Zarządzanie Projektami Informatycznymi

PODEJŚCIA ZWINNE

Scrum

Scrum – zespół i role

- **Właściciel Produktu**
  - jest centralnym punktem zarządzania – jest odpowiedzialny za to, co będzie wytwarzane produkowane i w jakiej kolejności
  - pośrednik między zespołem a klientem (użytkownikiem)
  - jego zadaniem jest zrozumienie potrzeb i priorytetów użytkowników oraz informowanie na bieżąco zespołu o tym, czego najbardziej potrzebują od oprogramowania i w jakiej kolejności
  - odpowiada za wymagania – zna wizję projektu, zarządza rejestrem produktu (listą wymagań)
  - ustala priorytety
  - określa kryteria akceptacji
  - odbiera pracę w sprintach
  - pilnuje aby powstało właściwe rozwiązanie – jest odpowiedzialny za całkowite powodzenie wytwarzanego lub rozwijanego rozwiązania
  - chcąc mieć pewność, że zespół buduje wyłącznie to, na czym mu zależy, aktywnie współpracuje z zespołem i ScrumMasterem – musi być dostępny i szybko odpowiadać na wszelkie pojawiające się pytania
  - może mieć asystenta

Scrum – zespół i role

Właściciel Produktu – główne obowiązki

- współpraca z przedstawicielami klienta – reprezentowanie ich interesów („głos klienta”)
- współpraca na co dzień z zespołem – zgłaszanie wytycznych, udzielanie odpowiedzi i wsparcia
- dbanie o podejmowanie właściwych ekonomicznie decyzji
- pielęgnacja rejestru produktu: tworzenie i uszczegóławianie elementów rejestru, nadawanie im ocen oraz priorytetów
- udział w planowaniu – współpraca z zespołem
- definiowanie kryteriów akceptacji i weryfikowanie czy zostały spełnione

Scrum – zespół i role

- **Właściciel Produktu – cechy i umiejętności**
  - posiada ekspercką wiedzę z zakresu biznesu i domeny produktu
  - umiejętności interpersonalne: dobre relacje z interesariuszami, jest dobrym negocjatorem (buduje konsensus) , umiejętności komunikowania się i przekazywania informacji, potrafi skutecznie motywować
  - zdolność podejmowania trudnych decyzji (wymagających często kompromisów), jest zdecydowany, potrafi zachować równowagę pomiędzy potrzebami biznesowymi i możliwościami inżynierskimi
  - odpowiedzialność – podejmowanie odpowiedzialności za dostarczenie dobrego produktu
  - pełne zaangażowanie i stała dostępność

Scrum – zespół i role

- **Właściciel produktu**
- **ScrumMaster** (Mistrz Młyna)
  - pomaga zespołowi w zrozumieniu i przestrzeganiu wartości, zasad i praktyk Scruma (zmianie mentalności)
  - nie jest kierownikiem projektu! (nie wyznacza bezpośrednio szczegółowych zadań, nie ma władzy nad zespołem)
  - wspomaga zespół i Właściciela Produktu jako doradca (trener metod zwinnych) poprzez zadawanie pytań, udzielanie wskazówek, rad itp.
  - „wielcy Mistrzowie Młyna niemal nigdy nie odpowiadają na pytanie w sposób bezpośredni, zamiast tego odpowiadają zawsze kolejnym pytaniem”
  - pomaga w sprawach organizacyjnych i usuwaniu przeszkód

Scrum – zespół i role

- **Właściciel Produktu**
- **ScrumMaster** – odpowiedzialny zarządzaniem procesem Scrum i usuwanie przeszkód
- **Zespół** - od 5 do 9 osób
  - Scrum bazuje na *samoorganizacji* zespołu i zaangażowaniu ludzi (pracy zespołowej)
  - podstawą realizacji prac jest *wspólne zobowiązanie* – nikt nie narzuca i nie przypisuje prac z góry (członkowie zespołu samodzielnie się ich podejmują i dzięki temu czują się bardziej zobowiązani)
  - zespół sam decyduje o metodach swojej pracy
  - wielofunkcyjny (interdyscyplinarny) i samowystarczalny – zróżnicowane i uzupełniające się umiejętności typu T
    - prostsza komunikacja
    - redukcja nieporozumień
    - szersza perspektywa

Scrum – zasady pracy zespołu

- wspólne zobowiązanie – każdy czuje się zobowiązany do realizacji celów projektu (wspólne zobowiązanie)
- wzajemny szacunek – kultura egalitaryzmu, otwartość na pomysły i uwagi innych
- skoncentrowanie na pracy – praca tylko w jednym projekcie, unikanie wielozadaniowości
- otwartość (przejrzystość komunikacji) – każdy członek zespołu ma pełny przegląd sytuacji, wczesne ujawnianie problemów, utrwalanie zaufania (zarówno do procesu, jak i pomiędzy członkami zespołu)
- odwaga – otwartość wymaga odwagi
- podejmowanie decyzji w ostatnim możliwym terminie – zapewnia to swobodę wyboru właściwej osoby do wykonania potrzebnej pracy w odpowiednim czasie

Scrum – praktyki

- **krótkie iteracje (sprinty)** do 30 dni ale stałej długości – efektem każdej iteracji jest przyrost funkcjonalności produktu

Zalety ograniczenia czasowego sprintu

- Ograniczenie pracy cząstkowej – szybszy przepływ prac i dostarczanie produktów
- Szybsza korzyść z inwestycji
- Wymuszenie ustalenia priorytetów i wykonywania pracy o największym znaczeniu – wytworzenie czegoś wartościowego w szybkim tempie (bez ograniczenia w czasie realizuje się więcej funkcji o niskiej wartości)
- Szybkie weryfikowanie produktów i założeń oraz uzyskiwanie informacji zwrotnych (np. sygnałów do zaprzestania prac zanim doprowadzą one do poważnych strat ekonomicznych)
- Motywowanie domykania prac – bez ograniczenia czasowego znika poczucie pilności zakończenia prac
- Poprawianie przewidywalności – łatwiejsze planowanie
- Unikanie zbędnego perfekcjonizmu („done is better than perfect”) – ograniczenie czasowe wymusza zakończenie prac i „pozłacania”
- Ograniczanie propagacji błędów poprzez szybsze ich wykrywanie

Scrum – praktyki

- **krótkie iteracje (sprinty)** – efektem każdej iteracji jest przyrost funkcjonalności produktu
- na początku iteracji – **spotkanie planujące sprint**
  - pierwsze 4 godz. – Właściciel Produktu prezentuje zespołowi wymagania i ich priorytety; zespół zadaje pytania i określa co da się zrobić (gra planistyczna)
  - kolejne 4 godz. – zespół rozplanowuje szczegóły działania

Scrum – praktyki

- **krótkie iteracje (sprinty)** – efektem każdej iteracji jest przyrost funkcjonalności produktu
- na początku iteracji – **spotkanie planujące sprint** (max 8 godz.)
- w trakcie iteracji **codzienne ok. 15 min. spotkania** wszystkich członków zespołu (scrumy)
  - omawiane są zadania zrealizowane poprzedniego dnia (co robiłem wczoraj?)
  - zadania do wykonania w dniu spotkania (co będę robił dzisiaj?)
  - pojawiające się problemy (co mi przeszkadza?)
  - zespół mówi o tym do siebie – nie raportuje do szefa
  - uaktualnienie rejestru postępu prac (na tablicy zadań i wykresie spalania)
  - celem spotkania jest codzienne zsynchronizowanie pracy członków całego zespołu
  - sposób na wczesne wykrywanie problemów i zatorów

Scrum – codzienne spotkania (scrumy)

- wszyscy powinni uczestniczyć w codziennych spotkaniach – włącznie z Właścicielem Produktu i ScrumMasterem
- na stojąco – z pełnym zaangażowaniem (bez odbierania e-maili, sprawdzania SMS itd.)
- przed tablicą – żeby każdy widział aktualny stan prac i wszystkie zadania do wykonania
- za każdym razem zaczyna kto inny – wszyscy uczestniczą na równych prawach (spotkania mają służyć całemu zespołowi)
- szczegółowe rozmowy i rozwiązywanie problemów po codziennym spotkaniu

Scrum – praktyki

- **30-dniowe iteracje (sprinty)** – efektem każdej iteracji jest przyrost funkcjonalności produktu
- na początku iteracji – **spotkanie planujące sprint** (max 8 godz.)
- w trakcie iteracji **codzienne ok. 15 min. spotkania** wszystkich członków zespołu (scrumy)
- na końcu iteracji
  - **spotkanie przeglądu sprintu**
    - prezentowany jest produkt wykonany podczas iteracji
    - ocena i adaptacja produktu
    - wspólne określanie tego, co zespół powinien robić w dalszej kolejności
    - rezultatem przeglądu sprintu mogą być decyzje o dodaniu nowych funkcjonalności lub zmiana w istniejących priorytetach
    - dwukierunkowy przepływ informacji (informacja zwrotna)

Scrum – praktyki

- **30-dniowe iteracje (sprinty)** – efektem każdej iteracji jest przyrost funkcjonalności produktu
- na początku iteracji – **spotkanie planujące sprint** (max 8 godz.)
- w trakcie iteracji **codzienne ok. 15 min. spotkania** wszystkich członków zespołu (scrumy)
- na końcu iteracji
  - **spotkanie przeglądu sprintu**
  - **retrospekcja**
    - ocena przebiegu iteracji
    - określenie działań w celu usprawnienia pracy
    - „oczyszczenie” atmosfery prac
    - celem jest nieustająca poprawa procesu, dzięki której zespół scrumowy stanie się zespołem jeszcze lepszym
    - dostosowanie Scruma do swoich własnych, unikatowych warunków

Scrum – praktyki

- **30-dniowe iteracje (sprinty)** – efektem każdej iteracji jest przyrost funkcjonalności produktu
- na początku iteracji – **spotkanie planujące sprint** (max 8 godz.)
- w trakcie iteracji **codzienne ok. 15 min. spotkania** wszystkich członków zespołu (scrumy)
- na końcu iteracji
  - **spotkanie przeglądu sprintu**
  - **retrospekcja**

Przegląd sprintu jest czasem przeznaczonym na ocenę i adaptację produktu (tego co tworzymy)

Retrospekcja sprintu stanowi okazję do oceny i adaptacji procesu (tego jak tworzymy)

Scrum – porównanie z podejściem tradycyjnym

Sekwencyjnie (kaskadowo)	Scrum
Koncentracja na formalnych procedurach	Nacisk na dostarczanie nadającego się do wdrożenia przyrostu produktu
Postęp mierzony liczbą ukończonych faz projektu	Postęp mierzony działającym oprogramowaniem
Tworzenie oprogramowania sterowane planem (z przewidywaniem)	Działanie w samą porę (just in time), adaptacja
Przewidywanie tego, co jest nieznane	Eksploracja i informacja zwrotna
Projekty o małej niepewności (daty są predykcją terminu zakończenia)	Projekty o dużej niepewności (daty stanowią momenty graniczne!)
Kontrolowanie, minimalizowanie i w miarę możliwości unikanie zmian	Otwartość na zmiany – zmiana to rzecz powszechna (nie da się wyeliminować niepewności)
Inwestowanie w zrobienie wszystkiego dobrze za pierwszym razem	Eksploracja i szybkie adaptacje w oparciu o informacje zwrotne od klienta
Wszystkie wymagania muszą posiadać ten sam poziom uszczegółowienia	Wymagania są uszczegółowiane w samą porę

Scrum – podsumowanie

- Scrum stanowi pewien szkielet (jak fundament i ściany dla domu) – środowisko ze zbiorem wartości, zasad i praktyk
- Pozwala zastosować własne metody realizacji prac inżynierskich i sposoby postępowania w ramach tego szkieletu
- Proces iteracyjny i przyrostowy
- Istotą Scrum jest ograniczenie czasowe sprintu (które mobilizuje do działania) oraz dostarczanie w tym czasie produktów o określonej funkcjonalności
- Filozofia Scrum – „sztuka rzeczy możliwych”
- Scrum bazuje na samoorganizacji zespołu i zaangażowaniu ludzi – wartościach uczciwości, szacunku, zaufania, mobilizacji i pracy zespołowej
- Równowaga kontroli i elastyczności

eXtreme Programming (XP)

eXtreme Programming (XP)

W odróżnieniu od Scrum określa też wiele praktyk programistycznych

eXtreme Programming (XP) – zespół i role

- Przedstawiciele klienta
- Programiści
- Testerzy
- Coachowie

Role w dojrzałych zespołach XP nie są stałe i ściśle określone

eXtreme Programming (XP) – praktyki

- krótkie i częste iteracje
- programowanie w parach
- wytwarzanie sterowane testami
- ciągła integracja i testowanie
- kolektywne prawo do zmian
- dążenie do prostoty
- stały kontakt z klientem
- projektowanie przyrostowe

eXtreme Programming (XP) – organizacja

- informacyjne miejsce pracy
- wspólna praca
- codzienne kilkuminutowe spotkania informacyjne
- zespół przez cały czas współpracuje z przedstawicielami klienta
- pełnoetatowość członków zespołu
- stopniowe zbieranie wymagań
- przestrzeganie ustalonych standardów pisania kodu
- szacowanie czasu na podstawie „prędkości” realizacji poprzedniej iteracji
- energiczna praca

eXtreme Programming (XP) – otwartość na zmiany

W XP liczy się możliwość szybkiego wprowadzania zmian
przy jak najmniejszej liczbie błędów

Praktyki wspierające łatwość wprowadzania zmian

- dążenie do prostoty kodu
- przestrzeganie ustalonych standardów pisania kodu
- projektowanie przyrostowe
- współwłasność kodu

Praktyki wspierające wykrywanie błędów:

- programowanie w parach
- programowanie sterowane testami
- ciągła integracja

eXtreme Programming (XP) – cykl pracy

- Planowanie iteracji
- Zobowiązanie się do dostarczenia nowej funkcjonalności
- Tworzenie oprogramowania
- Udostępnienie kodu
- Demonstracja iteracji
- Retrospekcja

eXtreme Programming (XP) – kwestie kontrowersyjne

- konieczna stała dostępność przedstawiciela klienta (klient „na miejscu”)
- „wspólna własność" kodu - każdy może zmieniać dowolny fragment systemu
- projektowanie przyrostowe