Programowanie Ekstremalne i PRINCE2

Jerzy R. Nawrocki, Michał Jasiński
Instytut Informatyki, Politechnika Poznańska
{Jerzy.Nawrocki, Michal.Jasinski}@cs.put.poznan.pl

Droga prosta jest najkrótsza, ale na ogół wymaga najwięcej czasu, aby dojść nią do celu.
Georg Christoph Lichtenberg (1742 - 1799)

Streszczenie

Jednym z głównych źródeł sukcesu współczesnych organizacji jest oprogramowanie. Jednocześnie producenci aplikacji są pod ciągłym naciskiem klientów, oczekujących spełnienia ich rosnących oczekiwań, z uwzględnie-niem szybko zmieniających się wymagań i znacząco zredukowanych budżetów przeznaczonych na inwestycje informatyczne. Lekkie metodyki tworzenia oprogramowania, takie jak Programowanie Ekstremalne (XP), zdobyły ogromną popularność umożliwiając sprostanie tym wymogom oraz oferując podejście pozwalające minimalizować koszty wytwarzania produktów programistycznych. Jednakże wiele organizacji zainteresowanych wdrażaniem zwinnych metodyk obawia się, że XP stoi w sprzeczności ze stosowaniem tradycyjnych metod i narzędzi inżynierii oprogramowania, a także podejść do zarządzania przedsięwzięciami. Celem rozdziału jest zaproponowanie sposobu połączenia lekkiej metodyki rozwoju oprogramowania z ela-stycznym zarządzaniem przedsięwzięciami opartym na metodzie PRINCE 2.

Wstęp

Ankieta przeprowadzona pod koniec 2001 roku wśród kierowników amerykańskich firm posiadających największe działy IT wykazała znaczący wzrost popularności lekkich metodyk wytwarzania oprogramowania. Cutter Consortium przewiduje w niej, że w 2003 roku aż 42% organizacji zamierza wykorzystywać zwinne (ang. agile) metodyki w dużych przedsięwzięciach informatycznych, podczas gdy w 2001 roku zainteresowanych było zaledwie 19% z nich [Cutter2001]. Ankieta wykazała ponadto, iż zarówno rozmiar, jak czas trwania przedsięwzięć realizowanych z wykorzystaniem lekkich metodyk systematycznie rośnie. Rodzi to nowe wyzwania, ale i nowe problemy, z którymi przyjdzie się zmierzyć ich uczestnikom. Popularność zwinnych metodyk, takich jak Programowanie Ekstremalne (XP) [Beck2000], [Beck2001] [Jeffries2001], rodzina metodyk Crystal [Cockburn2002], czy SCRUM [Schwaber2001] wynika przede wszystkim z ich

genezy. XP i inne lekkie podejścia, wyrosły na pragmatycznym gruncie sukcesu tzw. "dobrych praktyk" programistycznych. Praktyki te obejmują m. in. aktywny udział klienta w pracach rozwojowych, szybkie sprzężenie zwrotne pomiędzy formułowaniem wymagań biznesowych a ich realizacją, nieustanną pielęgnację jakości wytwarzanego oprogramowania poprzez intensywne testowanie, refaktoryzację oraz konstrukcję efektywnego zespołu (programowanie parami [Nawrocki2001]). Pojawienie się Programowania Ekstremalnego było również próbą rozwiązania odwiecznego dylematu pomiędzy wykorzystaniem podejść strukturalnych, jak CMM [Paulk1994], CMMI [CMMI2003], czy ISO 9001 [ISO9001-2000], a procesami ukierunkowanymi na ludzi. Metodyki PSP [Humphrey1995] i TSP [Humphrey2000] wywodzące się z klasycznego nurtu inżynierii oprogramowania, nie spełniły, niestety, pokładanych w nich w tym zakresie oczekiwań. Lekkie podejścia zaproponowały wreszcie odpowiedź na niepewność i ciągłą zmienność wymagań, stawianych przez organizacje zmuszone konkurować na coraz bardziej wymagających rynkach. Wszystkie te cechy zwinnych metodyk charakteryzują cztery wartości zawarte w Agile Manifesto [AM2003] [Cockburn2002]:

- Jednostki i interakcje są ważniejsze niż procesy i narzędzia,
- Działające oprogramowanie jest ważniejsze niż obszerna dokumentacja,
- Współpraca z klientem jest ważniejsza niż negocjacja kontraktu,
- Nadążanie ze zmianami jest ważniejsze niż trzymanie się planu.

Niestety, zmiany, o których myśleli twórcy Agile Manifesto nie zawsze dotyczą wyłącznie oczekiwań klientów. Niewielkie przedsięwzięcia, z myślą o których zaproponowano XP, z czasem mogą urosnąć do rozmiarów dużo większych niż początkowo zakładano. Tym samym zmieniają się nie tylko wymagania, ale i warunki realizacji przedsięwzięcia. Klasyczną odpowiedzią organizacji na rosnącą złożoność przedsięwzięć informatycznych są formalne metody zarządzania, takie jak PRINCE 2 [PRINCE2002], czy PMBOK [PMBOK1996]. Jednakże wytwarzanie oprogramowania rozumiane jest w ich ujęciu, jako ściśle zdefiniowany proces, nie zaś, charakterystyczne dla lekkiego podejścia, kreatywne działania pojedynczych osób, czy niewielkich zespołów. Ponadto brak im elastyczności i rozwiązań pozwalających szybko reagować na zmiany pojawiające się w procesach, produktach i środowisku rozwojowym. Stąd powstaje pytanie, czy i w jaki sposób można przygotować się na zmiany w sposobie prowadzenia przedsięwzięcia tak, aby połączyć zalety zwinności XP oraz kontroli oferowanej przez PRINCE 2. W kolejnym podrozdziale przedstawiamy krótko metodykę PRINCE 2. W sekcji 3. pokazujemy, w jaki sposób można pogodzić pozorne sprzeczności wynikające ze stosowania Programowania Ekstremalnego w przedsięwzięciu zarządzanym zgodnie z metody-ką PRINCE 2 oraz opisujemy zalety wynikające z wykorzystania tego podejścia. Ostatni rozdział przedstawia wyniki pierwszych doświadczeń związanych z rozwiązaniem zaproponowanym w niniejszym rozdziale.

PRINCE 2

PRINCE 2 [PRINCE2002] jest strukturalną metodyką oferującą organizacjom standardowe podejście do organizacji, zarządzania i kontroli przedsięwzięć. O jej wartości

świadczą przedsięwzięcia zrealizowane z powodzeniem na całym świecie, jak również w Polsce (np. "Projekt 2000" Telekomunikacji Polskiej S.A. [CRM2003]). Nazwa metodyki wywodzi się z angielskiego "PRojects IN Controlled Environments" (Przedsięwzięcia Realizowane w Kontrolowanych Środowiskach).

Przedsięwzięcie realizowane zgodnie z PRINCE 2 sterowane jest przez tzw. spojrzenie biznesowe, zawierające wszelkie uzgodnienia dotyczące produktów, które mają zostać wytworzone. Należy zaznaczyć, iż uzasadnienie biznesowe, jak również plany i analizy ryzyka, są na bieżąco weryfikowane, aby zapewnić, że przedsięwzięcie będzie kontynuowane tylko wówczas, gdy adresuje rozwiązania problemów danej organizacji.

PRINCE 2 dzieli każde przedsięwzięcie na zarządzane i kontrolowane etapy. Tym samym, kontrola oraz szacowanie kosztów i czasu realizacji nawet złożonych projektów są znacznie ułatwione. Ma na to również wpływ hierarchiczna struktura organizacyjna zespołu projektowego, sprzyjająca jednoznacznemu podziałowi obowiązków i odpowiedzialności za poszczególne obszary przedsięwzięcia. Podejście to jest na tyle elastyczne, iż może być stosowane w niemal każdym przypadku.

Warto w tym momencie zauważyć, że zarówno PRINCE 2, jak i XP dostarczają przejrzystych reguł komunikacji i podejmowania decyzji w ramach zespołu. Ponadto oba te podejścia zorientowane są raczej na produkty, niż czynności, które prowadzą do ich wytworzenia, co ułatwia komunikację pomiędzy dostawcą oprogramowania, a klientem. Wreszcie, planowanie zorientowane na produkt, umożliwia łatwiejszy pomiar zarówno bieżących postępów prac, jak i jakości końcowych produktów. Co więcej, PRINCE 2 wspiera standard ISO 9001:2000 citeISO9001-2000, stanowiąc dobry fundament do budowania systemu zarządzania jakością w organizacji programistycznej [Nawrocki2002a].

Niestety, PRINCE 2 ma również słabe strony. Najistotniejszą z nich, z punktu widzenia lekkich metodyk, jest duża ilość dokumentacji. Co gorsza, organizacja musi ją opracować od podstaw, ponieważ PRINCE 2 definiuje jedynie zawartość poszczególnych dokumentów, nie dostarczając jednocześnie odpowiednich szablonów. Ponadto, proces planowania proponowany przez PRINCE 2 wymaga określenia czasu trwania i zakresu przedsięwzięcia już na jego początku, co jest sprzeczne z wartościami Agile Manifesto. W dalszej części pracy zaprezentujemy nasze propozycje rozwiązania tych problemów.

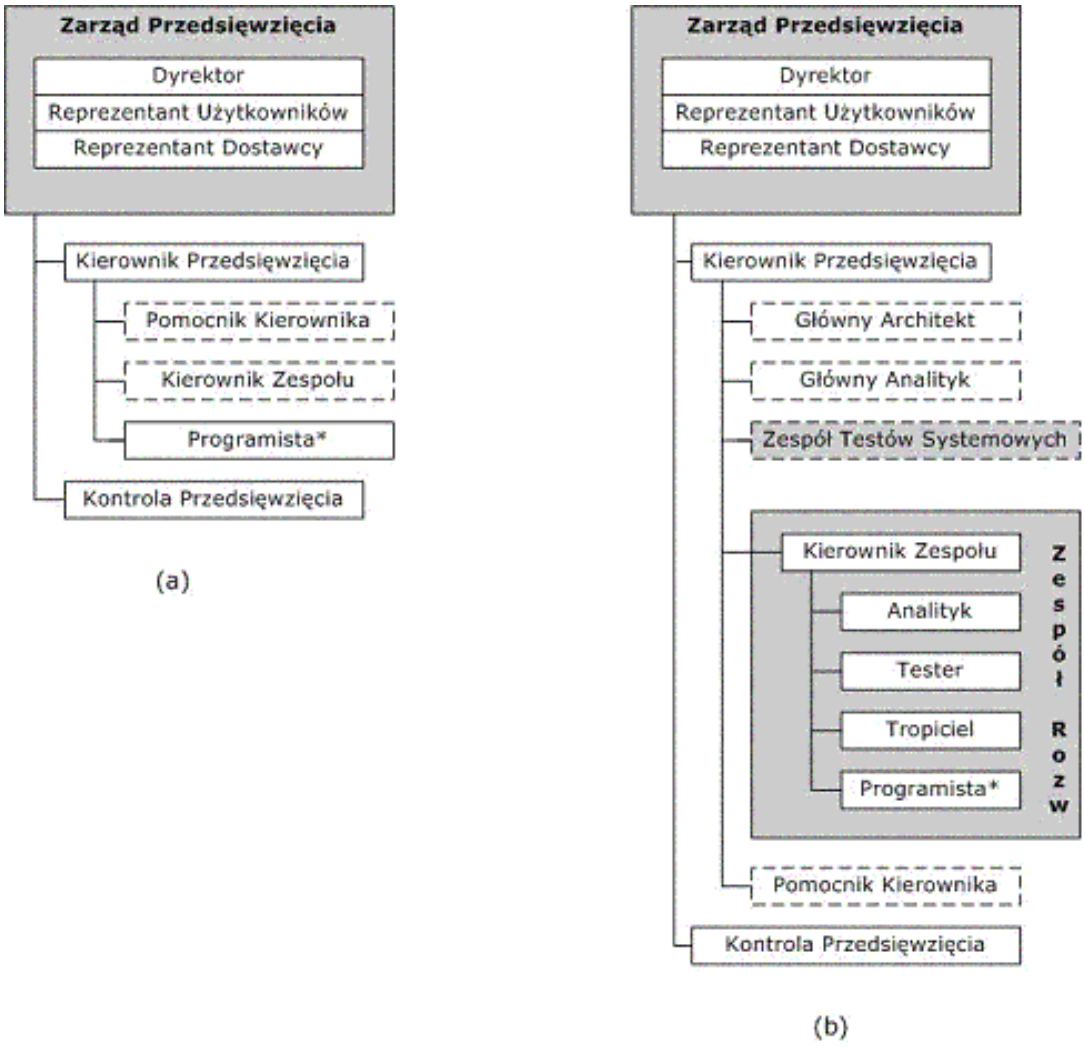
Zarządzanie przedsięwzięciami XP

Jak już wspomniano, PRINCE 2 doskonale współgra z systemami zarządzania jakością opartymi na ISO 9000. Warto jednak zauważyć, iż kłóci się to z jedną z elementarnych wartości Agile Manifesto, to jest preferowaniem jednostek oraz interakcji ponad procesy i narzędzia. Metody "ekstremalne" zostały stworzone, aby zarządzać lepiej przedsięwzięciami realizowanymi w "produkcyjnych" warunkach biznesowych, w których wymagania i potrzeby ciągle ulegają zmianie. Jeżeli system informatyczny ma poprawnie realizować założone funkcje, wówczas kluczem do jego sukcesu jest wystarczająco wczesne aktywne zaangażowanie użytkowników w przedsięwzięcie, zwłaszcza w zakresie testowania systemu i dostarczenia odpowiedniego sprzężenia zwrotnego dla

jego twórców. W momencie, gdy programista staje w obliczu zdefiniowanych procesów i ich ograniczeń, niezbędna okazuje się indywidualna kreatywność i grupowe "burze mózgów". Rozwiązania takie przynoszą zwinne metodyki, które koncentrują się na członkach zespołu i tworzeniu prostych, lecz użytecznych rozwiązań [Nawrocki2001]. Powstaje, zatem, pytanie, jak przystosować metodykę strukturalną tak, aby uczynić ją "zwinną". Ostatnie badania pokazują, iż lekkie podejścia do tworzenia oprogramowania oferują na tyle ogólną i uniwersalną perspektywę, iż połączenie ich z podejściem procesowym jest możliwe [Nawrocki2002a] [Paulk2001], pod warunkiem ich odpowiedniej adaptacji. Powstaje jednak problem, co i w jaki sposób należy dostosować, aby wypracowane rozwiązanie oferowało zalety obu tych podejść, minimalizując przy tym ich potencjalne niedoskonałości.

Struktura organizacyjna

PRINCE 2, jako ogólna metodyka zarządzania przedsięwzięciami, nie bierze pod uwagę specyficznych aspektów związanych z rozwojem produktów programistycznych. Koncentruje się wyłącznie na zagadnieniach związanych z prowadzeniem i kontrolą przedsięwzięcia. Struktura organizacyjna proponowana przez PRINCE 2 ogranicza się zatem wyłącznie do zespołu zarządzania przedsięwzięciem (rysunek 2a). Zarząd Przedsięwzięcia jest odpowiedzialny za ogólną kontrolę nad projektem i obejmuje trzy role: Dyrektora, reprezentującego klienta (inwestora), Reprezentanta Dostawcy, działającego z ramienia Zespołu Rozwojowego oraz Reprezentanta Użytkowników. Zarząd podejmuje strategiczne decyzje dotyczące przedsięwzięcia (np. w krytycznym momencie może zdecydować o zawieszeniu lub nawet przerwaniu projektu). Niestety, ze względu na niezbędne kompetencje członków Zarządu, role te pełnią zwykle osoby ze ścisłego kierownictwa organizacji, co często oznacza obciążenie wieloma innymi obowiązkami. Na co dzień, bezpośrednim zarządzaniem przedsięwzięciem zajmuje się Kierownik Przedsięwzięcia, który podlega bezpośrednio Zarządowi. Aby zagwarantować, iż raporty kontrolne przekazywane przez Kierownika Przedsięwzięcia nie są zbyt optymistyczne (lub zbyt pesymistyczne), Zarząd ma do dyspozycji zespól tzw. Kontroli Przedsięwzięcia, odpowiedzialny za weryfikację napływających danych. Wszystkie wymienione role są obowiązkowe. W przypadku naprawdę dużych przedsięwzięć konieczna może okazać się dystrybucja części obowiązków Kierownika na dedykowane podzespoły. Jednym z nich jest Pomocnik Kierownika, który przejmuje część zadań związanych z administracją przedsięwzięcia. Dodatkowo, jeżeli istnieje konieczność podziału zespołu rozwojowego na mniejsze grupy, konieczne będzie wyznaczenie stojących na ich czele Kierowników Zespołów.



a) Zespół zarządzania przedsięwzięciem w PRINCE2. b) Struktura przedsięwzięcia w Extreme PRINCE. Role oznaczone linią przerywaną są opcjonalne.

Koncepcję umieszczenia Programowania Ekstremalnego w obszarze zarządzania PRINCE 2 zademonstrowano na rysunku 2b. Jeżeli przedsięwzięcie jest niewielkie, wówczas można ograniczyć się wyłącznie do jednego zespołu rozwojowego. Wówczas role Dyrektora, Reprezentanta Użytkowników, Kontroli Przedsięwzięcia oraz Analityka mogą być przejęte przez jedną osobę, które z punktu widzenia XP jest "reprezentantem klienta." W konsekwencji "reprezentant klienta" przejmuje kontrolę i odpowiedzialność związaną z finansowaniem przedsięwzięcia, dostarczeniem wiedzy biznesowej i dziedzinowej, a w razie potrzeby również za wspomaganie i ciągłą dostępność dla Zespołu Rozwojowego. Jeżeli strategiczne decyzje biznesowe podejmuje ktoś inny, osoba ta automatycznie przejmuje rolę Dyrektora. Wówczas "reprezentant klienta" z punktu widzenia XP obejmuje role Reprezentanta Użytkowników oraz Analityka. Role Kierownika Przedsięwzięcia oraz Kierownika Zespołu mogą być pełnione przez tą samą osobę. Warto jednak zaznaczyć, iż Kierownik Zespołu musi być przygotowany do wykonywania funkcji "trenera" znanej z XP. W przypadku dużych przedsięwzięć uproszczona struktura organizacyjna nie spełni właściwie swojej funkcji. Rozwiązaniem problemu

jest zastosowanie reguły "dziel i rządź." Przykładowe rozwiązanie przedstawia rysunek 2b, uwzględniający dodatkowo konieczność powołania trzech zespołów: analityków, programistów oraz testerów. Niestety, jeżeli wszyscy analitycy pracują wspólnie istnieje groźba zaburzeń oraz zmniejszenia efektywności komunikacji z zespołem programistów (platformą współpracy mogłyby być wówczas udokumentowane wymagania [Nawrocki2002b]). Kolejnym problemem jest kwestia warunków pracy zespołu programistów. XP wymaga, aby wszyscy pracowali w jednym pomieszczeniu, co ze względu na liczbę osób może być trudne do realizacji, a w konsekwencji osłabi skuteczność komunikacji. Lepszym rozwiązaniem jest podział programistów na kilka mniejszych podzespołów, z których każdy zajmuje się wydzieloną, spójną częścią systemu. Wymaga to jednak doskonale zaplanowanego podziału systemu oraz zaprojektowania prostych, lecz użytecznych interfejsów między poszczególnymi komponentami. Zadanie to należy powierzyć doświadczonemu Głównemu Architektowi (który jest niczym Chirurg w koncepcji Zespołu Chirurgicznego zaproponowanego przez Brooksa [Brooks1995]). W ten sposób każdy zespół programistów będzie dysponował podzbiorem wymagań oraz definicjami interfejsów do części systemu tworzonych przez pozostałe grupy. Z kolei analitycy powinni być rozproszeni po zespołach programistycznych, w których mogą pełnić role "reprezentantów użytkownika" z XP. Jeden z nich - Główny Analityk - jest odpowiedzialny za ścisłe kontakty z użytkownikami końcowymi oraz dogłębną analizę domeny problemu. W tym ostatnim celu może również kontaktować się z zewnętrznymi ekspertami dziedzinowymi. Dodatkowo w przypadku dużych przedsięwzięć warto rozważyć utworzenie Zarządu Technicznego, składającego się z Głównego Architekta, Kierowników Projektu, Głównego Analityka oraz wszystkich analityków. Do zadań takiego zespołu należy utworzenie uzasadnienia biznesowego dla tworzonego systemu oraz ustalenie priorytetów dla każdego zespołu programistów. Dane te będą podstawą dla późniejszych działań analityków w pełniących na poziomie zespołów rozwojowych role "reprezentantów użytkownika." Spotkania Zarządu Technicznego powinny być stosunkowo częste (co najmniej raz w tygodniu), aby umożliwić Głównemu Architektowi oraz Głównemu Analitykowi kontrolę nad procesem projektowania całego systemu. Dzięki temu informacje o napotkanych "lokalnie" problemach mogą być błyskawiczne uwzględniane w pozostałych zespołach. Zespół Testów Systemowych odpowiada za przygotowanie i realizację zaawansowanych testów systemu, które z dużym prawdopodobieństwem nie zostaną przeprowadzone przez szeregowych użytkowników końcowych (np. testy bezpieczeństwa, wydajnościowe, odporności na ataki włamywaczy, obciążeniowe, etc.).

Dokumentacja

PRINCE 2 wprowadza do procesu zarządzania przedsięwzięciem około 30 typów dokumentów (i są to jedynie dokumenty służące celom zarządzania) [PRINCE2002]. Nasuwa się zatem wątpliwość, czy metodyka z tak dużymi "potrzebami" związanymi z dokumentacją może być "lekka" i "zwinna". Aby właściwie odpowiedzieć na to pytanie należy najpierw zastanowić się, kto odpowiada za tworzenie dokumentów i kto z nich później korzysta. Większość dokumentacji tworzona jest przez Kierownika Przedsię-

wzięcia, a następnie trafia ona do Zarządu Przedsięwzięcia. Oznacza to, iż, mimo że dokumenty te wpływają na członków zespołu rozwojowego nie stykają się oni z nimi bezpośrednio. PRINCE 2 wprowadza jedynie trzy dokumenty, które muszą być stworzone przez programistów:

- *Rejestr Spraw*, tworzony, kiedy wystąpi jakiś problem (np. błędy w działaniu zakupionych narzędzi),
- *Niezgodność ze Specyfikacją*, tworzony, kiedy dany fragment systemu nie działa zgodnie ze specyfikacją (np. podsystem stworzony przez inny zespół),
- *Propozycja Zmiany*, tworzony, kiedy należy zmienić specyfikację (wraz z odpowiadającą jej implementacją).

Należy podkreślić, iż to Zespół Zarządzania Przedsięwzięciem decyduje, czy wymagania oraz projekty systemu będą udokumentowane, czy też nie. PRINCE 2 niczego w tym względzie nie narzuca. Można zatem rozpocząć przedsięwzięcie mając do dyspozycji zaledwie jeden zespół rozwojowy i polegać wyłącznie na komunikacji ustnej. Takie podejście jest tym bardziej pożądane w sytuacji, kiedy organizacja pragnie zweryfikować ryzyko związane z przedsięwzięciem oraz uzyskać szybkie sprzężenie zwrotne ze strony użytkowników. Po pewnym czasie przedsięwzięcie z pewnością obejmie już kilka zespołów koordynowanych przez Głównego Architekta oraz Zespół Zarządzania Przedsięwzięciem. Wówczas będzie można ponownie rozważyć decyzję, czy wymagania oraz projekt powinny zostać udokumentowane.

Planowanie przedsięwzięcia

Metodyka PRINCE 2 wprowadza podział przedsięwzięcia na kilka etapów. Z drugiej strony, w Programowaniu Ekstremalnym mamy do czynienia z wydaniami, które dzielą się na przyrosty, zaś horyzont planowania jest ograniczony wyłącznie do jednego wydania. Proponujemy więc, aby każdy etap PRINCE 2 odpowiadał wydaniu XP. Jednakże główny problem polega na tym, że PRINCE 2 wymaga stworzenia planu całego przedsięwzięcia już na jego początku. Dodatkowo, proces planowania powinien również obejmować (między innymi) definicje produktów, które mają zostać stworzone, identyfikację zadań do realizacji oraz zależności między nimi, oszacowanie pracochłonności oraz stworzenie harmonogramu. Podejście takie wymaga planowania, które znacząco wykracza poza zakres jednego wydania (etapu) i doskonale nadaje się do przedsięwzięć o z góry ustalonym zakresie i podanych wymaganiach. Jednakże nawet, jeżeli wymagania znane są już na początku przedsięwzięcia, rozsądnie jest tworzyć oprogramowanie zgodnie z modelem przyrostowym proponowanym przez lekkie metodyki. Wstępna specyfikacja wymagań może zawierać błędy lub niejednoznaczności, które mogą być trudne do odkrycia podczas prostego czytania dokumentu. Dodatkowe ryzyko związane jest z zastosowaną technologią, wydajnością wykorzystywanych komponentów zewnętrznych, itp. Dlatego też sugerujemy połączenia podejścia PRINCE 2 z metodyką XP również w tym obszarze. Tym samym proces planowania przedstawiałby się następująco:

- Ustal, co jest bardziej istotne w rozpatrywanym przedsięwzięciu: czas, czy pieniądze? Załóżmy, że interesuje nas czas.

- Skorzystaj z danych historycznych dotyczących poprzednio zrealizowanych przedsięwzięć. Zidentyfikuj te, które są podobne do obecnego, a następnie ustal wartość współczynnika R. Na przykład, posiadając dane trzech podobnych przedsięwzięć z rzeczywistym oraz planowanym czasem realizacji równym odpowiednio: 24/20, 20/18 oraz 20/16 można wybrać wariant zachowawczy i ustalić wartość R=20/16. W celu dokładniejszego wyznaczenia współczynnika R można posłużyć się również metodami statystycznymi.
- Podczas rozmów z klientem, podziel wymagania na dwie części: istotne oraz opcjonalne. Im mniejszy jest pierwszy zbiór wymagań, tym większe prawdopodobieństwo stworzenia użytecznego produktu, a w konsekwencji sukcesu przedsięwzięcia.
- Oszacuj czas potrzebny do stworzenia zasadniczej części systemu oraz wszystkich zebranych funkcji (na tym etapie można skorzystać z metody Wide-band Delhi, COCOMO II, analizy statystycznej danych historycznych, itp.). Oznaczmy je odpowiednio, jako Z (część zasadnicza) oraz C (całość). Czas dostarczenia systemu, D, jest realistyczny, jeżeli zarówno C < D, jak i Z*R < D. Wówczas istnieje wysokie prawdopodobieństwo, że przynajmniej zasadnicza część systemu zostanie ukończona w założonym terminie.

Jeśli Z*R jest znacząco mniejsze od D, wówczas jesteśmy na dobrej pozycji. Rozpatrywane przedsięwzięcie jest elastyczne i mamy możliwość "naginania" jego ram czasowych tak, aby maksymalizować użyteczność. W takim przypadku przedsięwzięcie należy rozpocząć od stworzenia bardzo ogólnej architektury systemu oraz identyfikacji jego systemów składowych. Następnie należy oszacować priorytety podsystemów oraz określić minimalny zestaw funkcji końcowego produktu (te zagadnienia należy dokładnie przedyskutować z klientem oraz użytkownikami końcowymi). Teraz można uszeregować podsystemy od najważniejszych (obejmujących zasadnicze funkcje) do najmniej istotnych. Otrzymana w ten sposób lista jest dobrym punktem wyjścia do planowania zgodnie z wymaganiami PRINCE 2. Należy jednak pamiętać, aby na końcu każdego etapu odpowiednio modyfikować zawartość i kolejność elementów tej listy.

Wstępne wyniki

Od czterech lat w ramach Studium Rozwoju Oprogramowania prowadzone są na Politechnice Poznańskiej badania Programowania Ekstremalnego w kontekście metod i standardów postrzeganych w inżynierii oprogramowania jako zbyt "ciężkie". Każdy z realizowanych w ramach Studium projektów angażuje ośmiu studentów specjalności Inżynieria Oprogramowania citeNawrocki1998. Przedsięwzięcia, mimo że realizowane w warunkach akademickich, mają na celu wytworzenie oprogramowania dla zewnętrznych organizacji, a tym samym przypominają warunki panujące w niewielkich firmach programistycznych. Rezultaty przeprowadzonych dotąd badań [Nawrocki2002a] [Nawrocki2002b] [Nawrocki2002c] wskazują, iż adaptacja zwinnych procesów do wymagań istniejących modeli dojrzałości (CMM Poziom 2) oraz systemów zarządzania jakością (ISO 9000) jest możliwa, zaś jej rezultatem jest większa elastyczność tak zmodyfikowanych metodyk.

Dotychczasowe eksperymenty pokazały, iż mimo zalet Programowania Ekstremalnego przedsięwzięcia zgodne z tą metodyką rozwijane przez studentów trzeciego, zaś zarządzane i kontrolowane przez studentów czwartego i piątego roku borykały się z problemami opisanymi w pracy [Nawrocki2002c]. W tym roku, nasze badania były skoncentrowane na zdefiniowaniu procesów Studium tak, aby były one zgodne z wymaganiami stawianymi przez standard ISO 9001:2000. Podstawą do tych działań była analiza dotychczasowych przedsięwzięć. Analiza ta wykazała, iż niewątpliwe zalety Programowania Ekstremalnego, były odczuwalne przede wszystkim przez programistów, podczas gdy klienci musieli liczyć z pewnymi niedogodnościami. XP "uwolniło" twórców aplikacji od "budowania" procesu rozwoju oprogramowania (m. in. konieczność tworzenia obszernej dokumentacji procesowej), na rzecz aktywnego w nim uczestnictwa (prace koncepcyjne i programistyczne). Z drugiej strony, metodyka ta wymuszała na klientach częste spotkania z zespołem rozwojowym, co nie było zbyt korzystne ani dla osób podejmujących decyzje biznesowe, ani dla przedstawicieli użytkowników. Osoby te - kluczowe z punktu widzenia metodyki Programowania Ekstremalnego, są zarazem najbardziej zajęte, a tym samym mają trudności w przeznaczeniu odpowiedniej ilości czasu na spotkania z programistami XP. W konsekwencji, problemy związane z podejmowaniem na czas istotnych decyzji oraz weryfikacją wykonanych zadań bardzo utrudniały panowanie nad procesem rozwoju oprogramowania. Ponadto, część przedstawicieli organizacji, dla których powstawało oprogramowanie w ramach SRO sygnalizowała konieczność poprawy procesu zarządzania przedsięwzięciami.

Dlatego w tym roku naszym dodatkowym celem było rozwiązanie zidentyfikowanych problemów poprzez zastosowanie metodyki zarządzania przedsięwzięciem wykorzystującej PRINCE 2. Zaprezentowane podejście wyraźnie rozdziela obszar zarządzania przedsięwzięciem od rozwoju produktu, dzięki czemu zachowanie lekkości samego procesu tworzenia oprogramowania jest możliwe. Z drugiej strony, zastosowanie metodyki PRINCE 2 pozwala połączyć potrzeby biznesowe klienta z komfortem pracy zespołu rozwijającego oprogramowanie tak, aby możliwa była systematyczna kontrola przedsięwzięcia i aby zakończyło się ono sukcesem. Dodatkową zaletą strukturalnego podejścia do zarządzania, jest ułatwiony transfer wiedzy dotyczącej realizacji przedsięwzięcia od zespołu wykonawczego do klienta i użytkowników końcowych. Wreszcie, dzięki aktywnemu uczestnictwu wszystkich zainteresowanych stron, organizacja jest w stanie lepiej zarządzać swoimi oczekiwaniami i kontrolować ich realizację.

Dotychczasowy przebieg eksperymentu wskazuje, że objęcie rozwoju oprogramowania zgodnego z lekkim podejściem, zarządzaniem przedsięwzięciem w oparciu o wytyczne PRINCE 2 jest nie tylko możliwe, ale także przynosi wymierne korzyści. Badanie przeprowadzone wśród studentów odpowiedzialnych za zarządzanie przedsięwzięciem oraz zapewnianie jakości po sześciu miesiącach od wprowadzenia nowej metodyki wykazała ponad 74% poparcie dla połączenia PRINCE 2 i XP. Warto ponadto zauważyć, iż, pomimo, że przez większość osób objętość dokumentacji PRINCE 2 jest postrzegana jako "dość duża" (81%), to taka sama liczba badanych stwierdziła, że obciążenie programistów pozostało na takim samym poziomie. Co istotne, podobną opinię, ale dotyczącą klienta wyraziła jedna trzecia badanych, podczas, gdy prawie 52% stwierdziło, iż wprowadzenie PRINCE 2 zwiększyło obciążenie klienta "nieznacznie". Wyniki te

wskazują, iż większość pracy związanej z zarządzaniem przedsięwzięciem zgodnie z PRINCE 2 spadła przede wszystkim na Kierowników oraz zespół Kontroli Przedsięwzięcia, w niewielkim tylko stopniu zwiększając obciążenie klientów. Duża zgodność w ocenie bezpośredniego wpływu metody PRINCE 2 na lekkość procesów tworzenia oprogramowania pozwala domniemywać, iż zastosowane podejście w udany sposób łączy zalety zwinnych metodyk i strukturalnego podejścia do zarządzania przedsięwzięciami. Niezbędne są jednak dalsze badania, pozwalające przede wszystkim zweryfikować uzyskane wyniki z opiniami wyrażanymi przez przedstawicieli klienta oraz członków zespołów programistycznych. Konieczna jest również empiryczna weryfikacja skalowalności zaproponowanego podejścia, w przypadku przedsięwzięć o rosnącym stopniu złożoności.

Podsumowanie

Zwinne metodyki sięgające swymi korzeniami do Agile Manifesto oferują proste rozwiązania problemów związanych z "ciężkimi" procesami. Zwinność, rozumiana jako zdolność do szybkiej reakcji na zmieniające się wymagania uważana jest za synonim "lekkości". Naszym zdaniem cechy te nie są ze sobą sprzeczne - zwinność metodyki rozwoju oprogramowania może być zachowana pomimo korzystania z dokumentacji na poziomie zarządzania przedsięwzięciem informatycznym. Najwyższą wartością jest zdolność procesów do adaptacji, nie zaś ich "lekkość". Zwinne procesy mogą w pewnych sytuacjach okazać się "odporne" na zmianę - nie tyle w wymaganiach, co w zbiorze praktyk, do których się odnoszą. Dlatego potrzeba zmian powinna być odzwierciedlona także w samym procesie. W zaproponowanym podejściu PRINCE 2 uzupełnia XP, demonstrując, że adaptacja istniejących procesów staje się zyskowna. Ponadto pierwsze wyniki pochodzące badań w Studium Rozwoju Oprogramowania są obiecujące.

Bibliografia

[AM2003] *Manifesto for Agile Software Development*, Maj 2003, http://www.agilemanifesto.org/.

[Beck2000] K. Beck, *Extreme Programming Explained. Embrace change*, 2000, Addison-Wesley, Boston.

[Beck2001] K. Beck, *Planning Extreme Programming.*, 2001, Addison-Wesley, Boston.

[Brooks1995] F. P. Brooks, *The Mythical Man-Month: Essays on Software Engineering.*, 1995, Addison-Wesley.

[CMMI2003] CMMI? Web Site. http://www.sei.cmu.edu/cmmi/.

[Cockburn2002] A. Cockburn, *Agile Software Development.*, 2002, Addison-Wesley.

[CRM2003] *Centrum Rozwiązań Menadżerskich S. A.*, Maj 2003,

http://www.crm.com.pl/prince2_geneza.htm.

[Cutter2001] *projectmanagement.com*, grudzień 2001, http://www.projectmanagement.com/article/1,2462, 79285,00.html.

[ISO9001-2000] European , *Quality Management Systems - Requirements (ISO 9001:2000).*, 2000.

[Humphrey1995] *US DODI 5000.2*, 2000 kwiecień, Final Coordination Draft.

[Humphrey2000] W. Humphrey, *A Discipline for Software Engineering.*, 1995, Addison-Wesley, Massachusetts.

[Jeffries2001] R. Jeffries, A. Anderson i C. Hendrckson, *Extreme Programming Installed.*, 2001, Addison-Wesley, Boston.

[PMBOK1996] W. R. Duncan, *A Guide to the Project Management Body of Knowledge*, 1996, Project Management Institute, URL: http://www.pmi.org.

[PRINCE2002] *Managing Successful Projects with Prince 2.*, 2002, The Stationery Office Ltd., CCTA, Norwich.

[Nawrocki2002a] J. Nawrocki, M. Jasiński, B. Walter i A. Wojciechowski, *Combining Extreme Programming with ISO 9000.*, 786--794, October 2002, 1st EurAsian Conference on Advances in Information and Communication Technology, EurAsia-ICT 2002, Shiraz (Iran), A Min Tjoa (Ed.), Lecture Notes in Computer Science 2510.

[Nawrocki2002b] J. , *Extreme Programming Modified: Embrace Requirements Engineering Practices.*, 303-310, October 2002, Proceedings of the 10th IEEE Joint International Requirements Engineering Conference, IEEE Press, Inc., Los Alamitos.

[Nawrocki2002c] J. Nawrocki, M. Jasiński, B. Walter i A. Wojciechowski, *Comparison of CMM Level 2 and eXtreme Programming.*, 288--297, 2002, Software Quality - ECSQ 2002, Helsinki (Finland), Lecture Notes in Computer Science 2349, Springer Verlag}.

[Nawrocki2001] J. Nawrocki, A. Wojciechowski i K. , *Experimental Evaluation of Pair Programming., Project Control: Satisfying the Customer*, 269--276, 2001, Shaker Publishing, London.

[Paulk1994] M. C. Paulk, *The Capability Maturity Model: Guidelines for Improving the Software Process.*, 1994, Addison-Wesley.

[Paulk2001] M. C. Paulk, *Extreme Programming from a CMM perspective.*, 19--26, November/December 2001, IEEE Software.

[Schwaber2001] K. Schwaber i M. A. Beedle, *Agile Software Development with Scrum.*, 2001, Prentice Hall.