Inżynieria wymagań w metodach Agile

A. LIPSKI
e-mail:lipski.artur@gmail.com

Wydział Cybernetyki WAT
ul. S. Kaliskiego 2, 00-908 Warszawa

„Zwinne" wytwarzanie oprogramowania (Agile Software Development) stało się bardzo popularne na przestrzeni kilku ostatnich lat. Metody Agile zostały wymyślone w celu szybszego dostarczenia działającego oprogramowania do klienta oraz dostosowania się oprogramowania do zmiennych potrzeb klienta. Można zauważyć, że istnieje wiele technik i praktyk wymyślonych oraz opracowanych w kontekście tradycyjnej inżynierii wymagań, które obecnie wykorzystywane są z dobrym rezultatem przez metody Agile. Celem tego artykułu jest pokazanie, w jaki sposób techniki inżynierii wymagań są wykorzystywane przez metody Agile oraz w jaki sposób metody te mogą być udoskonalone za pomocą tych technik.

Słowa kluczowe: Agile, inżynieria wymagań, inżynieria oprogramowania

1. Wprowadzenie

„Zwinne" wytwarzanie oprogramowania (Agile Software Development) stało się bardzo popularne na przestrzeni kilku ostatnich lat. Metody Agile zostały wymyślone w celu szybszego dostarczenia działającego oprogramowania do klienta oraz dostosowania się oprogramowania do zmiennych potrzeb klienta. Ogólnie metody te dzielą między sobą wspólne praktyki: zwiększenie zadowolenia klienta, dostosowanie do zmieniających się wymagań, przyrostowe dostarczanie działającego oprogramowania oraz zapewnienie bliskiej współpracy pomiędzy klientem a producentem.

Inżynieria wymagań jest tradycyjnym procesem inżynierii oprogramowania, którego zadaniem jest zidentyfikowanie, zanalizowanie, udokumentowanie oraz walidowanie wymagań dla systemu, który ma być skonstruowany. Często zauważa się, że połączenie inżynierii wymagań z metodami Agile nie jest do siebie przystawalne. Inżynieria wymagań bardziej skupia się na zapisach w sporządzonej dokumentacji w celu dzielenia się wiedzą, niż współpracy „twarzą w twarz" pomiędzy klientem a producentem, aby osiągnąć wspólne cele. Ten artykuł ma pokazać, w jaki sposób techniki inżynierii wymagań są wykorzystywane przez metody Agile oraz w jaki sposób metody te mogą być udoskonalone za pomocą tych technik.

2. Inżyniera wymagań

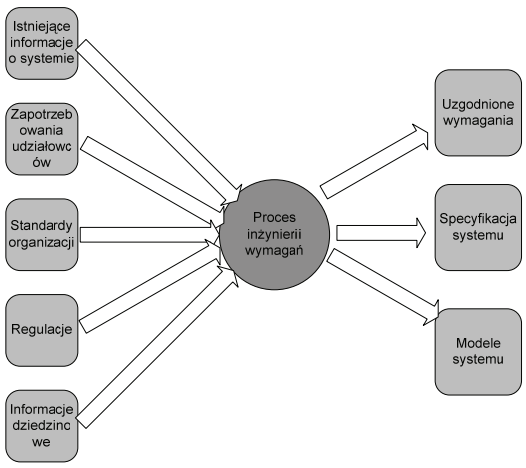
Inżynieria wymagań (IW) jest skoncentrowana wokół identyfikowania, modelowania, komunikowania oraz dokumentowania wymagań dotyczących systemu oraz kontekstu, w jakim system zostanie wykorzystany. Wymagania określają to, co ma być wytworzone, ale nie w jaki sposób ma być zaimplementowane. Istnieje wiele technik, które sprawiają, że wymagania są pełne, konsekwentne oraz istotne z punktu widzenia klienta.

2.1. Proces inżynierii wymagań

Proces ten składa się z pięciu głównych aktywności [1]:

- wydobycie
- analiza i negocjacje
- dokumentacja
- walidacja
- zarządzanie.

Na rysunku nr 1 zostały pokazane wyjścia oraz wejścia procesu:



Rys. 1. Proces inżynierii wymagań [1]

Pomimo tego że proces inżynierii wymagań może się różnić pomiędzy organizacjami, to zakres wejść i wyjść pozostaje niezmieniony w większości przypadków. IW pozwala na wcześniejsze znalezienie błędów, dzięki czemu możliwa jest redukcja kosztów związanych z procesem wytwórczym oprogramowania. Im później zostaną wykryte pomyłki, tym trudniej będzie je usunąć z systemu [1].

2.2. Wydobywanie wymagań

Wydobywanie odnosi się do identyfikacji wymagań oraz zakresu systemu poprzez konsultacje z udziałowcami (klientem, developerami, użytkownikami). Granice systemu wpływają na techniki wydobywcze oraz definiują kontekst tworzonego systemu. W tym rozdziale zostaną scharakteryzowane najbardziej znane techniki wydobywania wymagań.

1. Wywiad

Prowadzenie wywiadów jest metodą odkrywania faktów oraz opinii potencjalnych użytkowników oraz innych udziałowców na temat wytwarzanego systemu. Wszystkie pomyłki oraz niedopowiedzenia mogą zostać skorygowane za pomocą tej techniki. Rozróżnia się dwa różne rodzaje wywiadów:

- Zamknięty wywiad, w którym inżynier wymagań ma przygotowany zestaw pytań, na który będzie oczekiwał odpowiedzi.
- Wywiad otwarty, bez przygotowanych z góry pytań, podczas którego inżynier wymagań oraz udziałowcy dyskutują w otwartej rozmowie na temat oczekiwań odnośnie systemu.

2. Przypadki użycia/ Scenariusze

Przypadki użycia opisują interakcje między użytkownikami a systemem, skupiając się na ich potrzebach wobec niego, poprzez identyfikację najważniejszych funkcji. Opisują one sekwencję interakcji pomiędzy systemem a zewnętrznym „aktorem" (np. osobą, częścią sprzętową, innym systemem), uwzględniając warianty oraz wykluczenia, które może wykonać system. Przypadek użycia dostarcza wynik specyficznego zadania wykonanego przez jednego aktora. Analitycy oraz klienci powinni sprawdzić każdy zaproponowany przypadek w celu jego walidacji [1], [2].

3. Burza mózgów

Burza mózgów jest jednym ze sposobów, by wynaleźć kreatywne rozwiązania odnoszące się do określonego tematu. Normalnie burzą mózgów określa się grupową aktywność, jednak może być także przeprowadzana na osobności. Technika ta składa się z dwóch faz:

- Fazy twórczej – w której zbierane są pomysły.
- Fazy ewaluacji – w której zebrane pomysły są przedyskutowywane.

Każdy z pomysłów powinien być szybko wynaleziony oraz każdy z pomysłów może, ale nie musi prowadzić do nowych rozwiązań. Burza mózgów prowadzi do lepszego zrozumienia problemu przez każdego oraz daje poczucie wykreowania wspólnego rezultatu [4].

4. Obserwacje i analiza społeczna

Metody obserwacyjne angażują użytkowników dochodzeniowych, którzy pracują nad określonym obszarem, robią notatki z aktywności, które mają miejsce. Metoda jest przydatna we wczesnym stadium zbierania wymagań do pozyskania danych jakościowych.

5. Metoda miękkiego systemu (Soft System Methodology)

Metoda ta skupia się na wymyślonym systemie pod kątem jego osiągalności. Dostarcza ona analizy sytuacji problemowych z różnych perspektyw. SSM może zostać użyte, kiedy istnieje sytuacja w życiu codziennym i związany z nią problem, który jest dostrzegany przez przynajmniej jedną osobę pragnącą ten problem rozwiązać [1].

Konwencjonalny model metody składa się z siedmiu fundamentalnych etapów:

- opis sytuacji problemu
- zarys otaczającego świata
- główna definicja systemu
- model konceptualny
- porównanie modelu do otaczającego świata
- identyfikacja pożądanych i wykonalnych zmian
- działania w celu poprawy sytuacji problemu.

6. Ponowne użycie wymagań

Ponowne użycie wymagań odnosi się do wykorzystania specyfikacji wymagań wytworzonych w poprzednich projektach. Metoda ta może być użyta tylko w następujących przypadkach [4]:

- wymagania zawierają informacje dotyczące domeny aplikacyjnej
- wymagania skoncentrowane są na stylu prezentacji
- wymagania odzwierciedlają polityki firmy (np. politykę bezpieczeństwa).

7. Prototypowanie

Prototyp systemu jest początkową wersją systemu, który jest dostępny we wczesnym etapie procesu wytwórczego. Prototypy systemów są często wykorzystywane do wytypowania i walidacji wymagań systemowych.

Wyróżnia się dwa odmienne rodzaje prototypów:

- prototyp dwukierunkowy – pomaga w wyznaczeniu wymagań, które powodują problemy w ich rozumieniu
- prototyp ewolucyjny – dostarcza działający system do klienta i może być częścią finalnej wersji systemu.

2.3. Analiza wymagań i negocjacje

Analiza wymagań sprawdza wymagania pod kątem przydatności, zakresu, kompletności oraz wykonywalności. Konflikty w wymaganiach są rozwiązywane poprzez negocjację priorytetów. Wymagania, które wydają się problematyczne, są przedyskutowywane z udziałowcami w celu zebrania ich opinii. Głównymi technikami, które są wykorzystywane podczas analizy wymagań, są JAD, priorytetyzacja i modelowanie.

1. Joint Application Development (JAD)

JAD to funkcyjna sesja grupowa z podejściem analizy strukturalnej. Zadaniem tej techniki jest zdefiniowanie specjalnego projektu oraz jego monitorowanie na różnych poziomach szczegółowości, aż do momentu jego zakończenia [1]. Na sesjach wyróżnia się kilka głównych ról:

- przewodniczący sesji
- reprezentant użytkowników
- specjalista
- analityk
- reprezentant systemu informatycznego
- sponsor.

2. Priorytetyzacja wymagań

W projektach o krótkim harmonogramie, ograniczonych zasobach oraz wysokich wymaganiach klienta celowe jest jak najszybsze dostarczenie klientowi najbardziej wartościowych (z jego punktu widzenia) funkcji. W momencie, gdy zbliża się termin oddania projektu, a nie wszystkie funkcje zostały zaimplementowane, część z nich musi zostać wykluczona z zakresu projektu, co może być uzyskane właśnie dzięki priorytetyzacji funkcji. Priorytetyzacja powinna być wykonana przez klienta w porozumieniu z zespołem projektowym. Zarówno klient jak i programiści muszą dać swój wkład do priorytetyzacji wymagań. Klient oznaczy najwyższym priorytetem te funkcje, które będą dostarczały największą wartość dodaną dla użytkownika. Natomiast programista wskaże zagrożenia techniczne, koszt oraz problemy związane z realizacją danych funkcji [3].

3. Modelowanie

Modele systemu są ważnym połączeniem pomiędzy procesem analitycznym a wytwórczym. Wiele metod wykorzystuje różne techniki modelowania do sformułowania lub analizowania wymagań systemowych. Do najbardziej popularnych technik należą: modele przepływu danych (*data-flow models*), semantyczne modele danych (*semantic data models*) i podejścia zorientowane obiektowo [1], [3].

4. Quality Function Deployment (QFD)

Celem tej techniki jest zaprojektowanie satysfakcji klienta w postaci produktu przed tym jak zostanie on wyprodukowany, poprzez tłumaczenie wymagań klienta na specyfikacje inżynierskie. QFD zawiera sesje grupowe, na których tzw. diagram „House of Quality" używany jest w celu skupienia uwagi uczestników sesji [7].

2.4. Dokumentacja wymagań

Celem dokumentacji wymagań jest komunikacja rozumienia wymagań pomiędzy udziałowcami a programistami. Dokument z wymaganiami wyjaśnia domenę aplikacyjną oraz sam system, który będzie zbudowany. Może on być podstawą do rozwijania poszczególnych produktów i procesów (architektury systemu, przypadków testowych, aktywności weryfikacyjnych i walidacyjnych) [6].

2.5. Walidacja wymagań

Celem walidacji wymagań jest zapewnienie, że wymagania reprezentują akceptowalny opis implementowanego systemu. Wejściem procesu walidacyjnego jest dokument wymagań, standardy organizacyjne oraz wiedza organizacyjna. Wyjściem procesu jest lista problemów związanych z dokumentem i zaakceptowane czynności, mających na celu wyeliminowanie tych problemów. Wyróżnia się następujące techniki walidacji wymagań: przegląd wymagań i testy wymagań [1].

1. Przegląd wymagań

Przegląd wymagań zapewnia, że wymagania są:
- kompletne
- istotne
- nienadmiarowe
- testowalne.

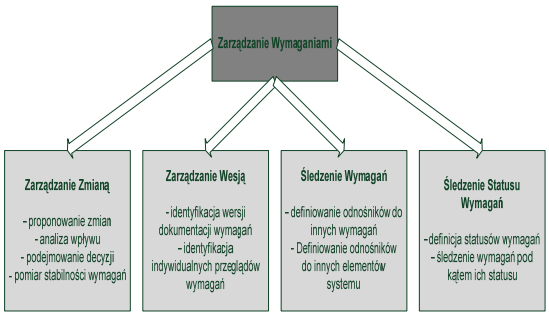
Wyróżnia się przeglądy formalne i nieformalne. Celem przeglądów nieformalnych jest krytyka i poprawa wymagań, natomiast celem przeglądów formalnych jest zaakceptowanie dokumentacji wymagań i autoryzacja projektu.

2. Testowanie wymagań

Testowanie systemu jest integralną częścią inżynierii oprogramowania, natomiast jeśli system jest zbudowany w oparciu o niezrozumiałe wymagania, wynik będzie nieadekwatny do postawionego wcześniej celu. Z tego powodu pisanie testów dla wymagań powinno zostać rozpoczęte zaraz po rozpoczęciu projektu.

2.6. Zarządzanie wymaganiami

W skład zarządzania wymaganiami wchodzą wszystkie zadania związane z Zarządzaniem Zmianą, Zarządzaniem Wersją, Śledzeniem Statusu oraz Zakresu Wymagań. Celem zarządzania wymaganiami jest znalezienie informacji, zachowanie ich oraz zarządzanie nimi. Zarządzanie informacją znaczy to samo co organizacja, analiza i śledzenie [9].



Rys. 2. Elementy wchodzące w skład zarządzania wymaganiami

3. Metody Agile

W porównaniu do standardowego wytwarzania oprogramowania, „zwinne" wytwarzanie oprogramowania jest bardziej zorientowane na kod niż na dokumentację. Nie jest to natomiast jego kluczowa charakterystyka, a głębsza refleksja dotycząca różnicy pomiędzy dwoma poniższymi stylami [3]:
- Metody Agile są bardziej adaptacyjne niż predykcyjne. W tradycyjnych metodach większość elementów procesu jest szczegółowo zaplanowana na dłuższy okres. Takie podejście się sprawdza, jeśli niewiele rzeczy się zmienia, a domena aplikacji i technologie oprogramowania są dobrze rozumiane przez zespół projektowy. Metody Agile powstały w celu przystosowania się do zmian i ich rozwijania.
- Metody Agile są bardziej zorientowane na ludzi, niż na proces wytwarzania. Polegają na ich ekspertyzie, kompetencjach i bezpośredniej współpracy, niż na rygorystycznych, zcentralizowanych na dokumentacji procesach do wytwarzania wysokiej jakości oprogramowania.

4. Techniki inżynierii wymagań w metodach Agile

Większość technik inżynierii wymagań opisanych wcześniej można odnaleźć w różnych podejściach Agile, jednak mogą być one używane w innym obszarze lub innym rozmiarze. Poniżej zostaną opisane sposoby wykorzystania tych technik przez metody Agile.

4.1. Udział klienta

Kluczowym elementem wszystkich metodyk Agile jest bezpośrednia dostępność klienta.

Jest to celowe w celu uzyskania szybkiej opinii i dobrej komunikacji. Bezpośredni udział klienta w procesie wytwarzania jest pierwszym celem podejść „zwinnego" wytwarzania oprogramowania.

Zaangażowanie udziałowców jest także kluczowym elementem obecnej inżynierii wymagań. Różne techniki wyznaczania wymagań pomagają w uzyskaniu jak największej liczby informacji o systemie od wszystkich zaangażowanych w projekt ludzi (użytkowników, ekspertów dziedziny, itd.).

4.2. Wywiady

Chociaż bezpośredni udział klienta w projekcie jest głównym celem „zwinnych" metod wytwarzania oprogramowania, można powiedzieć, że najbardziej powiązaną z inżynierią wymagań techniką jest wywiad. Dostarcza ona bezpośredni i nieprzefiltrowany dostęp do poszukiwanej wiedzy. Programiści, którzy mają bezpośredni kontakt z klientem i docelowymi użytkownikami, uzyskują wiedzę posiadającą najmniejszą liczbę nieporozumień. Dodatkowo dzięki bezpośredniej komunikacji nawiązuje się więź pomiędzy klientem a programistami, co wpływa na zwiększające się zaufanie i pewność siebie.

4.3. Priorytetyzacja

Priorytetyzację można odnaleźć we wszystkich podejściach Agile. Najczęściej wykorzystywaną praktyką jest implementacja funkcjonalności z najwyższym priorytetem w celu dostarczenia klientowi najbardziej wartościowego oprogramowania. Podczas wytwarzania projektu zwiększa się wiedza i sposób rozumienia projektu, co przekłada się na dokładanie do jego zakresu nowych funkcjonalności, przez co proces priorytetyzacji jest powtarzany okresowo w celu utrzymania aktualnych priorytetów.

4.4. JAD

Sesje JAD wykorzystywane są w Agile Software Development w celu zwiększenia udziału klienta w projekcie. Podczas tych sesji zarówno programiści, jak i klienci dyskutują o funkcjonalnościach tworzonego projektu. Takie typy dyskusji mogą być bardzo owocne, jeśli tylko przewodniczący sesji będzie pilnował, aby nie zboczyła ona na poboczne tematy. Wyniki sesji są przeważnie udokumentowane i dostępne, jeśli powstaną w przyszłości dodatkowe pytania.

4.5. Modelowanie

Celem modelowania w inżynierii wymagań jest powstanie wielu różnych modeli systemu począwszy od jego ogólnego opisu, skończywszy na bardziej szczegółowych modelach, które następnie stają się częścią dokumentacji systemu i muszą być utrzymywane na bieżąco. Podobny cel ma modelowanie w metodzie FDD, gdzie tworzony model reprezentuje system, a wytwarzanie opiera się na tym modelu. Ponieważ fazy analizy i tworzenia są iteracyjne, zmiany w wymaganiach i ich konsekwencje w modelu mogą być wykonane w późniejszej fazie procesu wytwórczego. Model odzwierciedla pierwszy krok, jaki musi być wykonany, aby rozpocząć wytwarzanie, ale nie opisuje finalnej wersji systemu.

4.6. Dokumentacja

Kompletna dokumentacja wymagań ma swoją nikłą reprezentację w metodach Agile. Niektóre z tych metod posiadają jakiś rodzaj dokumentacji albo rekomendują użycie dokumentacji wymagań (DSDM, Scrum), ale decyzja co do jej zawartości i ogólnego kontekstu jest pozostawiana programistom i nie jest szczegółowo opisana.

Brak dokumentacji w „zwinnym" wytwarzaniu oprogramowania może powodować długoterminowe problemy, natomiast w standardowym wytwarzaniu może przyśpieszać pracę. Dokumentacja jest wykorzystywana do dzielenia się wiedzą między członkami zespołu projektowego. Nowy członek zespołu będzie miał wiele pytań dotyczących projektu, dla których odpowiedź mógłby znaleźć w dobrze prowadzonej dokumentacji. Szukanie odpowiedzi wśród pozostałych członków zespołu może wpłynąć na czas realizacji ich zadań, co bezpośrednio może wpłynąć na czas realizacji projektu.

Z drugiej jednak strony metody Agile powinny być bardziej produktywne w zagadnieniach wytwarzanego kodu niż tradycyjne metodyki, ponieważ poświęcają one mniej czasu na tworzenie dokumentacji, co ma miejsce w tradycyjnych podejściach inżynierii wymagań.

Wady i zalety prowadzenia dokumentacji związane są także z rozmiarem zespołu projektowego. W przypadku większego zespołu lepsze jest prowadzenie dokumentacji niż wyjaśnianie wiele razy tej samej kwestii różnym członkom zespołu. Kiedy metody Agile próbują się dostosować do większych zespołów projektowych, często wprowadzają większą ilość

produkowanej dokumentacji w porównaniu do mniejszych zespołów.

4.7. Walidacja

Walidacja wymagań odgrywa bardzo znaczącą rolę we wszystkich metodach „zwinnego" wytwarzania oprogramowania. Tworzony system jest weryfikowany poprzez spotkania przeglądowe i testy akceptacyjne, co zwiększa poczucie bezpieczeństwa klienta dotyczące projektu oraz zespołu projektowego, ponieważ pokazywane jest prawidłowe jego działanie w określonych wcześniej granicach. Spotkania przeglądowe także pokazują, że system wytwarzany jest zgodnie ze wcześniej określonym zakresem i harmonogramem.

Każda z metodyk ma podobne podejście do sposobu walidacji wymagań. Wykorzystują różnego rodzaju spotkania przeglądowe w celu prezentacji tworzonego systemu. Klient może skorzystać z tworzonego systemu, szczegółowo doświadczyć sposobu jego funkcjonowania i sprawdzić, które funkcjonalności zostały już zaimplementowane. Podczas spotkań przeglądowych klient może dowiedzieć się o sile, słabości, korzyściach i limitach projektowych i technologicznych. Może on sprawdzić, czy zaimplementowane wymagania zostały poprawnie zrozumiane przez zespół projektowy, a w konsekwencji, czy zostały poprawnie zaimplementowane.

Powyższe zagadnienia są szczególnie widoczne w metodzie XP, gdzie klient powinien zawsze być obecny podczas procesu wytwarzania oprogramowania w celu zadawania pytań i sprawdzania, czy wykonywane przypadki testowe zwiększają zaufanie w zespole projektowym i tworzonym systemie.

4.8. Zarządzanie

Ograniczone zarządzanie wymaganiami jest typową „cechą" „zwinnych" metod wytwarzania oprogramowania. Z punktu widzenia inżynierii wymagań powinno być możliwe śledzenie zmian wprowadzonych do wymagań, architektury, czy dokumentacji w celu weryfikacji konieczności wprowadzonych zmian. Jednak śledzenie zmian powoduje konieczność prowadzenia dodatkowej dokumentacji i większego nakładu pracy. Dodatkowo nie zostało jeszcze udowodnione, że śledzenie zmian przynosi jakikolwiek pieniężny zysk tworzonemu projektowi.

Pomimo tego metody Agile stanowią solidną podstawę do zarządzania wymaganiami. Wszystkie wymagania zapisywane są na poindeksowanych kartach (lub scenariuszach użytkownika) albo są utrzymywane jako element zwrotny produktu. Różnica leży jedynie w poziomie szczegółowości, poprzez który opisywane są wymagania, scenariusze użytkownika, czy funkcje. Metody Agile zwykle omijają szczegóły i pozwalają, aby były zdefiniowane w momencie ich implementacji w procesie wytwarzania. Innymi słowy opóźniają one koszt związany z uzyskaniem szczegółów poszczególnych wymagań do czasu ich realizacji w kolejnym przyroście.

4.9. Wymagania niefunkcjonalne

W „zwinnych" podejściach radzenie sobie z wymaganiami niefunkcjonalnymi nie jest zdefiniowane. Klienci lub użytkownicy, którzy wyrażają się na temat tego, co by chcieli uzyskać poprzez wytworzony system, nie myślą w kategoriach zasobów, utrzymania, dostarczalności, bezpieczeństwa albo wydajności. Powinien on dostarczać bardziej zdefiniowane sposoby radzenia sobie z wymaganiami niefunkcjonalnymi, co i tak nie jest dość dobrze zdefiniowane przez inżynierię wymagań.

5. Podsumowanie

Jak widać trzy główne fazy procesu inżynierii wymagań − Faza Wydobywania, Faza Analizy i Faza Weryfikacji − są obecne we wszystkich metodach Agile, a techniki są różnie wykorzystywane w różnych podejściach. Ponieważ fazy te nie są tak dobrze odseparowane, jak w procesie inżynierii wymagań, są one także łączone ze sobą w odmienny sposób. Są one także powtarzane w każdej iteracji, co czyni je trudnymi w podzieleniu na fazy, przez co ciężko jest je porównać do tradycyjnych faz procesu inżynierii wymagań. Techniki używane w procesach wytwarzania Agile są często dość niedokładnie opisane a ich implementacja pozostaje w rękach programistów. Jest to wynik wpływu, jaki wywiera się na wykwalifikowanych ludzi w metodykach „zwinnych" (mówi się, że dobry programista zawsze będzie wiedział, co zrobić). Natomiast tradycyjne metodyki szczegółowo opisują to, co ma być zrobione, dzięki czemu dostarczają każdemu programiście bardzo dokładne wskazówki, jak zrobić coś „dobrze".
Jako że tradycyjna inżynieria wymagań bazuje na dokumentacji, nie jest to tak samo dobrze prezentowane w podejściach Agile, ponieważ

dokumentacja w „zwinnych" metodach wytwarzania oprogramowania jest tylko ich drobną częścią, która w dużej mierze zależy od programistów.

Podsumowując, w kluczowych obszarach tradycyjna inżynieria wymagań, jak i metodyki Agile zmierzają ku podobnym celom. Główną ich różnicą jest natomiast wpływ na ilość dokumentacji, jaka musi być wytworzona w celu zapewnienia wysokiej efektywności projektu. To częściowo jest także wynikiem różnic, jakie wypływają z podstawowych założeń stabilności wymagań.

6. Bibliografia

[1] G. Kotonya, I. Sommerville, *Requirements Engineering*, John Wiley & Sons, 1997.
[2] S. Adolph, P. Bramble, *Patterns for Effective Use Cases*, Alistair Cockburn, 2001.
[3] A. Cockburn, *Agile Software Development*, Alistair Cockburn, 2000.
[4] K. Beck, M. Fowler, *Planning Extreme Programming*, Addison Wesley, 2000.
[5] K. Schwaber, *Agile Project Management with Scrum*, Microsoft Press, 2004.
[6] V. Subramaniam, A. Hunt, *Practices of an Afile Developer*, Pragmatic Bookshelf, 2006.
[7] I.G. Stamelos, P. Sfetsos, *Agile Software Development Quality Assurance*, Information Science Reference, 2007.
[8] S. Hayes, M. Andrews, *An Introduction to Agile Methods*, Australia, 2002.
[9] K.E. Wiegers, *Software Requirements*, Microsoft Press, 1999.

Requirements Engineering in Agile Software Development

A. LIPSKI

Agile Software Development approaches have become increasingly popular during the last few years. Agile practises have been developed with the aim to deliver software faster and to ensure that the software meets changing needs of customers. We can find out that there are a lot of practices and approaches which are created and developed in context of traditional Requirements Engineering and which are used by Agile methods with a good result. The goal of this article is to show how the Requirements Engineering technics are used by Agile methods and how this methods can be improved by them.

Keywords: Agile, Requirements Engineering, Software Engineering