Metodyki programowania

Tomasz Kaszuba 2015
kaszubat@pjwstk.edu.pl

Wybrane metodyki zwinne

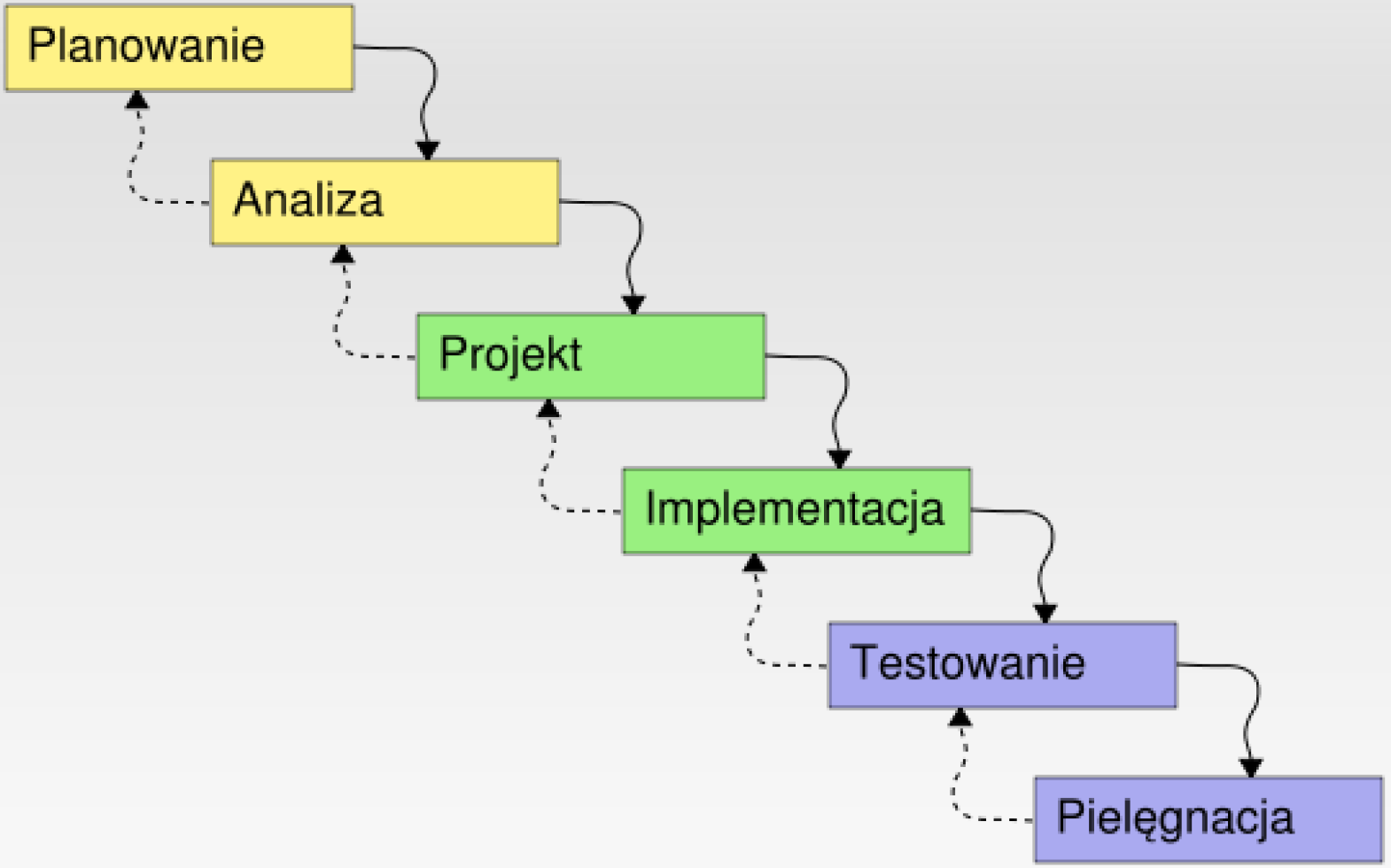
TRADYCYJNE:

- RUP (Rational Unified Process) – spiralny, rozbudowany
- PRINCE2 (Projects In Controlled Environments) – kaskadowy, rozbudowany

ZWINNE:

- Scrum
- XP
- Xprince (miks XP, PRINCE2 i RUP)

Model kaskadowy



Model kaskadowy

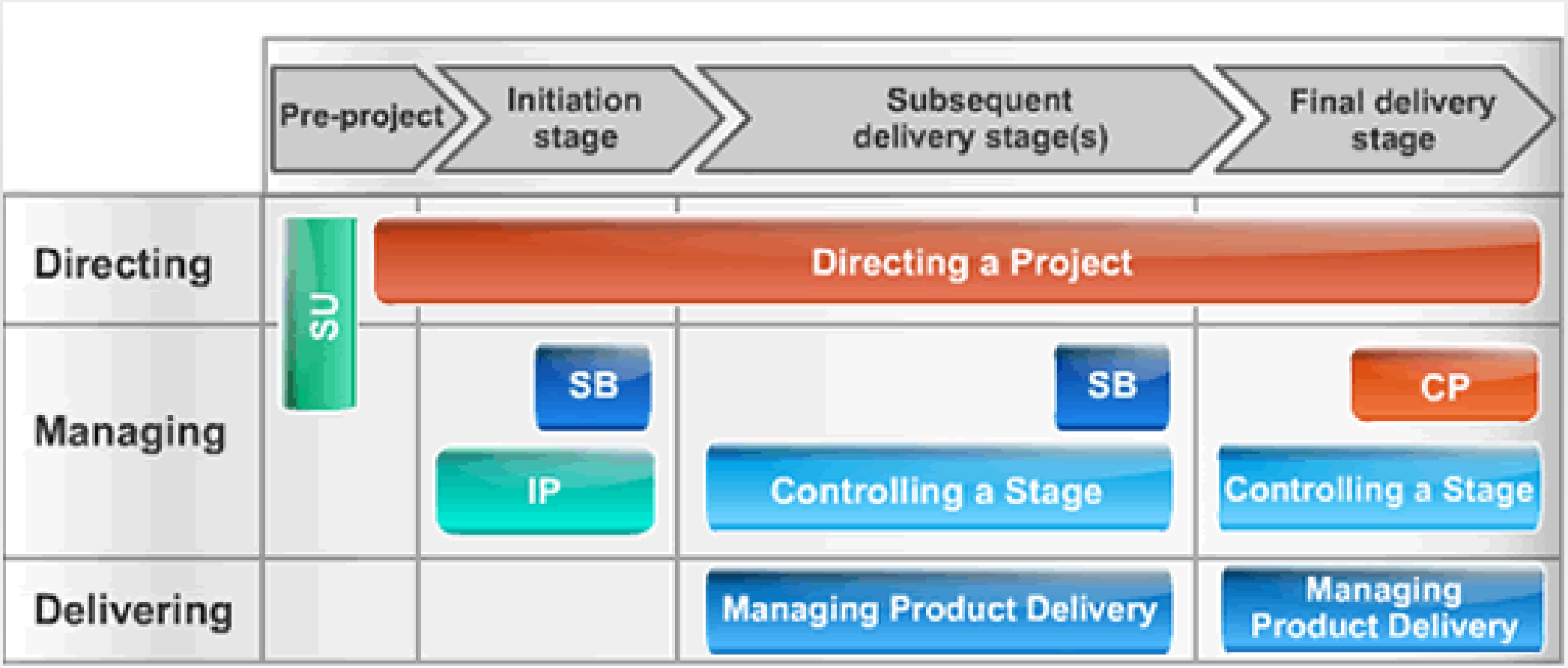
1. Planowanie systemu (w tym specyfikacja wymagań)
2. Analiza systemu (w tym Analiza wymagań i studium wykonalności)
3. Projekt systemu (poszczególnych struktur itp.)
4. Implementacja (wytworzenie kodu)
5. Testowanie (poszczególnych elementów systemu oraz elementów połączonych w całość)
6. Wdrożenie i pielęgnacja powstałego systemu.

- Nie można przejść do następnej fazy przed zakończeniem poprzedniej
- Model ten posiada bardzo nieelastyczny podział na kolejne fazy
- Iteracje są bardzo kosztowne - powtarzamy wiele czynności

PRINCE2

- Strategiczne zarządzanie projektem (ZS) – Directing a project (DP)
- Uruchamianie Projektu/Przygotowanie Założeń Projektu (PP) – Starting up a project (SU)
- Inicjowanie projektu (IP) – Initiating a project (IP)
- Sterowanie Etapem (SE) – Controlling a stage (CS)
- Zarządzanie Wytwarzaniem Produktów (WP) – Managing product delivery (MP)
- Zarządzanie Zakresem Etapu (ZE) – Managing stage boundaries (SB)
- Zamykanie Projektu (ZP) – Closing a project (CP)

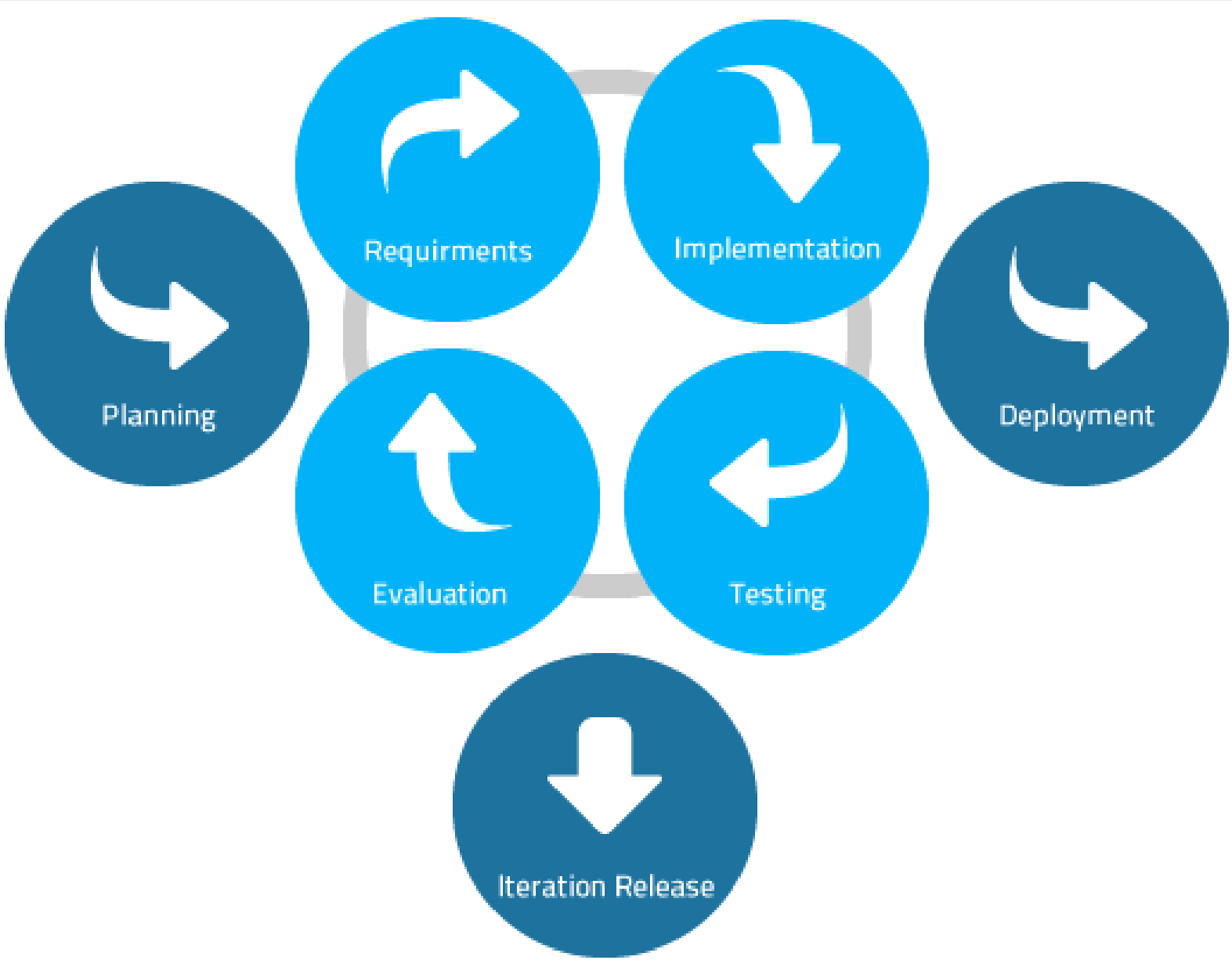
PRINCE2



Based on OGC PRINCE2® material. Reproduced under licence from OGC.

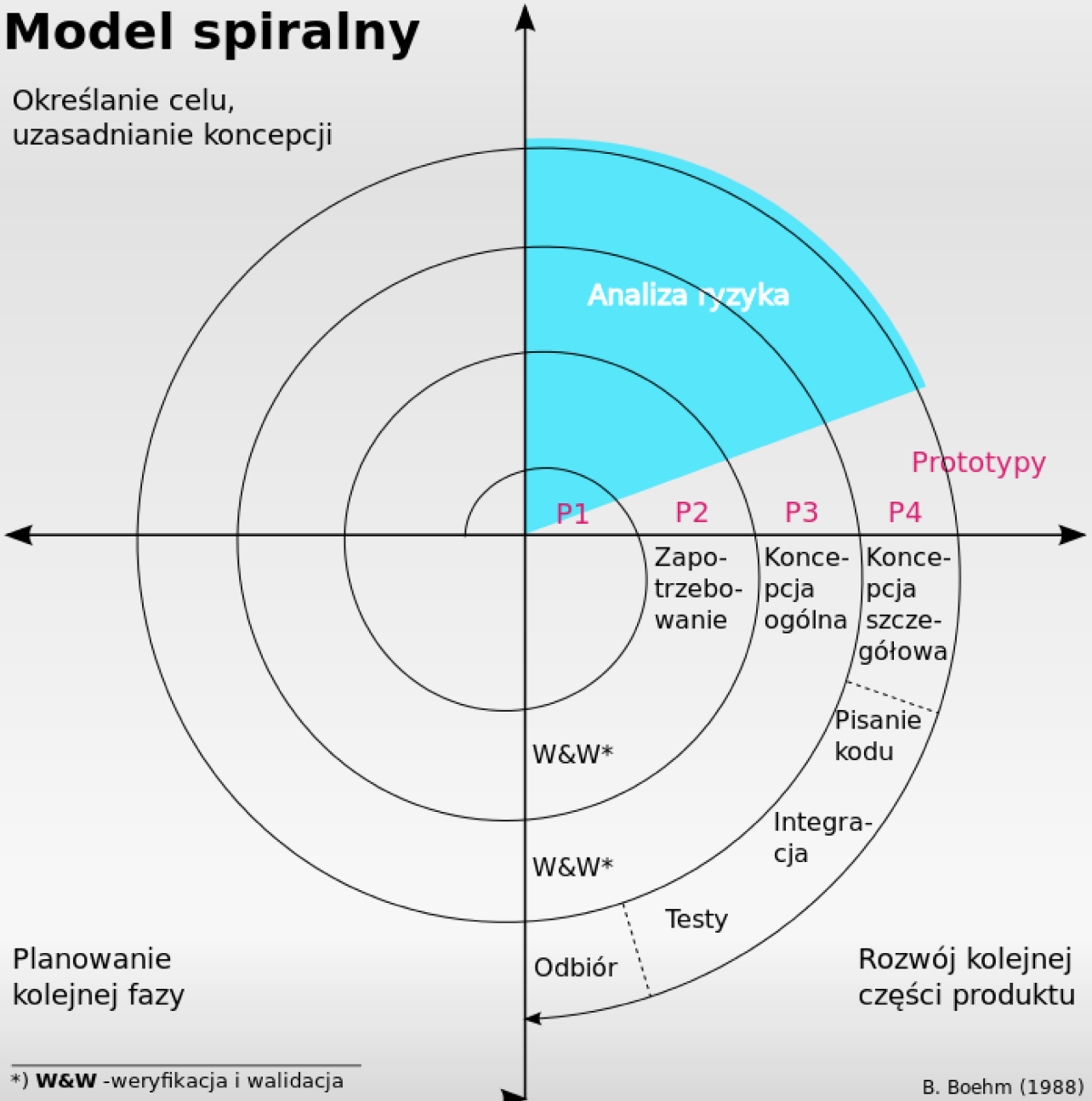
Key
SU = Starting up a Project
IP = Initiating a Project
SB = Managing a Stage Boundary
CP = Closing a Project

Model przyrostowy



Źródła: http://www.meranetworks.com/sites/default/files/IterativeIncrementalDevelopment.png

Model spiralny



W każdym cyklu:

- Ustalanie celów
- Rozpoznanie i redukcja zagrożeń
- Tworzenie i zatwierdzanie
- Ocena i planowanie

Źródła: wikipedia.pl

Model spiralny i przyrostowy

- Częste kontakty z klientem (w porównaniu do modelu kaskadowego)
- Faza oceny w każdym cyklu (pozwala uniknąć błędów lub wcześniej je wykryć)
- Wykorzystanie przez klienta fragmentów systemu (prototypy)
- Brak konieczności zdefiniowania z góry całości wymagań - cały czas istnieje możliwość rozwijania projektu
- Częste kontrole jakości w kolejnych cyklach - nastawienie na wykrywanie błędów i działania kontrolne, a nie na zapobieganie
- Orientacja na zarządzanie, czas i budżet.

- Trudności z określeniem pozbioru funkcji niezależnych (moduły)
- Konieczność implementacji szkieletu – dodatkowy nakład pracy
- Wysoki koszt usuwania błędów wykrytych w finalnych etapach projektu

RUP - Lista najczęstszych błędów

- Zarządzanie wymaganiami ad hoc (najczęściej brak zarządzania nimi)
- Niejednoznaczna, nieprecyzyjna komunikacja
- Architektura oprogramowania nieodporna na obciążenia (ang. Brittle architecture)
- Zbytnia, niepotrzebna złożoność oprogramowania
- Niewykryte niespójności w wymaganiach, projekcie oraz implementacji
- Brak lub niewystarczające testowanie
- Subiektywna ocena postępu projektu
- Brak zarządzania ryzykiem
- Brak automatyzacji prowadzenia projektu

Tradycyjne metodyki - wady

- Długoterminowe planowanie
- Mała elastyczność
- Sformalizowany zapis dokumentacji
- Programiści pełnią rolę drugorzędną

- Trudne zmiany w systemie
- Ograniczony kontakt z użytkownikiem / klientem (zebranie wymagań, testy)
- Brak wykorzystania potencjału programisty
- Rozsynchronizowanie budżetu, czasu pracy i oczekiwań klienta

Manifest agile

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Źródła: http://agilemanifesto.org/

Tradycyjne i zwinne metodyki cd

- Skupianie się na funkcjach oprogramowania (mniej dokumentacji) – oprogramowanie wytwarzane skokowo (za każdym razem dodawana jest działająca funkcjonalność)
- Współpraca z klientem – okresowe dostarczanie fragmentów oprogramowania (części funkcjonalności).
- Szybkie reagowanie na zmiany - częste spotkania kontrolne.

- Brak dokładnej specyfikacji (wiedza o projekcie w zespole)
- Trudne do przewidzenia koszty wytworzenia programu
- Klient musi być gotowy na częstą komunikację
- Potrzeba wysokich kwalifikacji programistów (kreatywność, rozwiązywanie problemów, decyzyjność)

Źródła: http://agilemanifesto.org/

XP – Extreme programming

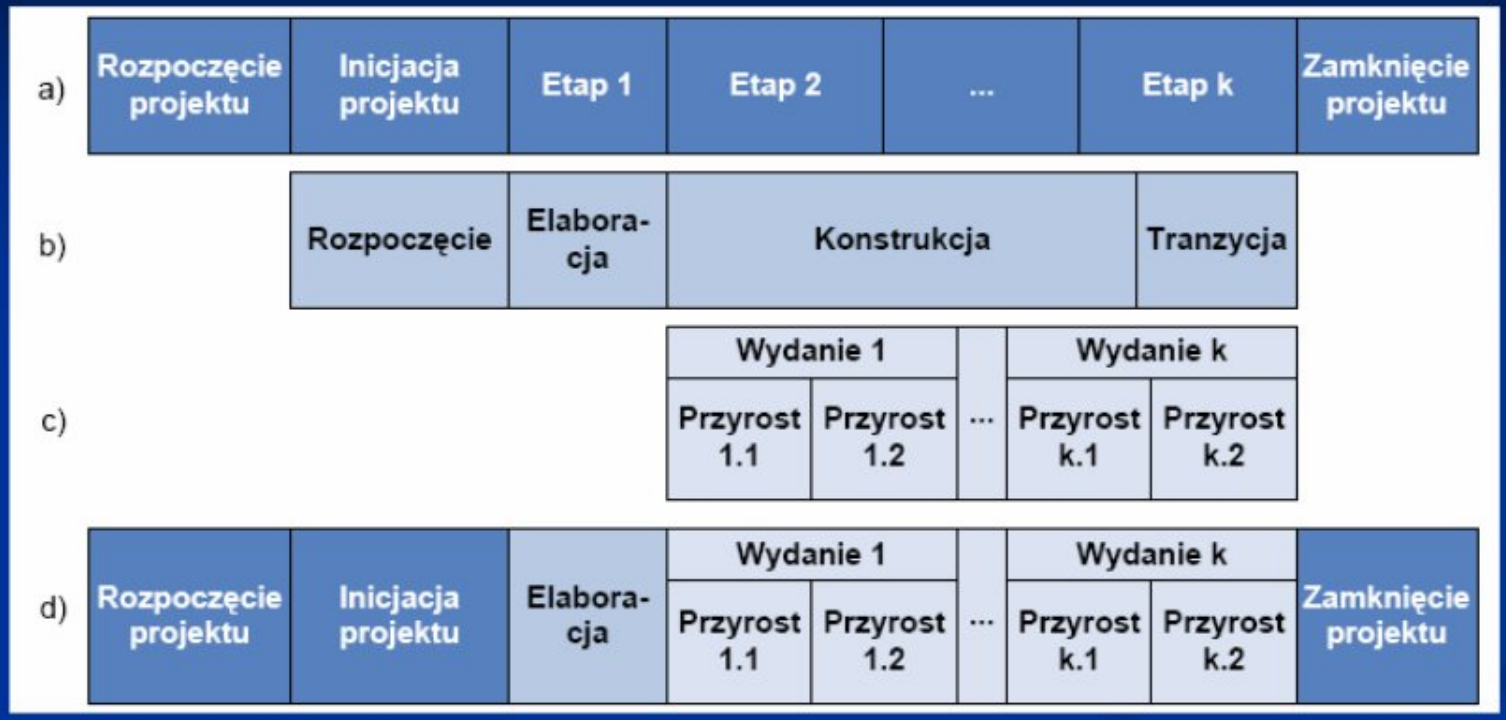
- Iteracyjność (krótkie kroki, bez planowania kolejnych iteracji).
- Brak ustalonej architektury – można zmieniać podczas trwania projektu
- Testy jednostkowe (podobne do Test Driven Developement)
- Programowanie parami (koder i obserwator), zmiany co kilkadziesiąt minut – ułatwia jasność kodu.
- Stały kontakt z klientem (czasem zamiast specyfikacji)

Test-Driven Developement

Fragment metodyki XP

- powstaje automatyczny test sprawdzający dodawaną funkcjonalność
- implementacja funkcjonalności by test się powiódł
- refaktoryzacja napisanego kodu, żeby spełniał on oczekiwane standardy.

XPrince



a) PRINCE2
b) RUP
c) XP
d) XPrince

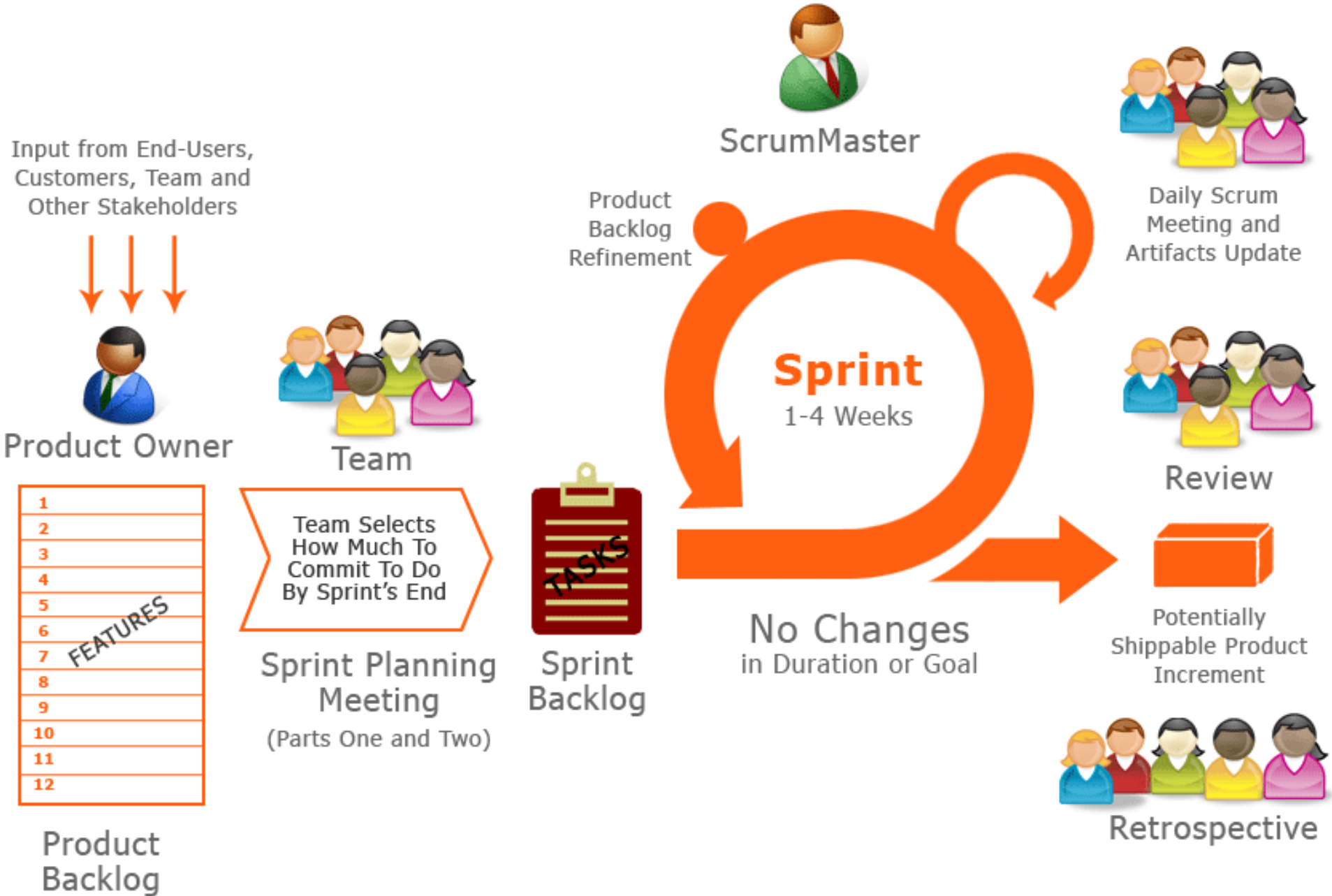
Źródła: http://www.mif.pg.gda.pl/homepages/sylas/students/sem_podypl/io-w05.pdf, slajd 40

SCRUM

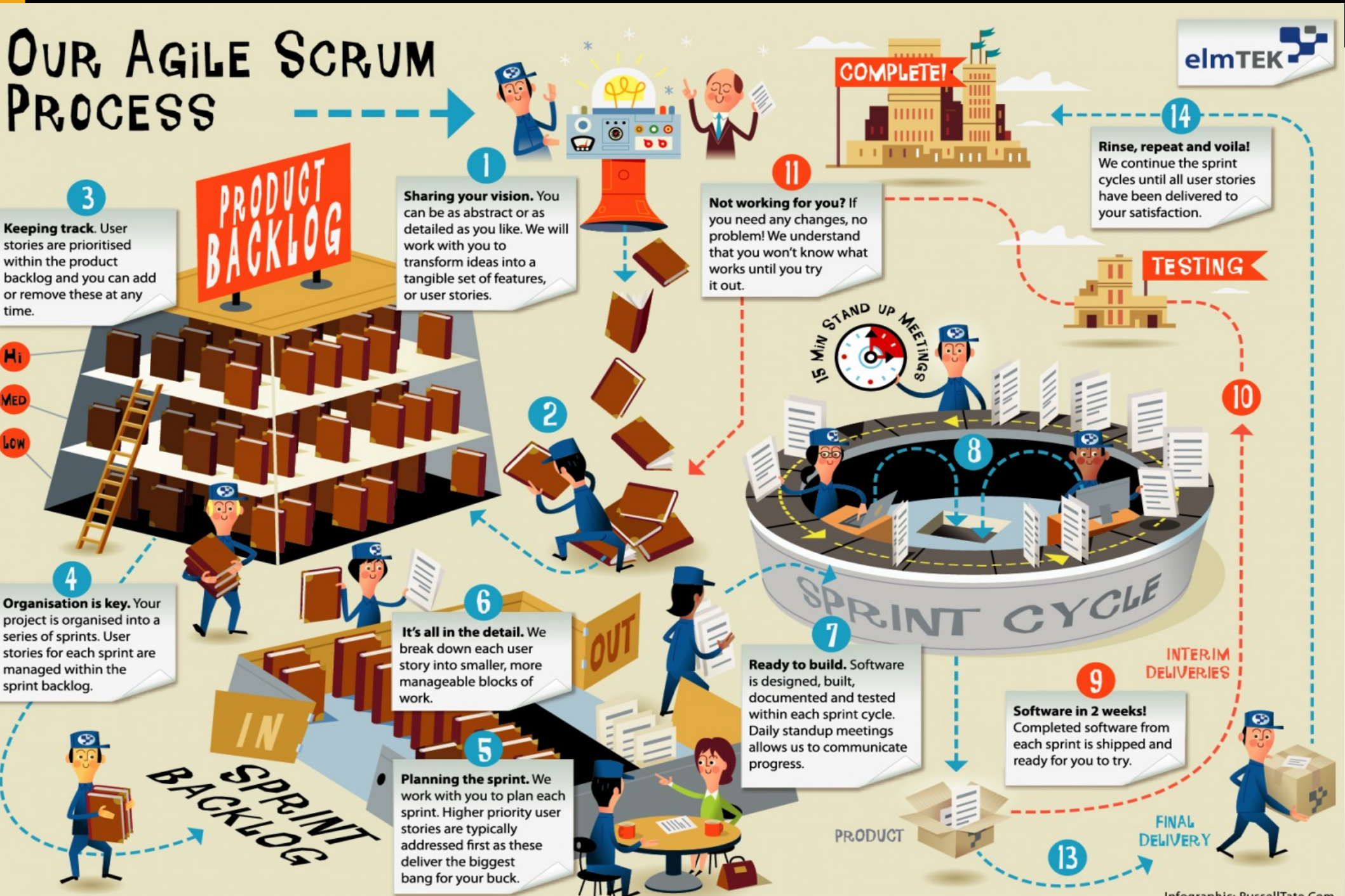
•Rozwój produktu podzielony na fazy (tzw Sprinty) trwające kilka tygodni
•Funkcjonalności zbierane od klienta (user stories), segregowane na sprinty (sprint planning).
•15-minutowe spotkania każdego dnia (daily scrum) opisujące taski na dany dzień i podsumowujące dzień poprzedni (sprint review)
•Działająca wersja wytwarzana po każdym sprincie (musi być widoczna zmiana z poprzedniej wersji). Zapis zmian w rejestrze (sprint backlog)

Właściciel produktu – ustala priorytety, ustala cel pierwszego przebiegu. (na podstawie tego tworzonyc jest backlog)
Zespół – tworzy produkt. Samodzielnie przypisuje zadania do wykonania
Scrum master – odpowiada za usuwanie problemów oraz poprawną implementacje procesu.

SCRUM - role


Input from End-Users, Customers, Team and Other Stakeholders
Product Owner
Team
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Product Backlog
Team Selects How Much To Commit To Do By Sprint's End
Sprint Planning Meeting
(Parts One and Two)
TASKS
Sprint Backlog
ScrumMaster
Product Backlog Refinement
Sprint
1-4 Weeks
No Changes
in Duration or Goal
Daily Scrum Meeting and Artifacts Update
Review
Potentially Shippable Product Increment
Retrospective

SCRUM


Our Agile Scrum Process
elmTEK
COMPLETE!
TESTING
PRODUCT BACKLOG
Hi
Med
Low
1
Sharing your vision. You can be as abstract or as detailed as you like. We will work with you to transform ideas into a tangible set of features, or user stories.
2
3
Keeping track. User stories are prioritised within the product backlog and you can add or remove these at any time.
4
Organisation is key. Your project is organised into a series of sprints. User stories for each sprint are managed within the sprint backlog.
5
Planning the sprint. We work with you to plan each sprint. Higher priority user stories are typically addressed first as these deliver the biggest bang for your buck.
6
It's all in the detail. We break down each user story into smaller, more manageable blocks of work.
7
Ready to build. Software is designed, built, documented and tested within each sprint cycle. Daily standup meetings allows us to communicate progress.
8
15 Min Stand Up Meetings
Sprint Cycle
9
Software in 2 weeks! Completed software from each sprint is shipped and ready for you to try.
10
11
Not working for you? If you need any changes, no problem! We understand that you won't know what works until you try it out.
12
Rinse, repeat and voila! We continue the sprint cycles until all user stories have been delivered to your satisfaction.
13
14
IN
OUT
Sprint Backlog
Interim Deliveries
Product
Final Delivery

Pytania na ćwiczenia: wybrać metodykę

- System reputacyjny dla stacji benzynowych
- Aplikacja mobilna – skaner wifi
- Aplikacja mobilna dla odwiedzających zoo
- Gra typu „angry birds”
- Aplikacja mobilna dla urzędu/rządu
- Rozbudowa systemu obsługi stanu magazynowego
- Rozbudowa strony internetowej (serwisu typu ebay)
- Budowa strony - wizytówki firmowej (landing page produktu)
- Budowa strony – bloga modowego