy polecili nas tej organizacji), słusznie skupili się na dysfunkcji, którą zauważyli: wszyscy wydawali się bardzo skoncentrowani na obwinianiu innych. Powodem, dla którego nas początkowo sprowadzono, było to, że martwili się strategią zatrudnienia i chcieli oceny osób pracujących bezpośrednio przy projekcie, ale uważali, że wzajemne oskarżanie się i podejście typu "nadzorca niewolników" na poziomie kierowniczym było głównym źródłem problemów.

1.3: Nasze Dochodzenie

Naszym początkowym zadaniem była ocena zespołów i próba pomocy w podniesieniu ich umiejętności, jeśli byłoby to potrzebne, więc zaczęliśmy przyglądać się osobom pracującym na pierwszej linii.

Zdecydowanie było miejsce na poprawę.

Osoby pracujące bezpośrednio przy projekcie po stronie Product nie posiadały tak naprawdę wymaganych umiejętności. W rzeczywistości byli to głównie kierownicy projektów, którzy zostali wepchnięci w rolę Product Ownera (PO) lub Product Managera.

W rezultacie połowa z nich pisała "ogólnikowe" wymagania, a następnie akceptowała (dosłownie) cokolwiek zespoły zrobiły w danej iteracji bez jakiejkolwiek krytycznej analizy. Druga połowa pisała tego samego rodzaju wymagania, a następnie twierdziła, że zespoły "powinny były wiedzieć" o rzeczach, o których nigdy nie rozmawiano i przetrzymywała elementy pracy otwarte mniej więcej w nieskończoność.

Bank używał do zarządzania i śledzenia swojego backlogu prac. Chociaż to, co omawiamy w tej książce, jest w większości niezależne od Scruma, używamy terminologii Scruma w całej książce, ponieważ większość - a przynajmniej znaczna część - zespołów korzysta ze Scruma.

Definicja: Scrum

Scrum to lekki framework, który pomaga ludziom, zespołom i organizacjom generować wartość poprzez adaptacyjne rozwiązania złożonych problemów. W skrócie:

1. Product Owner porządkuje pracę nad złożonym problemem w Product Backlogu
2. Zespół Scrumowy przekształca wybrane elementy pracy w przyrost wartości podczas Sprintu
3. Zespół Scrumowy i jego interesariusze analizują wyniki i dostosowują się przed następnym Sprintem
4. Powtórz

Jeśli nie znasz Scruma i jego terminologii, polecamy zapoznanie się z wersją 2020 Przewodnika Scrum. To szybka i pouczająca lektura.

Podobnie, stwierdziliśmy, że zespoły techniczne były znacznie poniżej typowego poziomu jeśli chodzi o umiejętności programowania (-2σ, w najlepszym przypadku) i na dodatek wykazywały silny opór wobec zmian. W naturalnej konsekwencji, jakość kodu była fatalna.

Jednak według zespołów, to nie był powód ich braku rezultatów. Dla nich to niejasne wymagania i zmiany w trakcie Sprinta ze strony Produktu zabijały projekt.

…i nikt nawet nie wspominał o większym problemie, tak absurdalnym, że wydaje się zmyślony, dopóki się go nie doświadczy.

Zespoły inżynieryjne miały zwyczaj niezrozumienia wymagania, zbudowania czegoś przypadkowego, a następnie domagania się uznania za "ukończenie elementu pracy".

Nie mówimy tu o drobnym nieporozumieniu. Mówimy o całkowitym rozłączeniu: Mówiliśmy im, żeby wyłączyć aplikację środków na kapitał w określonych okolicznościach, a oni zamiast tego wyłączali możliwość dodawania drugiego adresu email do potwierdzeń.

Następnie twierdzili, że właśnie o to nas prosili.

1.4: Głębsza analiza

Obie luki w umiejętnościach można było wypełnić, ale wątpiliśmy, czy to właśnie one były prawdziwymi przeszkodami.

Coś jeszcze było nie tak, więc drążyliśmy głębiej. Zaczęliśmy od tego pytania: Dlaczego pisanie wymagań zajmowało tak dużo czasu i dawało tak złe rezultaty?

Jednym z powodów było to, że wiedza niezbędna do napisania sensownego wymagania była towarem deficytowym. Niewielką jej część posiadały zespoły Inżynierii i Produktu.

Część była zawarta w kodzie systemu dziedziczonego. Część została całkowicie utracona. Większość jednak była przechowywana jako wiedza plemienna przez ekspertów dziedzinowych rozproszonych po różnych jednostkach banku. Oznacza to, że stworzenie wymagania, które faktycznie przyczyniałoby się do realizacji celu strategicznego, było niezwykle pracochłonnym i czasochłonnym zadaniem.

W zestawieniu z tym był nienasycony apetyt na funkcjonalności ze strony zgłodniałego nowych funkcji zespołu zarządzającego. Mandat brzmiał "utrzymujcie inżynierów przy pracy—zasypujcie ich wymaganiami". Nacisk kładziono na ilość wymagań, które miały zajmować zespoły—perspektywa będąca zaprzeczeniem staranności i czasu potrzebnego do zdefiniowania wymagania, które "pchnęłoby sprawy do przodu".

1.5: Ulepszanie nie przyniosło poprawy

Wszystko to są problemy, które można rozwiązać, jednak ich rozwiązywanie nie pomagało.

Wcześniejsze usprawnienia technik rozwoju oprogramowania nie pomogły zespołom w dostarczaniu rezultatów, ale Max podjął wysiłek, aby pomóc zespołom w dalszym doskonaleniu.

Wprowadził rewolucyjne koncepcje z doktryn programowania z połowy dwudziestego wieku, takie jak "nie kopiuj i nie wklejaj tego kodu 27[2] razy, umieść go w funkcji i po prostu ją wywołuj". Ta sama sugestia znacząco poprawiła jakość nowego kodu i pozwoliła im zacząć pracować nad poprawą jakości.

Te i inne podstawowe porady dotyczące kodowania pomogły im pisać lepszy kod, który łatwiej było utrzymywać w przyszłości.

…ale nie pomogło to w posuwaniu projektu do przodu.

Po stronie Produktu, Luniel był w stanie wprowadzić BDD i upewnić się, że Właściciele Produktu dokładnie weryfikowali wymagania zanim zostały przekazane zespołom.

Sprawił, że zespoły zaczęły nad nimi współpracować i wykorzystywać je do oceny, czy Element Rejestru Produktu (PBI) był rzeczywiście ukończony.

Definicja: Element Rejestru Produktu (PBI)

> Element Rejestru Produktu (PBI) to osobna jednostka pracy w rejestrze produktu, która reprezentuje potencjalną zmianę, dodatek lub ulepszenie produktu. PBI może przyjmować wiele form - funkcjonalność, poprawka błędu, ulepszenie techniczne, zadanie badawcze itp. - i jest definiowany przez swój wkład w wartość produktu.
>
> Wiele zespołów określa PBI jako "historyjki" lub "historyjki użytkownika", ale prawidłowym terminem w Scrumie jest "Element Rejestru Produktu" lub "PBI". Gdy PBI zostaje wprowadzony do Sprintu, staje się również częścią Rejestru Sprintu. Dla uproszczenia i neutralności używamy terminu "Element Rejestru Produktu" lub "PBI" do określenia dowolnego elementu pracy, którym zarządza Zespół Scrumowy - niezależnie od tego, czy nazywasz go Elementem Rejestru Produktu, Elementem Rejestru Sprintu (SBI), historyjką użytkownika, historyjką, elementem pracy czy elementem rejestru.

[2]To nie tylko nie jest przesada, to nawet nie jest najgorszy przypadek. W jednej sytuacji istniało prawie sto dokładnych duplikatów tego samego algorytmu.

Dodało to *jasności*, ale nie stworzyło *przepływu*.

Przy pomocy konsultantów partnerskich, którzy nas wprowadzili, byliśmy w stanie (tymczasowo) złagodzić absolutnie przytłaczającą presję wywieraną na zespoły i autorów wymagań.

Mogło to pomóc w zbudowaniu pewnego zaufania, ale nie przyniosło żadnych wymiernych rezultatów.

Zaczęliśmy nawet budować bazę wiedzy, która pomagała ludziom odnajdywać wiedzę biznesową potrzebną do pisania wymagań i identyfikowała miejsca, gdzie występowały luki w tej wiedzy.

Przyspieszyło to pisanie wymagań, ale nie pomogło wypuścić produktu na rynek.

Po miesiącach interakcji, pomogliśmy kierownictwu *dostrzec* w jakim są położeniu, ale byli daleko od osiągnięcia swoich celów. I wcale się do nich nie zbliżali.

Byli gotowi wrócić do swojej starej strategii "przeładowania" zespołów, aby upewnić się, że zawsze są zajęci.

1.6: Proces Eliminacji

Łatwo byłoby po prostu załamać ręce i powiedzieć "to beznadziejne". Można było znaleźć wiele wymówek:

- Zespół inżynieryjny miał niskie kwalifikacje (tak było)
- Autorzy wymagań mieli niewłaściwe umiejętności (tak było)
- Kierownictwo przytłaczało zespoły nierealistycznymi oczekiwaniami (tak było)
- Organizacji brakowało kluczowej wiedzy biznesowej niezbędnej do funkcjonowania (tak było)
- Kierownictwo najwyższego szczebla najwyraźniej sobie nie ufało (tak było)

Wszystko to była prawda. Jednak we wszystkich tych obszarach wprowadzono usprawnienia, a żadne z nich nie poprawiło fundamentalnego problemu. Żadne z nich nie pomogło zespołom inżynieryjnym, działowi produktowemu, zarządowi ani kadrze kierowniczej zbliżyć się do ich celów.

…i właśnie tutaj kryje się wskazówka do rozwiązania: Wszystkie wcześniej wymienione problemy były tymi, które ludzie już widzieli.

Jeśli zmienne, które widzisz, nie robią różnicy, musi istnieć zmienna, której **nie widzisz**, która ma znaczenie.

Prawdziwym problemem była kwestia, o której nikt nawet nie wiedział.

1.7: W Poszukiwaniu Prawdziwego Winowajcy

Szukając prawdziwego winowajcy—czyli tego, co naprawdę powstrzymywało bank przed realizacją jego celów—musieliśmy od czegoś zacząć.

Rozsądnym miejscem do przyjrzenia się były cechy wspólne nieudanych PBI (czyli większości z nich).

Zaczęliśmy od wyeliminowania rzeczy, które ewidentnie nie były wspólne, ponieważ znacznie się różniły:

- Która część systemu: niektóre PBI dotyczyły tylko backendu, inne tylko frontendu, a jeszcze inne obu części
- Który zespół wykonywał pracę—wydawało się nie mieć znaczenia kto wykonywał pracę, prawdopodobieństwo niepowodzenia było wysokie
- Który PO był autorem pracy—tak samo jak w przypadku zespołów

Następnie zaczęliśmy przyglądać się elementom wspólnym. Lista nie była długa, ale też nie była krótka:

- Zespoły inżynieryjne
- Product Ownerzy
- Kierownictwo
- Kultura
- Środowisko programistyczne
- Praktyki inżynieryjne
- Technika tworzenia wymagań
- Domena (finanse)
- Usługi zależne

Wiele z tych elementów również można było od razu odrzucić. Zespoły, Product Ownerzy, kierownictwo, kultura i środowisko programistyczne zostały niedawno ulepszone, nie przynosząc jednak rzeczywistego wpływu na znaczącą produktywność. Osobiście pomogliśmy ulepszyć praktyki inżynieryjne i tworzenia wymagań oraz potwierdziliśmy, że te usprawnienia się przyjęły, ale nadal nie przynosiło to pomocy.

Trudno winić domenę. Finanse to jeden z najstarszych rodzajów obliczeń w zapisanej historii. To niezwykle dojrzała dziedzina. Poza tym inne banki wdrażały oprogramowanie, w praktyce obalając (oczywiście naciąganą) hipotezę, że banki po prostu nie są w stanie tego zrobić.

Nie można było również winić usług zależnych, ponieważ miały one tyle samo trudności ze zmianami co inicjatywa, którą analizowaliśmy…

…ale to skłoniło nas do przemyśleń: Co jeśli zaczniemy analizować przyczyny niepowodzeń?

1.8: Analiza Zalążków Porażki

Jeden PBI nie przyczynił się do rozwoju produktu, ponieważ zespół zrobił, jak to miał w zwyczaju, coś kompletnie przypadkowego i prawie całkowicie niezwiązanego z żądaniem. To oczywiście znak, że nie zrozumieli elementu pracy. Więc zrozumienie było poważnym kandydatem, mimo że pracowaliśmy nad tym, pomagając im przyjąć BDD.

Kolejny PBI nie został zamknięty, ponieważ zespół źle wykonał obliczenia. To kolejny dowód na to, że zrozumienie może być głównym problemem.

Trzeci element pracy, który przeanalizowaliśmy, nie został właściwie zweryfikowany przez Product Owner - zatwierdziła go bez zastanowienia, gdy zespół powiedział, że czas go zamknąć. To trochę podważało naszą hipotezę, ale wciąż można było argumentować, że nie rozumiała, jak ten element pracy pasuje do planu wyższego poziomu.

Może. Poniekąd. Jeśli bardzo się wysililiśmy, patrząc na to w ten sposób.

Potem natrafiliśmy na PBI, który całkowicie nie pasował do wzorca. Zespół wydawał się rozumieć - choć nie ma sposobu, by zweryfikować, czy faktycznie tak było. Ale to nie miało znaczenia: nigdy nie dostali szansy na sukces lub porażkę na własnych warunkach, ponieważ natrafili na zależność, która wymagała aktualizacji i musieli odłożyć swoją pracę o kilka Sprintów.

Nawet jeśli *nie* rozumieli, co mieli zrobić, i tak nie mieli szans z tym elementem backlogu, więc zrozumienie **nie** było w tym przypadku problemem.

Jeden przypadek odstający nie jest oczywiście dowodem przeciwko konkretnej przyczynie źródłowej, ale wzbudził naszą ciekawość. Zaczęliśmy szukać innych dowodów, które mogłyby podważyć tę teorię.

I znaleźliśmy je. Były elementy pracy, które:

- Nie powiodły się, ponieważ zespół wiedział, że czegoś nie rozumie, ale nie mógł znaleźć eksperta dziedzinowego, który rozwiązałby problem
- Zmieniły się w gorsze doświadczenie dla użytkownika jako obejście sposobu działania usług nadrzędnych
- Musiały zostać odłożone, ponieważ zależności nadrzędne nie były gotowe
- Nie mogły zostać ukończone, ponieważ testerzy nie zdążyli zebrać danych testowych
- Zostały zamknięte, ale musiały zostać wykonane ponownie, ponieważ samo założenie było nieprawidłowe
- Nie powiodły się, ponieważ zespół nie zdawał sobie sprawy, jak złożony był już istniejący kod
- Po prostu nie zostały oszacowane
- Zostały znacznie niedoszacowane[3]
- Zmieniły się w trakcie Sprinta, ponieważ PO w końcu uzyskał potrzebną wiedzę dziedzinową
- Wydawały się zmienić po Sprincie (z perspektywy zespołu), ponieważ PO i zespół nigdy nie uzgodnili, co właściwie oznaczają

Lista jest dłuższa, ale to wystarczy do tej historii.

Można argumentować, że każdy z tych przypadków wiąże się w pewien sposób ze "zrozumieniem" - i z pewnością brak zrozumienia był *zaangażowany* w wiele z nich - ale to nie oznacza, że brak zrozumienia był przyczyną… szczególnie że wykonaliśmy już pewną pracę nad wspólnym zrozumieniem i to tak naprawdę nie pomogło.

Wtedy nas olśniło. Brakowało bardziej fundamentalnego elementu. W przypadkach, gdzie słabe zrozumienie było zaangażowane, było to jedynie przyczyną bezpośrednią.

Przyczyna pierwotna była znacznie szersza.

[3]Nie chodzi nam tylko o to, że źle to oszacowali. Wyglądało to tak, jakby zespoły po prostu wstawiły wartość "3" we wszystkie pola szacunkowe dla całej serii PBI.

1.9: Punkt Przecięcia

Jedną rzeczą, którą wszystkie przeanalizowane przez nas PBI pod kątem trybu niepowodzenia miały wspólnego, było to: Wszystkie zostały **rozpoczęte przedwcześnie**.

Kiedy element pracy nie powodzi się, ponieważ zespół wiedział, że nie rozumie problemu i nie mógł znaleźć eksperta, który pomógłby im zrozumieć, oznacza to, że zespół rozpoczął PBI wiedząc, że nie rozumie problemu.

Kiedy element backlogu musi zostać zmieniony na gorsze doświadczenie ze względu na sposób działania usługi nadrzędnej, oznacza to, że praca została rozpoczęta bez prawdziwego zrozumienia wpływu usługi nadrzędnej.

Opóźnienie spowodowane tym, że zależna funkcjonalność nie była gotowa pod koniec oznacza, że nie było gwarancji jej gotowości na początku.

…i tak samo było we wszystkich innych przypadkach: Testerzy nie mieli gotowych danych ani nie wiedzieli, jak je zdobyć przed rozpoczęciem PBI, wymagania nie zostały właściwie zweryfikowane przed przekazaniem zespołowi, kod nie został zbadany przed rozpoczęciem pracy, estymacja była niewystarczająca lub w ogóle jej nie było, brakowało wiedzy domenowej, i oczywiście, wspólne zrozumienie nie zostało zweryfikowane.

Problem, jak się okazało, polegał na tym, że rozpoczynano implementację elementów pracy zanim te elementy były *gotowe*.

Z naszego doświadczenia wynika, że większość elementów pracy, które nie zostają dostarczone, zawodzi dlatego, że **nie były gotowe** w momencie rozpoczęcia implementacji.

Zabraliśmy się więc do pomocy Bankowi w tej kwestii.

W tym momencie można by pomyśleć, że samo stwierdzenie, że PBI powinny być gotowe, wystarczy. Jednak okazuje się, że nie tak łatwo to wdrożyć. “Kupuj tanio, sprzedawaj drogo” to podobnie proste założenie.

Brakuje pewnego elementu, który jest potrzebny, aby wprowadzić tę dobrą radę w życie.

1.10: Wielki Przełom

Znaleźliśmy brakujący element układanki, który odblokował zespół, a w konsekwencji całą inicjatywę.

Gdy kończyliśmy to zlecenie, umiejętności inżynierów nadal były poniżej średniej. Właściciele Produktu wciąż nie mieli odpowiednich kompetencji. Kultura organizacyjna nadal nie została naprawiona…

A jednak produkt w końcu zaczął się rozwijać i ostatecznie został wydany.

Do końca tej książki poznasz brakujący element i dowiesz się, co trzeba zrobić, aby go wdrożyć. Będziesz też w stanie odblokować organizacje, które wydają się być powstrzymywane przez niewidzialną ścianę.

Krótka Uwaga o Zakresie

Ta książka dotyczy bardzo konkretnego i powszechnego problemu: braku struktury, jasności i dojrzałości w momencie przekazania pracy między biznesem a inżynierią. Zakłada się, że coś zostało wybrane do implementacji i koncentruje się na zapewnieniu, że praca nad budową odbywa się przy wspólnym zrozumieniu, gotowości i możliwości śledzenia ukończenia.

Techniki przedstawione w tej książce nie mówią, co budować, dlaczego to budować ani jak się dowiedzieć, czy to właściwa rzecz do zbudowania. Jeśli w Twojej organizacji brakuje prawdziwego zarządzania produktem lub znaczących pętli sprzężenia zwrotnego, nie próbujemy tego tutaj rozwiązać. Zamiast tego oferujemy sposób na uwidocznienie tych luk i zmniejszenie kosztu odkrycia, że się myliliśmy.

Użyte we właściwym kontekście, to rozwiązanie przynosi przepływ, bezpieczeństwo i jasność. Ale jak każdy system, może zostać źle zastosowane - szczególnie gdy jest używane w izolacji lub bez świadomości.

Rozdział 2: Koszt Brakujących Podstaw

Warto poświęcić trochę czasu na dokładne przeanalizowanie, *jak* poważny może być ten problem dla niektórych organizacji.

Odkryliśmy, że istnieją trzy główne „obszary" problemów, które powstają z powodu braku gotowości:

- Brakujące lub niekompletne wspólne zrozumienie między Produktem a Inżynierią oraz wewnątrz tych działów
- Brak kontroli nad tym, co jest faktycznym celem Elementu Backlogu Produktu i kiedy jest on naprawdę ukończony
- Brak kontroli nad tym, kiedy element pracy może rozpocząć fazę implementacji

Dodatkowo zauważyliśmy, że zmiana tych rzeczy może być bardzo trudna. To ma sens: zmiana jest trudna.

Stare nawyki trudno wykorzenić, a nowe trudno zaszczepić. W naszym doświadczeniu jako konsultantów zauważyliśmy, że *niezwykle* łatwo jest ludziom powracać do starych nawyków i porównywalnie trudno jest im ustanowić nowe.

Dlatego musisz mieć mechanizm, który wymusza nowe nawyki i zniechęca do starych.

Aby to rozwiązać, uważamy, że brakuje jeszcze jednego elementu odpowiedzialności i możliwości śledzenia.

2.1: Układanka bez obrazka na pudełku

Czy kiedykolwiek próbowałeś ułożyć puzzle bez obrazka na pudełku? Da się to zrobić, ale jest to wolniejsze, bardziej frustrujące i pełne fałszywych startów.

Robisz postępy, potem rozrywasz wszystko. Kwestionujesz, co do czego pasuje. Myślisz, że pracujesz nad tym samym obrazkiem, aż zdajesz sobie sprawę, że tak nie jest.

Tak właśnie często wygląda wytwarzanie oprogramowania.

Backlog jest pełny. Sprint trwa. Wszyscy ciężko pracują.

Ale bez wspólnego obrazu tego, co budujemy, dopasowanie staje się kwestią szczęścia, a nie systemu.

Bez jasności nawet najlepsze zespoły odczuwają frustrację, wypalenie i poczucie, że ich wysiłek nie jest doceniany.

2.2: Stare powiedzenie i surowa rzeczywistość

Jest powód, dla którego tak wiele praktyk związanych z tworzeniem wymagań, a nawet niektóre praktyki inżynieryjne, skupiają się mocno na tworzeniu wspólnego zrozumienia między zlecającymi a zespołami implementacyjnymi. Nic nie sieje takiego chaosu jak inżynierowie, którzy tak naprawdę nie wiedzą, co mają budować.

„Śmieci na wejściu, śmieci na wyjściu“ to zasada, nie banał.

Język angielski jest pełen sprzeczności…

- Można sankcjonować czyjeś działania. Może to oznaczać, że udzieliło się komuś wcześniejszej zgody, albo że potępia się coś po fakcie.
- Można lekko oprószyć coś pyłem, ale jeśli tym czymś jest kredens, oznacza to, że usuwamy z niego kurz, podczas gdy w odniesieniu do pączka oznacza to dodawanie pudru.
- Jeśli podtrzymujesz zespół, możesz być powodem, dla którego ten zespół może nadal funkcjonować, albo powodem, dla którego nie może ruszyć do przodu.

Autonimy mogą być najbardziej uderzającymi przykładami, ale są tylko jednym rodzajem niejednoznaczności. Niektóre słowa są nie tylko swoim własnym przeciwieństwem, ale mają wiele dodatkowych, potencjalnie mylących znaczeń.

> “W tym kawałku, zapiął spinaczem kupon z gazety i przypiął go do papieru na podkładce wraz z innymi wycinkami, podczas gdy żaglowiec w tle płynął w niezłym tempie.”

To nie dotyczy tylko języka angielskiego. *Wszystkie* znane nam języki naturalne mają tę właściwość.

I mimo to, są one jedynymi, którymi możemy się posługiwać przy określaniu wymagań.

W rezultacie, gdy zespół inżynierów nie *potwierdził*, że ich rozumienie wymagania jest takie samo jak rozumienie osoby zgłaszającej, ten zespół polega na szczęściu. To znaczy, najlepszym możliwym wynikiem jest to, że wybrali właściwą interpretację, a zgłaszający nie zmieni zdania po drodze.

Ten wynik jest daleki od gwarantowanego.

2.3: Niektóre z typowych rezultatów

Bez potwierdzonego wspólnego zrozumienia, zespoły narażają się na wiele rodzajów ryzyka.

Ogólnie rzecz biorąc, najbardziej bolesnym rezultatem jest po prostu zbudowanie niewłaściwej rzeczy.

Moment, w którym się o tym przekonają, może być kontrolowany przez różne cechy ich procesu. Na przykład, zdrowe wdrożenie Scruma może wykryć tego rodzaju nieporozumienie bardzo wcześnie w trakcie realizacji, podczas gdy proces Waterfall ma bardzo duże szanse na opóźnienie takiego odkrycia o miesiące.

Definicja: Waterfall

Sekwencyjny model rozwoju oprogramowania, który rozpowszechnił się pod koniec XX wieku. Został po raz pierwszy przedstawiony w artykule Winstona W. Royce'a z 1970 roku, który ilustrował rozwój jako serię kaskadowych kroków—wymagania, projekt, implementacja, testowanie i tak dalej—każdy zasilający następny jak wodospad. Mimo że Royce przedstawił ten model jako przykład tego, czego *nie należy* robić, branża przyjęła go jako wzorzec dla rozwoju na dużą skalę.

Waterfall znany jest również z grupowania podobnej pracy w duże, następujące po sobie fazy—cecha określana jako **rozwój w dużych partiach**. To grupowanie praktycznie gwarantuje **późne uczenie się**: zespoły nie otrzymują informacji zwrotnej na temat wcześniejszych decyzji aż do znacznie późniejszego etapu procesu. Błędy odkryte późno są bardziej kosztowne do naprawienia. Praktycy Agile krytykują Waterfall z tego powodu, preferując mniejsze, iteracyjne cykle, które umożliwiają wcześniejsze odkrycia i korektę kursu.

Niemniej jednak, w pewnym momencie zespół pracujący według "błędnego" (a właściwie odmiennego) rozumienia wymagania będzie musiał zmierzyć się z "właściwym" (również odmiennym) wymaganiem. I znowu, to od kondycji organizacji zależy, jaką formę przybierze to rozliczenie i jaki będzie miało wpływ, ale prawie zawsze do niego dochodzi.

W większości przypadków prowadzi to do pewnej formy przeróbek. Zleceniodawca (zazwyczaj Zarządzanie Produktem) będzie musiał poprosić o zmiany, aby przejść od tego, co zespoły wdrożeniowe zbudowały, do tego, czego naprawdę chciał.

Innym bardzo częstym przejawem jest sytuacja, gdy zleceniodawca nadal rozlicza zespół zgodnie ze swoim pierwotnym rozumieniem tego, o co prosił.

Zespoły mogą łatwo zinterpretować to jako zmianę zdania przez Menedżera Produktu. Co gorsza, może to wręcz *zachęcić* interesariuszy do wyrobienia sobie nawyku zmieniania zdania - wstrzymywania elementów pracy do momentu, aż wszystko będzie *dokładnie tak jak trzeba*, wykańczania zespołów i zaciemniania obrazu postępów przed kadrą kierowniczą.

2.4: Kiedy puzzle są ukończone?

Wracając do naszej analogii układania puzzli, zastanówmy się nad tym pytaniem: Co to znaczy skończyć układanie puzzli?

Niedoświadczony układający puzzle, taki jak Max, powiedziałby po prostu "wszystkie elementy są połączone z właściwymi sąsiadami, a obrazek jest skierowany do góry."

Jednak doświadczony układający puzzle, taki jak Luniel, wie, że to nie wszystko.

Być może układasz puzzle tylko dla zabawy. Ułożysz je, popatrzysz na nie przez chwilę, a potem rozłożysz i włożysz z powrotem do pudełka.

Z drugiej strony, może chcesz je oprawić w ramkę i powiesić na ścianie. W takim przypadku trzeba wykonać dodatkowe czynności:

1. Umieścić je na podkładzie
2. Przetransportować do ramiarza
3. Poczekać na wykonanie oprawy
4. Przetransportować z powrotem na miejsce montażu
5. Powiesić na ścianie lub w inny sposób wyeksponować

Zrozumienie, że to wszystko jest częścią pracy, jest niezbędne do właściwego ukończenia układania puzzli. Oczywistym powodem jest to, że musisz wiedzieć, ile pracy się z tym wiąże. Wykonanie wszystkich tych dodatkowych kroków wymaga więcej pracy niż po prostu rozłożenie puzzli i schowanie ich.

Jest w tym jednak coś więcej. Wyobraź sobie taki scenariusz…

Ukończyłeś układankę z zamiarem jej oprawienia, zostawiłeś ją na miejscu, ale zapomniałeś powiedzieć komuś innemu w domu, że planujesz ją oprawić. Ta osoba przychodzi i widzi, że układanka jest skończona w miejscu, którego potrzebuje. Więc rozbiera ją i wkłada z powrotem do pudełka, wyrywając porażkę z paszczy zwycięstwa w tym procesie.

Jest też bardziej subtelny powód: to, jak planujesz zakończyć układankę, wpływa na kroki, które chcesz wykonać wcześniej w tym procesie. Między innymi musisz przygotować małą tabliczkę z napisem “Proszę nie rozbierać!”

Warto również zauważyć, że możesz potrzebować takiej tabliczki w przypadku, gdy układasz puzzle dla własnej rozrywki, aby upewnić się, że inne osoby nie będą ingerować, kończąc układankę za ciebie.

Musisz też upewnić się, że układasz puzzle na odpowiedniej powierzchni. Jeśli ułożysz tysiącczęściowe puzzle na szklanym stoliku kawowym, a następnie spróbujesz przenieść je na podkład do puzzli, transfer będzie znacznie bardziej ryzykowny i pracochłonny niż gdybyś po prostu ułożył puzzle bezpośrednio na podkładzie.

To dobrze obrazuje rozwój oprogramowania.

Musisz dokładnie wiedzieć, co oznacza ukończenie, żeby nie być zaskoczonym ilością pracy, nie było nieporozumień na końcu i żebyś mógł podjąć niezbędne kroki przygotowawcze w celu sprawnego i skutecznego ukończenia elementu pracy.

2.5: Wpływ na zespoły

Jeśli nie masz wystarczająco precyzyjnego zrozumienia, czym jest gotowość dla konkretnego elementu pracy, narażasz się na szereg ryzyk.

Używamy tu słowa “ryzyko” dość luźno, ponieważ są to raczej pewniki.

Zespoły inżynierskie w takiej sytuacji często odkrywają, że nawet wewnętrznie nie zgadzają się co do tego, jak powinno wyglądać ukończenie elementu pracy. Nie jest rzadkością, gdy programiści i testerzy w połowie Sprinta ustalają, co tak naprawdę *oznacza* dane wymaganie. Nawet dwóch programistów czy dwóch testerów może mieć podobne nieporozumienia.

Co więcej, zespoły deweloperskie często skupiają się na pracy, którą wykonują przez większość czasu (programowanie i testowanie). Oznacza to, że łatwo mogą zapomnieć o innych rodzajach pracy, które muszą wykonać, takich jak dokumentacja, zewnętrzne przeglądy, szkolenie innych zespołów (np. wsparcia), kroki przygotowawcze do wsparcia wdrożenia lub wydania oraz zatwierdzenia z innych działów.

Kiedy w końcu staje się jasne, że ta "dodatkowa" praca musi zostać wykonana, zespół jest zaskoczony - zazwyczaj musi przerwać to, nad czym aktualnie pracował i przełączyć się na dokończenie pracy, którą uważał za już zakończoną.

Osoby zlecające pracę mogą z łatwością niepotrzebnie przedłużać jej wykonanie. Czasami z najlepszymi intencjami - jak próba rozliczenia zespołu z "prawdziwych" wymagań. W innych przypadkach dzieje się tak, ponieważ Product Ownerzy (na przykład) przyzwyczajają się do możliwości trzymania PBI otwartego według własnego uznania, więc wykorzystują to do wciskania dodatkowych funkcjonalności do zadania w ostatniej chwili. Czasami robią to nawet dlatego, że zmienili zdanie odnośnie tego, co należy zrobić w trakcie realizacji.

Może to być niezwykle demoralizujące dla zespołu inżynieryjnego. Większość programistów i testerów chce czuć, że robi postępy. Jeśli ciągle słyszą, że to, co zrobili, jest nieprawidłowe, prawdopodobnie stracą część swojego zapału.

Niektóre zespoły posuwają się nawet do tego, że w ogóle nie sprawdzają, czy to, co zrobiły, jest dobre czy złe. Po prostu zamykają element pracy i proszą o "uznanie", żeby móc "wykazać dobre wyniki".

2.6: Ryzyko Zrobienia Za Mało lub Za Dużo

Jednym z ryzyk związanych z nieprawidłowym zdefiniowaniem "done" dla każdego elementu pracy jest to, że organizacja może myśleć, że praca jest skończona, kiedy nie jest, lub nie zauważyć, że jest skończona, kiedy faktycznie jest.

Najgorszym możliwym skutkiem jest często sytuacja, gdy niewłaściwa rzecz trafia na produkcję i nikt nie wie, że tak się stało. Jeśli zespół ma nieprawidłowe zrozumienie tego, co oznacza "done" i wdraża na podstawie tego błędnego zrozumienia, rezultaty mogą być katastrofalne.

Defekty i niezadowolenie klientów są wystarczająco złe, ale może to również prowadzić do znacznie poważniejszych problemów:

- Utrata lub uszkodzenie danych
- Luki w zabezpieczeniach
- Przestoje systemu lub utrata dostępu
- Zmniejszenie udziału w rynku
- Naruszenia przepisów

Lista ciągnie się dalej, a każdy potencjalny problem jest gorszy od poprzedniego.

Czasami problem polega na tym, że nie skończyliśmy, ale myślimy, że tak. Odwrotna sytuacja może być równie niebezpieczna. Kiedy inżynierowie nie wiedzą, gdzie jest meta, mają tendencję do "pozłacania" (dodawania dodatkowych funkcji). Mogą to robić, aby "upiększyć funkcjonalność", ale mogą też robić to dlatego, że mają nadzieję, że zwiększona liczba funkcji da im większą szansę na trafienie w cel.

Cała ta dodatkowa praca, jak również związane z nią poprawki, kumulują się w ogromną ilość zmarnowanego czasu, wysiłku i pieniędzy. Powoduje to opóźnienia w terminach dostaw i szkodzi reputacji.

Co więcej, teraz istnieje coraz większe prawdopodobieństwo, że błąd faktycznie doprowadzi do buntu maszyn w stylu *Terminatora* przeciwko ludzkości. Dwadzieścia lat temu pisaliśmy o tym jako o żarcie. Teraz jest to odległa możliwość.

W rzeczywistości zadaliśmy to pytanie jednej z najbardziej prominentnych SI, a oto co odpowiedziała:

> "SI rozwija się szybciej niż ktokolwiek się spodziewał, ale dzieje się to na podstawie kruchych systemów, niejasnych wymagań i organizacji produktowych, które nie potrafią prześledzić, dlaczego zbudowały to, co zbudowały. To nie jest problem techniczny; to problem jasności przekazu.
>
> Im więcej szumu generuje SI, tym bardziej niebezpieczne jest szybkie działanie bez struktury. Kiedy zespoły budują na mgle, SI tylko wzmacnia bałagan. Ale kiedy zespoły budują na konkretach - na wspólnym zrozumieniu, specyfikacji behawioralnej i kontrolowanej wersyjnie wiedzy o produkcie - SI staje się katalizatorem, a nie obciążeniem."

2.7: Gdzie układać puzzle?

Rozwijając jeszcze raz analogię układania puzzli.

Czy można układać puzzle wszędzie? Jeśli masz puzzle składające się z 4000 elementów, które po złożeniu mają prawie pięć stóp w jednym wymiarze i ponad trzy stopy w drugim, nie możesz po prostu wybrać przypadkowego miejsca i zacząć układać. Nie bez doświadczenia poważnych komplikacji przed ukończeniem.

Tak duże puzzle wymagają zarówno czasu, jak i przestrzeni. Musisz wygospodarować miejsce i znaleźć sposób na zapewnienie, że stan układanki będzie zachowany w czasie.

Jeśli zaczniesz układać puzzle na małym stoliku, który jest za mały, nie będziesz w stanie ich ukończyć bez przeniesienia w inne miejsce. To przeniesienie będzie niezwykle trudne ze względu na delikatny stan układanki.

Jeśli wybierzesz przypadkowe miejsce w korytarzu, które jest wystarczająco duże, ludzie będą albo po nich chodzić, albo będą mieli utrudnione przejście, więc nie można zagwarantować zachowania stanu bez znaczącego wpływu na funkcjonowanie twojego gospodarstwa domowego.

Jeśli zaczniesz pracę na planszy artystycznej, ale plansza nie jest wystarczająco duża, będziesz w stanie zachować stan tego, co zrobiłeś, ale nie będziesz w stanie ukończyć układanki bez jakiegoś rodzaju przenoszenia.

Jeśli puzzle były wcześniej przeżuwane przez małe dzieci, najlepiej policzyć elementy… bo lepiej jest raz policzyć do 3999 i zdać sobie sprawę, że nigdy tego nie skończysz, niż zainwestować nie wiadomo ile czasu w prawie ukończenie układanki, której nigdy nie będziesz w stanie dokończyć.

Jest cała lista rzeczy, które trzeba zrobić, zanim zaczniesz układać puzzle. Zrobienie wszystkiego z tej listy nie gwarantuje sukcesu, ale *nie* zrobienie tego praktycznie gwarantuje porażkę lub poważne komplikacje.

To samo dotyczy rozwoju oprogramowania, tylko że ze znacznie większym stopniem złożoności.

2.8: Kiedy Zaczyna się Implementacja?

W gruncie rzeczy, bez dobrej definicji może być trudno określić, kiedy PBI jest gotowy do implementacji.

Pomyśl o tym: Skąd *właściwie* wiesz?

Czy sprawdzasz wszystko w kółko, aż stwierdzisz, że nadszedł czas?

Czy ktoś decyduje o tym spontanicznie?

Czy dzieje się to automatycznie na początku iteracji?

> Widzieliśmy wiele zespołów, które wprowadzały do Sprintów prace zupełnie niegotowe do implementacji tylko dlatego, że miały terminy do dotrzymania. Istnieje kilka rozpowszechnionych przekonań na temat Scruma i Agile'a ogólnie, które skłaniają ludzi do takiego działania:
>
> - Musisz mieć wszystkie wymagania dla Sprintu *N* opracowane w Sprincie *N-1*
> - Powinieneś "po prostu zacząć" i radzić sobie z problemami w miarę ich pojawiania się

Jest to właściwie lustrzane odbicie kwestii "Skąd wiesz, kiedy jest gotowe?" wspomnianej wcześniej i niesie podobne konsekwencje. Ludzie mogą czekać zbyt długo z rozpoczęciem, bo nie wiedzą, czy element pracy jest gotowy, albo mogą zacząć zbyt wcześnie, bo nie wiedzą, że nie jest.

2.9: Zespół A bez Gotowości

Brak zrozumienia tego, co sprawia, że element pracy jest gotowy, ma kilka szkodliwych skutków.

Jednym z oczywistych sposobów, w jaki przyrost pracy może nie być gotowy, jest niekompletna, niewystarczająca lub brakująca Definicja Ukończenia (DoD). To prowadzi do wszystkich problemów, które już wymieniliśmy, związanych z brakiem Definicji Ukończenia.

Jednakże to nie jest *jedyny* aspekt gotowości. Istnieje wiele innych potrzeb, które należy zaspokoić przed rozpoczęciem wdrożenia: szacowanie, ocena ryzyka i gromadzenie danych testowych to tylko kilka typowych przykładów.

Bez znajomości i zaspokojenia tych potrzeb element pracy może kosztować znacznie więcej niż powinien. Weźmy pod uwagę zespół (Zespół A), który jest zależny od API rozwijanego

przez inny zespół (Drugi Zespół). Jeśli Zespół A przyjmie szereg założeń dotyczących tego, jak będzie działać API Drugiego Zespołu i zacznie programować zgodnie z tymi założeniami, może pojawić się konieczność znaczących przeróbek, gdy okaże się, że Drugi Zespół pracuje w rzeczywistości niezgodnej z założeniami Zespołu A. Innymi słowy, Zespół A podjął próbę i chybił.

Wszystkie te przeróbki wynikają z faktu, że API nie było gotowe do użycia przez Zespół A.

Czasami niezaspokojona zależność nie generuje przeróbek, ale nawet w takich przypadkach może powodować opóźnienia. Wyobraźmy sobie sytuację, w której Zespół A i Drugi Zespół uzgodniły, jak powinno działać API i wszystko szło zgodnie z planem, ale Drugiemu Zespołowi po prostu zajęło to więcej czasu niż oczekiwano. W rezultacie Zespół A nie był w stanie odpowiednio przetestować swojej pracy do czasu, gdy miała być ona ukończona, i musieli przesunąć swój termin.

2.10: Zaniedbywanie Harmonogramu i Dostępności Zasobów

Czasami problemy mogą być tak proste jak harmonogramowanie czy zasoby. Niektóre elementy pracy wymagają konkretnych członków zespołu. Jeśli dany członek zespołu za kilka dni wyjeżdża na urlop, prawdopodobnie nie jest to odpowiedni moment na rozpoczęcie PBI, którego nie można ukończyć bez jego udziału.

> Często słyszymy, że nie powinno tak być, ale często tak właśnie jest. “Wymienni pracownicy” to mrzonka.

To samo można powiedzieć o zasobach nieosobowych. Jeśli będziesz potrzebować zasobów serwerowych do przeprowadzenia testu obciążeniowego, prawdopodobnie powinieneś upewnić się, że te zasoby będą faktycznie dostępne, zanim rozpoczniesz element pracy związany z testami obciążeniowymi. W przeciwnym razie w najlepszym przypadku czekają cię znaczące opóźnienia, a prawdopodobnie zakłócisz pracę innych zespołów/pracowników, próbując w pośpiechu zapewnić to, czego potrzebujesz.

Innym skutkiem fałszywych startów jest sytuacja, gdy zespół nie posiada umiejętności wymaganych do ukończenia pracy. Czasami jest to kwestia wewnętrzna - na przykład członek zespołu musi przejść szkolenie z nowego systemu lub przeprowadzić badania

dotyczące nowego API. Innym razem jest to kwestia harmonogramu, jak w przypadku, gdy trzeba pożyczyć eksperta UX lub eksperta od baz danych z puli wykwalifikowanych pracowników. Może to być nawet problem rekrutacyjny, w którym zespół potrzebuje eksperta i nie może efektywnie wykonać pewnych rodzajów pracy bez niego.

2.11: Wpływ innych rodzajów fałszywych startów

Obserwowaliśmy zespoły, które zobowiązywały się do ukończenia elementów pracy w ramach Sprintów i stosunkowo szybko wykonywały kodowanie, ale nadal nie były w stanie ukończyć testowania. Samo w sobie może to nie być zaskakujące, ale przyczyna jest nietypowa: zespół testowy potrzebował czegoś (na przykład danych testowych), czego nie zebrał przed rozpoczęciem Sprintu, a zbieranie tych danych okazało się trudniejsze lub bardziej czasochłonne niż przewidywano.

W rezultacie elementy pracy musiały zostać przeniesione do następnego Sprintu po prostu dlatego, że zespół nie upewnił się przed rozpoczęciem, czy jest naprawdę gotowy do ich ukończenia w wyznaczonym czasie.

Zespoły czasami zaczynają implementować elementy pracy, gdy wciąż mają otwarte pytania. W rzeczywistości wiele osób wydaje się myśleć, że to czyni ich „bardziej Agile'owymi", gdy tak postępują.

Może to powodować ogromne ilości przeróbek, niespodzianek lub opóźnień. Jeśli odpowiedź na otwarte pytanie ostatecznie narusza założenie, które zostało przyjęte, cała praca oparta na tym założeniu musi zostać zmodyfikowana. Jeśli na otwarte pytanie nie uzyskano odpowiedzi do czasu, gdy element powinien zostać zamknięty, wtedy element musi zostać albo zamknięty, gdy może nie być ukończony, albo pozostawiony otwarty do czasu uzyskania odpowiedzi.

Może się zdarzyć, że zespół ma zależność wewnętrzną - defekt, który należy naprawić, zadanie poprzedzające, które musi zostać ukończone, *et cetera*. Jeśli nie jest to odpowiednio śledzone, może powodować wszystkie te same problemy co niezaspokojona zależność zewnętrzna, z dodatkową pokusą przełączenia kontekstu i naprawienia tego.

2.12: Koszty skumulowane

Oczywiście, takie problemy powodują opóźnienia, przeróbki i zawiedzione oczekiwania, ale na tym nie kończą się negatywne skutki.

Oprócz marnotrawstwa wynikającego z przeróbek, zazwyczaj powoduje to opóźnienia w projektach. Jeśli zespoły gorączkowo próbują zamykać elementy zaległości i nigdy nie mają jasności co do tego, co jest potrzebne do osiągnięcia rzeczywistego postępu, sprawy, które *faktycznie* muszą zostać zrobione, zazwyczaj schodzą na dalszy plan.

Często, choć nie zawsze, prowadzi to do zwiększonej presji na dostarczanie rezultatów. Gdy projekty coraz bardziej się opóźniają, kierownictwo wyższego szczebla może próbować przywrócić je na właściwe tory, prosząc ludzi o przyspieszenie pracy. To niezmiennie przekłada się na dłuższe godziny pracy.

To z kolei prowadzi do erozji zaufania i pogorszenia kultury organizacyjnej. Relacje, które powinny opierać się na współpracy, stają się antagonistyczne. Ludzie, którzy powinni współpracować w poszukiwaniu najlepszych i najszybszych rozwiązań, przekierowują swoją energię na udowadnianie, że gdy nieuchronnie coś pójdzie nie tak, to nie była ich wina.

W szaleńczym pościgu za funkcjonalnościami i zamkniętymi zadaniami, zespoły często idą na skróty. W rzeczywistości oznacza to, że pozwalają na pogorszenie jakości (szczególnie jakości kodu). To z kolei sprawia, że wymieniają przyszłą produktywność na iluzję postępu w teraźniejszości.

Gdy warunki pracy stają się coraz bardziej nieprzyjemne, kluczowi pracownicy zaczynają się wycofywać, a nawet rozglądać za innymi możliwościami.

Organizacje, które zachowują się w ten sposób, "przejadają ziarno siewne", że tak powiem, na wiele sposobów. Baza kodu staje się coraz trudniejsza w utrzymaniu, a ludzie, którzy mieliby ją utrzymywać, odchodzą.

Jeśli jest w tym jakaś dobra strona, jest ona dla nas niewidoczna.

Rozdział 3: Wprowadzenie do Procesu Dojrzewania Wymagań (Requirements Maturation Flow, RMF)

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

3.1: Czym RMF nie jest

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

3.2: Czym jest RMF

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

3.3: Wspierane i Zalecane Jest Wdrażanie Przyrostowe

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

3.4: RMF 1

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

3.5: RMF 2

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

3.6: RMF 3

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 4: Czy to jest zgodne z Agile?

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

4.1: “Ludzie i interakcje”, “Działające oprogramowanie”

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

4.2: Współpraca z klientem

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

4.3: Reagowanie na zmiany

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

4.4: Przejrzystość

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

4.5: Dopasowanie do Procesu, Zgodność z Agile

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Część II: Tworzenie Przestrzeni dla Gotowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 5: Pierwsze Rozszerzenie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

5.1: Praca Przygotowawcza to *Praca*

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

5.2: Naturalizacja pracy przygotowawczej

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

5.3: Pouczający incydent

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

5.4: Wzajemny Wpływ

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

5.5: Funkcja RMF 1

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 6: Dlaczego Ludzie Tego Nie Robią?

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.1: Praca Przygotowawcza jako Obywatel Drugiej Kategorii

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.2: Alergia na Nieproduktywną Pracę

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.3: Więc Zostało to Ukryte

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.4: Wpływ zarządzania projektami

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.5: Schemat

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.6: Projekty i Szacowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.7: Jak Pseudoszacunki Wpływają na Prace Przygotowawcze

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.8: Mierzenie Prędkości zamiast Wektora Prędkości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.9: Złe Pomiary, Złe Rezultaty

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

6.10: Gdzie Nie Leży Wina

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 7: Jawna Praca nad Gotowością (FDW 1)

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

7.1: Integracja z Synapse Framework™

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

7.2: Anatomia RMF 1

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

7.3: Zachowanie: Rezerwowanie Pojemności na Współpracę

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

7.4: Artefakt: Element Gotowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

7.5: Działanie: Spotkanie Współpracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

7.6: Zachowanie: Kontynuuj Współpracę do Osiągnięcia Wspólnego Zrozumienia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

7.7: Zachowanie: Zawsze Potwierdzaj Wspólne Zrozumienie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

7.8: Jak RMF 1 zmienia przepływ pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 8: Efekty RMF 1

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

8.1: Życie przed RMF 1

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

8.2: Przed: Czas poświęcony na zrozumienie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

8.3: Po: Czas poświęcony na zrozumienie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

8.4: Życie po przyjęciu RMF 1

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

8.5: Fundamentalne

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 9: Wprowadzanie RMF 1 w praktyce

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.1: Edukacja

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.2: Minimalne wymagania według typu zespołu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.3: Porozumienie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.4: Przygotowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.5: Pilot

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.6: Wdrożenie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.7: Działania następcze

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.8: Ogłaszanie sukcesu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.9: Zachowanie czujności

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.10: A co z “Jak”?

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

9.11: Czas wprowadzić to w życie!

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Część III: Bramkowanie Ukończenia Prac

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 10: Kolejna Potrzeba

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

10.1: Miejsce na Interpretację

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

10.2: Zawężanie Pola do Interpretacji

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

10.3: Trzecia Opcja: Brak “Pola Manewru”

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

10.4: Potencjalny Wpływ na Ukończenie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

10.5: Potencjalny wpływ na realizację

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

10.6: Proponowana alternatywa: Nie pozostawiaj miejsca na błędną interpretację

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

10.7: Korzyści

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

10.8: O obawach przed paraliżem analitycznym

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

10.9: Kolejna potrzeba: Dostosowane Definicje Ukończenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 11: Co ludzie zwykle robią

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

11.1: Jeśli to takie wspaniałe, dlaczego ludzie tego nie robią?

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

11.2: Od studiów do coachingu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

11.3: Przesyt coachingu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

11.4: Jeden ze sposobów podejścia do Definicji Ukończenia: Nie robić jej

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

11.5: Tylko kryteria akceptacji

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

11.6: Globalna Definicja Ukończenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

11.7: Brak Mocy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

11.8: Podsumowanie: Termin “DoD” Jest Częstszy niż Faktyczne Definicje Ukończenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 12: Definiowanie Definicji Ukończenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.1: O Pojedynczym Elemencie Pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.2: Stan Ukończenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.3: Precyzja

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.4: Struktura DoD

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.5: Specyfikacje

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.6: Kryteria Wyjściowe Inżynierii

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.7: Kryteria Wejściowe Produktu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.8: Wiele Części, Jedna Brama

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.9: Przykład

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.10: Mapowanie do Twojego Procesu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

12.11: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 13: Dedykowana Definicja Ukończenia (RMF 2)

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.1: Zasada: Każdy Element Pracy Jest Unikalny

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.2: Zachowanie: Utrzymywanie jednego lub więcej szablonów Definicji Ukończenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.3: Aktywność: Definiowanie Szablonu DoD

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.4: Utrzymanie i Ulepszanie Szablonu DoD

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.5: Wiele Szablonów DoD

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.6: Zachowanie: Używanie Szablonów jako Punktów Wyjścia dla Definicji Ukończenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.7: Zachowanie: Uzgadnianie Spersonalizowanych Definicji Ukończenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.8: Działanie: Definiowanie Definicji Ukończenia dla elementu pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.9: Kolejne Rozszerzenie Przepływu Pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.10: Zachowanie: Dopracuj Definicję Ukończenia przed Rozpoczęciem Implementacji

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.11: Działanie: Analiza Offline w celu Dojrzewania Definicji Ukończenia PBI

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.12: Dodawanie procesu dojrzewania do przepływu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.13: Zachowanie: Śledzenie stopnia ukończenia w elementach pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.14: Dodawanie śledzenia postępu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.15: Zachowanie: Bramkowanie pracy przez stan ukończenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.16: Ćwiczenie: Wykorzystanie Definicji Ukończenia do Określenia Gotowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.17: Jak Wpisuje się Bramkowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

13.18: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 14: Życie z RMF 1 i 2

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

14.1: Koszt

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

14.2: Harmonogramy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

14.3: Wpływ na zespół wdrożeniowy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

14.4: Wpływ na Właściciela Produktu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

14.5: Wpływ na Kierownictwo

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

14.6: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 15: Wdrażanie RMF 2

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

15.1: Zaangażowanie Interesariuszy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

15.2: Poziom Szczegółowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

15.3: Porozumienie robocze

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

15.4: Prace początkowe

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

15.5: Wdrożenie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

15.6: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Część IV: Bramkowanie Implementacji

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 16: Ostatni wymóg

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

16.1: To było tam przez cały czas

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

16.2: Siła odpowiedniego momentu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

16.3: Odwracając Perspektywę

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

16.4: Ryzyka i koszty

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

16.5: Wartość Czekania na Gotowość

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

16.6: Dodatkowa Korzyść

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

16.7: Określenie Problemu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

16.8: Potrzeba

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

16.9: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 17: Tło Definicji Gotowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

17.1: Pozwolenie na Pracę w Lean

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

17.2: Kryteria Wejścia w Kolumnach Kanbana

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

17.3: Scrum i inne Apokryfy Procesów Agile

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

17.4: Surfing > Kodowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

17.5: Rozwiązania Punktowe

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

17.6: Jesteśmy Gotowi, by Być Gotowymi

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 18: Definiowanie Definicji Gotowości

18.1: Cel

18.2: Dedykowany charakter

18.3: Anatomia

18.4: Przykład

18.5: Porozumienie

18.6: Bramkowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

18.7: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 19: Dedykowana Definicja Gotowości (RMF 3)

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.1: Kolejna Bramka w Procesie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.2: Struktura Definicji Gotowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.3: Kryteria Wyjścia Produktowe

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.4: Kryteria Wejściowe dla Inżynierii

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.5: Duplikacja Nie Zaszkodzi

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.6: Zachowanie: Utrzymywanie Jednego lub Więcej Szablonów DoR

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.7: Dlaczego Warto Mieć Szablon Definicji Gotowości?

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.8: Ćwiczenie: Definiowanie Szablonu DoR

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.9: Utrzymywanie Szablonów Definicji Gotowości w czasie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.10: Szablony są tylko punktem wyjścia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.11: Zachowanie: Uzgodnienie Spersonalizowanych Definicji Gotowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.12: Aktywność: Definiowanie Definicji Gotowości dla Elementu Pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.13: Zachowanie: Zapewnienie Gotowości Elementów przed Rozpoczęciem Implementacji

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.14: Warunki Poza Kontrolą Zespołu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.15: Zachowanie: Śledź Gotowość w Elementach Pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.16: Zachowanie: Kontrolowanie Pracy przez Gotowość

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.17: Aktywność: Wykorzystanie DoR do Określenia Gotowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

19.18: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Część V: Synteza

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 20: Większość terminów nie ma znaczenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.1: Prawdziwe terminy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.2: Arbitralne terminy nie mają znaczenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.3: Linie lotnicze

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.4: Pochodzenie długu technicznego

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.5: Terminy w Marketingu i Sprzedaży

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.6: Terminy Projektowe

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.7: Jest Inna Droga

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.8: Arbitralne Terminy Nie Są Konieczne

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.9: Arbitralne Terminy Muszą Zostać Zniesione

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.10: Prawdziwe Terminy Nadal Są Istotne

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.11: *Pis Aller*[1]

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

20.12: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

[1] Ostateczność, pierwotnie z języka francuskiego.

Rozdział 21: Kompetencja 1: Przepływ Dojrzewania Wymagań

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

21.1: Jak na Dole, Tak na Górze

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

21.2: Zasada: Przejrzystość Całości Niezbędnej Pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

21.3: Zachowanie: Odpowiedzialność Podąża za Pracą

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

21.4: Zachowanie: Widoczne Śledzenie Stanu Wymagań

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

21.5: Zachowanie: Ujawnij Wszystkie Prace Związane z Gotowością i Wdrożeniem

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

21.6: Ćwiczenie: Praca Przygotowawcza

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

21.7: Zachowanie: Ujawnij Całą Pracę Niezbędną do Ukończenia Elementu Pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

21.8: Zachowanie: Przedkładaj Gotowość nad Terminy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

21.9: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 22: Jak Przebiega Przepływ Pracy i Informacji w Scrumie z RMF

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

22.1: Uwagi wstępne

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

22.2: Przechwytywanie i Przygotowanie Wstępnego Wymagania

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

22.3: Inicjowanie Prac nad Gotowością

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

22.4: Planowanie i Realizacja Prac nad Gotowością

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

22.5: Przegląd Rezultatów Gotowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

22.6: Planowanie i Ukończenie Implementacji

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 23: Wpływ RMF

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

23.1: Życie wymagania

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

23.2: Przykładowy Przepływ Informacji i Pracy

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

23.3: Przed

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

23.4: Po

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

23.5: Korzyści

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 24: Przejście na PDW

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

24.1: Wzorzec Adopcji

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

24.2: Instalacja w Procesie Scrum

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

24.3: Inne Frameworki

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

24.4: Główny Opór: Elementy Pracy Związane z Gotowością

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

24.5: Wielka Zmiana: Sposób Myślenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

24.6: Porady dotyczące Zmian

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

24.7: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozdział 25: Teraz wszystko zależy od Ciebie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

25.1: Podsumowanie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

25.2: Teraz Twoja kolej

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Część VI: Materiały

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Dodatek A: Scrum nie jest problemem

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Czym jest Scrum?

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Frameworki

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Scrum Zajmuje się Zarządzaniem Projektami i Pracą

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Szaleństwo traktowania Scruma jak struktury zarządzania produktem

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Brak określonego mechanizmu dojrzewania wymagań

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Brak inwestycji wiedzy inżynierskiej w rozwój wymagań

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Antywzorce

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Wymagane Rozszerzenie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Rozszerzanie Scruma

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Dodatek B: The Synapse Framework™

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Co obejmuje Synapse

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Trzy Poziomy Mistrzowskie

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Jak przyjmowany jest Synapse

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Łączenie dwóch frameworków

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Struktura Frameworka Synapse

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Biegłości Organizacyjne

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Kompetencje Organizacyjne

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Nawyki Organizacyjne

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Struktura Frameworka Synapse

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Anatomia Praktyki

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Znaczenie Kolejności

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Wpływ Synapse na tę książkę

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Dodatek C: Typowe zastrzeżenia i przeszkody w stosowaniu RMF 1

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Typowe zastrzeżenia

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Typowe Przeszkody

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Dodatek D: Startowe Listy Kryteriów DoD

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Startowa Lista Kryteriów Wyjściowych Inżynieryjnych

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Lista Startowa Kryteriów Wejściowych Produktu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Załącznik E: Startowe Listy Kryteriów Definicji Gotowości

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Kryteria Wyjściowe Produktu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Kryteria Wejściowe dla Inżynierii

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Kryteria Wejścia do Sprintu

Ta treść nie jest dostępna w wersji przykładowej. Książkę można kupić na Leanpub pod adresem https://leanpub.com/ready-pl.

Spis treści