Propozycja modelu dojrzałości dla Programowania Ekstremalnego*

Jerzy R. Nawrocki
Politechnika Poznańska, Instytut Informatyki
Jerzy.Nawrocki@put.poznan.pl

Bartosz Walter
Politechnika Poznańska, Instytut Informatyki
Bartosz.Walter@cs.put.poznan.pl

Adam Wojciechowski
Politechnika Poznańska, Instytut Informatyki
Adam.Wojciechowski@put.poznan.pl

Streszczenie

Rosnąca popularność lekkich metodologii rozwoju oprogramowania, takich jak Programowanie Ekstremalne (XP od ang. *eXtreme Programming*) skupia uwagę wielu zespołów programistycznych. Atrakcyjność tego stylu pracy płynie zarówno z ciekawego doboru praktyk programistycznych, jak i rezygnacji z wielu procedur formalnego dokumentowania postępów. Z naszych doświadczeń wynika jednak, że wiele osób traktuje XP jako wymówkę dla zaniedbań w realizacji tradycyjnych procedur zapewnienia jakości w budowie oprogramowania. Pełna realizacja praktyk proponowanych przez XP jest bardzo trudna. Potrzebny byłby więc model dojrzałości, który umożliwiłby ocenę stopnia zaawansowania zespołu wykonawców w stosowaniu praktyk XP. W pracy zaproponowany został prosty 4-poziomowy model dojrzałości dla zespołów stosujących XP w procesie wytwarzania oprogramowania.

* Praca wykonana w ramach projektu badawczego 43-364/01-BW.

1. Wprowadzenie

Metodologie wytwarzania oprogramowania określane mianem *lekkich*, reprezentowane m.in. przez Programowanie Ekstremalne (XP od ang. *eXtreme Programming*), szybko zdobywają popularność wśród zespołów programistów. Atrakcyjność takiego stylu pracy płynie częściowo z ciekawego doboru praktyk programistycznych. Praktyki XP zostały bowiem tak zdefiniowane, aby tworzyć zbiór wzajemnie uzupełniających się procedur wytwarzania oprogramowania oraz zapewnienie jakości i osiągnięcie satysfakcji klienta. Silnym argumentem przemawiającym za stosowaniem w praktyce podejścia XP jest rezygnacja z formalnych inspekcji oprogramowania oraz zaniechanie obowiązku gromadzenia szczegółowej i zwykle bardzo obszernej dokumentacji wymagań, projektu, raportów z testowania itp.

Z naszych obserwacji wynika, że dla wielu programistów XP stanowi wymówkę dla zaniedbań w zakresie realizacji tradycyjnych procedur zapewnienia jakości w procesie budowy oprogramowania. Porzucając klasyczny model dokumentowania prac nad budową oprogramowania, zespoły te nie realizując w pełni praktyk wymaganych w podejściu XP. Niestety, część z praktyk XP bywa trudna w realizacji. Dobrym przykładem może być tutaj wymaganie stałej obecności przedstawiciela klienta władnego podejmować decyzje o charakterze biznesowym odnoszące się do projektu. Ważne jest aby reprezentant klienta, stale współpracujący z zespołem programistów, odporny był na konflikty o charakterze politycznym w swej macierzystej organizacji [1]. Rola osób o takich kompetencjach i osobowości bywa jednak na tyle istotna dla firmy klienta, że pracownik taki jest trudno dostępny dla zespołu programistów.

Należy zatem rozróżnić stopnie zaawansowania w realizacji praktyk XP. Poziomy dojrzałości mogą stanowić miarę ryzyka związanego z realizacja projektu przez dany zespół programistów. Warto również, aby podobnie jak w modelu CMM(I) [2], poziomy dojrzałości wskazywały kierunek doskonalenia procesów budowy oprogramowania i wzrost satysfakcji klienta.

W pracy zaproponowany został prosty 4-poziomowy model dojrzałości dla zespołów realizujących projekty informatyczne wg metodologii XP. W rozdziale drugim przedstawione zostały zasadnicze praktyki modelu XP. Kolejny rozdział stanowi krótki przegląd uznanych modeli dojrzałości zespołów wytwarzających oprogramowanie. Z tego przeglądu wyciągnęliśmy wniosek, iż najbardziej odpowiedni będzie model przypominający CMMI i PSP. W czwartym rozdziale przedstawiona została propozycja modelu dojrzałości dla metodologii XP. Poszczególne poziomy proponowanego modelu zostały szerzej opisane w rozdziałach 5-8.

2. Krótka charakterystyka Programowania Ekstremalnego

Programowanie Ekstremalne jest *lekką* metodologią budowy oprogramowania [3]. XP jest zorientowane na szybką budowę oprogramowania poprzez przyrostowe dodawanie własności funkcjonalnych poczynając od tych, które posiadają najważniejsze znaczenie dla klienta. Elastyczny charakter tego podejścia do budowy oprogramowania płynie z doświadczeń kierowników projektów, programistów i osób odpowiedzialnych za logikę biznesową w projektach informatycznych. Model ten zakłada preferencję nieformalnej, ustnej komunikacji wewnątrz zespołu nad metodami formalnego dokumentowania pracy w formie dokumentów tekstowych. Stała obecność przedstawiciela klienta, który w czasie prac nad projektem staje się członkiem zespołu wykonawców daje szansę na szybkie wprowadzanie zmian do projektu na bardzo wczesnych etapach realizacji prac. XP jest podejściem proponowanym dla zespołów kilku- lub kilkunastoosobowych. Opiera się na pracy par programistów i zdarza się, że XP jest kojarzone wyłącznie z programowaniem parami. Zestaw podstawowych praktyk XP jest jednak zdecydowanie szerszy i obejmuje m.in.:

- Gra planistyczna (ang. *planning game*) – krótkie negocjacje i decyzje odnośnie priorytetów i wartości biznesowych płynących z realizacji poszczególnych funkcji systemu planowanych do realizacji w kolejnej edycji systemu.
- Metafora – prosty opis projektu, oparty na analogiach do innych systemów lub obiektów znanych klientowi i producentowi.
- Prostota projektu – zakłada implementację jedynie niezbędnej funkcjonalności odkładając kwestie optymalizacji do czasu, gdy będzie one faktycznie potrzebna.
- Małe wydania (ang. *small releases*) – budowa systemu poprzez przyrostowe zwiększanie funkcjonalności daje szansę na szybką reakcję klienta i uwzględnienie jego zastrzeżeń i uwag w kolejnych przyrostach.
- Programowanie parami – wspólna praca dwóch osób przy jednym komputerze stwarza warunki dla ciągłego przeglądu kodu. W ocenie programistów praca w parach jest oceniana jako styl dający więcej satysfakcji niż praca indywidualna [4]. Problemem jest wzrost pracochłonności [8].
- Współwłasność kodu – wszyscy członkowie zespołu programistów współdzielą cały kod. Każdy z nich może wprowadzać zmiany do dowolnego fragmentu oprogramowania.
- Testowanie – obowiązkowe jest opracowanie przypadków testowych przed rozpoczęciem kodowania. Podejście to wymusza dokładne poznanie dziedziny zastosowania i zrozumienie działania powstającego systemu.

- Ciągła integracja – każda para programistów może dowolnie często integrować swój kod z całością systemu dodając tym samym nową funkcjonalność do systemu. Dodanie nowych elementów do systemu pociąga za sobą konieczność ponownego wykonania wszelkich testów weryfikujących poprawność działania zmodyfikowanego systemu (testowanie regresyjne).
- Refaktoryzacja – jest tu miejsce na pewną optymalizację i modyfikację kodu bez zmiany realizowanej funkcjonalności. Procedura ta jest stosowana w przypadkach jasnej wizji korzyści płynących ze zmian lub poprawek systemu (np. eliminacja redundancji itp.).
- Standard kodowania – wszyscy programiści pracujący w zespole stosują wspólnie zaakceptowany sposób zapisu kodu źródłowego. Standard kodowania obejmuje zarówno sposób zapisu instrukcji języka programowania jak i włączania komentarzy, które stanowią ważną część technicznej dokumentacji systemu.
- Klient w zespole (ang. *on-site customer*) – stała obecność przedstawiciela przyszłych użytkowników systemu daje szansę na bezpośredni kontakt i konsultacje z klientem w chwili rozstrzygania napotykanych wątpliwości.
- 40-godzinny tydzień pracy – XP zakłada systematyczność przy jednoczesnym założeniu braku pracy „po godzinach".

3. Modele dojrzałości

Capability Maturity Model (CMM) zaproponowany przez Software Engineering Institute (SEI) stanowi schemat reprezentujący ścieżki poprawy procesów wytwarzania oprogramowania. Model CMM jest zorientowany na doskonalenie pracy organizacji i jest niezależny od dziedziny działalności. Hierarchiczna, 5-cio poziomowa struktura CMM stała się szeroko akceptowanym wyznacznikiem dojrzałości procesów stosowanych wewnątrz firmy. Z wyjątkiem poziomu pierwszego (ang. *Initial*), każdy z poziomów jest dekomponowany na obszary (ang. *Key Process Areas*, KPA). Każdy KPA definiuje zestaw czynności, których łączne wykonanie pozwala osiągnąć cele istotne dla dojrzałości procesu wytwórczego w firmie.

Innym podejściem do definiowania modelu dojrzałości organizacji został zaproponowany przez Sommerville'a i Sawyer'a [5]. W modelu tym zebrany został zestaw praktyk istotnych z punktu widzenia inżynierii oprogramowania, a następnie przyjęto zasadę, że realizacja poszczególnych praktyk może być oceniona w zakresie od 0 do 3 punktów w zależności od stopnia realizacji danej praktyki. Poziom dojrzałości oceniany jest na podstawie łącznej sumy punktów uzyskanych za realizację poszczególnych praktyk. Konsekwencją takiego kom-

pensacyjnego sposobu oceny jest możliwość identycznego sklasyfikowania dojrzałości organizacji realizujących różne praktyki.

W świetle prac nad modelem dojrzałości dla XP, z naszego punktu widzenia do istotnych charakterystyk CMM należą:

- Model CMM składa się z pięciu poziomów dojrzałości. Z każdym poziomem związany jest zbiór praktyk (rozumianych jako sposób realizacji zadań w poszczególnych KPA) i wszystkie praktyki związane z danym poziomem CMM muszą być wdrożone w celu uznania stopnia dojrzałości organizacji. Proponowany model dojrzałości w sensie XP, zakłada 4-poziomową strukturę obowiązkowych praktyk.
- Firmy, których sposób działania nie przystaje do modelu CMM klasyfikowane są na poziomie 1. Również w proponowanym modelu dojrzałości dla XP przyjęliśmy zasadę zaklasyfikowania na poziom 1 wszystkich tych organizacji, których działalność nie przystaje do obowiązkowych praktyk wymaganych na wyższych poziomach tego modelu.
- Poziom 2 modelu CMM jest zorientowany na projekt, podczas gdy na wyższych poziomach dojrzałości (3, 4, i 5) nacisk kładziony jest cele organizacyjne. Według naszej oceny, podejście XP jest zorientowane na projekty, nie zaś na cele organizacyjne. Kent Beck w swojej książce [3] jasno stwierdza, że metodologia XP najlepiej nadaje się dla małych zespołów programistów (do 10 osób). Zatem nasz model, w przeciwieństwie do CMM, będzie zorientowany na zespoły związane z projektami budowy oprogramowania.
- W modelu CMM, zorientowanym na doskonalenie procesów wewnątrz firmy, dojrzałość organizacji oceniana jest na podstawie zestawu praktyk, które są realizowane. Nie ma tu bezpośredniej oceny wyników pracy. Założeniami XP są m.in. satysfakcja klienta i brak pracy ponadnormatywnej. Zatem celem naszym jest, aby model dojrzałości dla XP wyraźnie rozgraniczał projekty „udane", czyli zrealizowane w założonym czasie i z pełną satysfakcją klienta od projektów zakończonych częściowym sukcesem lub porażką.

Model PSP zorientowany na doskonalenie pracy indywidualnego programisty [6] jest strukturalnie podobny do modelu CMM. Różna jest tu liczba poziomów dojrzałości i inne niż w CMM są praktyki wymagane od programisty w celu zaklasyfikowania jego stylu pracy na kolejne poziomy. Celem realizacji coraz bardziej zaawansowanych praktyk jest wzrost jakości wytwarzanego oprogramowania, doskonalenie metod planowania pracy i przewidywania efektów oraz ulepszanie procesu wytwarzania oprogramowania.

Celem procesu organizacji doskonalącej swój proces wytwarzania oprogramowania jest zrozumienie i powielanie sukcesów oraz poznanie i unikanie błędów [7]. Odrębne podejście do doskonalenia procesu wytwarzania oprogramowania zostało przedstawione przez NASA. Model ten zorientowany jest na końcowy

produkt i zakłada cykliczne doskonalenie procesów wytwórczych wewnątrz organizacji. W modelu tym nie ma zdefiniowanych poziomów dojrzałości, ani uniwersalnych zestawów praktyk. Każdy etap poprawy procesów jest podzielony na trzy fazy: poznanie (ang. *understanding*), ocena (ang. *assessing*) i opracowanie (ang. *packaging*). Dobre praktyki, które mają być wdrożone przez daną organizację są identyfikowane w fazie poznania. Następnie ich pilotażowe wdrożenie podlega ocenie. Jeśli uzyskane wyniki okazują się pozytywne opracowywane są procedury poprawy procesów, które następnie są dystrybuowane do poszczególnych działów firmy w celu wdrożenia gotowych procedur poprawy procesów. Ponieważ praktyki XP są ściśle określone i opisane, uznaliśmy, że model zaproponowany w NASA nie jest właściwym sposobem doskonalenia kompetencji w zakresie wytwarzania oprogramowania wg metodologii Programowania Ekstremalnego.

4. Proponowany model dojrzałości dla Programowania Ekstremalnego, XPMM

Trudno jest jednoznacznie ocenić, czy dany zespół pracuje zgodnie z regułami XP. Zbiór praktyk XP został przedstawiony m.in. w pracach [3, 9]. Nie ma tam jednak wskazówek ile z wymienionych praktyk lub jaka ich część musi zostać wdrożona, aby można było mówić o pracy zgodnej z XP. Jednoznaczne ocenianie pracy jako niezgodnej z duchem Programowania Ekstremalnego w sytuacji niespełnienia wymogów niektórych praktyk może być krzywdzące dla zespołu programistów. Brak pełnej realizacji niektórych praktyk bywa konsekwencją czynników zewnętrznych i nie musi oznaczać braku zaangażowania zespołu w realizację projektu wg zasad XP. Rozważmy następujące sytuacje:

- W wielu przypadkach programiści nie mają fizycznej możliwości, aby cały kod powstawał w czasie pracy w parach. Owszem, część oprogramowania powstaje w czasie wspólnej pracy par, ale warunek ten nie jest spełniony w stosunku do całości kodu. Czy w takim przypadku można mówić o pełnej realizacji praktyki XP?
- Nietrudno wyobrazić sobie sytuację, w której przedstawiciel klienta nie będzie mógł pozostawać z zespołem programistów przez cały czas pracy nad systemem.
- W idealnym przypadku zarządzanie wersjami oprogramowania powinno być w pełni zautomatyzowane poprzez zastosowanie narzędzi zarządzania wersjami. Czy jednak dużym uchybieniem jest powierzenie tego obowiązku członkowi zespołu, który będzie odpowiedzialny za administrowanie wersjami i integrację kodu?
- Ze względu na wewnętrzne i zewnętrzne ograniczenia, zespół programistów może stanąć przed koniecznością pracy w godzinach nadliczbowych. Warto

tu zauważyć, że praktyka ta nie należy do rzadkości w wielu firmach wytwarzających oprogramowanie.

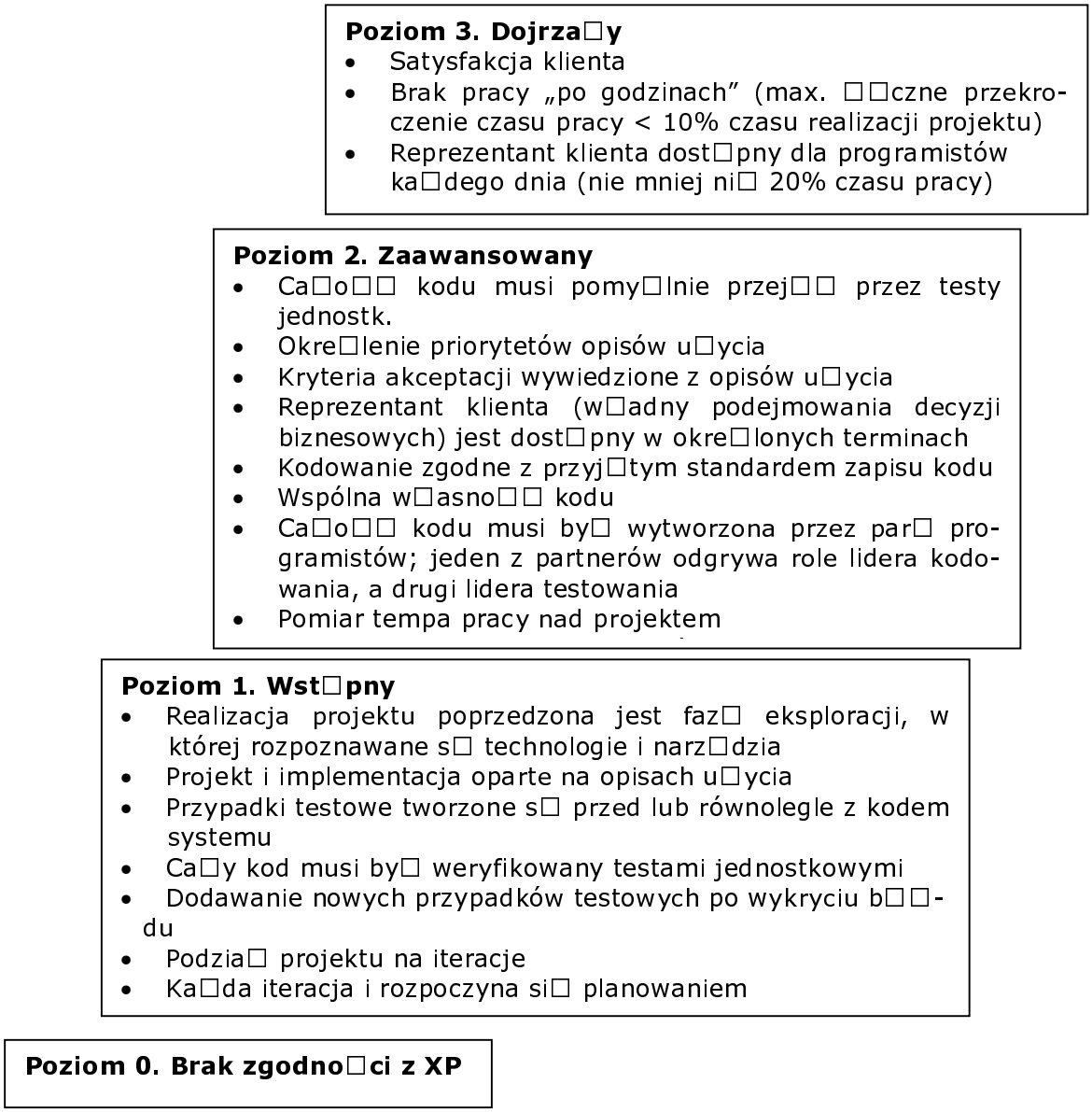
Wszystkie przedstawione sytuacje stanowią pewne odstępstwa od praktyk XP. Należy jednak zauważyć, że spełniając pozostałe warunki określane zestawem praktyk, taki zespół programistów byłby bardzo bliski ideałowi pracy zgodnej z XP. W ocenę taką wkrada się jednak pewien relatywizm i subiektywne odczucia odnośnie znaczenia poszczególnych praktyk dla XP. Rozwiązaniem obiektywnej oceny stopnia realizacji XP byłby hierarchiczny model dojrzałości określający zakres obowiązkowych praktyk wymaganych na poszczególnych poziomach.

Konstruując model dojrzałości dla XP, który miałby być stosowany w firmach produkujących oprogramowanie warto wziąć pod uwagę wymiar praktycznej użyteczności modelu dla firmy i jej klientów. Do istotnych czynników wpływających na możliwość zaakceptowania modelu w środowisku przemysłowym należą:

- Niski koszt wdrożenia i oceny pracy zespołu.
- Jasna wizja związku pomiędzy poziomem dojrzałości zespołu (w sensie przyjętego modelu) a wytwarzanymi produktami.
- Czytelna hierarchiczna struktura obowiązkowych praktyk XP związanych z każdym poziomem dojrzałości.
- Osiągnięcie wyższego poziomu dojrzałości powinno wiązać się z uzyskaniem korzyści płynących z poprawy procesu i lepszego wykorzystania dostępnych technologii (wzrost kompetencji).
- Poprawa procesu wytwarzania oprogramowania musi iść w parze ze wzrostem jakości produktów (gwarancja jakości dla klienta).
- Model dojrzałości powinien być równie *lekki* jak metodologia XP.

Struktura zaproponowanego modelu dojrzałości (por. rys. 1) jest zbliżona do CMM lub PSP w takim sensie, że osiągnięcie kolejnych poziomów dojrzałości wiąże się ze wzrostem kompetencji zespołu programistów. Zbiór obowiązkowych praktyk rozrasta się z poziomu na poziom. Poziom 0 jest kategorią, do której klasyfikowane są zespoły nie realizujące praktyk XP w swojej pracy. Na poziomie 1 (wstępnym) wymagany jest od członków zespołu raczej wąski zakres realizacji praktyk skupiający się na zbudowaniu ram organizacyjnych dla pracy zespołowej. Warto tutaj zauważyć, że realizacja praktyk z poziomu 1 jest raczej kwestią zmiany nawyków w pracy, a nie pociąga za sobą konieczności dodatkowych inwestycji. Na poziomie 2 modelu dojrzałości (zaawansowany) obowiązkowe praktyki zostały rozszerzone o działania służące zapewnieniu jakości pracy programistów oraz minimalizacji ryzyka niezadowolenia klienta. Praktyki na najwyższym poziomie w modelu dojrzałości, poziom 3 (dojrzały), zorientowane są na osiągnięcie swoistej doskonałości w pracy nad projektami informatycznymi. Miarą sukcesu

jest tutaj zadowolenie klientów, umiejętność konstruowania wykonalnych planów i terminowa ich realizacja.



Rysunek 1. Model dojrzałości Programowania Ekstremalnego

5. Poziomy dojrzałości w proponowanym modelu

5.1. Poziom 0

Na najniższym poziomie dojrzałości proponowanego modelu (brak zgodności z XP) klasyfikowane są te projekty i zespoły realizujących je programistów, które nie realizują poprawnie żadnych praktyk Programowania Ekstremalnego.

5.2. Poziom 1

Poziom 1 (wstępny) przewidziany jest dla zespołów realizujących podstawowe praktyki XP. Dobór praktyk nie jest przypadkowy. Wzięto tutaj pod uwagę niski (w zasadzie zerowy) koszt adaptacji praktyk do realizacji projektu. Część praktyk XP zostaje pominięta na poziomie wstępnym a zostają one przeniesione na wyższe poziomy. Celem takiego postępowania jest przekonanie zespołu wykonawców do praktyk XP i wykonanie pierwszego kroku na drodze do dojrzałości w sensie XP. Inne praktyczne względy skłoniły nas do ograniczenia niektórych praktyk. Przykładowo jeśli stała obecność przedstawiciela klienta wśród zespołu wykonawców nie jest możliwa należy zastąpić ją spotkaniami, prezentacjami, dyskusjami itp. Niestety atmosfera takich spotkań bywa bardziej formalna od codziennych kontaktów. Na tym poziomie dopuszcza się możliwość pracy bez reprezentanta klienta o ile nie wpłynie to na niewykonalność projektu.

Praktyki XP wybrane na poziomie 1 obejmują gromadzenie opisów użycia dostarczanych przez klienta lub rejestrowanych w czasie spotkań z reprezentantem klienta. Opisy użycia stanowią podstawę dla projektu i weryfikacji poprawności realizacji. Przypadki testowe muszą powstawać nie później, niż równolegle z kodem. Taki styl pracy wymusza dokładne zrozumienie dziedziny i algorytmów działania aplikacji jeszcze przed jej napisaniem. Równoległa praca nad przypadkami testowymi i kodem jest również wprowadzeniem do pracy w parach. Należy tutaj zauważyć, że na tym poziomie modelu dojrzałości nie ma silnego wymagania dla pracy w parach.

Lekki charakter XP znajduje swój wyraz w swobodzie eksperymentowania z kodem. Zamiast długich dyskusji i czytania obszernej dokumentacji podejście XP sugeruje fazę eksploracji, w której zespół wykonawców poznaje narzędzia i technologie, w których realizowany będzie projekt. Praca zespołowa wymaga dobrej koordynacji i właściwego przydziału zadań do członków zespołu, aby w efekcie odnosić korzyści płynące z synergii wysiłków całego zespołu. Począwszy od poziomu wstępnego, w modelu dojrzałości XP zakłada się budowę krótko- i długo-

terminowych planów poddawanych dyskusji na etapie planowania iteracji i wydań.

5.3. Poziom 2

Na poziomie 2 (zaawansowany) modelu dojrzałości, zestaw praktyk z poziomu 1 rozszerzony jest o działania, których realizacja zorientowana jest na jakość produktu.

Programowanie parami, które było rekomendowane, ale nie obowiązkowe na poziomie niższym staje się tutaj praktyką obowiązkową. Para programistów współdzieli własność i odpowiedzialność za wytwarzany kod. Jeden z programistów pełni rolę lidera kodowania, a drugi kieruje testowaniem. Takie rozwiązanie, wprowadzające podział obowiązków, pozwala na rozłożenie w czasie momentów największego skupienia obu partnerów. Choć podejście XP nie wymaga formalnych przeglądów kodu to stała obserwacja działań partnera spełnia taką właśnie funkcję. Wspólna praca całego zespołu nad kodem programu (skład par programistów ulega zmianie) wymusza programowanie zgodnie z przyjętym stylem.

Rola przedstawiciela klienta została jasno zdefiniowana na poziomie zaawansowanym modelu dojrzałości. Bez jego zaangażowania w realizację projektu znacznie rośnie ryzyko niepowodzenia. Wprowadzone praktyki określają zadania realizowane wspólnie z reprezentantem klienta. Obejmują one: konstrukcję opisów użycia, określanie priorytetów dla cech funkcjonalnych projektowanej aplikacji oraz konsultowanie bieżących postępów pracy. Ważnym elementem współpracy z klientem jest określenie terminów i czasu cyklicznych spotkań. Zgodnie z zasadą 20-80, 20% procent pracy i wysiłków programistów zaspokaja 80% funkcjonalności projektowanej aplikacji potrzebnej klientowi.

Rejestrowanie postępów pracy zespołowej może być realizowane na co najmniej dwa sposoby. Po pierwsze zespół może oddelegować jedną osobą do obserwacji prac poszczególnych par i zbierania uwag nt. zrealizowanych zadań i napotkanych problemów. Dobrym rozwiązaniem jest, kiedy tym, obserwatorem jest członek zespołu odpowiedzialny za integrowanie kodu, albowiem regularnie spotyka się on ze wszystkimi parami programistów, co stwarza okazję do nieformalnych rozmów o projekcie. Innym sposobem na zbieranie danych o tempie realizacji projektu jest przygotowywanie okresowych raportów. Podejście to narzuca obowiązek wypełniania formularzy, co jest to obce osobom pracującym w duchu XP. Dlatego pierwsze z podejść wydaje się nam lepsze.

5.4. Poziom 3

Najwyższy poziom modelu dojrzałości (dojrzały) podkreśla istotę satysfakcji klienta i zespołu programistów. Dla osiągnięcia tych celów proponowane jest minimalizowanie godzin pracy dodatkowej (XP zakłada 40-godzinny tydzień pracy) oraz łatwy kontakt z klientem, który poprzez swoją codzienną obecność wśród programistów integruje się z zespołem służąc jednocześnie wiedzą dziedzinową i podejmując decyzje o charakterze biznesowym.

6. Podsumowanie

Model dojrzałości dla Programowania Ekstremalnego, przedstawiony w niniejszej pracy, został zaprojektowany w celu ułatwienia etapowego wdrożenia XP do małych zespołów programistów oraz stworzenia warunków porównywania ze sobą zespołów głoszących, że realizują projekty wg zasad XP. Proponowany model dojrzałości został poddany eksperymentalnej ocenie w ramach prac nad projektami inżynierskimi realizowanymi na Politechnice Poznańskiej. Pięć zespołów 6-osobowych (2 menadżerów i 4 programistów) realizowało swoje prace w oparciu o metodologię XP. Ich zadaniem była realizacja możliwie wielu praktyk XP pozwalając na osiągnięcie jak najwyższego poziomu dojrzałości pracy w sensie przyjętego modelu. Rezultaty eksperymentów zostaną poznane w lipcu 2001r.

Bibliografia

[1]. Yourdon E., *Death March: The Complete Software Developer's Guide to Surviving 'Mission Impossible' Projects*, Prentice Hall, 1999.

[2]. CMU/SEI-93-TR-24 ESC-TR-93-177, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, USA.

[3]. Beck K., *Extreme Programming Explained: Embrace Change*, Addison Wesley Longman, Reading Mass. 2000.

[4]. Williams L., Kessler R., Cunningham W., Jeffries R., *Strengthening the Case for Pair Programming*, IEEE Software, July/August 2000.

[5]. Sommerville I., Sawyer P., *Requirements Engineering. A good practice guide,* John Wiley & Sons, 1998.

[6]. Humphrey W., *Discipline for Software Engineering*, Addison-Wesley, Reading, 1995.

[7]. http://www.ivv.nasa.gov/SWG/docs/process, February 2001.

[8]. Nawrocki J., Wojciechowski A., *Experimental Evaluation of Pair Programming*, Proceedings of the 12th European Software & Metrics Conference ESCOM'01, London, 2001, 269-276.

[9]. http://www.extremeprogramming.org, luty 2001.