Symulacja w Badaniach i Rozwoju
Vol. 4, No. 1/2013

Tomasz PROTASOWICKI, Jerzy STANIK
Wojskowa Akademia Techniczna, 00-908 Warszawa, ul. Kaliskiego 2
E-mail: tomasz.protasowicki@wat.edu.pl, jstanik@wat.edu.pl

Model jakości oprogramowania wytwarzanego w zwinnym procesie produkcji

1 Wprowadzenie

Współczesne projekty informatyczne w coraz większej części prowadzone są w oparciu o metodyki zwinne. Ich zastosowanie ma z definicji zapewnić szybkie dostarczanie klientowi kolejnych wersji funkcjonującego oprogramowania charakteryzującego się określoną jakością. Jakość oprogramowania stanowi zagadnienie wieloaspektowe, związane z tym, że twórcy oprogramowania muszą sprostać jawnym i ukrytym potrzebom klientów, oczekiwaniom komitetu sterującego, partnerów i audytorów. Ponadto, produkt programowy musi być konkurencyjny i przede wszystkim opłacalny, stworzony w terminie i z uwzględnieniem ograniczeń budżetowych. Jakość można zatem zdefiniować jako stopień spełnienia przez produkt, obiekt lub system stwierdzonych i przewidywanych oczekiwań klientów i innych partnerów. Wysokiej jakości produkt programowy powinien być:

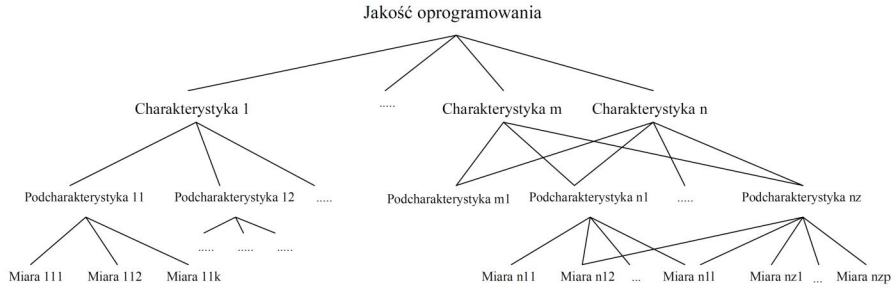
- funkcjonalny,
- bezpieczny,
- niezawodny,
- wydajny.

Jednak jakość jest pojęciem subiektywnym. W związku z tym różne osoby mogą przyjmować odmienne kryteria jej oceny, a jednocześnie kryteria te mogą ewoluować w czasie, co dodatkowo utrudnia osiągnięcie ich dostatecznego poziomu. W związku z powyższym od dawna wskazywano potrzebę stworzenia modeli jakości jak najbardziej abstrahujących od subiektywnej oceny użytkowników. W ramach pracy, której wyniki przedstawiono w niniejszym artykule, autorzy zaprezentowali podejście do zagadnienia jakości oprogramowania na tle normy ISO 9126, którą scharakteryzowali w punkcie 2 niniejszego artykułu. Następnie autorzy omówili model jakości oprogramowania wytwarzanego w klasycznym procesie produkcji i przeciwstawili mu podejście do zagadnienia jakości oprogramowania powstającego w procesie zwinnym. Podjęli próbę zbadania wpływu odpowiedniego doboru personelu do zespołów wytwarzających oprogramowanie w oparciu o podejście zwinne na jakość powstającego produktu. W tym celu opracowali model symulacyjny, który obejmuje swoim zakresem obszary: zarządzania przepływem pracy, detekcją i naprawą błędów oraz budową zespołów w projekcie zarządzanym według metodyki Scrum. Zbudowany model, czerpiąc z osiągnięć nauk humanistycznych i inżynierii oprogramowania, pozwala między innymi badać, jak dopasowanie cech osobowości poszczególnych członków zespołu do różnych ról i procesów wpływa na jakość produktu w którego tworzenie są oni zaangażowani. Przeprowadzone przy jego użyciu eksperymenty symulacyjne dają ciekawe spojrzenie na aspekty dotyczące zarządzania zasobami ludzkimi i ich wpływu na przebieg projektu oraz jakość otrzymanego w rezultacie

oprogramowania. Wykorzystanie zaproponowanego modelu w praktyce może przyczynić się do lepszego zarządzania projektami prowadzonymi w zgodzie z filozofią Agile poprzez dobór kadry charakteryzującej się odpowiednimi zdolnościami zarówno technicznymi, jak i interpersonalnymi.

2 Standardy jakości

Próby stworzenia modelu tzw. jakości technicznej trwają od drugiej połowy lat 70-tych ubiegłego stulecia. Wszystkie z dotychczas zaproponowanych modeli jakości technicznej mają zbliżoną budowę. Typowy schemat modelu jakości technicznej przedstawiony został na poniższym rysunku (rys. 1).



Rys. 1. Ogólna postać modeli jakości technicznej (Źródło: A. Kobyliński; „ISO/IEC 9126 – Analiza modelu jakości produktów programowych", s. 2)

Fig. 1. The general form of technical quality models

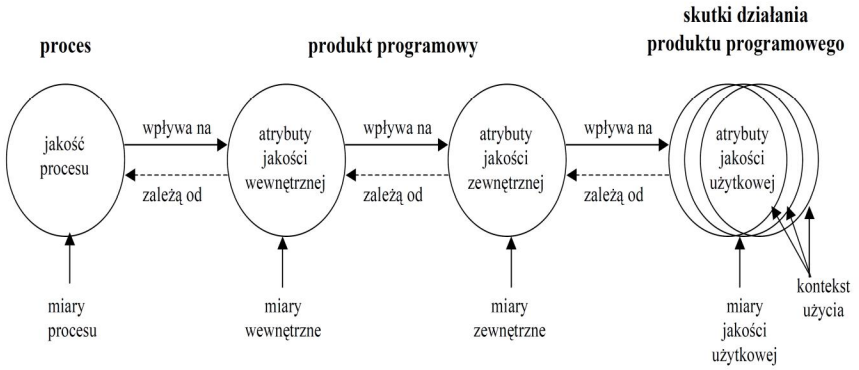
Schematy poszczególnych modeli różnią się nieznacznie. Wszystkie wyróżniają charakterystyki i pod charakterystyki, jak również sugerują (chociaż nie zawsze wprost ujawniają) istnienie miar określających poziom cech i/lub właściwości. Istnieje kilka standardów jakości (np. ISO 9000, CMM, CMMI, itp.). Normą istotną dla jakości oprogramowania jest norma ISO/IEC 9126. Norma ta określa standardy opisu wymagań dla oprogramowania. Jej najnowsza wersja złożona jest z czterech części:

- model jakości (norma ISO/IEC 9126-1:2001) opisuje charakterystyki i atrybuty służące do definiowania wymagań niefunkcjonalnych; czynniki te mogą też być wykorzystywane do weryfikacji oczekiwań użytkownika;
- miary zewnętrzne (norma ISO/IEC 9126-2:2003) – zestaw metryk zewnętrznych służących do pomiaru charakterystyk określających zachowanie się systemu; miary mogą być dedykowane różnym fazom życia oprogramowania w wersji wykonywalnej, np. fazie testowania lub później implementacji;
- miary wewnętrzne (norma ISO/IEC 9126-3:2003) – zewnętrzne metryki jakości służące do pomiaru charakterystyk oprogramowania będącego w postaci niewykonywanej; może być to na przykład faza projektowana lub wczesnej implementacji;
- miary jakości użytkowej (norma ISO/IEC 9126-4:2004) – zbiór metryk jakości, który może zostać użyty do wygenerowania raportu technicznego zawierającego rozszerzalną listę atrybutów.

Standard ISO 9126 wspomaga definiowanie wymagań jakościowych oprogramowania, podając charakterystyki i atrybuty cech, na które twórcy oprogramowania powinni zwrócić uwagę. Cechy oprogramowania definiowane przez standard z powodzeniem mogą posłużyć do oceny jakości tworzonego oprogramowania. Ponadto model jest uniwersalny i daje się zastosować do dowolnego typu oprogramowania i może być użyty do [3]:

- walidacji kompletności definicji wymagań,
- identyfikacji wymagań oprogramowania,
- identyfikacji celów projektu oprogramowania,
- identyfikacji celów implementacji oprogramowania,
- identyfikacji kryteriów zapewniania jakości,
- identyfikacji kryteriów akceptacji produktu przez użytkownika po zakończeniu prac nad produktem.

Norma ISO/IEC 9126 jest zgodna z wieloma normami jakości, niezawodności, bezpieczeństwa, ergonomii, cyklu życia oprogramowania itp. W normie stwierdza się, że dobry jakościowo produkt programowy jest efektem realizacji dobrych jakościowo procesów wytwórczych, a jakość produktu ma wpływ na jakość użytkową oprogramowania. Jakość tego produktu mierzy się zgodnie z normą ISO/IEC 9126, natomiast jakość procesów wytwórczych można oceniać za pomocą poziomu CMM/CMMI lub ISO/IEC15504. Wobec tego można stwierdzić, że ocena i poprawa procesu jest środkiem do poprawy jakości produktu, a ocena i poprawa jakości produktu jest środkiem do poprawy jakości użytkowej/eksploatacyjnej. Sytuację tę przedstawiono na poniższym rysunku (rys. 2).



Rys. 2. Jakość w cyklu życia produktu programowego (Źródło: „ISO/IEC FDIS 9126-1:2000(E)" s. 3)

Fig. 2. The quality of the software product

3 Charakterystyka zarządzania jakością oprogramowania wytwarzanego w tradycyjnym procesie produkcji

Tradycyjne metodyki zarządzania projektami informatycznymi powstawały w różnych miejscach i ośrodkach, w związku z czym mają różne korzenie [7]. Można je pogrupować następująco:

- metodyki ogólne,
- metodyki firmowe,
- metodyki specjalne.

Metodyki ogólne, uniwersalne, to takie, które znajdują zastosowanie w wielu obszarach aktywności, nie tylko dla projektów informatycznych. Do nich zalicza się przede wszystkim:

- amerykańską metodykę PMI (ang. Project Management Institute),
- brytyjską - APM (ang. Association for Project Management),
- japońską - P2M (ang. Project & Program Management for Enterprise Innovation),
- podejście Kepner Tregoe'a.

Metodyki firmowe, to takie, które powstały w laboratoriach poszczególnych firm informatycznych i zostały upublicznione a nawet przyjęte jako standardy. Są one ukierunkowane na zarządzanie projektami informatycznymi realizowanymi dla konkretnego klienta w pełnym cyklu wytwarzania (przy wykorzystaniu różnych modeli cyklów życia: od kaskadowego do iteracyjnego) i wdrożenia, ze szczególnym uwzględnieniem dużych projektów. Zaliczamy do nich m.in.:

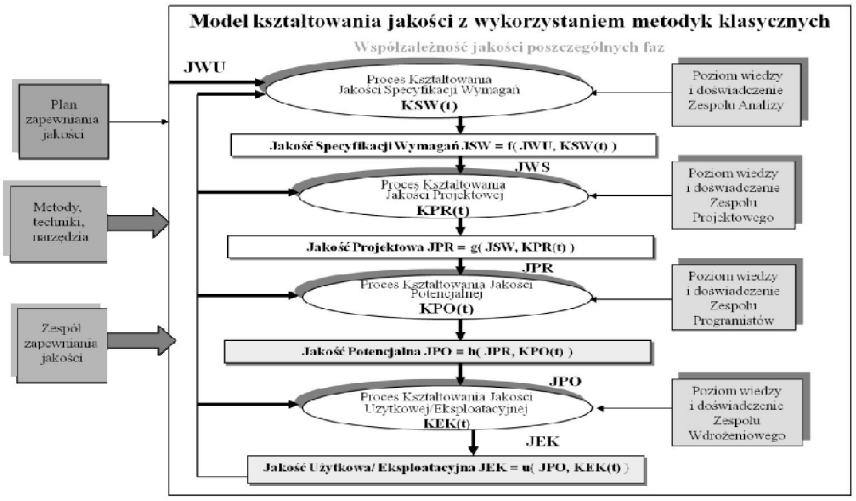
- metodyki firmy IBM po nazwą PMM (ang. Project Management Methodology),
- brytyjską metodykę PRINCE2 (ang. PRojects IN Controlled Environments) pierwotnie opracowaną dla Central Computer and Telecommunication Agency (CCTA)
- oraz RUP (ang. Rational Unified Process) metodykę opracowaną przez firmę Rational Software (obecnie jest to dział IBM).

Metodyki specjalne, które są ukierunkowane na specyficzne typy projektów informatycznych (zarówno produkcji, jak i wdrożenia oprogramowania). Są to m.in.:

- MSF (ang. Microsoft Solution Framework) firmy Microsoft dla realizacji projektów wytwarzania oprogramowania
- oraz ASAP (ang. Accelerated SAP) firmy SAP dla projektów wdrożenia systemów klasy ERP (ang. Enterprise Resource Planning).

Metodyki tradycyjne charakteryzują się dużą restrykcyjnością, wysokim poziomem formalizmu i dbania o kontrolę procesów. Przewidują ścisłą kontrolę większości procesów, zarówno planistycznych, jak i realizacyjnych czy rozliczeniowych. Metodyki ukierunkowane są zatem na kontrolowanie (a w zasadzie restrykcyjne zarządzanie) zakresu, czasu, zasobów, ludzi, budżetu, zmian, komunikacji, jakości i ryzyka. Metodyki klasyczne opisują i formalizują procesy oraz procedury ich realizacji na wszystkich etapach projektu. Podobnie definiują struktury organizacyjne, tj. sposoby zorganizowania pracy ludziom - członkom zespołów realizujących projekty. Są to w większości wielopoziomowe, zhierarchizowane struktury organizacyjne, składające się z Komitetu Sterującego, Kierownika Projektu, Zespołów Wykonawczych, uwzględniające takie specjalne komórki, jak Zespół Zapewnienia Jakości czy Biuro Projektu.

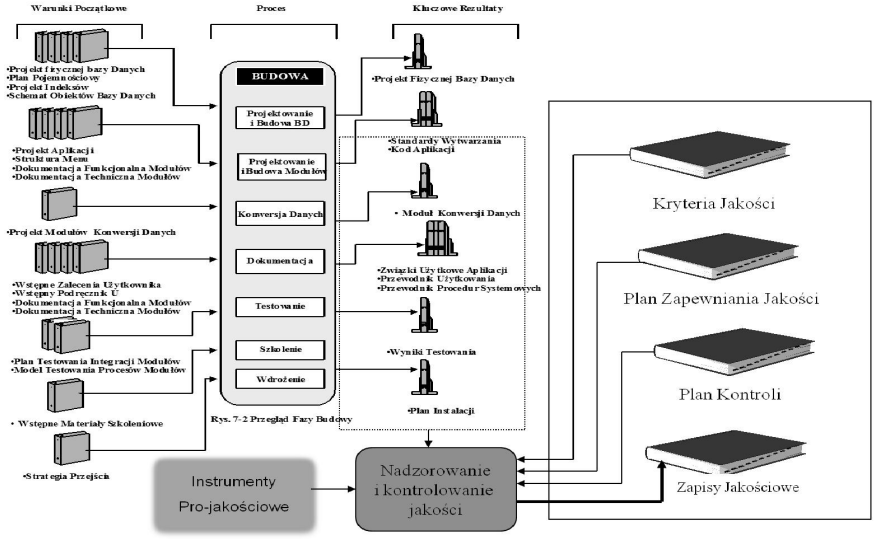
Model kształtowania jakości z oprogramowania wytwarzanego w tradycyjnym procesie produkcji przedstawiono na poniższym rysunku (rys. 3)



Rys. 3. Model kształtowania jakości z wykorzystaniem metodyk klasycznych

Fig. 3. Model of the quality assurance for software product developed with use of the traditional approach

Przykład zapewniania jakości na etapie budowy oprogramowania zgodnie ze strukturalną metodyką wytwarzania przedstawiono na kolejnym rysunku (rys. 4).



Rys. 4. Przykład zapewniania jakości na etapie budowy oprogramowania

Fig. 4. Example of quality assurance process applied during software development

4 Metody i zasady zapewniania jakości oprogramowania wytwarzanego w zwinnym procesie produkcji

Współczesne projekty informatyczne w coraz większej części prowadzone są w oparciu o metodyki zwinne. Ich zastosowanie ma z definicji zapewnić szybkie dostarczanie klientowi kolejnych wersji funkcjonującego oprogramowania charakteryzującego się określoną jakością. Zwinny proces produkcji oprogramowania stanowi zatem odpowiedź na:

- zbyt restrykcyjne procesy,
- „niepotrzebne" tracenie sił i środków,
- ograniczenie elastyczności w działaniach
- oraz zbyt długotrwałe oczekiwanie na rezultat projektu przy zmieniających się wymaganiach.

Zwinny proces produkcji oprogramowania oparty jest na zdyscyplinowanym zarządzaniu projektem, które zakłada częste inspekcje wymagań i rozwiązań wraz z procesami adaptacji (zarówno specyfikacji, jak i oprogramowania). Kolejne etapy wytwarzania oprogramowania zamknięte są w iteracjach, w których za każdym razem przeprowadza się testowanie wytworzonego kodu, zebranie wymagań, planowanie rozwiązań itd. Skład zespołów jest zazwyczaj wielofunkcyjny oraz samozarządzalny, bez zastosowania jakiejkolwiek hierarchii korporacyjnej. Członkowie zespołu biorą odpowiedzialność za zadania postawione w każdej iteracji i sami decydują, jak osiągnąć postawione cele. Metoda nastawiona jest na szybkie wytwarzanie oprogramowania wysokiej jakości i bezpośrednią komunikację pomiędzy członkami zespołu, minimalizującą potrzebę tworzenia dokumentacji. Jeśli członkowie zespołu są w różnych lokalizacjach, to planuje się codzienne kontakty za pośrednictwem dostępnych kanałów komunikacji (wideokonferencja, e-mail itp.). Zwinność w procesach wytwarzania oprogramowania związana jest z:

- chęcią zmiany podejścia do procesu wytwórczego,
- odrzuceniem pewnych elementów formalizmu metodyk tradycyjnych,
- chęcią wykorzystania najlepszych praktyk pracy zespołowej, zwiększających jego efektywność i jakość rezultatu.

Ponadto zwinność w procesach wytwarzania oprogramowania:

- dynamizuje proces wytwarzania oprogramowania,
- wykorzystuje efekt synergii wynikający ze ścisłej współpracy z klientem w trakcie całego projektu,
- umożliwia dynamiczne sterowanie przyrostowym wytwarzaniem oprogramowania przy pomocy zmieniających się wymagań klienta.

Jakość oprogramowania jest jednym z fundamentów podejścia zwinnego. Modele jakości, specjalne techniki tworzenia kodu, wielopoziomowe testowanie oraz standaryzacja rozwiązań to tylko niektóre techniki jej osiągnięcia. Zaangażowanie zdopingowanego zespołu, poczucie własności i równocześnie współdzielenie kodu jest dodatkowym czynnikiem wzrostu jego jakości, w tym i podatności na modyfikacje. Istotnym elementem podejścia zwinnego, silnie odróżniającego go od tradycyjnych metodyk realizacji projektów, jest sposób traktowania zespołu i organizacji jego pracy. Przyjęte zasady stałej współpracy wszystkich osób zaangażowanych w projekt oraz

zwiększenia swobody pracy zespołu programistów całkowicie zmieniają strukturę organizacyjną, techniki pracy i zarządzania projektem. Zasady współpracy osób w czasie projektu zawierają bowiem postulaty:

- samodzielnej organizacji zespołu,
- współodpowiedzialności całego zespołu za rezultat pracy,
- silnego zmotywowania (samomotywowania) członków,
- zaspokojenia potrzeb członków zespołu,
- obdarzenia zespołu dużym zaufaniem.

5 Model symulacyjny badania jakości oprogramowania wytwarzanego w zwinnym procesie produkcji

Proces tworzenia modelu symulacyjnego obejmował następujące etapy:

- określenie symulowanego systemu,
- sformułowanie modelu,
- przygotowanie danych,
- zaprogramowanie modelu,
- ocena adekwatności modelu,
- planowanie eksperymentów symulacyjnych,
- wprowadzenie eksperymentów,
- interpretacja uzyskanych wyników.

Autorzy zbudowali model symulacyjny, który stanowi próbę całościowego podejścia do symulacji procesów wytwórczych w projektach prowadzonych w oparciu o metodykę Scrum [5,11]. Stanowi on własną propozycję autorów i uwzględnia, oprócz elementów skupionych wokół aspektów związanych przebiegiem zwinnych procesów produkcyjnych [10], również:

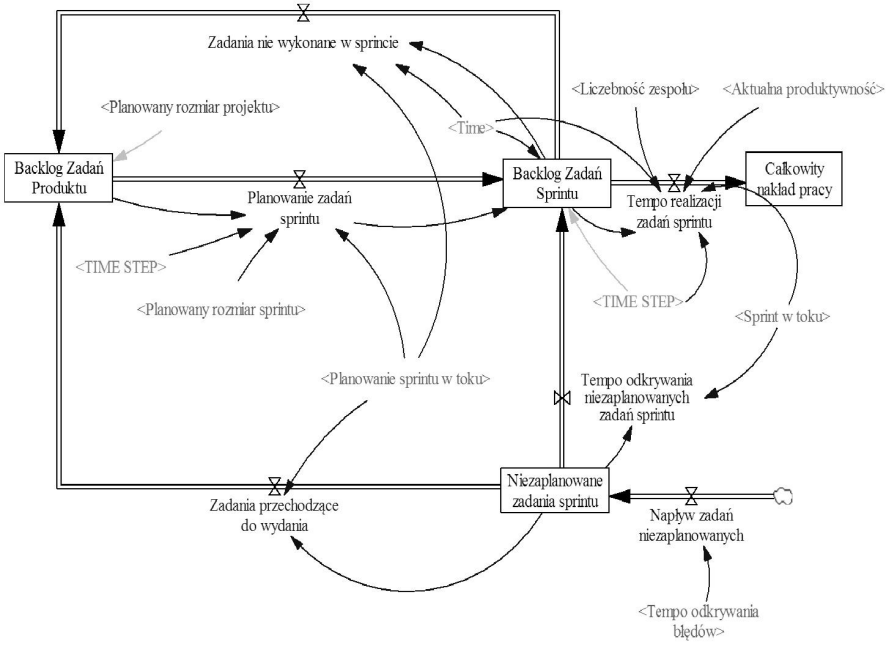
- aspekt jakości tworzonego oprogramowania,
- aspekt jakości zespołu wytwórczego.
- aspekt jakości podstawowych parametrów projektu prowadzonego zgodnie z zaleceniami metodyki zwinnej.

5.1 Sformułowanie modelu

Model symulacyjny zbudowany został na bazie założeń metodyki Scrum, AgileEVM oraz teorii małych grup (*ang. small group theory*) z wykorzystaniem środowiska symulacyjnego Vensim® [12]. Bazuje na modelu matematycznym zapisanym w postaci programu komputerowego. W skład oprogramowania wchodzą moduły i pakiety umożliwiające odtwarzanie w przestrzeni wirtualnej następujących działań:

- przebieg procesu Scrum;
- planowanie i przepływ sprintu;
- analiza i ocena jakości produktu programowego,
- badanie wybranych parametrów (charakterystyk) zespołu wytwórczego;
- badanie i ocena procesu wytwórczego oprogramowania według ustalonych i/lub zdefiniowanych metryk,
- AgileEVM (wartość wypracowana);
- zarządzanie czasem.

W niniejszej pracy badanie symulacyjne jakości oprogramowania sprowadza się do odtwarzania przebiegu procesu produkcji oprogramowania w oparciu o metodykę zwinną Scrum (patrz rys. 5). Równocześnie w procesie symulacji są zbierane różnego typu charakterystyki zespołu wytwórczego (patrz rys. 6) oraz jakości procesu wytwórczego (patrz rys. 7) poprzez dodatkowe wykorzystanie odpowiednich pakietów środowiska symulacyjnego Vensim® [6].

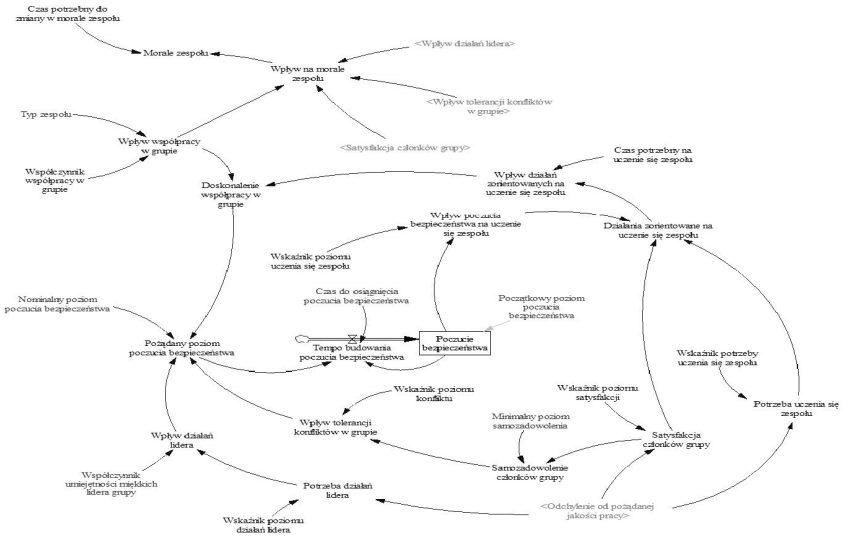


Rys. 5. Symulacyjny model przebiegu procesu produkcji oprogramowania w podejściu Scrum

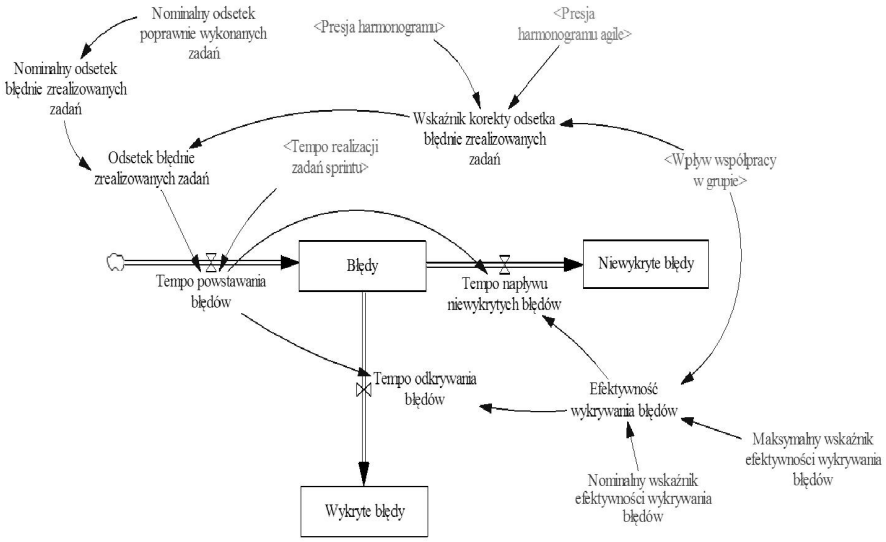
Fig. 5. Simulation model of the software production process with use of the Scrum approach

W procesie symulacji wyznaczane są wartości następujących charakterystyk:

- dla zespołu wytwórczego:
  - produktywność członka zespołu:
  - produktywność całego zespołu,
  - czas potrzebny do zmiany morale zespołu,
  - czas potrzebny na uczenie się zespołu,
  - nominalny poziom poczucia bezpieczeństwa;
- oprogramowania:
  - odsetek błędnie zrealizowanych zadań,
  - efektywność wykrywania błędów,
  - wskaźnik korekty odsetka błędnie zrealizowanych zadań.



Rys. 6. Charakterystyki zespołu wpływające na jakość wytwarzanego oprogramowania
Fig. 6. Team characteristics affecting the quality of the software



Rys. 7. Model jakości oprogramowania
Fig. 7. Model of the software quality

5.2 Ocena adekwatności modelu

W celu oceny prawdziwości zbudowanego modelu i jego zdolności do odwzorowywania badanej rzeczywistości w wybranym zakresie dokonano sformalizowanego badania adekwatności przyjętego modelu. Problematyka ta szczegółowo została opisana w [13]. Najogólniej rzecz ujmując, polegała ona na opracowaniu testów pozwalających stwierdzić poprawność: strukturalną i behawioralną modelu oraz jego odpowiedzi na zmiany reguł decyzyjnych.

Wielu autorów podnosi kwestię istotności budowania zaufania do modelu symulacyjnego przez przeprowadzenie szerokiego zakresu testów w wymiarach:

- struktury modelu – testy obejmujące m.in. weryfikację: struktury modelu, parametrów modelu, warunków ekstremalnych, granic modelu, zgodności wymiarów parametrów;
- zachowania modelu – testy obejmujące m.in. poprawność reprodukowania zachowań obiektu rzeczywistego (wraz z anomaliami), analizę wrażliwości na zmiany parametrów, zgodność statystyczną wartości na wyjściach modelu z wartościami pochodzącymi z obserwacji systemu (obiektu) rzeczywistego;
- stabilności reakcji na zmiany reguł decyzyjnych – testy obejmujące m.in. wrażliwość reguł decyzyjnych na zmianę parametrów.

Autorzy ci marginalizują równocześnie kwestię statystycznego dopasowania danych generowanych na wyjściu modelu do danych historycznych zarejestrowanych w wyniku obserwacji systemów (obiektów) w rzeczywistości. Podkreślają oni, że każdy zbiór danych wynikowych można zawsze dopasować do wymaganego poziomu dokładności, dlatego modele powinny być oceniane pod kątem ich adekwatności głównie według innych kryteriów niż dopasowanie statystyczne. W opinii autorów niniejszej pracy nie można do końca zgodzić się z takim stanowiskiem z kilku niżej wymienionych powodów:

- większość odbiorców spodziewa się istnienia formalnej metody oceny prawdziwości dostarczonych im modeli symulacyjnych;
- pomimo że diagnoza statystyczna dopasowania modelu dynamiki systemowej do danych uzyskiwanych z obserwacji systemów (obiektów) rzeczywistych nie jest kryterium wystarczającym do stwierdzenia, że opracowany model jest poprawny, to jej rola w procesie budowania zaufania do tegoż modelu jest nieoceniona, gdyż pozwala na częściowe sformalizowanie procesu oceny jego adekwatności;
- dobrze zbudowany model dynamiki systemowej musi poprawnie odwzorowywać historyczne (przeszłe) zachowania systemu (obiektu) rzeczywistego na bazie dostępnych danych historycznych oraz umożliwiać równie prawidłową predykcję jego przyszłych zachowań na bazie założonych parametrów.

Ocenę adekwatności zaproponowanego modelu symulacyjnego dokonano przez zastosowanie metod statystycznych do oceny jego poprawności behawioralnej. Problem autorzy sprowadzili do opracowaniu reguły decyzyjnej, która na podstawie wyników eksperymentów przeprowadzonych na bazie modelu pozwoliła wyznaczyć stopień jego "prawidłowości" na podstawie dopasowania generowanych przy jego użyciu wartości do posiadanych danych historycznych [12]. Do oceny błędu dopasowania modelu symulacyjnego dynamiki systemowej wykorzystano statystyki niezgodności Theila.

Podejście to zasługuje na szczególną uwagę pośród całego zbioru testów służących temu celowi, ponieważ pozwala poznać nie tylko całkowitą wielkość błędu, lecz również umożliwia wskazanie przyczyny jego powstawania. Konstrukcja tych statystyk opiera się na zaobserwowanych w praktyce budowy modeli symulacyjnych przyczynach powstawania ich niedopasowania do charakterystyki obiektów rzeczywistych, z których jedna to słabość opracowanego modelu (błąd systematyczny), zaś druga to przypadkowość występująca w danych historycznych (błąd losowy).

Oceny adekwatności modelu symulacyjnego dokonano na podstawie wielkości (1), (2) i (3):

$$U^M = \frac{(\bar{S} - \bar{A})^2}{\sigma^2}, \tag{1}$$

$$U^S = \frac{(S_S - A_A)^2}{\sigma^2}, \tag{2}$$

$$U^C = \frac{2(1-r)S_S A_A}{\sigma^2} \tag{3}$$

za pomocą funkcji decyzyjnej określonej następująco:

$$d(U^M, U^S, U^C) = \begin{cases} h^0, jeżeli\ U^M \in \langle 0, \quad 0.1\rangle \wedge U^S \in \langle 0, \quad 0.2\rangle \wedge U^C \in \langle 0.8, 1\rangle \\ h^1, jeżeli\ U^M \notin \langle 0, \quad 0.1\rangle \vee U^S \notin \langle 0, \quad 0.2\rangle \vee U^C \notin \langle 0.8, 1\rangle \end{cases}, \tag{4}$$

gdzie:

h^0 – nie ma podstaw do odrzucenia hipotezy o adekwatności opracowanego modelu symulacyjnego do procesów zachodzących w rzeczywistości;

h^1 – uzyskane wyniki zaprzeczają prawdziwości hipotezy o adekwatności opracowanego modelu symulacyjnego do procesów zachodzących w rzeczywistości,

przy czym:

U^M – mierzy udział błędu systematycznego w wartości błędu średniokwadratowego,

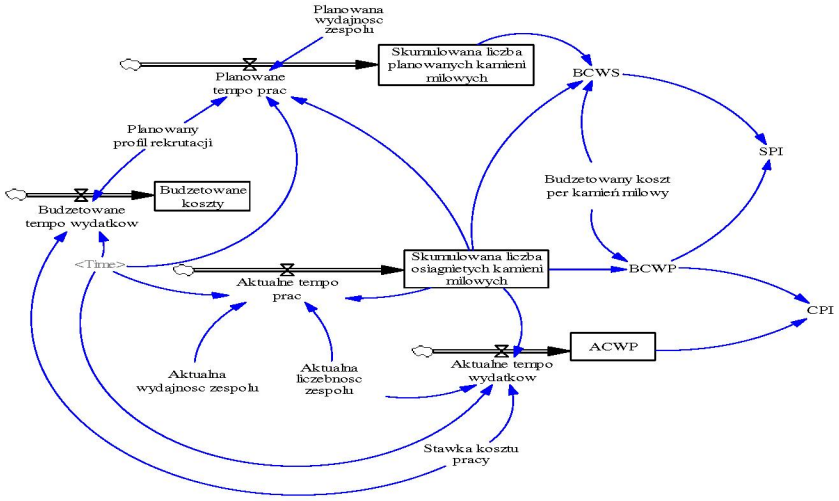
U^S – mierzy udział błędu spowodowanego przez niezgodności wariancji w wartości błędu średniokwadratowego,

U^C – mierzy udział błędu spowodowanego przez niezgodność kowariancji w wartości błędu średniokwadratowego.

5.2 Schemat procesu walidacji

Jako podstawę prac walidacyjnych przyjęto porównanie wartości wypracowanej „Earned Value". Nazwa "Earned Value" pochodzi od idei, że każdy produkt dostarczany przez projekt posiada planowany koszt, czyli swoją „wartość".

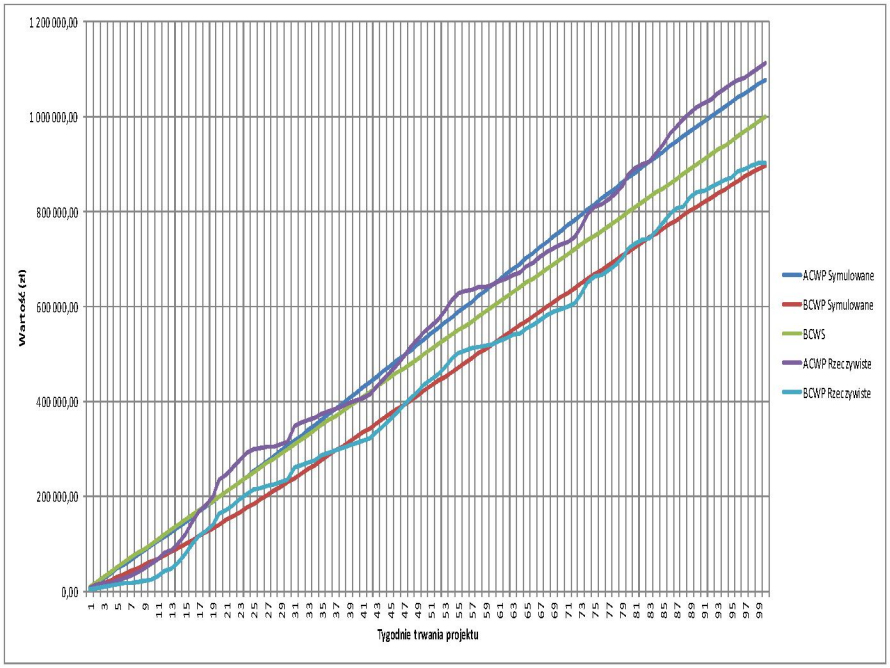
Model symulacyjny zbudowany zgodnie z założeniami metody Earned Value przy użyciu środowiska symulacyjnego Vensim® przedstawiono na rysunku 8.



Rys. 8. Model symulacyjny Earned Value
Fig. 8. Simulation model Earned Value

Gdy produkt jest ukończony, jego „wartość" jest „zarobiona" w ramach projektu. Podejście takie, oparte na ciągłym monitorowaniu projektów na każdym etapie ich realizacji, pozwala wcześnie dostrzec ewentualne odchylenia harmonogramu oraz przekroczenia kosztów i na tej podstawie podjąć odpowiednie działania, aby zminimalizować ryzyko niepowodzenia realizowanego przedsięwzięcia. Kontrola projektu obejmuje również wycenę wartości prac, produktów, usług itp. zrealizowanych do chwili kontroli. Pozwala to na porównanie trzech parametrów, tj. kosztów planowanych (BCWS), kosztów rzeczywistych (ACWP), wartości uzyskanej (BCWP) i określenia wskaźników odchyleń zawiązanych z realizacją projektu, takich jak planowany czas (harmonogram) - SPI - i budżet (koszt projektu) – CPI.

Na bazie skonstruowanego modelu został przeprowadzony eksperyment symulacyjny, którego wyniki posłużyły następnie jako materiał wejściowy do procesu walidacji a następnie do oceny jego adekwatności. Na poniższym wykresie (rys. 9) zestawiono ze sobą wyniki symulacji oraz dane zebrane podczas realizacji rzeczywistego projektu IT, realizowanego metodyką Scrum.



Rys. 9. Porównanie wyników symulacji z wartościami zaobserwowanymi w rzeczywistości

Fig. 9. Comparison of simulation and real values

Na ich podstawie dokonano obliczenia statystyk Theila, których wartości zaprezentowano w tabeli 1.

Tab. 1. Uzyskane charakterystyki Theila

Tab. 1. Theil's characteristics

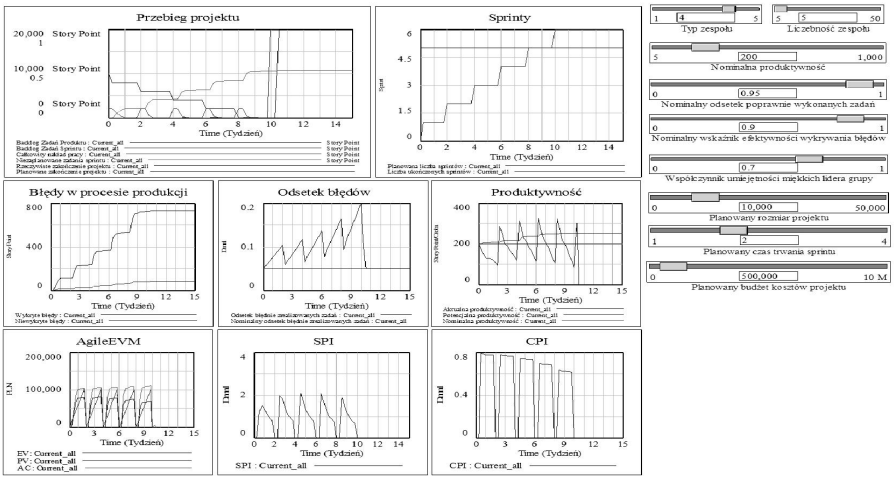
Zmienna	$\mathbf{U^M}$	$\mathbf{U^S}$	$\mathbf{U^C}$
ACWP	0,03	0,13	0,84
BCWP	0,00	0,10	0,90

Duża wartość U^C i relatywnie małe wartości U^M i U^S pozwalają stwierdzić występowanie błędu niesystematycznego, który może być spowodowany dużą fluktuacją w zakresie parametrów wpływających na aktualne tempo prac zespołu. Na podstawie powyższych wyników oraz przyjętej postaci funkcji decyzyjnej (4) można stwierdzić, że nie ma podstaw do odrzucenia hipotezy o adekwatności opracowanego modelu symulacyjnego do procesów zachodzących w rzeczywistości.

6 Uzyskane wyniki symulacji

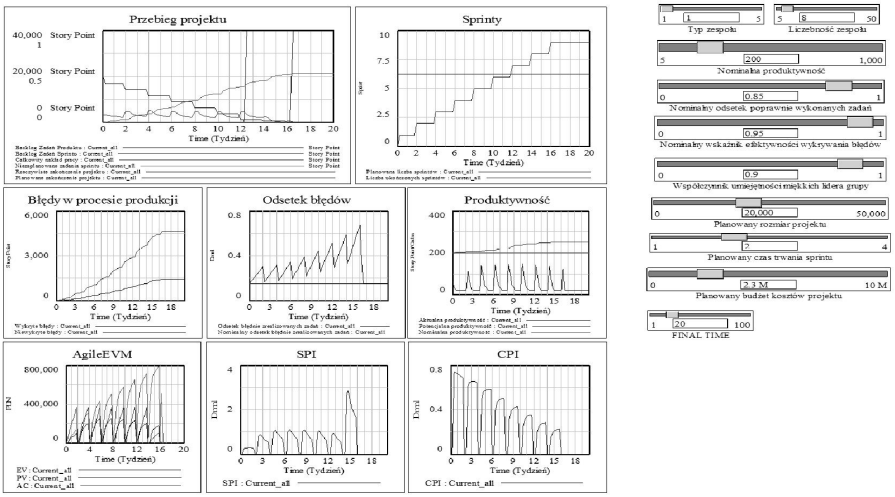
W oparciu o zbudowany przez autorów model przeprowadzono eksperymenty symulacyjne dla dwóch projektów. Główne parametry modelu i uzyskane wyniki

symulacji przedstawiają kolejne rysunki (rys. 10 i 11). Natomiast podsumowanie wyników symulacji przedstawia w sposób syntetyczny zamieszczona poniżej tabela (tab. 2).



Rys. 10. Wyniki symulacji dla projektu 1

Fig. 10. Results of the simulation for project 1



Rys. 11. Wyniki symulacji dla projektu 2

Fig. 11. Results of the simulation for project 2

Tab. 2. Podsumowanie wyników symulacji
Tab. 2. Simulation run summary

Charakterystyka	Projekt 1	Projekt 2
Wartość projektu	0,5 mln zł	2.0 mln zł
Zespół	• zespół doświadczonych specjalistów dziedzinowych, • każdy starał się jak najlepiej wykonywać swoje zadania, • żadna nie wniosła do projektu nic ponad stawiane mu wymagania	• doświadczonych specjalistów dziedzinowych, • Każda z ośmiu osób wchodzących w skład zespołu miała bardzo duże problemy z pracą w nieznanej sobie grupie, • osoby skrajnie egocentryczne
Lider zespołu	• umiejętności miękkie lidera określono na poziomie 0,7 (w skali 0 - 1), • osoba posiadająca doświadczenie w kierowaniu zespołami wytwórczymi, • osoba została dobrze przyjęta przez pozostałych współpracowników	• umiejętności miękkie lidera określono na poziomie 0,9 (w skali 0 - 1), • osoba posiadająca doświadczenie w kierowaniu zespołami wytwórczymi, • osoba zdołała tylko częściowo przezwyciężyć trudności pozatechniczne w zespole
Złożoność projektu	10 000 story point	40 000 story point
Odsetek błędów	5%	15%
Skuteczność wykrywania błędów	95%	90%
Opóźnienie projektu	2 tygodnie	6 tygodni
Produktywność	Wysoka	Słaba
Jakość pracy	Dobra	Bardzo niska
Rezultat projektu	Sukces	Porażka (znaczny wzrost kosztów)

Wnioski nasuwające się w wyniku oceny rezultatów uzyskanych przez autorów w drodze symulacji komputerowej są następujące:

- typ zespołu realizującego projekt wraz z presją harmonogramu stanowią główne czynniki wpływające na jakość pracy wykonywanej w projekcie wyrażoną jako odsetek błędnie zrealizowanych zadań;
- im gorszy był zespół, tym wyższy odnotowano odsetek błędów i niższą efektywność (%) ich wykrywania przez zespół (założono, że w projektach realizowanych zgodnie ze Scrum nie istnieje oddzielny zespół odpowiadającego za testy, a role testerów pełnią cyklicznie kolejni członkowie);

- z kolei wyższy odsetek błędów wpływa negatywnie na satysfakcję zespołu, a przez to również na poczucie bezpieczeństwa i działania związane z samodoskonaleniem, co z kolei przekłada się morale zespołu i obniżenie jego produktywności.

7 Podsumowanie

Nie ulega wątpliwości, że obecnie do oceny jakości procesów wytwórczych i jakości pracy zespołów przykłada się o wiele większą uwagę niż do oceny produktów programowych. Świadczy o tym chociażby fakt, że w Polsce wszystkie znaczące firmy wytwarzające oprogramowanie uzyskały certyfikat systemu zarządzania jakością ISO 90001, kilka zdecydowało się na ocenę CMM/CMMI, natomiast krótka norma dotycząca oceny jakości produktów programowych – ISO 9126, praktycznie nie jest stosowana, nie została nawet przetłumaczona na język polski. Sytuacja ta wydaje się niezrozumiała, chociaż istnieją argumenty przemawiające za takim postępowaniem, np. pojawienie się zwinnego procesu produkcji oprogramowania lub stosowanie symulacyjnych modeli jakości zespołu. Można jednak przypuszczać, że w niedalekiej przyszłości sytuacja musi ulec zmianie – w rzeczywistości dla odbiorcy oprogramowania wcale nie jest istotne, jak dojrzałe i ustabilizowane procesy realizuje wykonawca i/lub jakie stosuje symulacyjne modele jakości pracy zespołu, natomiast żywotne znaczenie ma jakość nabytego produktu programowego.

Literatura:

1. Abrahamsson P. et al.: *Agile software development methods*, Review and Analysis, 2002
2. Elssamadisy A.: *Agile. Wzorce wdrażania praktyk zwinnych*, Gliwice, Helion 2010,
3. *Information technology – Software product evaluation, Parts 1-6*, ISO, Geneva, 1998-2001
4. *ISO/IEC 9126:2001 Software engineering - Product Quality, Part 1: Quality model*, ISO, Geneva, 2001
5. Sterman J. D.: *Business Dynamics. System thinking and modeling for a complex world*. McGraw Hill, 2000
6. Matuszyńska M., Wawrzyniak A., Wąsikowska B.: Przegląd oprogramowania do modelowania systemowo-dynamicznego, *Studia Informatica*, nr 12, Wydawnictwo Naukowe Uniwersytetu Szczecińskiego, Szczecin 2000
7. Martin R., Martin M.: *Agile. Programowanie zwinne: zasady, wzorce i praktyki zwinnego wytwarzania oprogramowania w C#*, Gliwice, Helion, 2008
8. Protasowicki T., Stanik J.: *Modelowanie wartości Earned Value w zarządzaniu projektami z wykorzystaniem podejścia Agile. Część 1 - Opis dla potrzeb symulacji*, XV Krajowa Konferencja Inżynierii Oprogramowania, 2013
9. Protasowicki T., Stanik J.: *Modelowanie wartości Earned Value w zarządzaniu projektami z wykorzystaniem podejścia Agile. Część 2 - Model symulacyjny*, XV Krajowa Konferencja Inżynierii Oprogramowania, 2013
10. Protasowicki T., Stanik J.: *Ocena adekwatności modeli symulacyjnych dynamiki systemowej na przykładzie modelu Earned Value*, XIX Warsztaty Naukowe PTSK "Symulacja w Badaniach i Rozwoju", 2012
11. Madachy R. J.: *Software process dynamics*, Wiley, New Jersey, 2008
12. Schwaber K.: *Sprawne zarządzanie projektami metodą Scrum*, Warszawa, 2005

13. Vensim®. Ventana® Simulation Environment. User's Guide Version 5, Ventana Systems, Inc. 1988-2002, http://www.vensim.com

Streszczenie

Współczesne projekty informatyczne w coraz większej części prowadzone są w oparciu o metodyki zwinne. Ich zastosowanie ma z definicji zapewnić szybkie dostarczanie klientowi kolejnych wersji funkcjonującego oprogramowania charakteryzującego się określoną jakością. W artykule podjęto próbę zbadania wpływu odpowiedniego doboru personelu do zespołów wytwarzających oprogramowanie w oparciu o podejście zwinne na jakość powstającego produktu. W tym celu opracowano model symulacyjny, który - czerpiąc z osiągnięć nauk humanistycznych i inżynierii oprogramowania - obejmuje swoim zakresem obszary: zarządzania przepływem pracy, detekcją i naprawą błędów oraz budową zespołów w projekcie zarządzanym według metodyki Scrum. Zbudowany model pozwala między innymi zbadać, jak dopasowanie cech osobowości poszczególnych członków zespołu do różnych ról i procesów wpływa na jakość produktu, w którego tworzenie są oni zaangażowani. Przeprowadzone przy jego użyciu eksperymenty symulacyjne dają ciekawe spojrzenie na aspekty dotyczące zarządzania zasobami ludzkimi i ich wpływu na przebieg projektu oraz jakość otrzymanego w rezultacie oprogramowania. Wykorzystanie zaproponowanego modelu w praktyce może przyczynić się do lepszego zarządzania projektami prowadzonymi w zgodzie z filozofią Agile poprzez dobór kadry charakteryzującej się odpowiednimi zdolnościami zarówno technicznymi, jak i interpersonalnymi.

Słowa kluczowe: zarządzanie projektami, metodyki zwinne, jakość oprogramowania, dynamika systemowa

Quality model for agile software development process

Summary

IT projects currently more often base on agile management methods. This approach should ensure that project will, by the definition, provide quick delivery of the next versions of the software on the specific level of the quality. This paper is an attempt to examine the impact of appropriate team composition on the quality of software developed in the agile oriented environment. The simulation model proposed in this paper spans social sciences and software engineering and covers following areas: workflow management, defects detection and repair, and construction of teams in projects run with Scrum. The model allows to examine how personality fit of the individual team members to different roles and processes affects the quality of the software. Simulation experiments provide an interesting insight at the aspects of human resource management and its impact on the course of the project and the quality of developed software. The use of the proposed model in practice can contribute to better management of projects carried out in accordance with the philosophy of Agile

through the selection of staff characterized by appropriate set of skills both technical and interpersonal.

Keywords: project management, earned value method, system dynamics, simulation