

SCRUM / KANBAN

Leszek Grocholski

Inżynieria oprogramowania, 2023r.

SCRUM/KANBAN

Literatura:

- <http://www.poddrzewem.pl>
- <http://www.scrumvival.com>
- <http://www.scrum.org>

- Scrum Guide. Przewodnik po Scrumie. Ken Schwaber, Jeff Sutherland, 2013
- Scrum Shortcuts Without Cutting Corners: Agile Tactics, Tools & Tips. Ilan Goldstein
- Delight Their Customers, And Leave Competitors In the Dust. Ken Schwaber, Jeff Sutherland, Wiley 2012
- Succeeding with Agile: Software Development Using Scrum. Mike Cohn, Addison–Wesley 2009
- The Mythical Man–Month: Essays on Software Engineering. Frederick P. Brooks, Jr., Pearson, 1975–1995
- The New New Product Development Game. Hirotaka Takeuchi, Ikujiro Nonaka, Harvard Business Review, Jan-Feb 1986

Plan wykładu

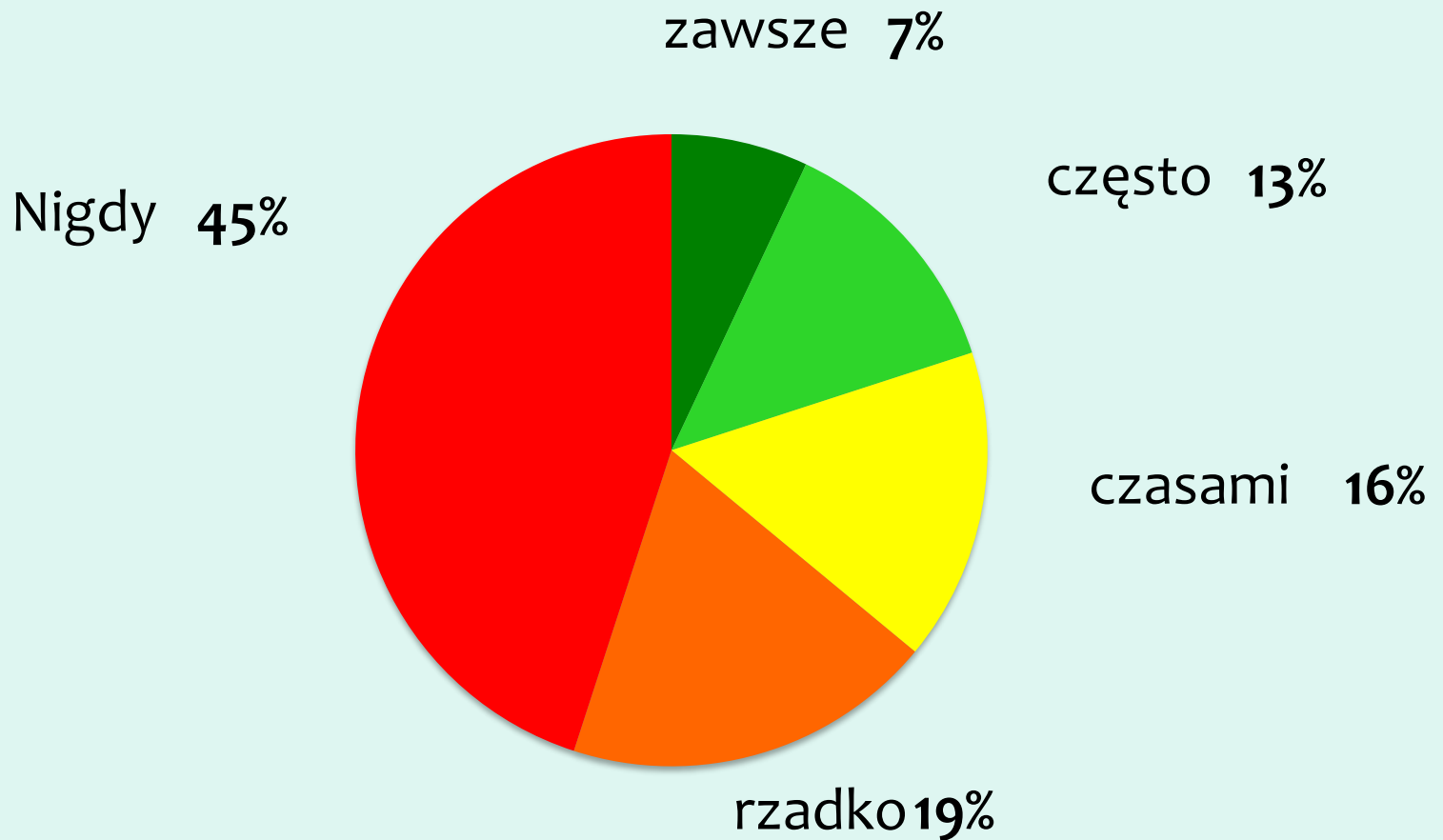
Pochodzenie

Scrum

Kanban

Podsumowanie

Chaos Report: Feature Use



The Standish Group International, 2004

Definicja Scruma

Scrum (rzeczownik): ramy postępowania (ang. framework), dzięki którym ludzie mogą z powodzeniem rozwiązywać złożone problemy adaptacyjne, by w sposób produktywny i kreatywny wytwarzać produkty o najwyższej możliwej wartości.

Scrum jest:

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

Teoria Scruma

Scrum został osadzony w teorii empirycznego sterowania procesem, lub — krócej — w teorii empiryzmu. Empiryzm reprezentuje pogląd, iż wiedza wynika z *doświadczenia i podejmowania decyzji* w oparciu o to, co poznane. Scrum wykorzystuje podejście iteracyjne i przyrostowe w celu zwiększenia przewidywalności i lepszej kontroli ryzyka.

Każda realizacja empirycznego sterowania procesem opiera się na **trzech filarach: przejrzystości, inspekcji i adaptacji**.

Pochodzenie

Prekursorzy

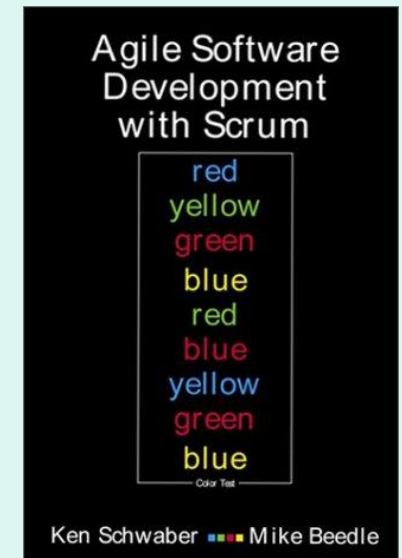
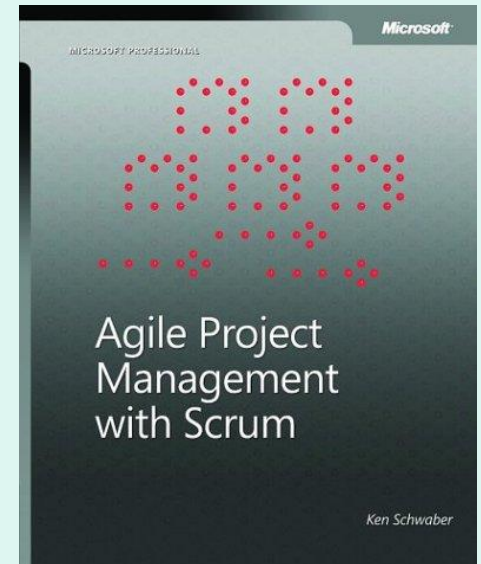
Hiroataka Takeuchi, Ikujiro Nonaka

artykuł: The New Product Development Game (1986)

Scrum – młyn w rugby. Scrum wznawia grę.

Autorzy

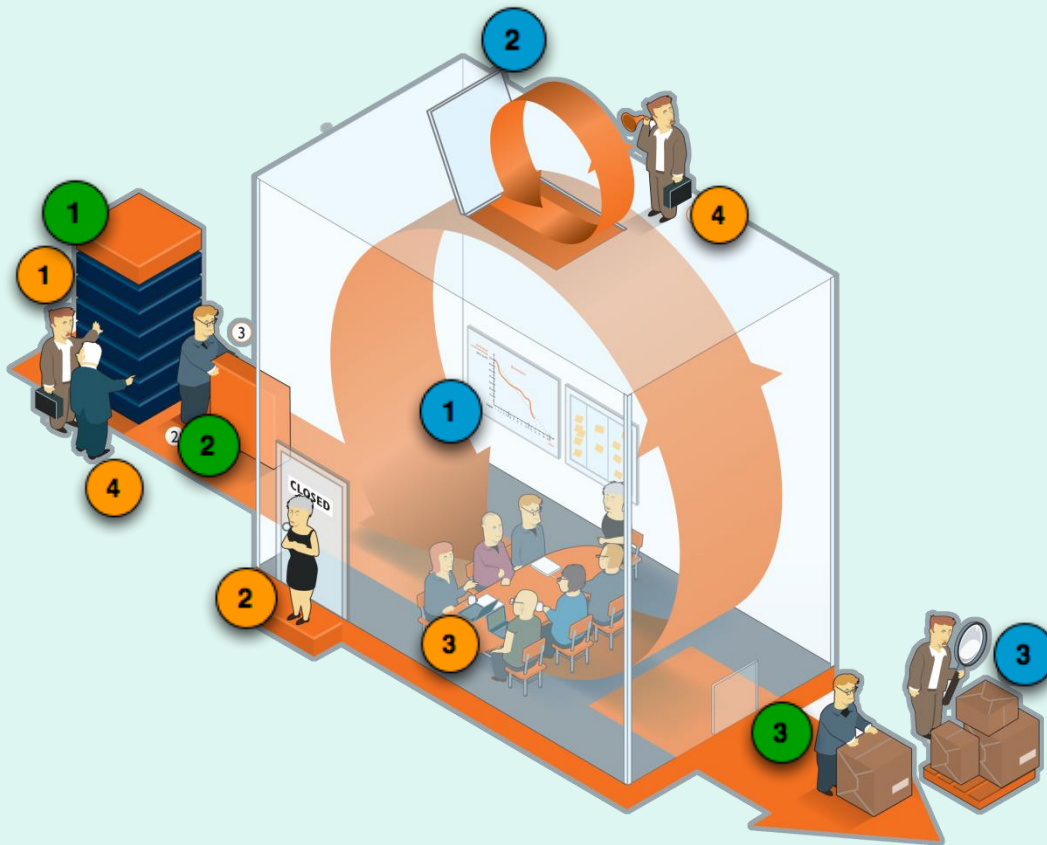
- Jeff Sutherland
 - PatientKeeper
 - Initial Scrums at Easel Corp in 1993
 - IDX and nearly 600 people doing Scrum
 - Not just for trivial projects
 - FDA-approved, life-critical software for x-rays and MRIs
- Ken Schwaber
 - Advanced Development Methods (ADM)
 - Initial definitions of Scrum at OOPSLA 96 with Jeff Sutherland



Wartości Scrum

- **Commitment.** Umiejętność podejmowania zobowiązania rozumiana jako silne poczucie związku z wyrażanymi opiniami i podejmowanymi działaniami oraz konsekwencja w ich realizacji.
- **Focus.** Koncentracja i uważność, bez nich nie jest możliwe ograniczanie się do najważniejszych tematów i dogłębne ich rozumienie. Bez tego przygotowywane rozwiązania charakteryzują się przeciętnością i bylejąkością.
- **Openess.** Otwartość (szczerłość) wskazywana jako gotowość do dzielenia się prawdziwą informacją ze wszystkimi, bez względu na jej charakter – zgodny lub nie – z oczekiwaniami odbiorcy.
- **Respect.** Poszanowanie oznaczające pozytywne odczucia względem innych i umiejętność budowania synergii przy uwzględnieniu różnic w doświadczeniu, wykształceniu i kulturze oraz cechach charakteru.
- **Courage.** Odwaga (co do wprowadzania zmian) jest elementem niezbędnym do ciągłego doskonalenia się. Scrum domaga się zmiany dotychczasowego porządku, przełamywania zwyczajów i sposobów wykonywania pracy. Odwaga potrzebna jest do postępowania zgodnie z nowym modelem pracy.

Cykl wytwarzania oprogramowania wg Scrum



Role

- 1 Product Owner
- 2 Scrum Master
- 3 Członek zespołu Scrum
- 4 Udziałowiec (intersariusz)

Artefakty

- 1 Product Backlog
- 2 Sprint Backlog
- 3 Product Increment

Ceremonie

- 1 Sprint Planning
- 2 Daily Scrum
- 3 Sprint Review

Sprint Retrospective

Role w Scrum

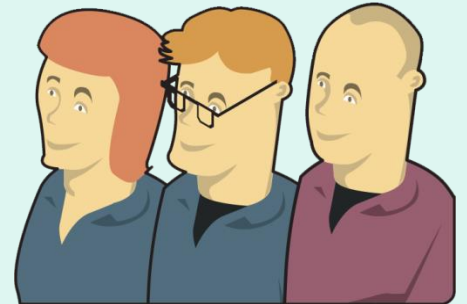
Scrum Master pomaga organizacji i zespołowi wykorzystać Scrum do osiągnięcia celów.



Product Owner zapewnia wytworzenie tego co organizacja w danej chwili najbardziej potrzebuje i maksymalizuje ROI.



Scrum Team wytwarza oprogramowanie i organizuje sobie prace w ramach Scrum.



Scrum Master



- **Odpowiedziany za:**
 - **Sukces Scrum**
 - **Wprowadzenie i postępowanie zgodne z praktykami Scrum i ról**
 - **Monitorowanie prac**
 - **Usuwanie przeszkód**
 - **Organizację prac, która zachęca do pracy zespołowej, samoorganizacji i odpowiedzialności**

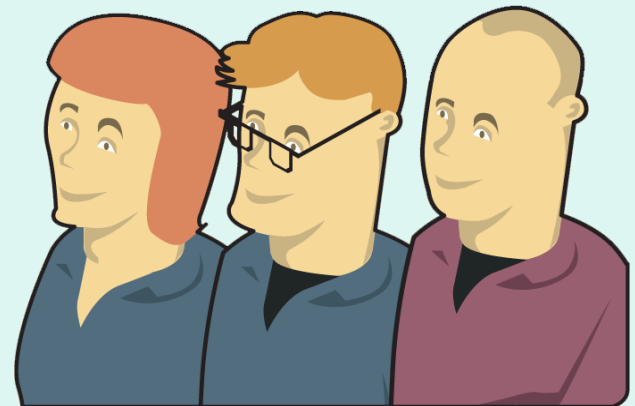
Product Owner



- **Ustala harmonogram produkcji poprzez priorytetowe listy zadań do wykonania (product backlog).**
- **Jest jedną osobą która zapewnia, że tylko jeden zestaw zadań jest realizowany w danym Sprincie.**
- **Eliminuje zamieszanie wywoływane przez wielu szefów, różnych opinie i inne zakłócenia.**
- **Jest jedną osobą, która w wyniku kontaktów z interesariuszami klienta i producenta, określa priorytety zadań.**
- **Stanowi podstawową siłą napędową realizacji wizji produktu i zapewnienia osiągnięcia celu.**

Scrum Team

- 6 +/- 3 osoby
- Cel działania: przyrostowe wytwarzanie oprogramowania
- Samoorganizująca się grupa
- Brak specjalizacji (wszyscy są wytwórcami - developerami)
- Możliwość wszechstronnego rozwoju każdej osoby
- (Samo) Zobowiązani do wykonywania zadań
- Mają prawo (i władzę), robić wszystko co potrzeba aby wykonać zadania

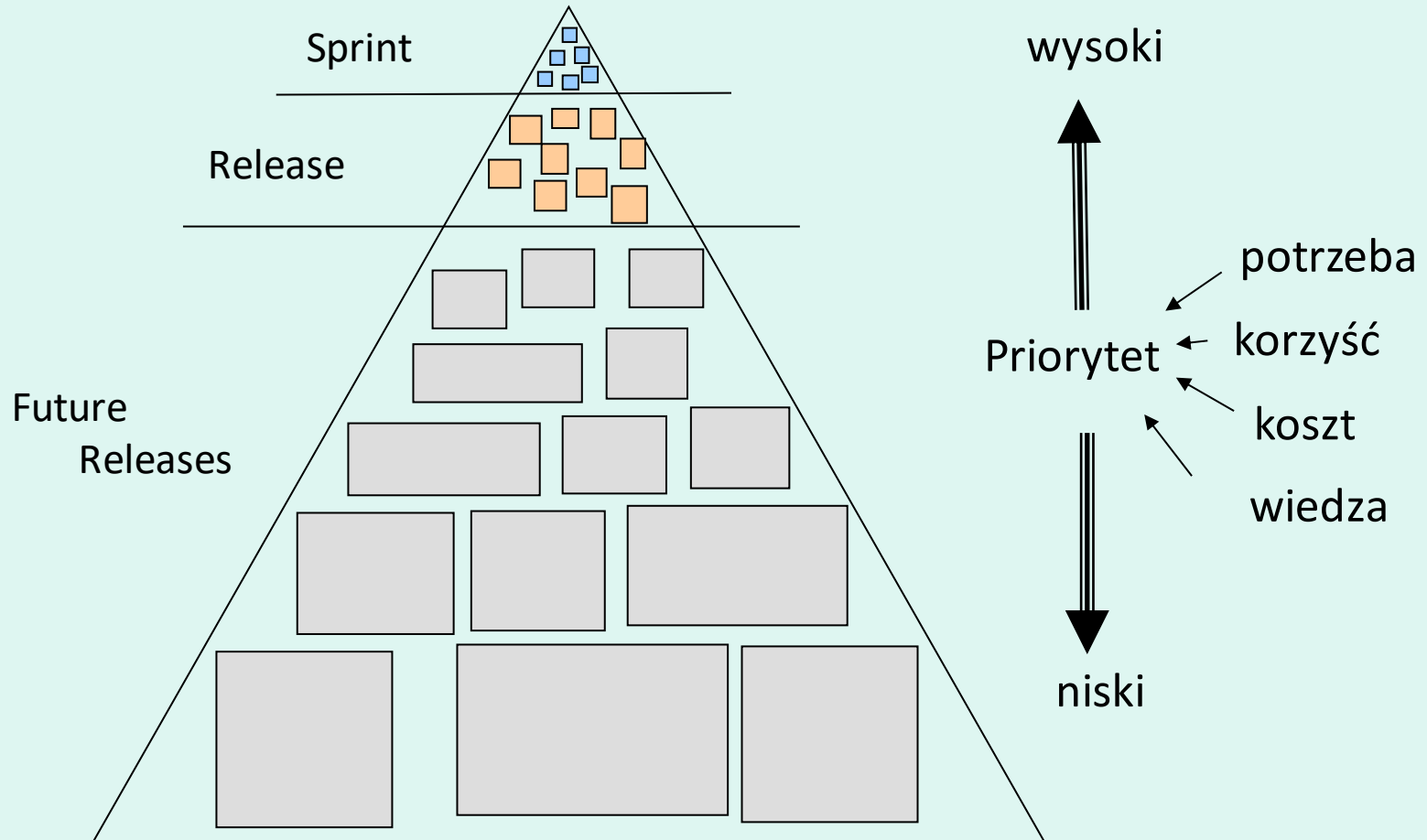


Product Backlog

- **Lista wszystkich wymaganych zadań (prac) dotyczących produktu – np:**
 - dostarczenie funkcjonalności (“let user search and replace”)
 - modyfikowanie funkcjonalności
 - wykonanie zadania (“upgrade to Oracle 11”)
 - wykonanie testów i usuwanie błędów
- **Lista jest porządkowana przez Product Ownera wg zadań/prac najbardziej potrzebnych w danej chwili**
- **Pozycje listy ciągle ulegają zmianie (dodawanie, usuwanie, modyfikowanie)**
 - Pozycje o wyższej pozycji są bardziej szczegółowo opisane
- **Jedna lista dla wielu zespołów**

Góra lodowa Product Backlog

(The Product Backlog Iceberg)



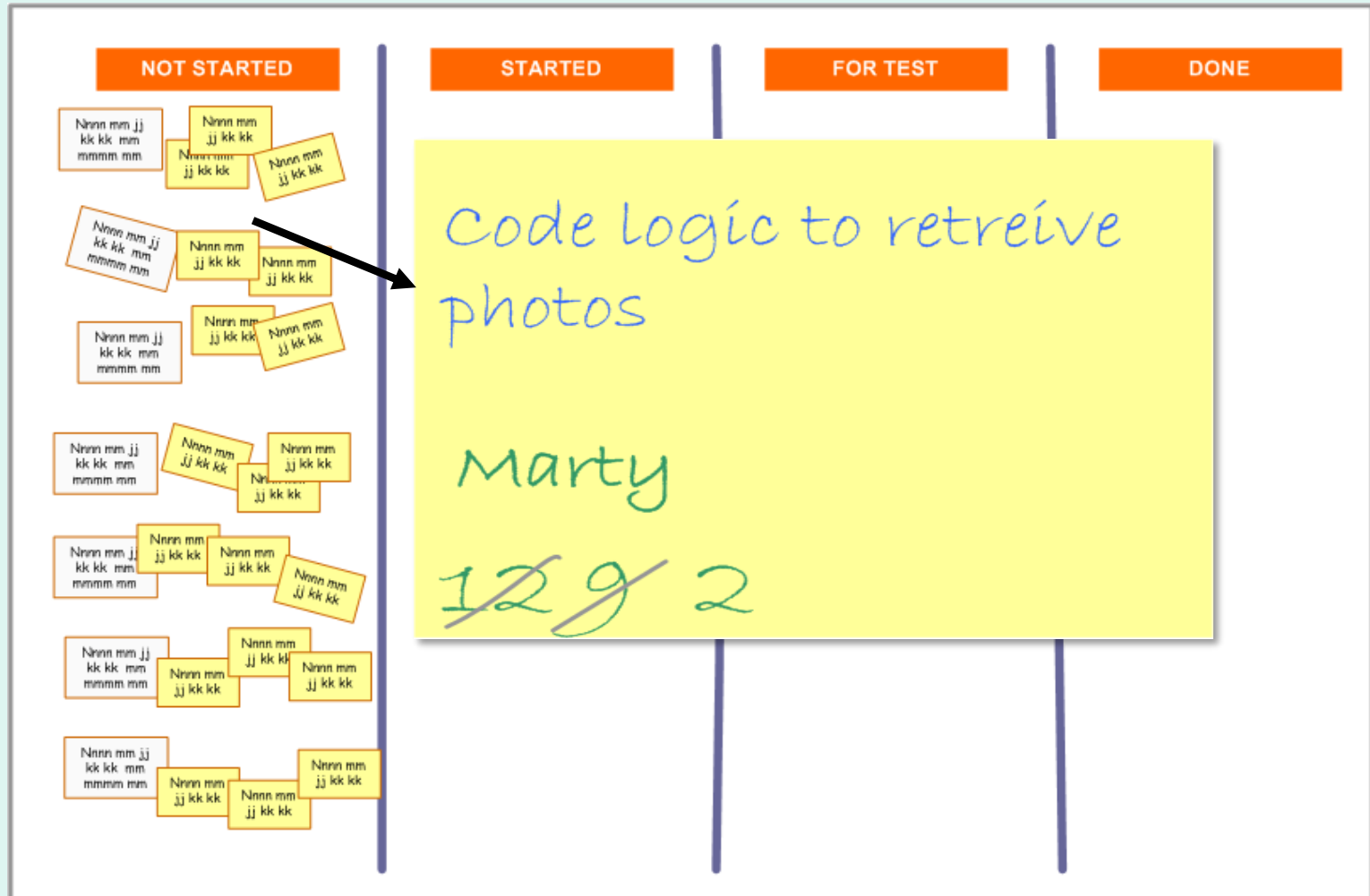
Pozycje Sprint Backlog

Sprint Backlog Items (= TASKS)

- **Zadania, które umożliwiają transformację pozycji Backlog w produkt (kod, testy, dokumentacja, zmiany konfiguracji, itd. ...)**
- **Pracochłonność 4-16 godzin. Jeżeli większa to podział zadań na mniej pracochłonne.**
- **Sprint Backlog jest tworzony w zespole i jego właścicielem jest Scrum Team.**
- **Lista zadań pozostałych do wykonania jest codziennie uaktualniana.**
- **Tylko zespół może dodać elementy do Sprint Backlog.**
- **Członkowie zespołu przypisują się do zadań zaraz po rozpoczęciu sprintu.**

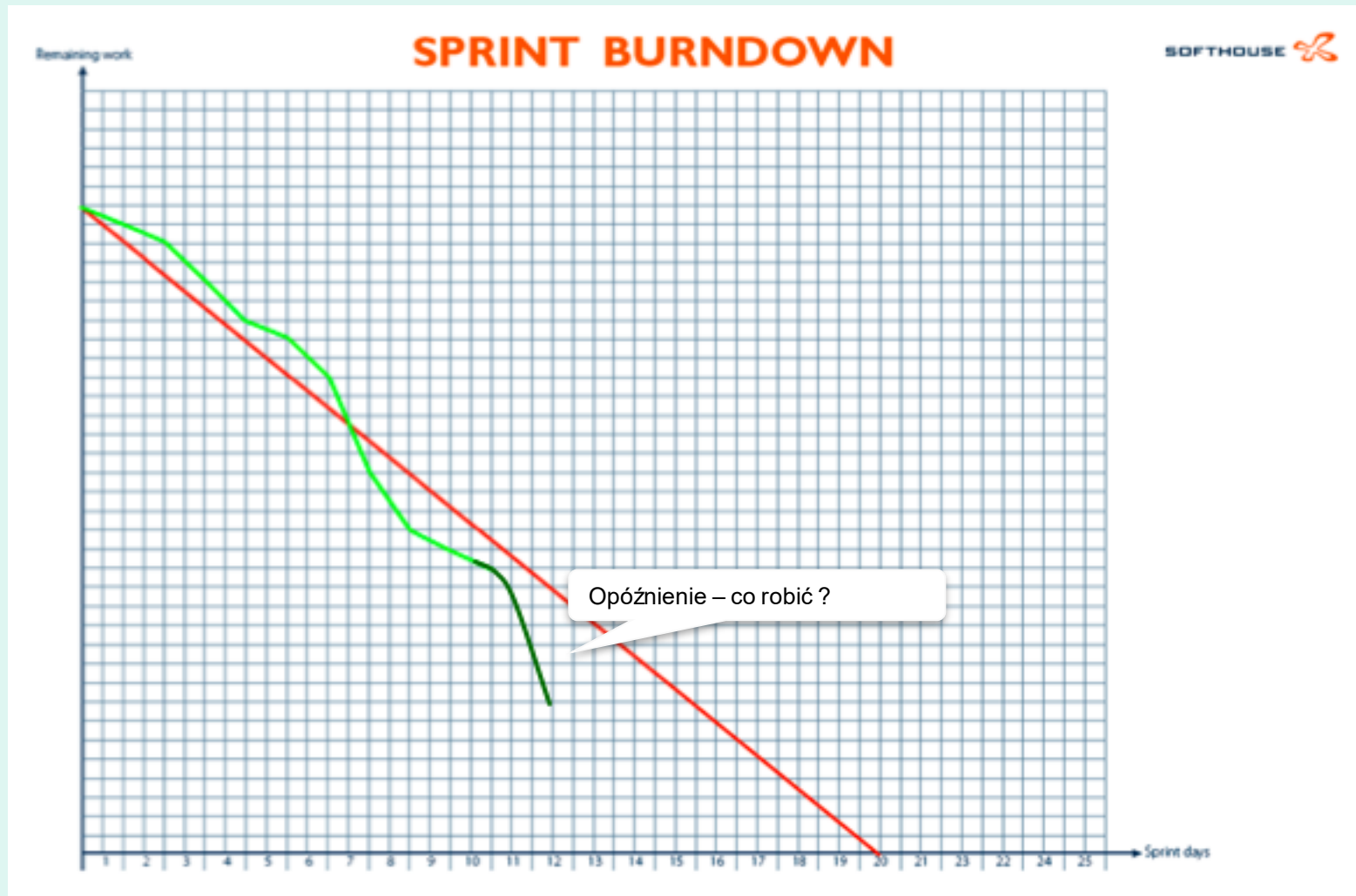
Tablica zadań

(Task board)



WYKRES WYPALANIA

(Sprint Burndown Chart)



Planowanie Sprintu

(Sprint Planning)

Uczestnicy:



Cel:

- przydzielenie pozycji z product backlog i zdefiniowanie celu sprintu
- utworzenie sprint backlog
- osiągnięcie porozumienia co do celu sprintu i przydziału zadań

Przebieg:

- Product Owner prezentuje wizję produktu
- wybór pozycji z product backlog
- product owner i Scrum team definiują cel sprintu
- -//- określają prędkość zadań sprint backlog
- -//- określają harmonogram prac (ludzie, zdania, czas)
- Członkowie zespołu wzajemnie zobowiązują się wykonać zadania

CODZIENNY SCRUM

(Daily Scrum)

Uczestnicy:



Cel:

- synchronizacja pracy zespołu
- przy okazji zapewnienie przejrzystości

Przebieg:

Czas trwania, maks 15 minut, na stojąco!

Każdy członek zespołu odpowiada na następujące 3 pytania

- które zadania skończył od ostatniego spotkania ?
- które zadania planuje skończyć przed następnym spotkaniem?
- które zadanie aktualnie wykonuje
- oraz zgłasza ew. problemy

Uaktualnienie tablicy zadań, wykresu wypalania

Identyfikacja problemów

**Kluczowym wynikiem jest synchronizacja,
Określenie stanu jest uzyskiwane przy okazji.**

Przegląd Sprintu

(Sprint Review)

Uczestnicy:

udziałowcy +



Cel:

przegląd wszystkich zadań wykonanych w danym sprint
sprawdzenie czy cel sprintu został osiągnięty
przedstawiane jest tylko działające oprogramowanie

Przebieg:

zespół prezentuje wyniki pracy
zespół demonstruje nowe/zmodyfikowane funkcjonalności
Product owner + representanci klienta oceniają wyniki

Ocena Sprintu

(Sprint Retrospective)

Uczestnicy:



Cel:

Umożliwienie oceny co było dobre a co złe w sprincie.

Uczenie się na podstawie doświadczenia

Osiągnięcie porozumienia zespołu co do priorytetu i kolejnych działań naprawczych

Przebieg:

Omówienie tego jak postępowano

Znalezienie odpowiedzi na 3 pytania

- jakie praktyki powinniśmy **kontynuować**?
- co musimy **zaprzestać** robić ?
- co powinniśmy **zacząć** robić?

Zbudowanie plany działań naprawczych w przyszłym sprincie

Scrum



Łatwy do zrozumienia
ale
trudny to opanowania ...

Co jest trudne w SCRUMie?

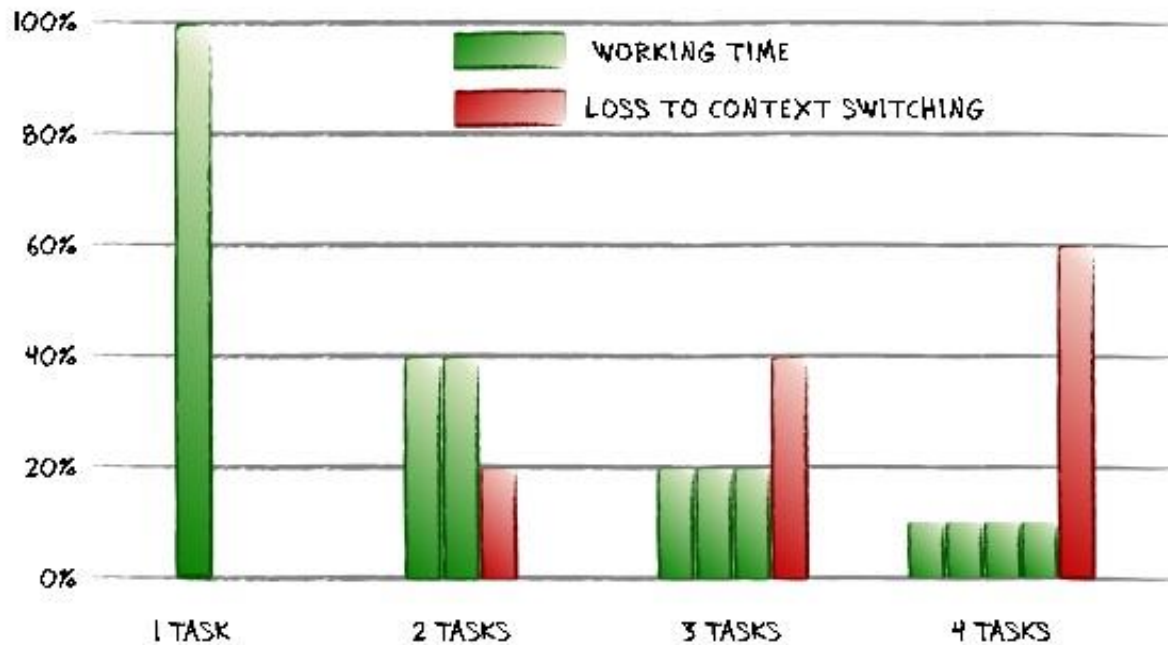
- Zapewnienie ciągłej dostępności przedstawicieli klienta
- Zbudowanie samo-organizującego się, składającego się z członków o wszechstronnych umiejętnościach zespołu.
 - Scrum nie da się wdrożyć, jeżeli nie ma woli pracy zespołowej.
- Zapewnienie udziału naprawdę zaangażowanego Product Ownera
 - Scrum nie da się wdrożyć jeżeli klient/Product Owner nie rozumie specyfiki pracy w Scrum i/lub nie zapewni odpowiednich zasobów w czasie przedsięwzięcia
- Wiarygodne szacowanie pracochłonności zadań i ponownych oszacowań.
- Synchronizacja pracy zespołów międzynarodowych
 - Trzeba zapewnić ten sam sposób podejścia do pracy, i stosowanie tych samych narzędzi.

(dowcipna) zagadka

- Co programistom najbardziej przeszkadza w pracy?


Koszt wielozadaniowości

Cost of multitasking



Source: Gerald Weinberg, Quality Software Management: Vol. 1 System Thinking

lunar logic

 **kanbanery**



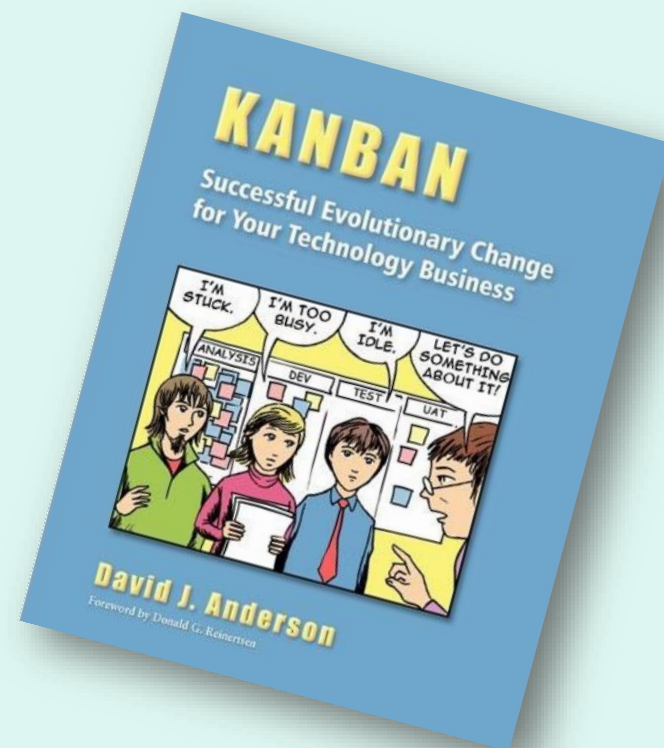
看板 – Kanban Background

- 看板 – W pierwotnym znaczeniu – w języku japońskim – oznacza: „szyld”, „tabliczkę z napisem informującym”, „billboard”.
- Charakterystyczną cechą tej metody jest praktyczna likwidacja magazynów przedprodukcyjnych (cały zapas znajduje się na stanowisku roboczym), międzyoperacyjnych i wyrobów gotowych.
- **Metoda KANBAN opiera się na poszczególnych kartach wyrobów, ich cyrkulacji i analizie.**
- Kanban jest przejrzystym, ograniczającym prace i zapasy, zaciągającym prace system organizacji pracy.



KANBAN w wytwarzaniu oprogramowania

- Zastosowanie Kanban do wytwarzania oprogramowania, zostało pierwszy raz opisane w książce przez Davida Andersona w roku 2007.
- Książkę opublikowano w kwietniu 2010
- Zawartość - ok. 72,000 słów wprowadzających w tematykę Kanban



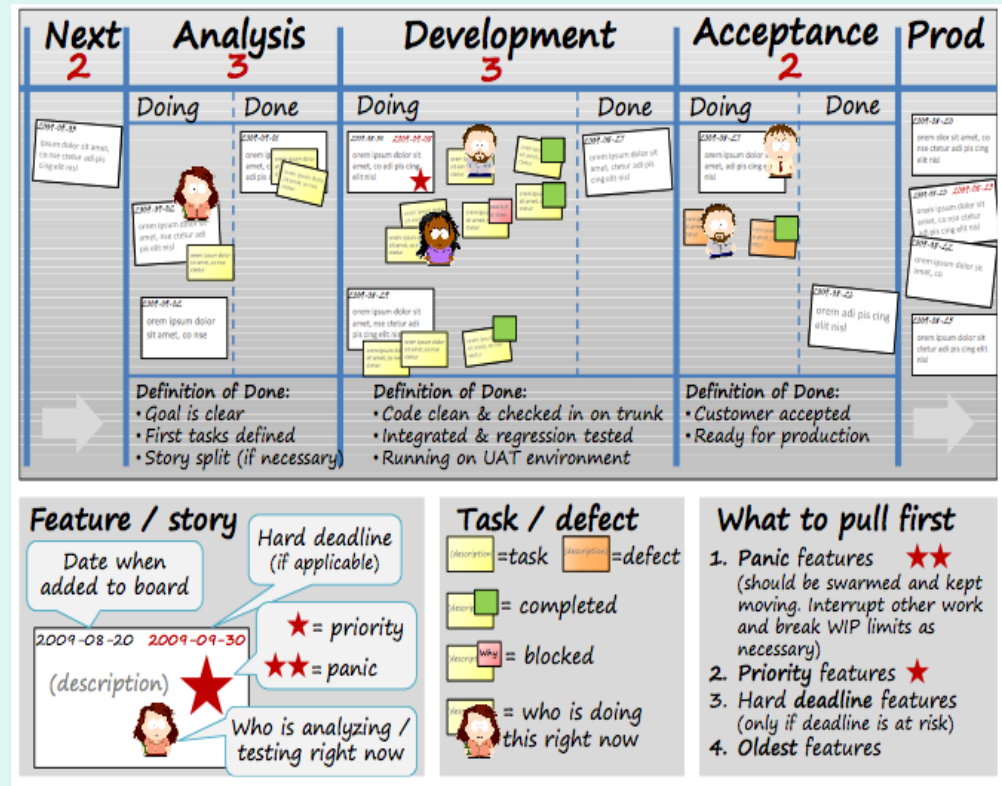
Kanban w pigułce

„płytkie
wdrozenie

Podstawowe praktyki Kanban :

- wizualizacja pracy
- ograniczanie Work in Progress (WIP) pracy w toku
- śledzenie przepływu pracy
- oczywista, prosta organizacja pracy
- stymulacja pracy zespołowej

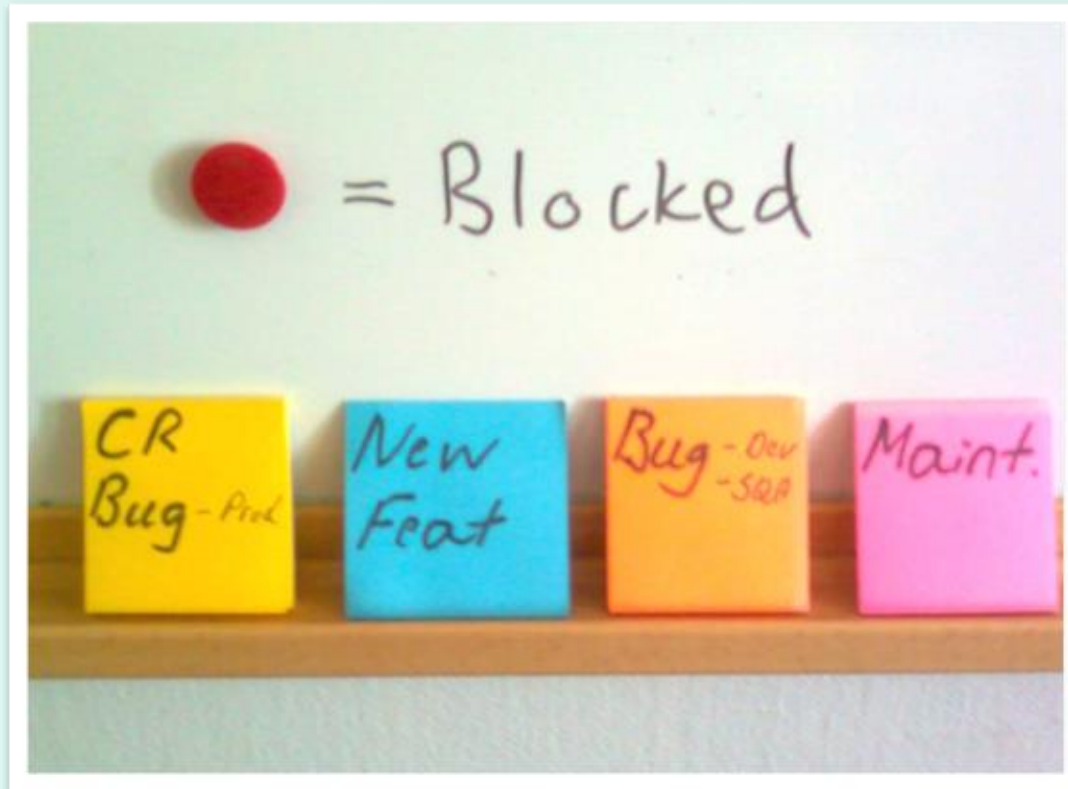
„głębokie
wdrozenie



5 podstawowych praktyk zaobserwowanych
w udanych wdrożeniach Kanban

Karty

Różne nazwy: karty, żetony, żelki, bilety, tikety

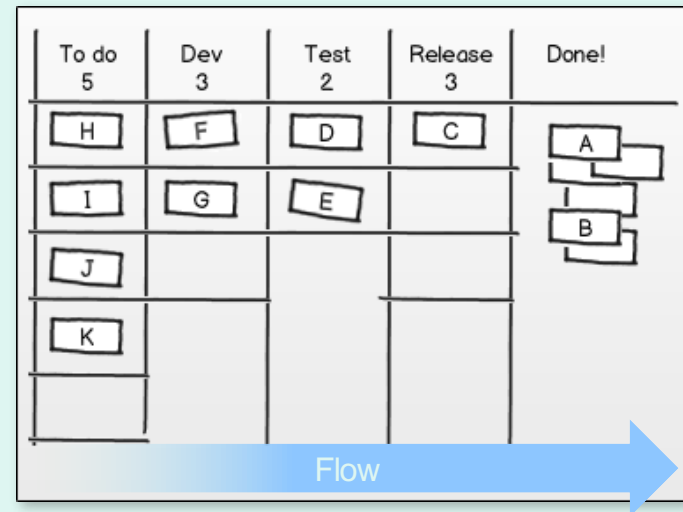
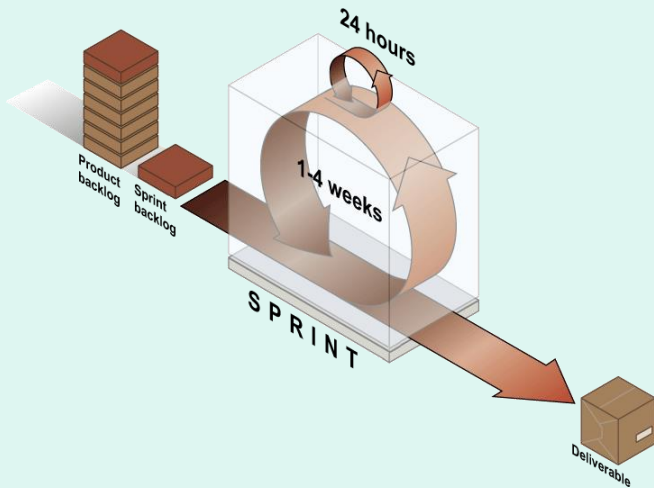


Przykład tablicy Kanban - White Board



Scrum vs. Kanban podobieństwa

- Praca jest zaciągana przez osoby.
- Ogranicza prace w toku (WIP work in progress).
- Wizualizuje planowaną, wykonywaną, wykonywaną pracę.
- Stymuluje doskonalenie procesu wytwarzania.
- Zorientowany na dostarczanie wydań oprogramowania szybko i często.
- Wymaga podziału prac na „małe” zadania.



Scrum vs. Kanban

różnice

Scrum	Kanban
Praca zorganizowana w iteracje (sprinty).	Interakcja nie jest w standardzie.
Wymaga porozumienie zespołu co do wspólnego wykonania w iteracji określonych prac.	Zespół niezależnych pracowników. Porozumienie zespołu co do wspólnego wykonywania prac nie jest w standardzie.
Stosuje pracochłonność jako domyślną metrykę planowania oraz doskonałości procesu.	Stosuje faktyczny czas realizacji jako domyślną metrykę planowania i doskonalenia procesu
Zalecane wszechstronne zespoły.	Wszechstronne zespoły mogą wystąpić. Standard mówi o zespołach specjalistów indywidualistów.
Nakazuje 3 role (PO, SM, Development Team).	Żadne role nie są nakazane.
Nakazuje oszacowania pracochłonności.	Oszacowania pracochłonności mogą wystąpić.
Sprint backlog is własnością jednego zespołu	Tablica kanban może być wspólna dla wielu zespołów indywidualistów

Scurum i Kanban są tylko ramami

Wytwarzając oprogramowanie, też zgodnie z Scrum czy Kanban należy realizować następujące etapy:

- zbudowanie wizji,
- analiza,
- projektowanie,
- kodowanie,
- testowanie,
- wdrożenie.

Na pytanie w jaki sposób powyższe etapy realizować, Scrum i Kanban jedynie podpowiadają:

- **w sposób zwinny, tzn elastycznie i skutecznie, wykonując jedynie niezbędne prace zapewniające wytworzenie potrzebnego klientowi oprogramowania.**

Istnieje wiele zwinnych praktyk dot. wytwarzania oprogramowania.

Na ostatnich slajdach omówiono:

- zwinne szacowanie pracochłonności,
- zwinne zbudowanie wizji.

Uwaga: w/w nie są składowymi standardów Scum, Kanban.

SZACOWANIE PRACOCHOŁONNOŚCI

W zwinnym wytwarzaniu oprogramowania zauważa się, że niemożliwe jest precyzyjne (w godzinach, osobodniach) oszacowanie pracochłonności.

„Szacowanie na wysokim poziomie to zgadywanki
(i to zazwyczaj bardzo nieudane, a przy tym zbyt optymistyczne)”.

Bywa, że dokładne oszacowanie jest niemożliwe i musimy przestać udawać, że jest inaczej.

Jedynym pytaniem na które początkowe oszacowania mogą próbować odpowiedzieć na pytania, jest: Czy ten projekt da się wykonać!
(w takim czasie i takimi środkami, jakimi dysponujemy)

Ale do zbudowania planu wytwarzania szacowanie pracochłonności (w tym też planu współpracy z klientem) jest bardzo ważne.

To czego potrzeba w zwinnym wytwarzaniu to sposób szacowania, który:

- umożliwia planowanie przyszłości;
- przypomina nam, że nasze szacunki to tylko zgadywanie;
- bierze pod uwagę wewnętrzną złożoność tworzenia oprogramowania.

SZACOWANIE PRACOCHOŁONNOŚCI ...

!! 1 dzień przeliczeniowy <> 1 dzień kalendarzowy !!

Badania psychologiczne) pokazały, że ludzie są dobrzy w szacowaniu względnym. Grupowe szacowanie względne prowadzi do lepszych wyników.

Szacowanie zwinne potrzebuje dwóch rzeczy:

- wzorcowych zadań, których rozmiary można porównywać z innymi zadaniami
- systemu punktowego szacowania pracochłonności i śledzenia postępów.

W metodach zwinnych zaleca się zamknięcie oszacowań w prostym, łatwym do wykorzystania systemie punktowym i unikanie wiązania ich z czasem kalendarzowym.

Skala punktowa 1, 3, 5 punktów. Nie więcej. Duże zadania dzielimy na mniejsze.

Planistyczny poker to gra, w której zespół programistyczny szacuje najpierw zadania indywidualnie (1,3, 5 pkt) a następnie wspólnie porównuje wyniki.

Rzeczywista szybkości pracy osoby i zespołu mierzymy w punktach.

Te dane używamy do planowania przyszłości

WIZJA SYSTEMU

Dzięki wizji członkowie zespołu rozumieją kto, co i dlaczego potrzebuje.

Dlatego mogą myśleć samodzielnie:

- podejmować lepsze, bardziej świadome decyzje,
- w lepszy sposób godzić przeciwieństwa i wypracowywać kompromisy,

I w efekcie tworzyć lepsze, bardziej innowacyjne rozwiązania !

Tablica koncepcyjna – krótkie, najważniejsze informacje o przedsięwzięciu.

Tablica zbudowana po to aby zespół i interesariusze wspólnie wiedzieli:

1. Co zespół buduje i po co?
2. Co jest ważne w projekcie (jakich korzyści spodziewa się klient, dlaczego kupi)?
3. Co jest w zakresie przedsięwzięcia a co poza nim ?
4. Jakie są wymagane najważniejsze funkcjonalności ?
5. Kto jest w sąsiedztwie (interesariusze – osoby u klienta zainteresowane wynikiem) ?
6. Jak wygląda rozwiązanie od strony technicznej (architektura) ?
7. Czego należy się obawiać (ryzyka) ?
8. Jak duży jest projekt (zwymiarowanie wszystkiego) ?
9. Kiedy i gdzie trzeba będzie się elastycznie dostosować (co trzeba odrzucić)?
10. Ile w przybliżeniu to pochłonie (pracowników, czasu i pieniędzy) ?

Survey & Questions?

