Politechnika Warszawska

# Metodyki wytwarzania oprogramowania

dr hab. inż. Michał Śmiałek, prof. uczelni
dr inż. Kamil Rybiński

Wydział Elektryczny
POLITECHNIKA WARSZAWSKA

Inżynieria oprogramowania

1

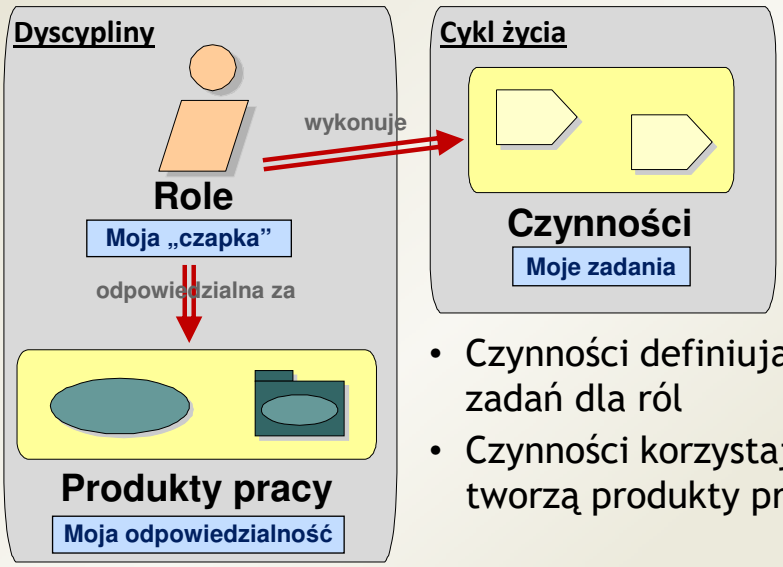
## Co to jest metodyka?

- Definicja metodyki wytwarzania oprogramowania
  - Usystematyzowany i zorganizowany zbiór technik objętych cyklem wytwórczym (patrz poprzedni wykład), którego produkty wyrażone są przy pomocy notacji
- Metodyka powinna być opisana i wdrożona
  - Wdrożenie metodyki w organizacji i/lub projekcie jest złożonym procesem
  - Wdrożenie metodyki obejmuje szkolenia, mentoring, wybór narzędzi, ustalenie kultury pracy, …
- Podstawowe rodzaje metodyk
  - Metodyki sformalizowane
  - Metodyki zwinne (agilne)



2

## Jak opisujemy metodykę?



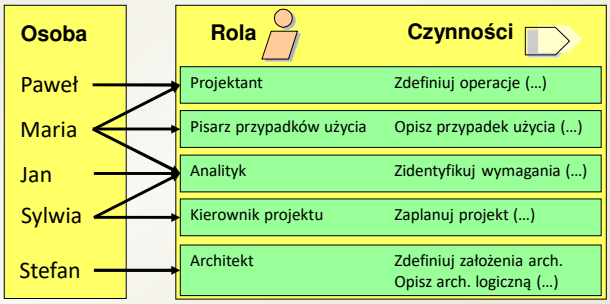

- Czynności definiują zakres zadań dla ról
- Czynności korzystają i tworzą produkty pracy



3

## Role w metodyce

- Rola: definicja pewnego zakresu obowiązków wymagających określonych kwalifikacji
- Role są pełnione przez określone osoby
- Jedna osoba może pełnić kilka ról w projekcie






4

## Produkty pracy i ich notacje

- Produkt pracy: wynik działań w projekcie, np. dokument, model, element modelu, kod, element montażowy
- Produkty pracy wykonują role w ramach dyscyplin
- Produkty tworzymy stosując jednolite **notacje**
  - Łatwiejsze porozumienie w projekcie
  - Poprawa jakości pracy w zespole
- Produkty pracy tworzymy używając odpowiednich narzędzi
  - Zintegrowane środowiska deweloperskie (IDE)
  - Narzędzia wspierające modelowanie (CASE)
  - Narzędzia do zarządzania projektami, …



5

## Metodyki zwinne (agile)

- Bazują na manifeście zwinności (patrz wykład 1)
- Stosują cykl iteracyjny
  - Krótkie iteracje (nawet od 1 tygodnia)
- Koncentrują się na technikach organizacji zespołu i współpracy z klientem
  - Częste dostarczanie działającego oprogramowania
  - Ciągła weryfikacja systemu przez klienta (klient zawsze dostępny)
- Podkreślanie czynnika ludzkiego
  - Stworzenie dobrego środowiska pracy
  - Zapewnienie spójności dobrze zmotywowanego zespołu
  - Pomoc w przestrzeganiu metodyki
  - Dobra komunikacja – codzienna praca w zespole z klientem



6

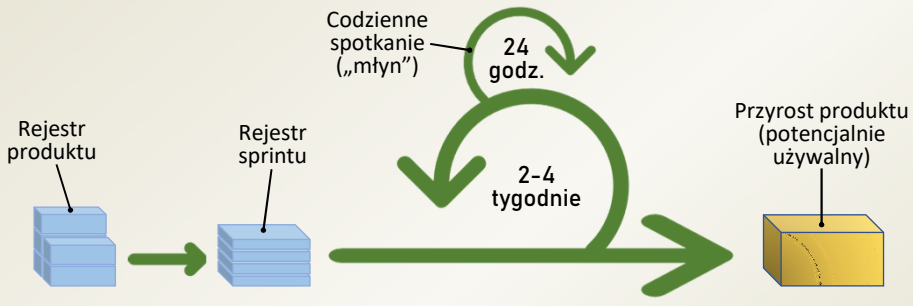
## Rodzaje metodyk zwinnych

- Większość tych metyodyk powstała w połowie lat 90. lub na początku lat 00.
- Scrum (Ken Schwaber, Jeff Sunderland)
- Extreme Programming XP (Kent Beck)
- Crystal (Alistair Cockburn)
- Lean Development (Mary i Tom Poppendieck)
- Kanban (Taiichi Ohno, David Anderson)
- Lean i Kanban to raczej zbiory praktyk niż metodyki
  - Opracowane na bazie zasad produkcji samochodów w firmie Toyota



7

## Metodyka Scrum

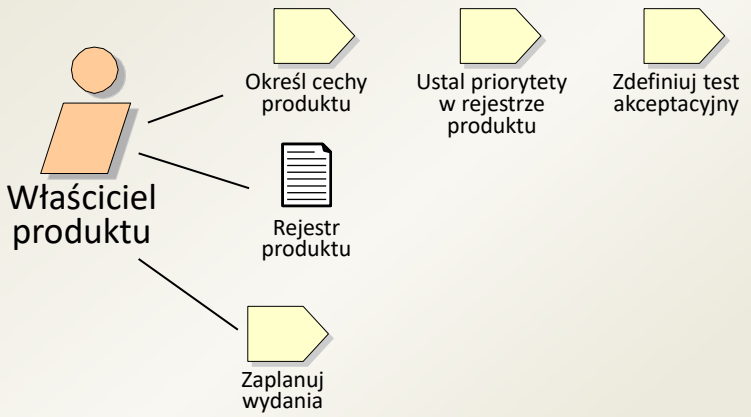



- Iteracja = sprint
- Codzienna kontrola postępu prac w sprincie
- Efektem sprintu jest przyrost produktu
- Produkty zarządcze: rejestr produktu, rejestr sprintu



8

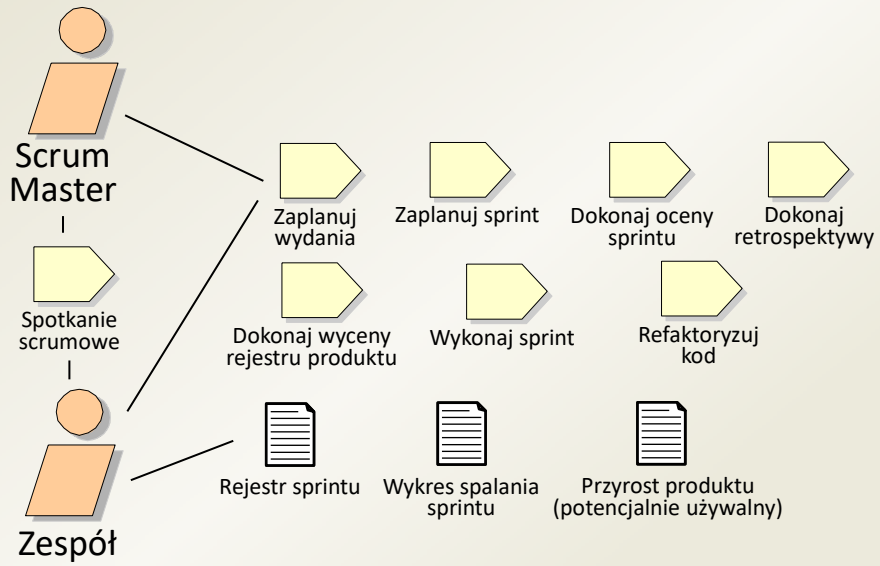
## Role w metodyce Scrum (1)






9

## Role w metodyce Scrum (2)






10

## Zespół deweloperski

- Zespół sam się organizuje
  - Kilka osób (do ok. 10)
  - Metodyka nie określa wymaganych kompetencji
  - Typowe kompetencje obejmujące wszystkie dyscypliny
- Działania zespołu toczą się wokół rejestru sprintu
  - Plan sprintu tworzony przez zespół
  - Ciągle aktualizowana lista zadań
  - Zadania dotyczą działań technicznych we wszystkich dyscyplinach (od wymagań do wdrożenia)
  - Postęp prac mierzony wykresem spalania sprintu (graficzna ilustracja zadań pozostających do wykonania)
- Efektem końcowym jest przyrost produktu



11

## Właściciel produktu

- Reprezentuje klienta
  - Określa cechy produktu (systemu), ustala priorytety i planuje wydania (przyrosty produktu gotowe do użycia)
  - Definiuje testy akceptacyjne
- Działania właściciela produktu koncentrują się na rejestrze produktu
  - Rejestr produktu zwiera cechy produktu
  - Cechy podstawowe: jednostki funkcjonalne (historie użytkownika, przypadki użycia – patrz dalsze wykłady)
  - Cechy uzupełniające: cechy jakościowe (j.w.)
- Rejestr produktu służy do planowania kolejnych sprintów
  - Przydzielanie cech podstawowych do sprintów
  - Dla cech określamy pracochłonność i wagę dla klienta



12

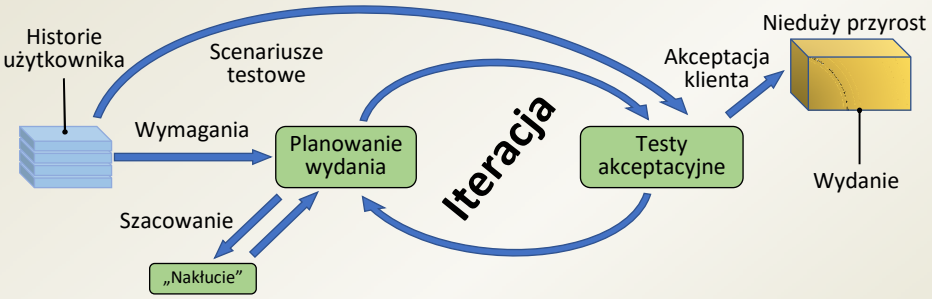
## Mistrz młyna

- Kluczowa rola dla zapewnienia jakości systemu
  - Pomaga w przestrzeganiu reguł metodyki
  - Stwarza zespołowi dobre warunki pracy
  - Uczestniczy (pomaga) we wszystkich czynnościach
- Codzienne spotkania scrumowe
  - Przedstawienie postępów i zaplanowanie pracy
  - Na stojąco, maksimum 15 minut
  - Przedstawienie problemów i zadań (bez dyskusji)
- Retrospektywy na koniec sprintu
  - Identyfikacja problemów i dokonanie korekt w działaniu zespołu
- Oceny sprintów
  - Przedstawienie postępu prac nad produktem końcowym
  - Ocena jakości systemu przez właściciela produktu



13

## Metodyka Extreme Programming (XP)




- Iteracje sterowane historiami użytkownika oraz testami
- Każda iteracja kończy się niedużym przyrostem systemu



14

## Podstawowe cechy XP

- Dwie role podstawowe
  - Klient i Programista
- Trzy role uzupełniające
  - Tester, Tropiciel i Trener
- Wspólna przestrzeń pracy
  - Klient ciągle na miejscu, deweloperzy pracują w jednym pomieszczeniu i mają ciągły dostęp do klienta
- Codzienne spotkania na stojąco
- „Zwinne” techniki dotyczące działań technicznych i zarządczych



15

## Klient


- Pisze historie użytkownika
  - Niewielkie fragmenty funkcjonalności systemu
  - Realizują konkretny cel dla użytkownika
- Definiuje wydania
  - Dostarczenie klientowi konkretnej wartości
- Definiuje testy akceptacyjne
  - Szczegółowe scenariusze historii użytkownika wraz z danymi



16

## Programista (1)


- Realizuje system na podstawie historii użytkownika
  - Podstawą działań jest oszacowanie pracochłonności wykonania wydań systemu





## Programista (2)

- Technika: nakłucie (ang. spike)
  - Szybkie rozpoznanie możliwości technicznych poprzez wykonanie bardzo wąskiego fragmentu systemu
- Zasada: kolektywna własność kodu
  - Kod nie ma autora – za kod odpowiada cały zespół
  - Każdy może wykonać zmiany w całym kodzie
  - Konieczne są wspólne standardy kodowania
- Technika: programowanie parami
  - Cały kod tworzony w zespołach dwuosobowych
  - Osoba pisząca i sprawdzająca
  - Pary zmieniają się co kilka godzin
  - Pozornie nieefektywna, ale znacznie podnosi efektywną jakość kodu

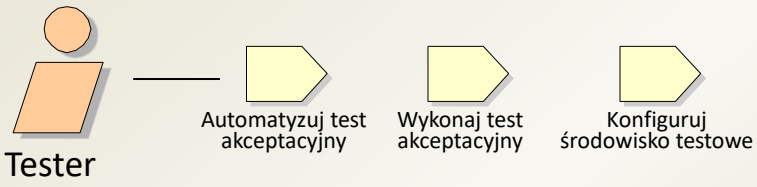
## Programista (3)

- Zasada: prosty projekt
  - Łatwość testowania, przeglądania i wyjaśniania
  - Metafory – opis komponentów systemu
  - Technika: karty CRC (klasy, odpowiedzialność, współpracownicy)
- Testy jednostkowe
  - Sprawdzenie przez programistę działania fragmentu kodu (klasy, komponentu)
- Zasada: ciągła integracja
  - Codzienna (nawet co kilka godzin) integracja kodu do wspólnego repozytorium
  - Automatyczne testy jednostkowe na zintegrowanym kodzie



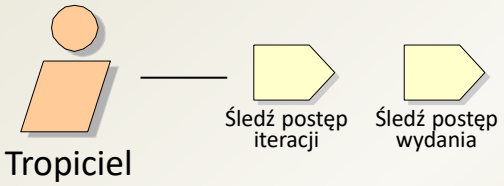


## Tester




- Implementuje testy akceptacyjne zadane przez Klienta
- Zasada: automatyzacja testów
  - Odpowiednio skonfigurowane środowisko testowe
  - Stosowanie narzędzi do testowania metodą czarnej skrzynki
  - Tworzenie skryptów testowych
  - Testy regresyjne – powtarzane w kolejnych iteracjach

## Tropiciel

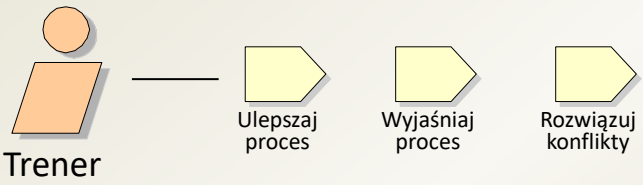



- Śledzi postępy pracy zespołu
  - Zarządza planem projektu
  - Kontroluje realizację zadań technicznych
  - Kontroluje realizację historii użytkownika
  - Komunikuje zespołowi postępy w budowie systemu





## Trener




- Dbanie o jakość pracy zespołu
  - Podobna rola do Scrum Mastera

## Metodyki sformalizowane

- Posiadają szczegółowo opisane reguły, często zawarte w formalnych standardach
- Stosowane dla systemów zamawianych formalnie
- Mogą stosować cykl wytwórczy podobny do metodyk zwinnych
  - Metodyki formalne można dostosowywać do warunków projektu
- Odpowiednie dla dużych projektów
  - Zespoły liczące dziesiątki (setki) osób
  - Sformalizowane procedury odbioru systermu





## Rodzaje metodyk sformalizowanych

- Metodyki stosowane w zamówieniach publicznych
  - SSADM – metodyka o cyklu wodospadowym stosowana w latach 80. w Wielkiej Brytanii
  - DOD-STD-2176A, MIL-STD-498 – metodyki stosowane do dziś przed Departament Obrony USA (wodospadowe, potem iteracyjne)
  - V-Modell – metodyka stosowana przez rząd RFN oparta na cyklu V
- Metodyki stosowane przez duże firmy software'owe
  - ALM – firma Borland (już nie istnieje)
  - MSF – firma Microsoft
  - Select Perspective – firma Select Business Solutions
  - OPEN – konsorcjum firm pod tą samą nazwą
  - Essence – inicjatywa SEMAT

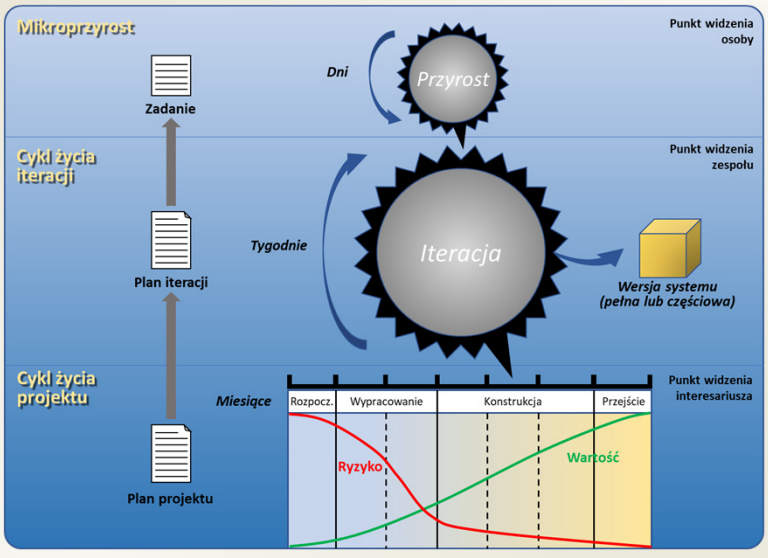
## Metodyki zunifikowane

- Metodyka oryginalna: Rational Unified Process (RUP)
  - Opracowana przez twórców języka UML
  - Produkty pracy w istotnej części wykorzystują język UML
  - Firma Rational obecnie została przejęta przez IBM
- Metodyka RUP jest bardzo obszerna
  - Implementacja metodyki wymaga sporego wysiłku w dostosowaniu jej do konkretnej firmy/projektu
- Metodyka uproszczona: OpenUP
  - Znacznie ograniczona liczba ról i produktów
  - Dostosowana do małych i średnich zespołów
  - Stosuje wiele zasad metodyk zwinnych



25

## Metodyka OpenUP






26

## Cechy metodyki OpenUP

- Podział na iteracje zawarty w planie projektu
- Projektem sterują wymagania – przypadki użycia
- Cztery fazy projektu
  - Każda faza zawiera od 1 do kilku iteracji
  - Nie należy mylić z cyklem wodospadowym!
- Każda iteracja kończy się wykonywalną wersją systemu
- Co kilka dni jest oddawany przyrost systemu
- Harmonogram iteracji zawarty jest w planie iteracji

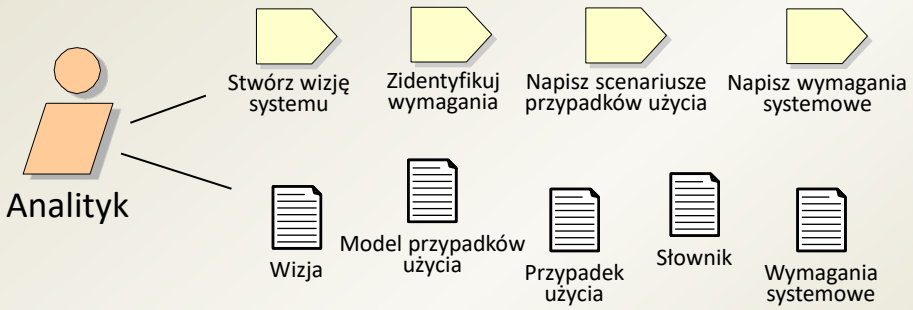




## Fazy metodyki OpenUP

- Rozpoczęcie (zazwyczaj 1 iteracja)
  - Dobre rozpoznanie zakresu funkcjonalnego
  - Efektem jest bardzo ograniczony prototyp
- Wypracowanie (1-3 iteracje)
  - Ustabilizowanie architektury systemu
  - Weryfikacja rozwiązań technicznych w praktyce
- Konstrukcja (3-7 iteracji)
  - Stopniowa realizacja przyrostów funkcjonalności
- Przejście (1-2 iteracje)
  - Uzupełnienie funkcjonalności (zmiany itp.)
  - Przekazanie systemu klientowi
- Uwaga: w każdej fazie iteracje kończą się kolejnymi wersjami systemu
  - Każda iteracja zawiera czynności wymagań, projektowania, implementacji, testowania, ew. wdrożenia

## Analityk




- Formułuje i zarządza wymaganiami
- Pięć produktów pracy
  - Wizja, model przypadków użycia, przypadek użycia, wymagania systemowe, słownik
  - Patrz dalsze wykłady

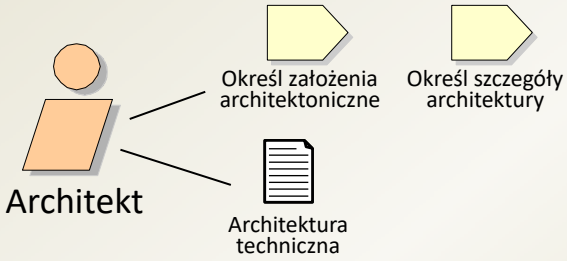




## Przypadek użycia

- Podstawowa jednostka realizacji systemu w metodykach zunifikowanych (RUP, OpenUP)
- Nieduży, kompletny fragment funkcjonalności systemu
- Stanowi określoną wartość biznesową dla użytkownika
  - Kluczowy jest cel zawarty w nazwie
- Zawiera zestaw scenariuszy
  - Szczegóły interakcji użytkowników z systemem
  - Mogą być uzupełnione o wygląd ekranów (scenopis)

## Architekt




- Główna rola projektowa
- Produkt pracy: architektura techniczna
  - Podział systemu na komponenty
  - Definicja interfejsów
  - Definicja fizycznych jednostek wykonawczych
  - Decyzje dotyczące stosowanych technologii



31

## Deweloper


- Deweloper łączy kompetencje projektanta i programisty
- Produkt pracy: projekt
  - Definicja struktury i działania konkretnego komponentu
- Rola podobna do programisty w XP



32

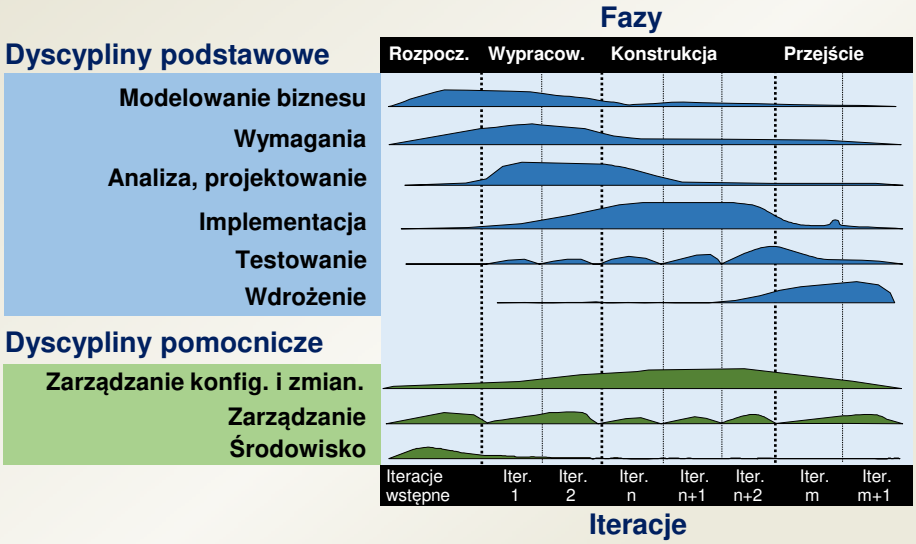
## Inne role OpenUP

- Tester
  - Rola podobna do testera w XP
- Kierownik projektu
  - Typowa rola w projektach
  - Zarządza planem projektu i planami iteracji
- Inżynier procesu
  - Rola podobna do Trenera w metodyce XP
- Specjalista narzędziowiec
  - Instaluje i konfiguruje narzędzia
  - Wspiera zespół w codziennej pracy z narzędziami
  - Obecnie rola przekształca się w specjalistę DevOps



33

## Rational Unified Process






34