Zarządzanie projektami i modelowanie procesów

Redakcja naukowa

Zdzisław Szyjewski
Karolina Muszyńska

Konferencje naukowe organizowane przez

Polskie Towarzystwo Informatyczne:

VIII edycja Sejmiku Młodych Informatyków

XV edycja Krajowej Konferencji Inżynierii Oprogramowania

XX edycja Systemów Czasu Rzeczywistego

zostały dofinansowane

przez Ministra Nauki i Szkolnictwa Wyższego

w ramach programu związanego z realizacją zadań upowszechniających naukę

(decyzja nr 1064/P-DUN/2013 z dnia 24.07.2013)

Dziękujemy!

POLSKIE TOWARZYSTWO INFORMATYCZNE

Zarządzanie projektami i modelowanie procesów

Redakcja naukowa

Zdzisław Szyjewski
Karolina Muszyńska

Warszawa 2013

Recenzenci
prof. dr hab. inż. Zbigniew Banaszak, dr hab. inż. Leszek Borzemski, dr inż. Włodzimierz Dąbrowski, prof. dr hab. inż. Krzysztof Goczyła, dr hab. Tomasz Królikowski, dr hab. Leszek A. Maciaszek, prof. dr hab. Andrzej Marciniak, Wiesław Paluszyński, dr Witold Staniszkis, dr hab. Jakub Swacha

Redakcja naukowa
prof. dr hab. Zdzisław Szyjewski
dr Karolina Muszyńska

Autorzy
mgr inż. Tomasz Protasowicki, dr inż. Jerzy Stanik – **ROZDZIAŁY 1, 2**
dr inż. Robert Waszkowski – **ROZDZIAŁ 3**
dr inż. Artur Ziółkowski, dr inż. Tomasz Sitek – **ROZDZIAŁ 4**
prof. dr hab. Zdzisław Szyjewski – **ROZDZIAŁ 5**
dr Jan Trąbka – **ROZDZIAŁ 6**
mgr inż. Rafał Wojszczyk – **ROZDZIAŁ 7**
mgr Tomasz Komorowski – **ROZDZIAŁ 8**

Redakcja techniczna
Jakub Swacha

Projekt okładki
Łukasz Piwowarski

ISBN 978-83-7518-599-7

Wydanie: I. Nakład: 200 egz. Ark. wyd. 6,5. Ark, druku 8,1.
Wydawca, druk i oprawa: PPH ZAPOL, al. Piastów 42, 71-062 Szczecin

Spis treści

Wstęp **9**

1. Modelowanie wartości Earned Value w zarządzaniu projektami z wykorzystaniem podejścia Agile. Część 1 - Opis dla potrzeb symulacji **13**

1.1. Istota problemu adaptacji klasycznego modelu EVM do realiów projektów prowadzonych zgodnie z podejściem Agile . 14

1.2. Przegląd metody EVM 15

1.3. Podejście Agile w zarządzaniu projektami 18

1.4. Model AgileEVM 20

Podsumowanie 22

Bibliografia 23

2. Modelowanie wartości Earned Value w zarządzaniu projektami z wykorzystaniem podejścia Agile. Część 2 - Model symulacyjny 25

2.1. Dodatkowe założenia do budowy modelu symulacyjnego 26

2.2. Struktura modelu symulacyjnego 30

2.3. Uzyskane wyniki 33

Podsumowanie 37

Bibliografia 38

3. Metody pozyskiwania danych do oceny produktywności poszczególnych członków zespołu projektowego w projektach informatycznych **39**

3.1. Zakres informacyjny danych gromadzonych podczas realizacji projektu ... 40
3.2. Wskaźniki produktywności dla projektów IT ... 43
3.3. Metody pozyskiwania danych do oceny produktywności ... 45
Podsumowanie ... 48
Bibliografia ... 48

4. Projekt-czynnik-decyzja. Badanie czynników decyzyjnych w projektach informatycznych i ich wpływu na powodzenie projektów ... 49

4.1. Czynniki decyzyjne a decyzje projektowe ... 51
4.2. Wpływ czynników decyzyjnych na katalog decyzji ... 56
4.3. Projekt zarządzany świadomie ... 59
Podsumowanie ... 60
Bibliografia ... 61

5. Strategie zarządzania ryzykiem w projektach ... 63

5.1. Ryzyko w projekcie ... 64
5.2. Zarządzanie ryzykiem w projekcie ... 65
5.3. Budżet projektu ... 69
5.4. Działania zapobiegawcze ... 71
Podsumowanie ... 73
Bibliografia ... 74

6. Analiza procesów workflow jako narzędzie opisu wymagań funkcjonalnych – propozycja metodyki ... 77

6.1. Podstawowe pojęcia związane z systemami workflow ... 79
6.2. Założenia i narzędzia proponowanej metodyki ... 81
6.3. Krótka charakterystyka wybranych narzędzi metodyki SPARD. 83

6.4. Przykład zastosowania metodyki SPARD ... 87

Podsumowanie ... 93

Bibliografia ... 94

7. Zestawienie metryk oprogramowania obiektowego opartych na statycznej analizie kodu źródłowego ... 95

7.1. Cel stosowania metryk ... 96

7.2. Przegląd wybranych najpopularniejszych metryk ... 96

7.3. Przegląd wybranych pozostałych metryk ... 101

7.4. Propozycja kierunków dalszego rozwoju metryk ... 102

Podsumowanie ... 105

Bibliografia ... 106

8. Analiza wymagań w metodologiach zarządzania projektami informatycznymi ... 109

8.1. Wybrane metodologie zarządzania projektami informatycznymi ... 110

8.2. Zarządzanie wymaganiami i klasyfikacja wymagań ... 114

8.3. Wybrane metodyki zarządzania projektami informatycznymi - charakterystyka i podejście do analizy wymagań ... 116

Podsumowanie ... 131

Bibliografia ... 132

Autorzy i afiliacje ... 135

Wstęp

Rozwój nowych technologii teleinformatycznych i ich możliwości zastosowań otwierają nowe obszary badawcze. Nie tylko szybkość obliczeniowa komputerów i niedostępne dotychczas możliwości komunikacyjne stanowią wartość do wykorzystania w działalności gospodarczej, ale również nowe możliwości analityczne, eksploracja wiedzy zapisanej w hurtowniach danych czy też wkomponowywanie rozwiązań teleinformatycznych w realne procesy biznesowe w celu ich efektywnego wspomagania i usprawniania.

Definicja określonego celu w zdefiniowanym czasie i zaplanowanych budżecie to podejście projektowe do prowadzenia biznesu. Coraz częściej spotykamy się z takim sposobem prowadzenia działalności gospodarczej, gdzie cele cząstkowe osiągane w ramach zdefiniowanych przedsięwzięć składają się na długofalowy rozwój dynamicznie funkcjonującej firmy. Definiowanie celów projektowych nie jest ograniczone do obiektów fizycznych, w wyniku, których powstają określone budowle lub obiekty inżynierskie, ale coraz częściej dotyczy to celów niematerialnych. Szczególnym przypadkiem przedsięwzięć projektowych, gdzie cel jest niematerialny są projekty informatyczne. W przypadku projektów budowlanych czy konstrukcyjnych procesy planowania, szacowania pracochłonności i wreszcie realizacji zadań z kontrolą postępu prac korzystały z w miarę stabilnych warunków wytwarzania i bogatych doświadczeń wcześniejszych realizacji.

Projekty informatyczne maja odmienną charakterystykę. Szybko zmieniająca się technologia teleinformatyczna powoduje, że doświadczenia z wcześniej realizowanych projektów, są tylko w niewielkim stopniu przydatne w pracach nad nowymi przedsięwzięciami. Niewidzialny produkt w postaci oprogramowania komputerowego, jest trudny do kontroli postępu prac oraz sprawdzeniu jego faktycznej funkcjonalności. Dotychczasowe metody wytwarzania i kontroli stają się mało przydatne i w związku z tym prowadzone są prace mające na celu wypracowanie nowych rozwiązań, adekwatnych do zmienionych warunków zarządzania projektami, mających na celu wytwarzanie systemów informatycznych.

Ustabilizowane procesy produkcyjne wytwarzania długich serii określonego wyrobu pozwalają na korzystanie z ustabilizowanych procesów wytwarzania. Obecne warunki funkcjonowania firm zmuszają do częstych modyfikacji procesów wytwarzania, częstego modyfikowania produktu, co stanowi określoną trudność i istotną zwyżkę kosztów wytwarzania. Nowe technologie teleinformatyczne dostarczają narzędzi do szybkiego modelowania procesów produkcyjnych, ich analizy i symulowania zachowania, co w znaczący sposób podnosi sprawność funkcjonowania firm i przynosi określone korzyści biznesowe. Budowa modelu nowego procesu i sprawdzenie jego funkcjonowania z wszystkimi aspektami analizy biznesowej nie wymaga dużych nakładów fizycznych i skomplikowanych procesów badawczych, ale może być sprawnie i relatywnie prosto realizowana z wykorzystaniem oprzyrządowania informatycznego.

Problematyce zarządzania projektami informatycznymi i modelowaniu procesów biznesowych poświęcamy niniejszą monografię. Pierwsze dwa rozdziały opisują koncepcję modelowania wartości Earned Value w projektach zarządzanych z wykorzystaniem metodyk zwinnych. Pierwsza część to opis metody Earned Value i propozycje jej modyfikacji na potrzeby zastosowania jej w projektach Agile, natomiast w części drugiej autorzy prezentują model symulacyjny służący do badania przebiegu projektów IT zarządzanych w oparciu o metody zwinne. Rozdział trzeci poświęcony jest analizie źródeł dla oceny produktywności członków zespołu projektowego oraz prezentacji wskaźników do oceny tej produktywności. W rozdziale czwartym autorzy zwracają uwagę na potrzebę usystematyzowania i skodyfikowania wiedzy teoretycznej i praktycznej związanej z zarządzaniem projektami informatycznymi w obszarze podejmowania decyzji kierowniczych i w tym celu podejmują próbę zdefiniowania zestawu czynników decyzyjnych mających wpływ na powodzenie projektów. Rozdział piąty prezentuje czytelnikowi strategie zarządzania ryzykiem w projektach oraz uwarunkowania zarówno aktywnego jak i reaktywnego podejścia do zarządzania ryzykiem. W rozdziale szóstym autor proponuje metodykę analizy procesów workflow, jako narzędzia opisu wymagań funkcjonalnych, wykorzystującą narzędzia analityczne i informatyczne, oraz opisuje jej zastosowanie w analizie procesów przykładowej organizacji. Autorzy rozdziału siódmego dokonują zestawienia metryk oprogramowania obiektowego, opartych na statycznej analizie kodu źródłowego i

prezentują autorską propozycję dalszego rozwoju metryk, jako jednego z czynników potrzebnych do oceny jakości zaimplementowanych wzorców projektowych. Monografie zamyka rozdział ósmy, który poświęcony jest problematyce analizy wymagań w wybranych metodykach zarządzania projektami informatycznymi. Omówiono w nim nie tylko charakterystyczne cechy takich metodyk jak: PRINCE2, PMBoK, RUP, MSF czy eXtreme Programming, ale wskazano także na różnice w podejściach do analizy wymagań, jej formalizacji i umiejscowieniu w procesach zarządzania projektami.

Życzymy czytelnikom interesującej lektury i dziękujemy wszystkim osobom, które przyczyniły się do powstania niniejszej monografii.

Zdzisław Szyjewski
Karolina Muszyńska

Rozdział 1

Modelowanie wartości Earned Value w zarządzaniu projektami z wykorzystaniem podejścia Agile. Część 1 - Opis dla potrzeb symulacji

W dwóch kolejnych rozdziałach przedstawiono koncepcję modelowania wartości Earned Value w projektach zarządzanych z wykorzystaniem metodyk zwinnych. W pierwszym rozdziale - opis dla potrzeb symulacji, autorzy dokonali przeglądu metody Earned Value (EVM) pod kątem możliwości jej zastosowania w projektach IT bazujących na nowoczesnych metodach zarządzania opartych na filozofii Agile. Przedstawione treści objęły sformułowanie potrzeby modyfikacji metody EVM w kontekście problemu zarządzania projektami zgodnie z tą filozofią. Zaprezentowano również podstawy związane z podejściem Agile w zarządzaniu projektami. Autorzy zaproponowali modyfikacje polegające na zmianach w sposobie kalkulacji zmiennych klasycznego modelu EVM umożliwiające jego przekształcenie do wersji uwzględniającej specyfikę projektów Agile tj. AgileEVM.

Podstawy koncepcyjne metody Earned Value powstały około 1890 roku i wywodzą się z naukowego nurtu zarządzania, który skupiony był głównie na wydajności pracy, czyli poprawie wyników osiąganych przez poszczególnych pracowników. Metoda Earned Value (EVM) pozwala na skuteczną integrację zakresu, kosztów i czasu [6], stanowi więc nieocenione narzędzie wsparcia kierownika projektu w mierzeniu efektywności w projekcie. Jest ona ponadto rekomendowana przez PMBoK (Project Management Body of Knowledge) opracowane w Project Management Institute (PMI), które określa ją jako jedną z najskuteczniejszych metod mierzenia postępów projektu [5]. EVM jest jednak krytykowana w kontekście jej przydatności do wspomagania

zarządzania projektami realizowanymi z wykorzystaniem koncepcji Agile. U podstaw tej krytyki leżą założenia poczynione w EVM na bazie projektów realizowanych w sposób tradycyjny, które były fundamentem do jej stworzenia. EVM zakłada, że projekt jest realizowany w oparciu o dobrze zdefiniowaną strukturę zadań do wykonania, która wynika ze znanego i z reguły niezmiennego zakresu projektu. Założenia te zderzone z rzeczywistością projektów prowadzonych w oparciu o podejście zwinne często stanowią podstawę do odrzucenia EVM jako nie nadającej się do zastosowania w warunkach, w których zakres projektu jest niedookreślony, wymagania są zmienne, a wykonana praca często może wymagać jej częściowego powtórzenia (refaktoringu).

W niniejszej publikacji autorzy nakreślili istotę problemu adaptacji klasycznego modelu EVM do wymagań podejścia zwinnego w zarządzaniu projektami. W celu rozwiązania tego problemu autorzy przeprowadzili przegląd tradycyjnego modelu EVM oraz przedstawili istotę podejścia Agile do zarządzania projektami związanymi z wytwarzaniem oprogramowania na przykładzie metodyki Scrum. Na koniec autorzy zaproponowali modyfikację modelu EVM tj. AgileEVM uwzględniającą specyfikę tego podejścia.

1.1. Istota problemu adaptacji klasycznego modelu EVM do realiów projektów prowadzonych zgodnie z podejściem Agile

Autorzy zgadzają się z tezami przedstawionymi w literaturze [2][3][8], że zastosowanie EVM w jej niezmienionej postaci może, w przypadku projektów związanych z wytwarzaniem oprogramowania, prowadzić do generowania błędnych wartości poszczególnych wskaźników. W rezultacie sprowadza się to do uzyskania błędnej oceny stanu realizacji projektu.

W związku z powyższym wymagane jest takie zmodyfikowanie sposobu kalkulacji wybranych zmiennych modelu EVM, aby umożliwić jego zastosowania w badaniu postępów pracy w projektach realizowanych w oparciu o Scrum, XP i inne metodyki zwinne. Zaproponowana przez autorów modyfikacja modelu EVM koncentruje się na dokonywaniu pomiarów na poziomie poszczególnych sprintów zamiast na poziomie wydania (składającego się z

kilku sprintów) lub całego produktu. Wskutek wprowadzenia w/w zmian powstał model AgileEVM.

Autorzy kontynuowali prace, podjęte w niniejszym rozdziale, z zamiarem weryfikacji poprawności przyjętych hipotez w drodze budowy modelu symulacyjnego. Implementacja modelu symulacyjnego i weryfikacja uzyskanych na jego podstawie wyników wykracza poza ramy niniejszej publikacji, mającej charakter wstępnych rozważań teoretycznych.

1.2. Przegląd metody EVM

Earned Value jest metodą zarządzania projektami w oparciu o regularne porównywanie faktycznych kosztów projektu do kosztów planowanych i wykonanych prac. Nazwa "Earned Value" wywodzi się z założenia, które mówi, że każdy produkt dostarczany przez projekt cechuje się planowanym kosztem, czyli posiada swoją "wartość". W momencie ukończenia produktu, jego "wartość" jest "zarobiona" w ramach projektu. Podejście takie, zakłada ciągłe monitorowanie projektu na każdym etapie jego realizacji. Pozwala to wcześnie dostrzec ewentualne odchylenia harmonogramu oraz przekroczenia kosztów. Dzięki temu można podjąć odpowiednie działania prowadzące do zminimalizowania ryzyka niepowodzenia realizowanego przedsięwzięcia. Kontrola projektu obejmuje wycenę wartości prac, produktów, usług itp. zrealizowanych do chwili kontroli. Umożliwia to porównanie trzech parametrów, tj. kosztów planowanych (PV), kosztów rzeczywistych (AC), wartości uzyskanej (EV) i określenia wskaźników odchyleń zawiązanych z realizacją projektu takich, jak planowany czas (harmonogram) - SPI - i budżet (koszt projektu) - CPI [7].

Na rysunku 1.1 przedstawiono graficznie parametry PV, AC i EV.

Rys. 1.1. Parametry Earned Value.
Źródło: opracowanie własne.

Kluczowe parametry projektu zdefiniowane są w następujący sposób [6]:

- koszty planowane (PV) - budżetowane koszty pracy przewidzianej do wykonania zgodnie z harmonogramem;
- koszty rzeczywiste (AC) - całkowite koszty pracy wykonanej w badanym okresie, zawierające realne koszty zmienne i stałe;
- wartość uzyskana (EV) - suma planowanych budżetów wykonanych zadań.

Na podstawie w/w wartości możliwe jest wyliczenie kluczowych wskaźników EVM [6]:

- wskaźnika odchyleń harmonogramu:
$$SPI = EV/PV \qquad (1.1)$$
- wskaźnik odchylenia kosztów:
$$CPI = EV/AC \qquad (1.2)$$
- odchylenie harmonogramu:
$$SV = EV - PV \qquad (1.3)$$
- odchylenie kosztów:
$$CV = EV - AC \qquad (1.4)$$

W celach prognozowania realizacji planowanych kosztów i/lub harmonogramu na podstawie w/w elementów EVM wykorzystywane są następujące wskaźniki EVM [6]:

- estymacja ostatecznych kosztów:
$$EAC = AC + \big((BAC - EV)/CPI\big) \qquad (1.5)$$

- estymacja kosztów pozostałych do ukończenia pracy:

$$\mathrm{ETC} = (\mathrm{BAC} - \mathrm{EV})/\mathrm{CPI} \qquad (1.6)$$

SPI jest miarą efektywności, która pokazuje jaka część planowanej pracy została wykonana. SPI równe 1 oznacza, że praca w projekcie jest wykonywana zgodnie z harmonogramem. SPI mniejsze od 1 oznacza, że występuje opóźnienie względem harmonogramu. SPI większe od 1 oznacza, że praca jest wykonywana szybciej niż planowano.

CPI jest miarą efektywności, która pokazuje jak faktycznie wydatkowany jest budżet, w porównaniu do tego jak planuje się go wydać. CPI równe 1 oznacza, że budżet jest zużywany w tempie, które zostało zaplanowane. CPI mniejsze od 1 oznacza, że budżet jest zużywany mniej wydajnie, niż planowano, ponieważ rzeczywiste koszty są większe niż zaplanowano. CPI większe od jeden wskazuje na wykonywanie pracy w sposób powodujący powstanie nadwyżki wartości uzyskanej nad kosztami rzeczywistymi.

EAC jest prognozą łącznej kwoty, którą należy wydać, aby zrealizować planowane prace. Prognoza ta uwzględnia wartość rzeczywiście wykonanej pracy.

ETC jest prognozą pozostałych kosztów, które należy ponieść w ramach realizacji zaplanowanej pracy.

Interpretację wartości wybranych wskaźników przedstawiono syntetycznie w tabeli 1.1.

Tab. 1.1. Interpretacja wybranych wskaźników EVM.

CPI > 1	CPI = 1	CPI < 1
EV > AC	EV = AC	EV < AC
Poniżej budżetu	**Zgodnie z budżetem**	**Powyżej budżetu**
SPI > 1	SPI = 1	SPI < 1
EV > PV	EV = PV	EV < PV
Przed czasem	**Na czas**	**Po czasie**

Źródło: opracowanie własne.

1.3. Podejście Agile w zarządzaniu projektami

Dynamiczny rozwój technologiczny, mający miejsce przez ostatnie kilkanaście lat, spowodował zbliżenie się dostawców rozwiązań informatycznych (IT) do środowiska biznesowego. Użytkownicy (biznes) wymagają obecnie od IT wsparcia we wszystkich aspektach związanych z zarządzaniem informacją w ich organizacjach. Niestety pomimo ogromnych nakładów pracy i środków pieniężnych, nadal ponad połowa projektów IT kończy się niepowodzeniem. Na niepowodzenie projektów związanych z wytwarzaniem oprogramowania wpływa wiele czynników powiązanych w dużej mierze z ich specyfiką. Odpowiedzią na powtarzające się w projektach IT problemy było powstanie nowego nurtu prowadzenia projektów – podejścia Agile. Zwinne (ang. agile) prowadzenie projektów charakteryzuje się skupieniem uwagi na jak najsprawniejszym dostarczeniu odbiorcy działającego oprogramowania oraz na ścisłej współpracy z klientem i reagowaniu na jego potrzeby.

Do grupy zwinnych metodyk wytwarzania oprogramowania zaliczamy m.in.: eXtreme Programming (XP), Scrum, Crystal family, Adaptive Software Development (ASD), Dynamic Systems Development Method (DSDM) oraz Feature Driven Development (FDD). Mają one pewne cechy wspólne, które wynikają z przyjętej ideologii tworzenia oprogramowania zapisanej w Manifeście Agile (Manifesto for Agile Software Development). Mówi on, że:

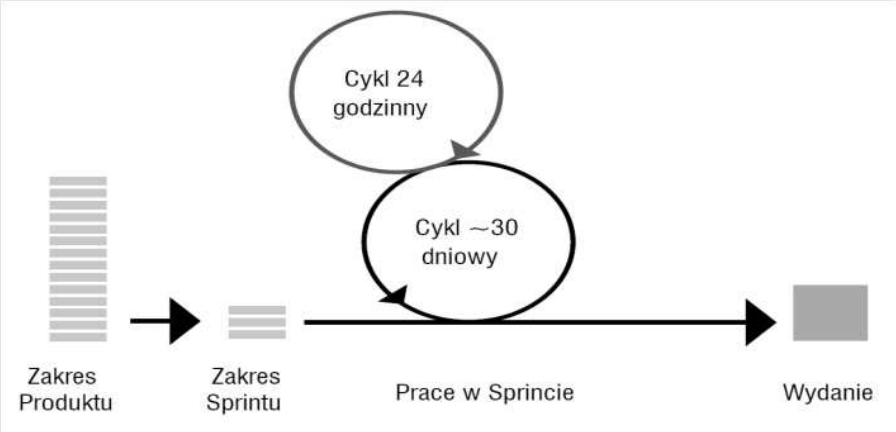
1. ludzie i ich wzajemne interakcje ponad procedury;
2. działające oprogramowanie ponad dokumentację (często ograniczaną do niezbędnego minimum);
3. współpraca z klientem ponad negocjację umów;
4. otwarcie na zmianę zakresu projektu, jako reakcję na zmieniające się wymagania i priorytety klienta, co w przypadku innych metodyk jest bardzo trudne do realizacji.

W niniejszym rozdziale autorzy skupili swoją uwagę na dopasowaniu kalkulacji EVM do projektów zarządzanych w oparciu o metodykę Scrum. W związku z tym poniżej zostały przedstawione jej główne założenia umożliwiające powiązanie procesów wytwórczych oprogramowania z modelem EVM.

Scrum jest jedną z najpowszechniej stosowanych metodyk zwinnych. Znajduje ona zastosowanie w zarządzaniu projektami innowacyjnymi, o trudnym do przewidzenia zakresie prac z dużą zmiennością wymagań. Skupia się

ona przede wszystkim na włączaniu się przyszłych użytkowników w proces wytwórczy oraz na samoorganizacji zespołu projektowego [1].

Przebieg procesu wytwórczego według metodyki Scrum przedstawiono na rysunku 1.2.



Rys. 1.2. Przebieg procesu wytwórczego wg Scrum.
Źródło: opracowanie własne.

Na początku projektu, przed rozpoczęciem pracy, odbywa się spotkanie planistyczne, które ma na celu opracowanie zarysu systemu oraz wstępnego planu jego wytwarzania. Opisuje on ostateczny cel projektu, główne zagrożenia, które mogą wystąpić podczas jego realizacji oraz wyznacza orientacyjną datę zakończenia pracy i planowany koszt wytworzonego produktu. Na tym etapie planowania powstaje tzw. rejestr produktu (ang. product backlog), stanowiący pełną listę funkcjonalności produktu, który jest celem projektu. Zawartość tego rejestru jest przygotowywana przez Właściciela Produktu (ang. Product Owner), czyli osobę reprezentującą klienta (eksperta znającego biznesową funkcjonalność produktu, który może podejmować decyzje związane z produktem). Zazwyczaj rejestr ten zawiera ułożoną według priorytetów listę funkcjonalności, które często są zapisane jako tzw. historyjki użytkownika (ang. user stories). Są to napisane z punktu widzenia klienta wymagania w postaci: „Jako użytkownik systemu chciałbym aby ... (tutaj opis funkcjonalności), która ... (szczegółowy opis cech funkcjonalnych)”. Najważniejszą częścią rejestru produktu są priorytety historyjek, które są im nadawane przez Właściciela Produktu. To według nich poszczególne zadania są wybierane

do realizacji w ramach kolejnych iteracji (ang. sprint). Ponadto historyjki mają przypisane miary złożoności wyrażone w punktach (ang. story points), stanowiących abstrakcyjną jednostkę pracy. Rejestr produktu nie jest bytem statycznym i ewoluuje przez cały czas trwania projektu.

Zespół realizuje pracę w następujących po sobie iteracjach (ang. sprint). Przed rozpoczęciem każdego sprintu odbywa się spotkania planistyczne, którego celem jest utworzenie podzbioru wymagań i wygenerowanie zadań dla zespołu wraz z oszacowaniem czasu potrzebnego na ich realizację. Na tym etapie planowania tworzony jest rejestr sprintu (ang. sprint backlog) określający zakres sprintu. Czas trwania pojedynczego sprintu wynosi od jednego do czterech tygodni, przy czym zaleca się stosowanie przebiegów o tym samym czasie trwania w ramach całego projektu. Pomaga to zespołowi w wypracowaniu regularnego rytmu pracy oraz upraszcza zarządzanie i śledzenie procesu wytwórczego. Skład osobowy zespołu projektowego, jak i cele jakościowe muszą pozostać niezmienne przez cały czas trwania sprintu. Główną zasadą pracy w metodyce Scrum jest to, że każda iteracja musi dostarczyć użytkownikowi kolejną działającą wersję produktu wnoszącą nową wartość. Iteracja kończy się więc z reguły wydaniem kolejnej wersji oprogramowania, niekiedy na jedno wydanie może składać się więcej niż jeden sprint.

1.4. Model AgileEVM

Metoda Earned Value (EVM), omówiona w punkcie 2 niniejszego rozdziału, zakłada, że projekt jest realizowany w oparciu o dobrze zdefiniowaną strukturę zadań do wykonania, która wynika ze znanego i z reguły niezmiennego zakresu. Projekty związane z wytwarzaniem oprogramowania prowadzone w sposób tradycyjny stwarzają wiele problemów dotyczących jakości, kosztów i czasu. Podejście zwinne do wytwarzania oprogramowania, omówione w punkcie 3 niniejszego rozdziału jest wskazywane, zarówno przez naukowców jak i praktyków, jako alternatywa pozwalająca na przezwyciężenie tych problemów. Metodyki zwinne kładą silny nacisk na iteracyjny cykl wytwórczy połączony z dużym zaangażowaniem użytkowników i ich silnym oddziaływaniem na powstający produkt. Powoduje to, że zakres nie może być w pełni zdefiniowany w sposób oddolny na początku projektu, jak to ma miej-

sce w przypadku podejścia tradycyjnego. Stanowi to źródło wyzwań związanych z zastosowaniem metody EVM do zarządzania takimi projektami. Bezpośrednie zastosowania EVM będzie prawdopodobnie skutkować nieprawidłowymi wartościami kosztów planowanych (PV) objawiającymi się ich niedoszacowaniem lub przeszacowaniem występującym w trakcie realizacji projektu.

Autorzy na podstawie swoich doświadczeń związanych z zarządzaniem projektami IT uważają, że po dokonaniu pewnych modyfikacji metoda EVM nadaje się do zastosowania w badaniu postępów pracy w projektach realizowanych w oparciu o metodyki zwinne np. Scrum. Zaproponowana przez autorów modyfikacja klasycznej metody EVM koncentruje się na dokonywaniu pomiarów na poziomie poszczególnych sprintów zamiast na poziomie wydania (składającego się z kilku sprintów) lub całego produktu. W opinii autorów jest to najwłaściwszy sposób, aby wykorzystać formuły EVM w projektach prowadzonych w oparciu o podejście zwinne. W porównaniu z wymogami tradycyjnego modelu EVM, AgileEVM wykorzystuje elementy, które są integralnym elementem procesu Scrum. Pomiary są wykonywane w trakcie każdego sprintu, gdy znane są rzeczywiste wartości produktywności zespołu i koszty. Zamieszczona poniżej tabela 1.2 przedstawia zestawienie elementów EVM w wersji tradycyjnej i zaproponowanej przez autorów AgileEVM.

Tab. 1.2. Zestawienie elementów EVM i AgileEVM.

Element	**EVM**	**AgileEVM**
BAC	Planowany budżet projektu.	Planowany budżet sprintu.
APC	Stosunek kosztu pracy wykonanej w projekcie do planowanego budżetu projektu.	Stosunek liczby story point ukończonych sprincie do liczby story point planowanych w ramach danego sprintu.
EPC	Planowany % ukończonej pracy na danym etapie projektu.	Planowany % zrealizowanych story pointów na danym etapie sprintu.
AC	Realny koszt pracy wykonanej w projekcie.	Całkowity koszt pracy wykonanej w danym sprincie.
PV	Planowany koszt realizacji pracy zgodnie z harmonogramem.	Planowany koszt realizacji sprintu zgodnie z harmonogramem.

Element	EVM	AgileEVM
ETC	Prognoza pozostałych kosztów, które należy ponieść w ramach realizacji zaplanowanej pracy w projekcie.	Prognoza pozostałych kosztów, które należy ponieść w ramach realizacji zaplanowanej pracy w danym sprincie.
EAC	Prognoza łącznej kwoty, którą należy wydać, aby zrealizować planowane prace w projekcie	Prognoza łącznej kwoty, którą należy wydać, aby zrealizować planowane prace w danym sprincie.

Źródło: opracowanie własne.

Podsumowanie

W wyniku dynamicznego rozwoju technologicznego, który spowodował zbliżenie się dostawców rozwiązań informatycznych (IT) do środowiska biznesowego wymagającego obecnie od IT wsparcia we wszystkich aspektach związanych z zarządzaniem informacją w ich organizacjach rośnie ilość kompleksowych projektów mających na celu wdrażanie coraz bardziej skomplikowanego oprogramowania. Projekty te coraz częściej prowadzone są zgodnie z podejściem Agile skupiającym uwagę na jak najsprawniejszym dostarczeniu odbiorcy działającego oprogramowania oraz na ścisłej współpracy z klientem i reagowaniu na jego potrzeby. Rodzi to potrzebę adaptacji już istniejących narzędzi wypracowanych do zarządzania projektami tak, aby można je było wykorzystywać również w projektach prowadzonych według metodyk zwinnych. Przedstawiona w niniejszym rozdziale koncepcja adaptacji metody EVM polegająca na dokonywaniu pomiarów na poziomie poszczególnych sprintów nie wyklucza użycia innych mechanizmów śledzenia postępu projektów realizowanych w oparciu o metodyki zwinne, typowych dla takiego podejścia. Pozwala zaś na łatwe wprowadzenie w tych projektach aspektu ich oceny finansowej uwzględniającej specyficzne uwarunkowania wynikające z podejścia zwinnego. Podjęte w niniejszym rozdziale prace będą stanowiły podstawę do zbudowania przez autorów modelu symulacyjnego i weryfikacji poprawności przyjętych hipotez na bazie uzyskanych wyników symulacji komputerowej.

Bibliografia

1. Abrahamsson P. et al.: *Agile software development methods. Review and analysis*, VTT Publications, Vuorimiehentie, 2002.
2. Cabri A., Griffiths M.: *Earned value and agile reporting*, Agile Conference, Minneapolis, 2006.
3. Christensen D., Ferens D.: *Using Earned Value for Performance Measurement on Software Development Projects*, Air Force Institute of Technology, Ohio,1995.
4. Madachy R. J.: *Software process dynamics*, Wiley, New Jersey, 2008.
5. Project Management Institute: *A Guide to the Project Management Body of Knowledge (PMBoK)*,PMI, Newtown Square, 2004.
6. Project Management Institute: *Practice Standard for Earned Value Management*, PMI, Newtown Square, 2005.
7. 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.
8. Sulaiman T., Barton B., Blackburn T., *AgileEVM - Earned Value management in Scrum Projects*, Agile Conference, Minneapolis, 2006.
9. Sutherland J.; Schwaber K,: *Business object design and implementation*, OOPSLA '95 Workshop Proceedings, University of Michigan, 1995.
10. Takeuchi H., Ikujiro N.: *The New New Product Development Game*, Harvard Business Review, 1986.

Rozdział 2

Modelowanie wartości Earned Value w zarządzaniu projektami z wykorzystaniem podejścia Agile. Część 2 - Model symulacyjny

W niniejszym rozdziale, stanowiącym kontynuacje rozdziału pierwszego, zaproponowano podejście integrujące w ramach jednego modelu aspekty związane m.in. z: przebiegiem procesów wytwórczych według Scrum oraz kalkulacją zmiennych Agile-EVM. Autorzy zbudowali model symulacyjny i poddali go badaniu na podstawie danych zebranych przez siebie w trakcie realizacji licznych projektów IT. Uzyskane wyniki umożliwiły stwierdzenie, że zaproponowana metoda pozwala opracować modele symulacyjne, które mogą być doskonałym narzędziem służącym do badania przebiegu projektów IT zarządzanych w oparciu o metody zarządcze wywodzące się z ruchu Agile. Zaproponowany model symulacyjny stanowi własną propozycję autorów i uwzględnia oprócz elementów skupionych wokół aspektów związanych z kalkulacją AgileEVM również elementy dotyczące wpływu prawidłowego doboru członków zespołu na przebieg procesów produkcyjnych oraz inne zagadnienia znane z zarządzania projektami IT.

Metoda Earned Value (EVM) pozwala na skuteczną integrację zakresu, kosztów i czasu [6], stanowi więc nieocenione narzędzie wsparcia kierownika projektu w mierzeniu efektywności w projekcie. EVM jest jednak krytykowana w kontekście jej przydatności do wspomagania zarządzania projektami realizowanymi z wykorzystaniem koncepcji Agile. EVM zakłada, że projekt jest realizowany w oparciu o dobrze zdefiniowaną strukturę zadań do wykonania, która wynika ze znanego i z reguły niezmiennego zakresu projektu.

Założenia te zderzone z rzeczywistością projektów prowadzonych w oparciu o podejście zwinne często stanowią podstawę do odrzucenia EVM jako nie nadającej się do zastosowania w warunkach, w których zakres projektu jest niedookreślony, wymagania są zmienne, a wykonana praca często może wymagać jej częściowego powtórzenia (refaktoringu).

Autorzy uważają, że po dokonaniu pewnych modyfikacji metoda EVM nadaje się do zastosowania w badaniu postępów pracy w projektach realizowanych w oparciu o Scrum, XP i inne metodyki zwinne. Modyfikacje te zostały przedstawione w poprzednim rozdziale, stanowiącym część 1 - opis dla potrzeb symulacji [4]. W tym rozdziale autorzy przedstawili wyniki osiągnięte w drodze kontynuacji tam podjętych prac teoretycznych. Prace te kontynuowano z zamiarem weryfikacji poprawności przyjętych poprzednio hipotez. Dokonano tego w drodze budowy modelu symulacyjnego w środowisku Vensim i zbadania adekwatności uzyskiwanych za jego pomocą wyników. Badania te przeprowadzono na podstawie danych zebranych przez autorów w trakcie realizacji licznych projektów IT.

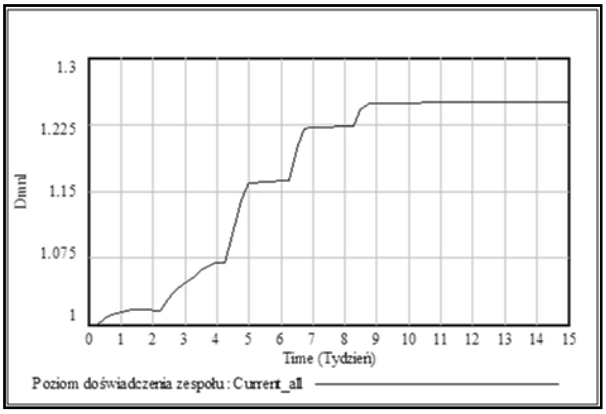
Zbudowany przez autorów model stanowi próbę podejścia całościowego do symulacji procesów wytwórczych w projektach prowadzonych w oparciu o metodykę Scrum. Jest on własną propozycją autorów i uwzględnia oprócz elementów skupionych wokół aspektów związanych z kalkulacją AgileEVM również elementy dotyczące wpływu prawidłowego doboru członków zespołu na przebieg procesów produkcyjnych oraz inne zagadnienia znane z zarządzania projektami IT. Nie stanowi on jednak gotowego rozwiązania i wymaga adaptacji i kalibracji do specyfiki wykorzystującej go organizacji.

2.1. Dodatkowe założenia do budowy modelu symulacyjnego

Oprócz przedstawionych w poprzednim rozdziale założeń dotyczących procesu wytwórczego wg metodyki Scrum oraz kalkulacji zmodyfikowanych wartości EVM autorzy podczas tworzenia modelu symulacyjnego uwzględnili również aspekty dotyczące wpływu prawidłowego doboru członków zespołu na przebieg procesów produkcyjnych oraz inne zagadnienia znane z zarządzania projektami IT, których podsumowanie zawarto w niniejszym punkcie.

2.1.1. Wpływ uczenia się zespołu na projekt

Uczenie się jest nieodłącznym elementem produkcji oprogramowania związanym z pozyskiwaniem nowych umiejętności i/lub wiedzy wraz postępem projektu. Członkowie zespołu projektowego wraz z upływem czasu i akumulowaniem doświadczenia stają się coraz bardziej produktywni. Zjawisko to jest dobrze udokumentowane w literaturze, na co wskazuje m.in. Madachy [1] podając szereg przykładów modeli krzywej uczenia się. W modelu opracowanym przez autorów została zastosowana krzywa [1] skalibrowana na podstawie danych zgromadzonych przez autorów podczas licznych projektów zarządzanych zgodnie z filozofią Agile. Na rysunku 2.1 przedstawiono przykładowy przebieg krzywej uczenia się wygenerowany podczas symulacji.

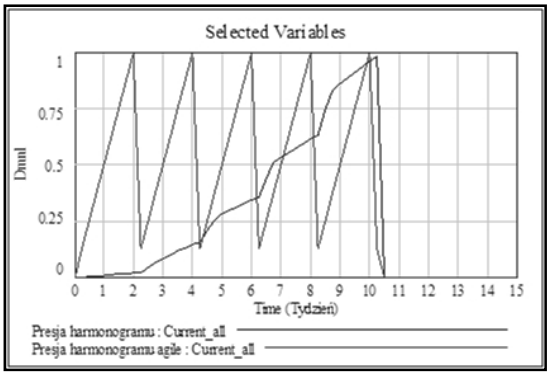


Rys. 2.1. Krzywa uczenia się dla projektu.
Źródło: opracowanie własne.

2.1.2. Wpływ presji harmonogramu na projekt

Presja harmonogramu jest kolejnym dobrze udokumentowanym zjawiskiem występującym w projektach związanych z produkcją oprogramowania. Pojawia się ona zazwyczaj w momencie, gdy czas pozostały do ukończenia projektu jest, w odczuciu członków zespołu, nieadekwatny do zakończenia pracy przy utrzymaniu wymaganych parametrów projektu takich jak czas,

koszt i jakość. Źródłem tego stanu mogą być m.in. błędy w estymacji pracochłonności, która w tego typu projektach zazwyczaj jest niedoszacowana, poprzez nieuwzględnienie lub marginalizację zdawałoby się błahych elementów systemu, które przeradzają się w toku prac w najbardziej kosztochłonne pozycje. Kolejnym powodem powstawania presji harmonogramu jest napływ zadań związanych z naprawą powstających w procesie produkcji błędów, których obsługa konsumuje czas zaplanowany na wykonywanie przez zespół zasadniczych zadań. Praca pod dużą presją harmonogramu silnie oddziałuje na proces produkcji oprogramowania, ponieważ wydatnie przyczynia się do wzrostu liczby powstających błędów. Projekty realizowane w sposób klasyczny (sekwencyjnie) charakteryzują się narastającą presją harmonogramu z silnym wzrostem pod koniec projektu. W projektach opartych na filozofii Agile zjawisko to rozłożone jest w czasie całego projektu, w taki sposób, że narasta cyklicznie w każdej iteracji, aby pod jej koniec osiągnąć wartości graniczne. Przykład zawierający ilustrację omawianego zjawiska w projektach klasycznych i opartych na podejściu zwinnym przedstawia rysunek 2.2.



Rys. 2.2. Presja harmonogramu w ujęciu klasycznym i Agile.
Źródło: opracowanie własne.

2.1.3. Wpływ typu zespołu na projekt

We współcześnie tworzonych modelach związanych z symulacją przebiegu projektów związanych z wytwarzaniem oprogramowania uwzględnia się szereg parametrów związanych z zespołem projektowym. W wyniku przeglądu literatury stwierdzono, że wpływ dopasowania cech osobowości członków

zespołu do pracy w środowisku projektowym opartym na fundamencie Agile jest całkowicie pomijany w stworzonych do tej pory modelach. W opinii autorów takie modele są pozbawione kluczowego elementu pozwalającego symulować i ocenić przebieg prac projektowych. Wniosek ten wywodzi się z obserwowanej przez autorów w praktyce prawidłowości wskazującej, że specyficzne połączenie różnych typów osobowości osób biorących udział w projekcie może prowadzić zarówno do poprawy jak i drastycznego obniżenia efektywności zbudowanego w ten sposób zespołu projektowego. Jest to o tyle istotne, że w projektach prowadzonych zgodnie z metodyką zwinną kluczową rolę odgrywa wysoka dyscyplina i „zgranie" zespołu, który jest zdolny do samoorganizacji, skupiony na wyznaczonym celu i zdeterminowany, aby go osiągnąć.

Nauki humanistyczne dostarczają pewnych podstaw teoretycznych, na bazie których zostały przez autorów stworzone dodatkowe elementy modelu symulacyjnego. Elementy te powstały w oparciu o teorię dotyczącą małych grup (ang. small groups theory). Ich szczegółowe omówienie wykracza poza ramy niniejszego opracowania, ograniczono się zatem do przedstawienia najważniejszej idei. Teoria małych grup dzieli zespoły na następujące kategorie:

1. gromada - zbiór przypadkowych ludzi działający skrajnie nieefektywnie;
2. zespół wrogi - zespół nastawiony wrogo do projektu, mogący dyskutować godzinami i niezdolny do efektywnego realizowania postawionych zadań;
3. zespół zbyt zgrany - zespół zdolny do rozwiązywania skomplikowanych problemów, którego członkowie zbyt wiele uwagi poświęcają na dyskusję zamiast na implementację pomysłów w praktyce;
4. zespół - grupa zdolna do realizacji pracy z normalną w danej organizacji efektywnością, którego członkowie kierują się zasadą, że każdy robi
   to, co należy do jego obowiązków i nic ponadto;
5. zespół zgrany - zespół zdolny do rozwiązywania skomplikowanych problemów i nastawiony na szybkie wypracowywanie rozwiązań i ich implementacji w praktyce.

Kategorię wyznacza się poprzez określenie jakie cechy osobowości dominują w badanym zespole. Dokonuje się tego na bazie modelu MBTI (May-

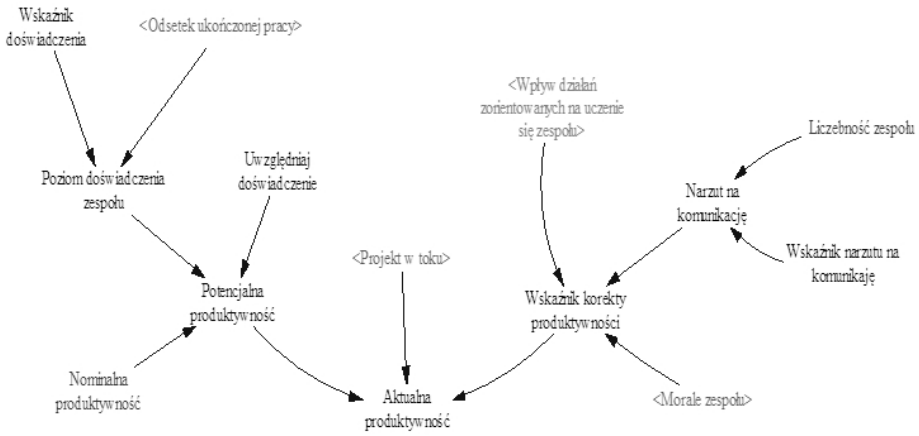
ers-Briggs Type Indicator), poprzez zidentyfikowanie preferencji badanych osób dotyczących sposobów: kierowania swojej energii, zbierania informacji, podejmowania decyzji oraz nastawienia do świata zewnętrznego. Każda z osób wchodzących w skład zespołu ma przypisany 4 literowy kod opisujący dany typ psychologiczny. Rozkład typów psychologicznych w badanym zespole determinuje rodzaj zespołu przypisując mu jedną z w/w kategorii. Każdej z nich przypisano z kolei odpowiednie współczynniki przekładające się na parametry modelu określające morale zespołu oraz doskonalenie pracy w grupie. Determinują one ogólną efektywność zespołu i wpływają tym samym na przebieg projektu. W modelu symulacyjnym uwzględniono również wpływ tzw. umiejętności miękkich lidera zespołu odwzorowujący jego zdolności do radzenia sobie z pojawiającymi się w zespole problemami natury pozatechnicznej.

2.2. Struktura modelu symulacyjnego

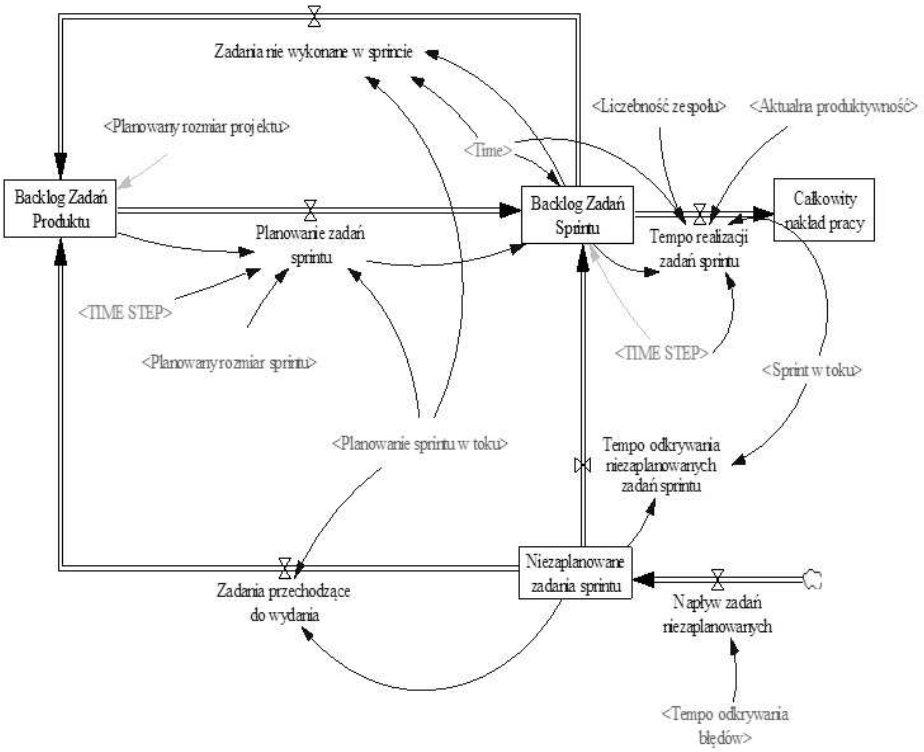
Model symulacyjny oparty m.in. na bazie założeń metodyki Scrum oraz AgileEVM [4] został zbudowany przy użyciu środowiska symulacyjnego Vensim. Ten kompleksowy model składa się z następujących części:

- przebieg procesu Scrum;
- planowanie i przepływ sprintu;
- zespół;
- produktywność;
- jakość;
- AgileEVM;
- zarządzanie czasem.

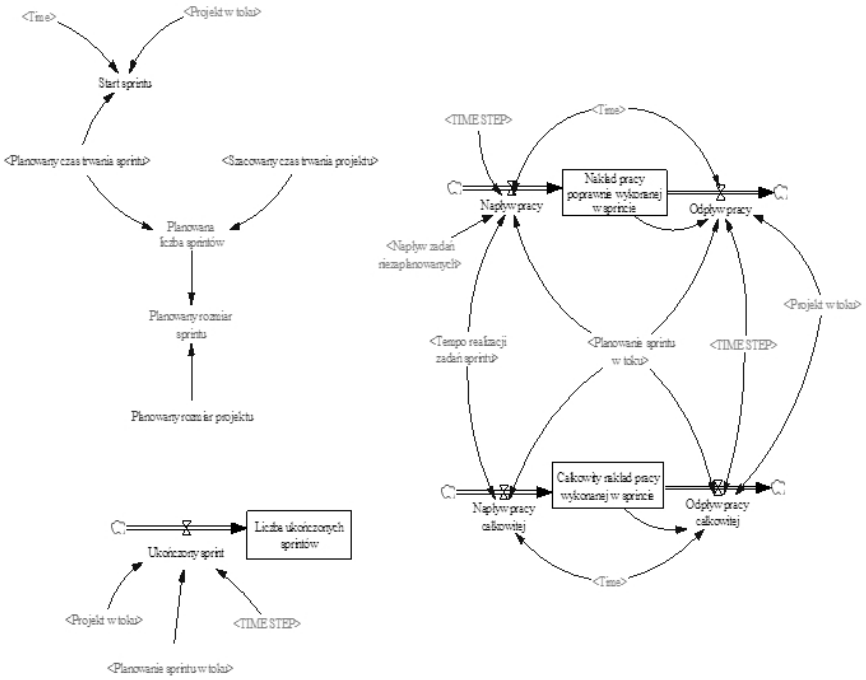
Wybrane fragmenty modelu istotne z punktu widzenia niniejszej publikacji zostały zaprezentowane na rys. 2.3-2.6.



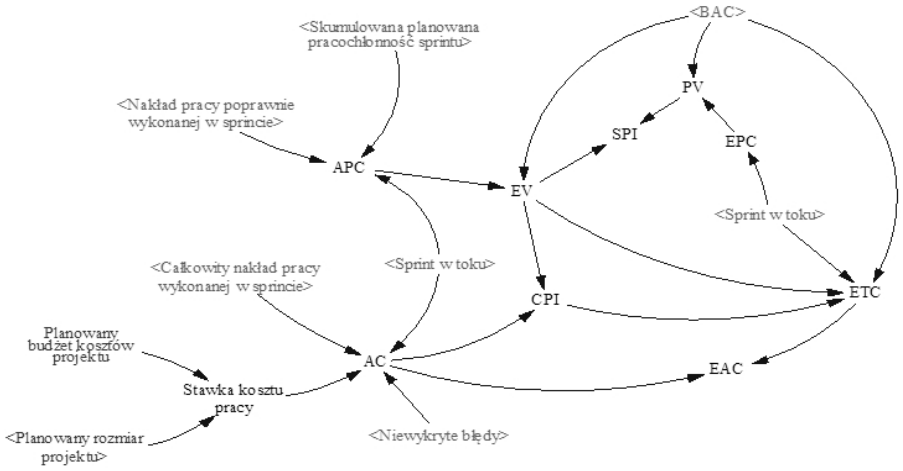
Rys. 2.3. Model symulacyjny - przebieg procesu Scrum.
Źródło: opracowanie własne.



Rys. 2.4. Model symulacyjny - produktywność zespołu.
Źródło: opracowanie własne.



Rys. 2.5. Model symulacyjny - planowanie sprintów.
Źródło: opracowanie własne.



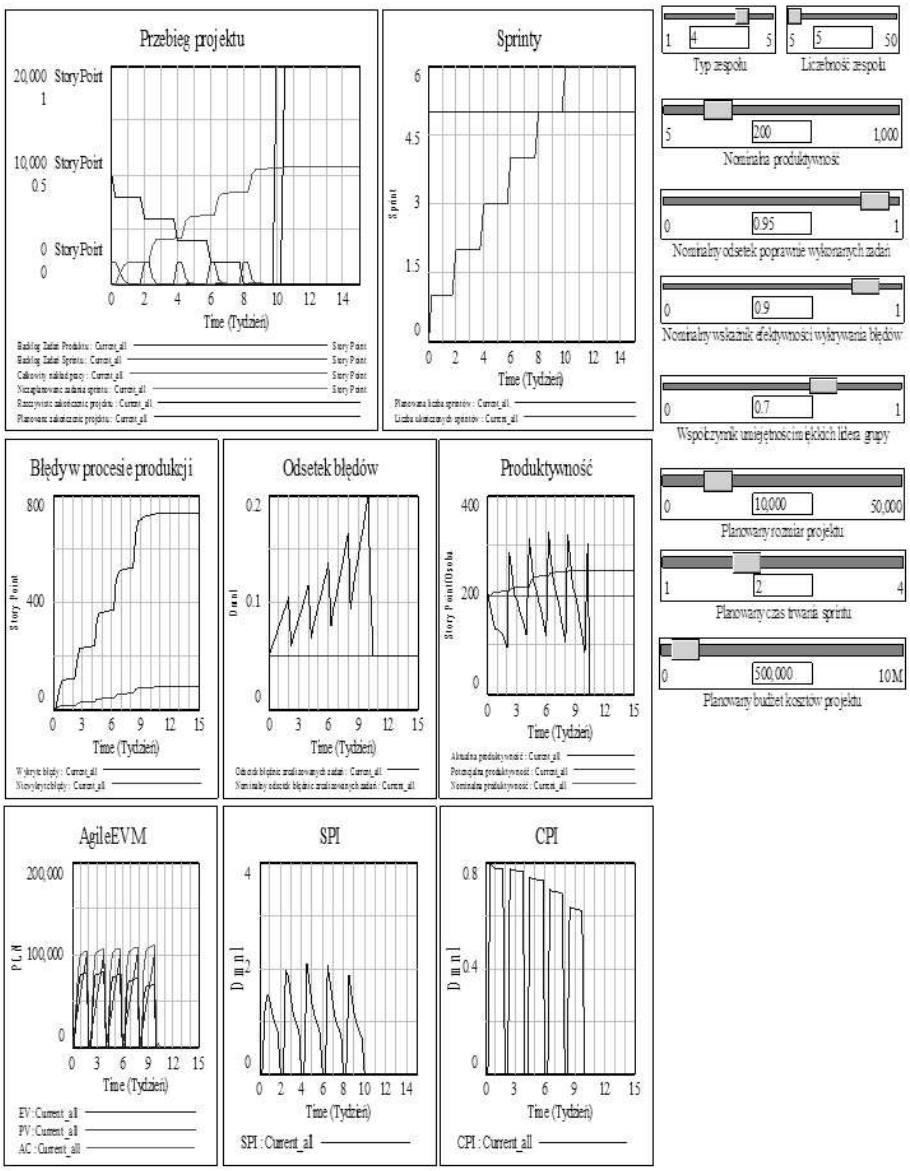
Rys. 2.6. Model symulacyjny - AgileEVM.
Źródło: opracowanie własne.

2.3. Uzyskane wyniki

Podczas oceny zbudowanego modelu symulacyjnego wzięto pod uwagę dwa projekty realizowane w oparciu o metodykę Scrum.

Projekt nr 1 był wartym pół miliona złotych przedsięwzięciem realizowanym przez zespół doświadczonych specjalistów dziedzinowych, w którym każdy starał się jak najlepiej wykonywać swoje zadania, jednak nie wnosił do projektu nic ponad stawiane mu wymagania. Projekt charakteryzował się złożonością około 10 000 story point i zaplanowano jego realizację w sprintach trwających dwa tygodnie każdy. Liderem została osoba posiadająca doświadczenie w kierowaniu zespołami wytwórczymi, która, z racji posiadanych cech osobowości i usposobienia, została dobrze przyjęta przez pozostałych współpracowników. Umiejętności miękkie lidera określono na poziomie 0,7 (w skali od 0 do 1, gdzie 0 oznacza brak umiejętności miękkich, a 1 wysokie zdolności do współdziałania w grupie i przewodzenia jej). Odsetek błędów w tym projekcie wynosił 5%, zaś skuteczność ich wykrywania 90%. Projekt zakończył się sukcesem, jednak jego realizacja wymagała przedłużenia czasu ostatecznej dostawy o około 2 tygodnie i naprawy niekrytycznych błędów po jego produkcyjnym uruchomieniu u klienta. Wyniki symulacji zamieszono na rys. 2.7.

Projekt nr 2 był wartym ponad dwa miliony złotych przedsięwzięciem realizowanym przez zespół doświadczonych specjalistów dziedzinowych. Osoby te zostały wybrane z różnych pionów przedsiębiorstwa dostarczających rozwiązań klientom z różnych sektorów. Każda z ośmiu osób wchodzących w skład zespołu miała bardzo duże problemy z pracą w nieznanej sobie grupie. Ponadto dużą część można scharakteryzować jako osoby skrajnie egocentryczne, które nie czuły się w żadnym stopniu odpowiedzialne za sukces projektu i doszukiwały się winy za kłopoty w projekcie u innych, tylko nie u siebie. Projekt charakteryzował się złożonością około 20 000 story point i zaplanowano jego realizację w sprintach trwających po dwa tygodnie. Liderem została osoba posiadająca doświadczenie w kierowaniu zespołami wytwórczymi, która, pomimo posiadanych cech osobowości, zdołała tylko częściowo przezwyciężyć trudności pozatechniczne w zespole. Umiejętności miękkie lidera określono na poziomie 0,9.

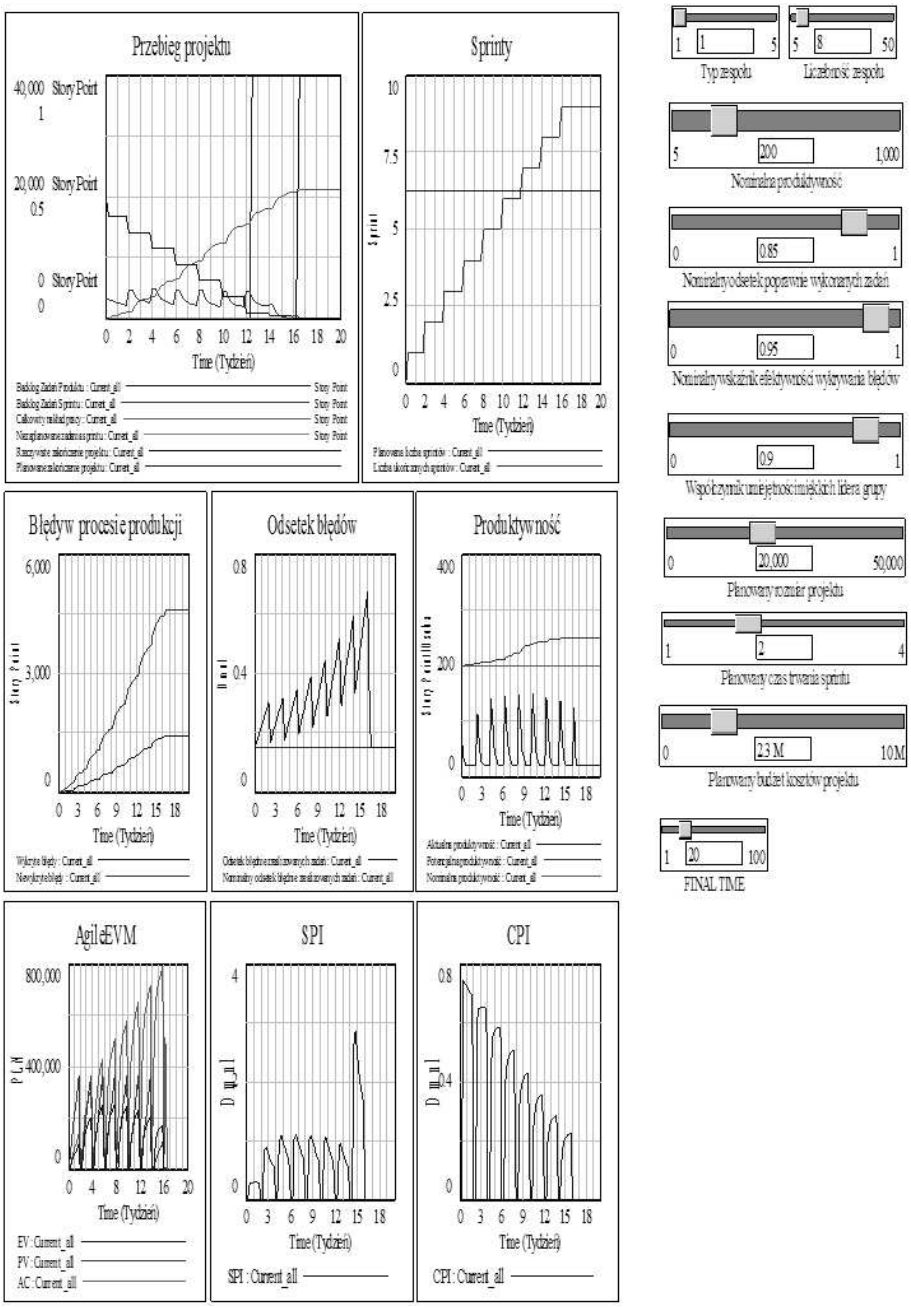


Rys. 2.7. Wyniki symulacji oraz główne parametry dla projektu nr 1.
Źródło: opracowanie własne.

Odsetek błędów w tym projekcie wynosił 15%, zaś skuteczność ich wykrywania 95%, z powodu zaangażowania zewnętrznego zespołu zapewniania jakości. Projekt zakończył się około 6 tygodni po czasie, w wyniku słabej pro-

duktywności i niskiej jakości pracy jego koszty wzrosły znacząco, przez co nie uzyskano satysfakcjonującej marży. Wyniki symulacji zamieszono na rysunku 2.8.

Uzyskane wyniki zostały poddane analizie zgodnie z zaproponowaną przez autorów metodą badania adekwatności modeli symulacyjnych opartą o badanie statystycznego dopasowania danych generowanych na wyjściu modelu do danych historycznych zarejestrowanych w wyniku obserwacji systemów (obiektów) w rzeczywistości [7]. Na podstawie przeprowadzonych badań i porównania uzyskanych wyników symulacji z danymi pochodzącymi z opisanych powyżej rzeczywistych projektów, można stwierdzić, że zbudowany przez autorów model dość dobrze oddaje rzeczywistość. Ocenę jego adekwatności przeprowadzono z wykorzystaniem zaproponowanej przez autorów funkcji decyzyjnej opartej o wartości statystyk niezgodności Theil'a [7]. Na tej podstawie można stwierdzić, że wskaźniki AgileEVM w sposób satysfakcjonujący odwzorowują stan projektu i na ich podstawie możliwe byłoby podejmowanie odpowiednich akcji zmierzających do skorygowania prowadzonych prac. Opracowany model stanowi pierwszą próbę kompleksowego podejścia do symulacji dynamiki procesów wytwórczych zarządzanych w oparciu o podejście zwinne i wymaga dalszego doskonalenia oraz kalibracji. W związku z tym będzie on nadal rozwijany przez autorów w toku dalszej pracy naukowej w tym zakresie. Niemniej jednak autorzy uważają, że już na tym etapie dojrzałości stanowi on dobre narzędzie dla kierownika projektu do prowadzenia analiz typu "co jeśli" na etapie planowania projektów i budowania zespołów.



Rys. 2.8. Wyniki symulacji oraz główne parametry dla projektu nr 2.
Źródło: opracowanie własne.

Podsumowanie

Dynamiczny rozwój technologiczny spowodował zbliżenie się dostawców rozwiązań informatycznych (IT) do środowiska biznesowego, które wymaga obecnie od IT wsparcia we wszystkich aspektach związanych z zarządzaniem informacją w ich organizacjach. Skutkuje to coraz większą liczbą kompleksowych projektów mających na celu wdrażanie coraz bardziej skomplikowanego oprogramowania. Projekty te coraz częściej prowadzone są zgodnie z podejściem Agile, które charakteryzuje się skupieniem uwagi na jak najsprawniejszym dostarczeniu odbiorcy działającego oprogramowania oraz na ścisłej współpracy z klientem i reagowaniu na jego potrzeby. Rodzi to potrzebę adaptacji istniejących narzędzi wypracowanych do zarządzania projektami, aby można je było wykorzystywać również w projektach prowadzonych według metodyk zwinnych [4]. W niniejszym rozdziale podjęto próbę adaptacji metody Earned Value do uwarunkowań wynikających z metodyki Scrum. Zaproponowano podejście integrujące aspekty związane m.in. z: przebiegiem procesów wytwórczych według Scrum oraz kalkulacją zmiennych AgileEVM w ramach jednego modelu symulacyjnego. Zbudowany model autorzy poddali badaniu na podstawie danych zebranych przez siebie w trakcie realizacji licznych projektów IT. Uzyskane wyniki pozwoliły stwierdzić, że zaproponowana metoda umożliwia opracowanie modeli symulacyjnych, które mogą być doskonałym narzędziem służącym do badania przebiegu projektów IT zarządzanych w oparciu o metody zarządcze wywodzące się z ruchu Agile. Opracowany model stanowi pierwszą próbę kompleksowego podejścia do symulacji dynamiki procesów wytwórczych zarządzanych w oparciu o podejście zwinne, nie stanowi zatem gotowej recepty ani rozwiązania, lecz pomaga kierownikowi projektu ocenić wpływ różnych czynników na przebieg planowanego lub realizowanego przedsięwzięcia.

Bibliografia

1. Madachy R. J.: *Software process dynamics*, Wiley, New Jersey, 2008.
2. Project Management Institute: *A Guide to the Project Management Body of Knowledge (PMBoK)*,PMI, Newtown Square, 2004.
3. Project Management Institute: *Practice Standard for Earned Value Management*, PMI, Newtown Square, 2005.
4. Protasowicki T., Stanik J.: *Modelowanie wartości Earned Value w zarządzaniu projektami z wykorzystaniem podejścia Agile, Część 1 - Opis dla potrzeb symulacji*, 2013, w druku.
5. 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.
6. Sulaiman T., Barton B., Blackburn T., *AgileEVM - Earned Value management in Scrum Projects*, Agile Conference, Minneapolis, 2006.

Rozdział 3

Metody pozyskiwania danych do oceny produktywności poszczególnych członków zespołu projektowego w projektach informatycznych

W rozdziale przeanalizowano i przedstawiono przykładowe źródła danych dla bieżącej oceny produktywności na podstawie zmierzonych dotychczasowych dokonań członków zespołu projektowego oraz zaprezentowano wskaźniki do bieżącej oceny produktywności członków zespołu projektowego definiowane, jako funkcje danych pozyskiwanych z opisanych wcześniej źródeł. W dalszej kolejności opisane zostały algorytmy wyznaczania produktywności na podstawie danych historycznych zebranych i opracowanych na przykładzie rzeczywistych projektów informatycznych.

Kierownik projektu, w projektach informatycznych, zwykle dysponuje zespołem złożonym zarówno z pracowników doświadczonych jak i pracowników początkujących. Nie pozostaje więc bez znaczenia dla realizacji projektu fakt, które zadania zostaną przydzielone któremu pracownikowi. Dobry dobór wykonawców do zadań może zaowocować znaczącym skróceniem czasu realizacji całego przedsięwzięcia.

W praktyce wiedza kierownika projektu na temat produktywności poszczególnych członków zespołu projektowego rośnie wraz z czasem realizacji przedsięwzięcia. Jeżeli kierownik będzie potrafił zmierzyć produktywność pracowników, to jest w stanie tak przebudować harmonogram w kolejnych etapach realizacji projektu, żeby osiągać optymalne rezultaty. Dlatego niezwykle istotne jest zapewnienie kierownikowi projektu narzędzi, dzięki którym możliwe będzie szybkie, zautomatyzowane wyznaczanie produktywność danego pracownika.

Dzięki zastosowaniu opisanych w rozdziale metod i narzędzi informatycznego wspomagania oceny produktywności zespołu w projektach informatycznych możliwe jest określenie jakości prac wykonywanych przez poszczególnych pracowników, a więc ich produktywności. Wyznaczona produktywność, w dalszej kolejności, znajduje zastosowanie przy planowaniu prac projektowych w sposób zapewniający minimalizację czasu potrzebnego na wykonanie zadań projektowych przez zespół pracowników o różnych kwalifikacjach.

3.1. Zakres informacyjny danych gromadzonych podczas realizacji projektu

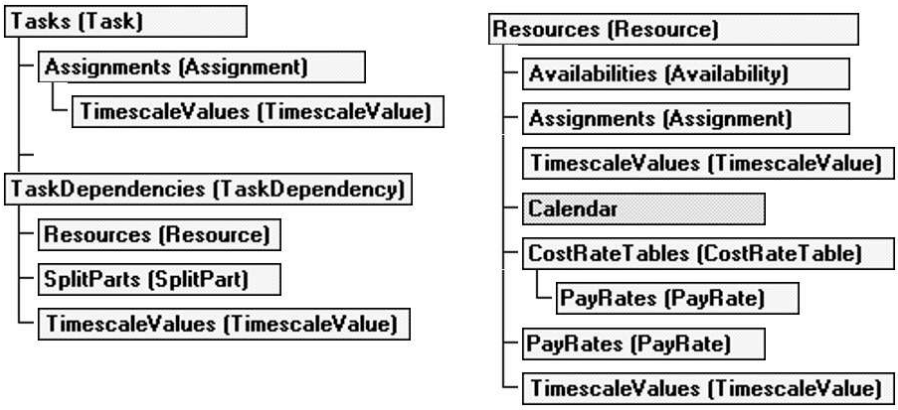
Podczas realizacji projektu informatycznego kierownik projektu posługuje się różnorodnymi narzędziami, które pozwalają mu na planowanie, organizowanie i monitorowanie projektu.

W zakresie planowania kierownik projektu korzysta z oprogramowania takiego jak Microsoft Project lub Primavera. Do organizowania prac bardzo często używany jest system dekretacji zadań. W tym zakresie różne organizacje używają różnych systemów. Niektóre posługują się systemem CRM, inne wykorzystują do tego celu WorkFlow. Ostatnio coraz częściej używane są systemy klasy BPM (Business Process Management).

W efekcie użytkowania tego typu systemów, wraz z postępami projektu gromadzą się dane historyczne stanowiące bardzo cenne źródło do oceny wielu parametrów projektu.

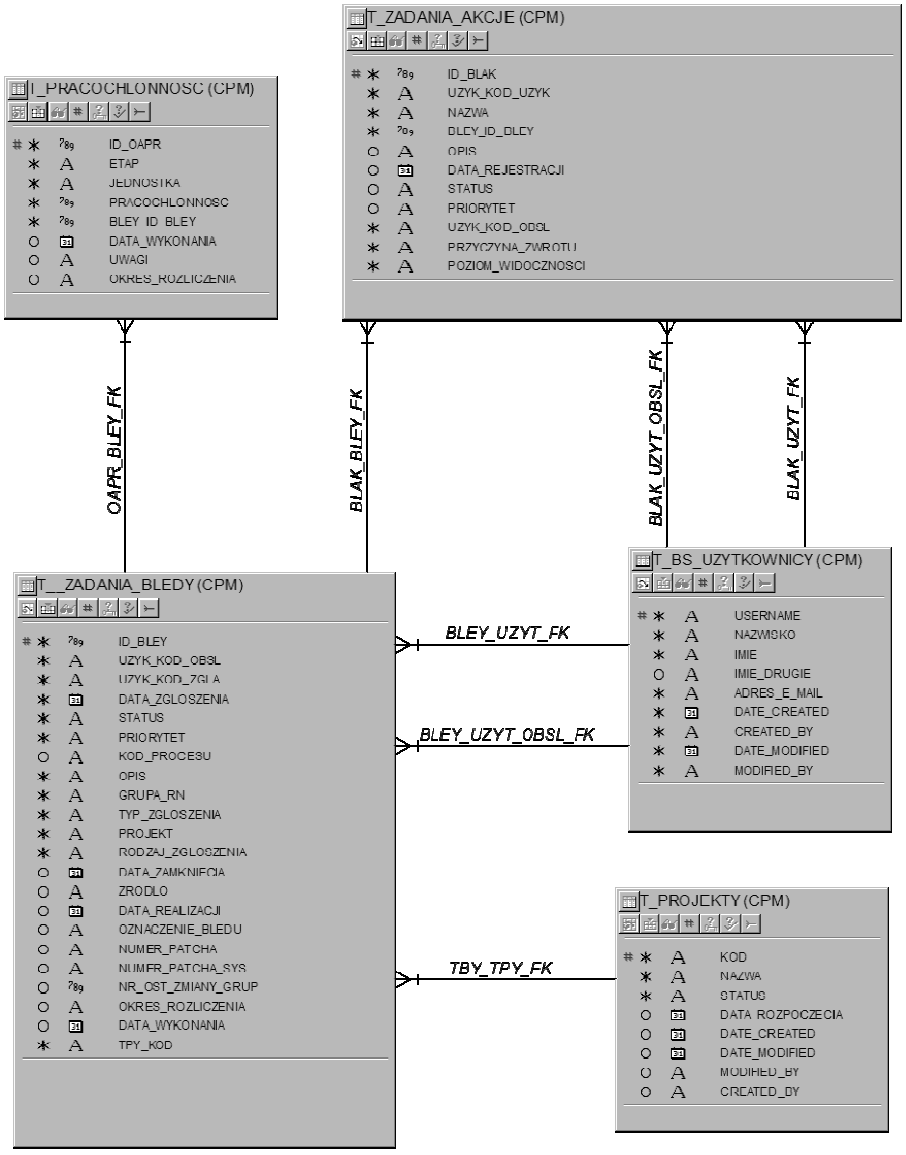
Zakres informacyjny systemu do harmonogramowania można prześledzić na podstawie Microsoft Project (rys. 3.1). System zawiera dane o wszystkich zadaniach projektowych oraz szereg atrybutów opisujących te zadania. Pozwala na bieżąco aktualizować terminy realizacji poszczególnych zadań i śledzić ich odstępstwa od planu bazowego [1]. Prowadzi również rozliczenie zasobów zużywanych do realizacji zadań. Na podstawie danych zawartych w systemie do harmonogramowania można planować kolejne etapy projektów. Ze względu na brak automatyzacji oraz bezpośredniego sprzężenia systemu z systemami, za pomocą, których produkuje się oprogramowanie nie ma możli-

wości śledzenia na bieżąco postępów i jakości prac. Wszystkie dane muszą być wpisywane do systemu harmonogramującego oddzielnie.



Rys. 3.1. Zakres informacyjny programu do zarządzania projektami – Microsoft Project – zadania i zasoby.
Źródło: Dokumentacja programu MS Project.

Dekretacja zadań w wielu organizacjach odbywa się po prostu za pomocą poczty elektronicznej. Zaawansowane zespoły projektowe dysponują często dedykowanym systemem pozwalającym na opisywanie wszystkich zadań projektowych, przekazywanie ich do wykonania, rejestrowanie czasu pracy poświęconego na realizację zadania, przekazywanie produktu cząstkowego do kolejnej fazy realizacji, zatwierdzanie produktu cząstkowego, rejestrowanie zmian w odniesieniu do elementów systemu, obsługę zmian w oprogramowaniu oraz poprawy błędów, jako kolejnych zadań projektowych. Dysponując takim systemem dekretacji pracy (SDP) kierownik projektu jest w stanie śledzić na bieżąco postępy prac oraz odpowiednio przetwarzając dane wyznaczać interesujące go parametry systemu, tak w odniesieniu do całego zespołu projektowego jak i konkretnych grup zadaniowych a nawet poszczególnych pracowników.



Rys. 3.2. Zakres informacyjny wybranego systemu do zarządzania produkcją oprogramowania.

Model danych przykładowego systemu do zarządzania produkcją oprogramowania oraz zarządzania poprawkami błędów na bazie platformy Aurea BPM [2] przedstawia rys. 3.2.

W odróżnieniu do systemu harmonogramowania system dekretacji pracy może być w bardzo dużym zakresie powiązany z systemami do produkcji oprogramowania. Pozwala to na bieżąco, automatycznie obliczać rozmiar poszczególnych elementów produkowanego systemu, np. metodą punktów funkcyjnych lub punktów obiektowych. Daje również możliwość dokładnego obliczania czasu potrzebnego na zrealizowanie zadania oraz określania jakości, z jaką zadanie zostało zrealizowane.

3.2. Wskaźniki produktywności dla projektów IT

Poniżej przedstawiono możliwe miary produktywności dla poszczególnych członków zespołu projektowego w projektach informatycznych.

Tab. 3.1. Miary produktywności dla poszczególnych członków zespołu projektowego.

Stanowisko	**Miary produktywności**
Analityk	(+) Liczba przeanalizowanych i opisanych modułów systemu w miesiącu (+) Liczba opracowanych analitycznie zmian w systemie w miesiącu (–) Liczba błędów w systemie, wynikających z powodu niespójnej, nielogicznej lub niedokładnej, zbyt pobieżnej analizy
Projektant	(+) Liczba zaprojektowanych i opisanych modułów systemu w miesiącu (+) Liczba opracowanych analitycznie zmian w systemie w miesiącu (–) Liczba błędów w systemie, wynikających z powodu niespójnego, nielogicznego lub niedokładnego projektu
Programista	(+) Liczba zbudowanych i przekazanych do testów modułów systemu w miesiącu (+) Liczba względna (w sensie rozmiaru oprogramowania) wykonanych poprawek systemu w miesiącu (–) Liczba błędów w stworzonym oprogramowaniu wykrytych na testach cząstkowych (–) Liczba błędów w stworzonym oprogramowaniu wykrytych na testach systemowych

Stanowisko	Miary produktywności
	(–) Liczba błędów w stworzonym oprogramowaniu wykrytych przez Użytkownika
Tester	(+) Liczba przetestowanych modułów systemu (w sensie rozmiaru oprogramowania) na miesiąc (–) Liczba błędów, które nie zostały ujawnione na etapie testowania i zostały wykryte przez Użytkownika w stosunku do rozmiaru przetestowanego oprogramowania
Wdrożeniowiec	(+) Liczba stworzonych przypadków testowych (stanowiących część składową scenariusza testów) na miesiąc (+) Ilość danych testowych przygotowanych do przeprowadzenia testów oprogramowania na miesiąc (+) Liczba stron dokumentacji użytkownika na miesiąc (–) Liczba błędów systemu wykrytych przez Użytkownika a niemożliwych do stwierdzenia na podstawie przygotowanych przypadków testowych w stosunku do rozmiaru oprogramowania

Źródło: opracowanie własne.

Weźmy zmienną losową c_z oznaczającą czas, w jakim pracownik zrealizuje zadanie z. Podczas wykonywania projektu informatycznego zapisywane są dane o momentach rozpoczynania i kończenia zadań, a tym samym znany jest czas wykonania każdego z nich. W związku z tym przy wyznaczaniu produktywności do dyspozycji jest zbiór realizacji zmiennej losowej c_z.

Dla każdego zadania projektowego określona jest pracochłonność wzorcowa – c_z^w. Jest to podstawowy parametr zadania. Pracochłonność określa kierownik projektu na podstawie wiedzy o wielkości zadania. Metody badania rozmiaru oprogramowania są powszechnie znane i używane w planowaniu projektów informatycznych. Oszacowana za pomocą tych metod wzorcowa pracochłonność jest zmienną losową.

Produktywność definiuje się, jako stosunek wzorcowego czasu realizacji zadania do faktycznego czasu realizacji zadania [3]:

$$\beta_z = \frac{c_z^w}{c_z}.$$

Produktywność, jako iloraz zmiennych losowych, jest zmienną losową. Znane są realizacje historyczne tej zmiennej losowej. Znając realizacje historyczne określa się wartość oczekiwaną, na podstawie której wyznacza się harmonogram zadań na kolejne etapy projektu. Jako estymator wartości oczeki-

wanej $E(\beta_z)$ przyjęto średnią arytmetyczną niezalenych pomiarów wielkości β_z:

$$E(\beta_p) = \frac{1}{Z_p}\sum_{z=1}^{Z_p}\beta_z, p \in P,$$

gdzie Z_p – liczba zadań zrealizowanych przez pracownika p w rozpatrywanym okresie czasu.

3.3. Metody pozyskiwania danych do oceny produktywności

Poniżej przedstawiono sposób wyboru danych za pomocą zapytania w języku SQL oraz otrzymane rezultaty [4].

Zapytania dla projektu budowy i wdrożenia systemu informatycznego dla 3 funduszy emerytalnych w Rosji, przy realizacji którego pracowało 54 pracowników przez 2,5 roku, realizując 8117 zadań trzech typów, przedstawiają się następująco:

```
SELECT uzyk_kod_uzyk,
  ROUND(AVG(pracochlonnosc/czas_realizacji), 2) produktyw-
nosc
FROM
  (SELECT ba.BLEY_ID_BLEY,
    ba.UZYK_KOD_UZYK,
    ba.PRIORYTET,
    ba.data_rozpoczecia,
    ba.data_zakonczenia,
    ROUND(((data_zakonczenia - data_rozpoczecia)*24 -
(TRUNC(data_zakonczenia)-TRUNC(data_rozpoczecia))*16))
czas_realizacji,
    pr.pracochlonnosc
  FROM
    (SELECT b1.BLEY_ID_BLEY,
      b1.UZYK_KOD_UZYK,
      b1.PRIORYTET,
      (SELECT MAX(b2.data_rejestracji)
      FROM t_bledy_akcje b2
      WHERE b2.bley_id_bley   = b1.bley_id_bley
      AND b2.data_rejestracji < b1.data_rejestracji
      ) data_rozpoczecia,
      b1.DATA_REJESTRACJI data_zakonczenia
    FROM T_BLEDY_AKCJE b1
    WHERE b1.nazwa = 'POP'
```

```
    ) ba,
    t_pracochlonnosc pr
  WHERE pr.bley_id_bley    = ba.bley_id_bley
  AND pr.etap              = 'I'
  AND ba.data_rozpoczecia IS NOT NULL
  )
WHERE czas_realizacji > 0
GROUP BY uzyk_kod_uzyk /
```

Otrzymane w wyniku powyższych zapytań rezultaty zawarto w tab. 3.2.

Tab. 3.2. Wyniki wyznaczania produktywności w realnym projekcie 1 (CPM).

Lp	PRACOWNIK	PRODUKTYWNOŚĆ
1	Mirosław_A	0,111
2	Janusz_B	2,584
3	Joanna_B	0,144
4	Anna_B	1,812
5	Oksana_B	2,832
..	…	…
50	Wojciech_S	1,405
51	Piotr_S	0,642
52	Olena_T	0,238
53	Marzena_W	3,029
54	Sylwester_Z	0,999

Zapytania dla projektu budowy i wdrożenia systemu zarządzania procesami biznesowymi dla firmy informatycznej w Polsce, przy realizacji którego pracowało 6 pracowników przez 12 miesięcy, realizując 128 zadań trzech typów, przedstawiają się następująco:

```
SELECT name,
  ROUND(SUM(workload)/SUM(exec_time), 2) productivity
FROM
  (SELECT vh.rt_proc_nr,
    vh.proc_name,
    vh.task_name,
    ROUND((SUM(vh.exec_time)/60)+1) exec_time,
    vh.name
    ||'_'
```

```
    || SUBSTR(vh.ev_user,1,1) name,
    vh.ev_desc,
    tp.id,
    tecna.awf_runtime.get_parameter(tp.id, 'SYS_WORKLOAD')
workload,
    vh.ev_user
  FROM tecna.v_history vh,
    tecna.t_rt_processes tp
  WHERE vh.rt_proc_id  = tp.id
    AND vh.task_name     = 'Rozwiązanie problemu'
    AND vh.ev_desc       = 'Zatwierdzono'
  GROUP BY vh.rt_proc_nr,
    vh.proc_name,
    vh.task_name,
    vh.name
    ||'_'
    || SUBSTR(vh.ev_user,1,1),
    vh.ev_desc,
    tp.id,
    tecna.awf_runtime.get_parameter(tp.id,
'SYS_WORKLOAD'),
    vh.ev_user
  ORDER BY vh.rt_proc_nr
  )
GROUP BY name
/
```

Otrzymane w wyniku powyższych zapytań rezultaty zawarto w tab. 3.3.

Tab. 3.3. Wyniki wyznaczania produktywności w realnym projekcie 2 (Aurea)

Lp.	PRACOWNIK	PRODUKTYWNOŚĆ
1	Andrzej_W	0,59
2	Andrzej_H	1,18
3	Piotr_O	1,63
4	Tomasz_T	0,5
5	Michał_W	0,7
6	Kamil_K	2,24

Otrzymane rezultaty dla projektu budowy i wdrożenia systemu obsługi procesów przetwarzania danych dla firmy sektora bankowego w Polsce, przy

realizacji którego pracowało 5 pracowników przez 2 lata, realizując 157 zadań trzech typów, zawarto w tab. 3.4.

Tab. 3.4. Wyniki wyznaczania produktywności w realnym projekcie 3 (Aurea)

Lp.	PRACOWNIK	PRODUKTYWNOŚĆ
1	Andrzej_W	0,81
2	Andrzej_H	2,08
3	Piotr_O	0,79
4	Adam_K	2,25
5	Tomasz_T	1,42

Podsumowanie

Na przykładzie dwóch systemów przydziału i rozliczania pracy zostały zebrane i przeanalizowane dane z małych i dużych projektów informatycznych. Zaproponowane sposoby obróbki danych pozwoliły na wyznaczenie produktywności członków zespołu projektowego. Osiągnięte rezultaty potwierdzają, iż systemy dekretacji pracy stanowią cenne źródło do oceny produktywności pracowników.

Bibliografia

1. Wilczewski S.: *MS Project 2010 i MS Project Server 2010. Efektywne zarządzanie projektem i portfelem projektów*, Helion, 2011.
2. Tecna Sp. z o.o.. System Aurea BPM: www.aurea-bpm.com, 2012.
3. Kosieradzka A., Lis S.: *Produktywność. Metody analizy oceny i tworzenia programów poprawy*, Oficyna Wydawnicza Politechniki Warszawskiej, 2000.
4. Waszkowski R.: *Metody i narzędzia informatycznego wspomagania oceny produktywności zespołu w projektach informatycznych*, WAT, 2012.

Rozdział 4

Projekt-czynnik-decyzja. Badanie czynników decyzyjnych w projektach informatycznych i ich wpływu na powodzenie projektów

Autorzy rozdziału wskazują na konieczność pogłębienia, usystematyzowania oraz skodyfikowania wiedzy teoretycznej i praktycznej dotyczącej zarządzania projektami informatycznymi w obszarze podejmowania decyzji kierowniczych. Zakres podejmowanych decyzji przez kierowników podlega stałemu przyrostowi. Kierownicy projektów podejmują decyzje już nie tylko w ramach klasycznego trójkąta ograniczeń, na który składa się harmonogram, budżet i zakres projektu, ale również w ramach wielu innych problemów występujących podczas realizacji projektów. Zauważona potrzeba wyodrębnienia i zdefiniowania wszystkich czynników występujących w projektach ma doprowadzić do identyfikacji skończonego zestawu czynników decyzyjnych oraz samych decyzji, a następnie do określenia ich wzajemnych korelacji.

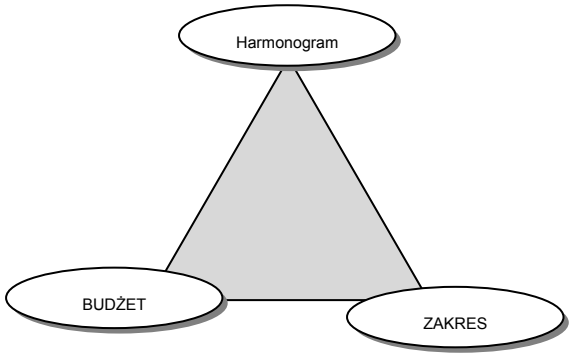
Zarządzanie projektami stanowi jedno z podstawowych wyzwań współczesnej gospodarki. Dynamicznie zmieniające swe struktury organizacje stanowiące trzon tej gospodarki implikują zmiany zarówno w procesach, jak i strukturach zarządzania. Elementem tych zmian jest odmienne od dotychczasowego podejście do wytwarzania nowoczesnych, szybko zmieniających się (wraz z wymaganiami rynku) produktów. W ramach tych zmian klasyczne struktury organizacyjne zastępowane są przez dedykowane zespoły projektowe. Wraz z rosnącym znaczeniem projektów dla funkcjonowania organizacji, nabiera znaczenia problematyka zarządzania nimi. W szczególności problematyka ta jest widoczna w branży IT, w której złożone (trudne do wstępnego zdefiniowania), niepowtarzalne projekty obarczone wysokim ryzykiem wyko-

nawczym są bardzo często realizowane przez rozproszone (trudne do zarządzania) zespoły projektowe [por. 2].

Publikowane przez firmy konsultingowe wyniki badań wykazują, że ponad połowa projektów informatycznych jest nieudana z racji przekroczenia harmonogramu czy budżetu [1]. Wskazuje się także, że jedną z podstawowych przyczyn nieudanych projektów stanowią błędy kierowników projektów. Błędy te są konsekwencją niewłaściwie realizowanych procesów zarządzania projektami, co w przypadku projektów informatycznych sprowadza się do błędnych decyzji dotyczących takich obszarów, jak np. dobór metod zarządzania projektami informatycznymi do specyfiki danych projektów, dobór zespołu, tworzenie relacji z klientem w projekcie informatycznym [8].

Zainteresowanie taką tematyką wynika zatem z faktu, że zakres podejmowanych decyzji przez kierowników podlega stałemu przyrostowi. Kierownicy projektów podejmują decyzje już nie tylko w ramach klasycznego trójkąta ograniczeń, na który składa się harmonogram, budżet i zakres projektu (rys. 4.1).

Podczas realizacji projektów pojawia się również wiele innych problemów, m.in. kwestie kadrowe związane z konstruowaniem (i późniejszym zarządzaniem) zespołów projektowych, decyzje dotyczące komunikacji z klientem i sposobu przeprowadzania analiz biznesowych czy decyzje wynikające z typowej dla projektów informatycznych zmienności (nie tylko wymagań, ale warunków np. technologicznych), w których prowadzony jest projekt.



Rys. 4.1. Trójkąt warunków brzegowych w projekcie informatycznym
Źródło: opracowanie własne na podstawie [1].

Potrzeba wyodrębnienia i zdefiniowania wszystkich czynników występujących w projektach mogłaby pozwolić na dokładniejszy opis dziedziny zarządzania projektami. Może ona stanowić jednocześnie wstęp do poszukiwania kluczowych czynników sukcesu gwarantujących powodzenie projektów. Celem autorów jest wobec tego pogłębienie, usystematyzowanie oraz skodyfikowanie wiedzy teoretycznej i praktycznej dotyczącej zarządzania projektami informatycznymi w obszarze podejmowania decyzji kierowniczych.

4.1. Czynniki decyzyjne a decyzje projektowe

4.1.1. Nowy trójkąt ograniczeń w projektach informatycznych

Jak wspomniano, podstawowym kryterium sukcesu projektu informatycznego jest jego zakończenie w ramach klasycznych ograniczeń projektowych rozumiane jako zrealizowanie projektu w ramach [1]:

- ustalonego harmonogramu,
- przyjętego budżetu,
- w pełnym założonym zakresie.

Nie oznacza to jednak, że decyzje kierowników zapadają wyłącznie w ramach tych trzech obszarów. Klasyczny trójkąt ograniczeń powstaje bowiem w następstwie wcześniejszych działań i decyzji podejmowanych przez kierownika projektu. Zakres projektu określany jest przeważnie na podstawie rozmów z klientem, harmonogram ustala się na podstawie oceny możliwości zasobowych firmy realizującej projekt informatyczny (dostawca), budżet jest najczęściej wypadkową oceny pracochłonności zespołu realizującego projekt po stronie dostawcy oraz możliwości finansowych klienta. Stąd też można wysnuć wniosek, iż zanim kierownik projektu zacznie podejmować decyzje odnośnie realizacji projektu w ramach klasycznego trójkąta ograniczeń, zapada szereg decyzji wcześniejszych - dotyczących klienta czy zespołu.

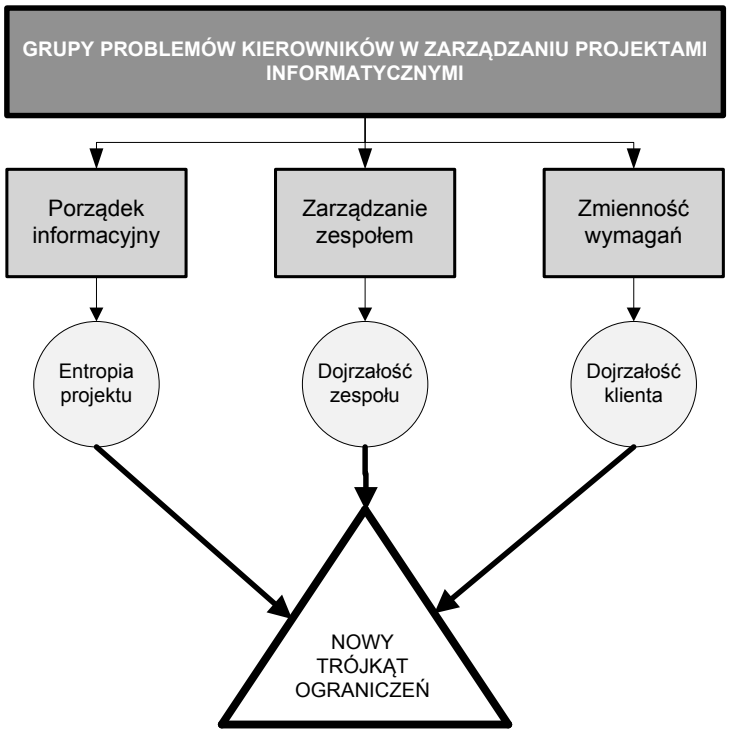
Biorąc pod uwagę dobre praktyki zawarte w metodach zarządzania projektami (takimi jak np. RUP czy SCRUM) oraz konieczność uzupełnienia klasycznego trójkąta ograniczeń dodatkowym zestawem zmiennych decyzyj-

nych, autorzy zaobserwowali trzy obszary projektu niezbędne do przeprowadzenia analizy przed jego rozpoczęciem.

Obszary te można określić mianem kategorii problemów decyzyjnych (zwanych dalej obszarami decyzyjnymi), względem których kierownicy podejmują decyzje. Do takich grup zaliczono:

- klienta i jego dojrzałość,
- zespół realizujący projekt i dojrzałość tego zespołu,
- zmienność i uporządkowanie informacji niezbędnych do realizacji projektu (entropia projektu).

Nowy trójkąt ograniczeń przedstawiono na rysunku 4.2.



Rys. 4.2. Obszary decyzyjne jako nowy trójkąt ograniczeń w projektach informatycznych.
Źródło: opracowanie własne.

4.1.2. Dojrzałość klienta, czyli świadomość własnych wymagań

Pierwszy z tych obszarów obejmuje procesy realizowane przy udziale klienta w projektach informatycznych. Klient jest najczęściej (choć nie zawsze) głównym odbiorcą produktu powstałego w wyniku projektu informatycznego. Dekompozycja metod zarządzania projektami (wykonana przez autorów na potrzeby tworzenia adaptacyjnego podejścia do zarządzania projektami) [5,6] pozwoliła na ekstrakcję tych dobrych praktyk zarządzania, które dotyczą wyłącznie kierowników projektów. Fragment dekompozycji jednej z metod zarządzania projektami z punktu widzenia zarządzania relacjami z klientem przedstawiono poniżej.

Tab. 4.1. Dekompozycja metody RUP na potrzeby decyzji kierowniczych.

DOBRA PRAKTYKA	UZASADNIENIE
Definiowanie produktów pracy jako wejścia/wyjścia do/z zadań	Pozwala kierownikowi na przekazanie zespołowi oczekiwań klienta co do zrealizowania określonego zadania.
Definiowanie precyzyjnych ról projektowych	Pozwala kierownikowi zorganizować zespół w zależności od określonych działań. Kontakty z klientem powierza się jednej osobie (analityk biznesowy)
Stosowanie modeli wysokopoziomowych	Pozwala kierownikowi zaprezentować produkt projektu informatycznego w sposób bardziej zrozumiały przez klienta.

Źródło: opracowanie własne.

Inne metody zarządzania projektami również zawierają i zalecenia związane z obecnością klienta. W wielu wypadkach zaleca się wręcz włączenie go w procesy wytwórcze i zarządcze (*Scrum, XP Programming*), co implikuje podejmowanie określonych decyzji przez kierowników projektów. Takie postrzeganie projektu gdzie konieczna jest ocena dojrzałości klienta i podjęcie decyzji odnośnie jego miejsca w projekcie wymaga dogłębnego poznania stanu klienta. Zadaniem kierowników jest więc podjęcie określonych decyzji zmierzających do budowania właściwych relacji z klientem i właściwego

umiejscowienia klienta w ramach realizowanego projektu. Brak znajomości specyfiki klienta może doprowadzić do błędnych decyzji odnośnie sposobu realizacji projektu. W przypadku, gdy klient niedojrzały będzie często zmieniał wymagania, kierownik projektu powinien podjąć określone decyzje zmierzające do uelastycznienia procesów wytwórczych, czyli dążyć do stosowania metod zwinnych. Takie decyzje będą jednak możliwe po dokładnej analizie wszystkich czynników związanych z klientem implikujących zmienność. Stąd proponowana przez autorów konieczność tworzenia pełnego zbioru czynników, w ramach których kierownicy podejmują decyzje związane zarówno z klientem jak i dalszymi pracami projektowymi [7].

4.1.3. Dojrzałość zespołu dostawcy, czyli właściwy przebieg prac

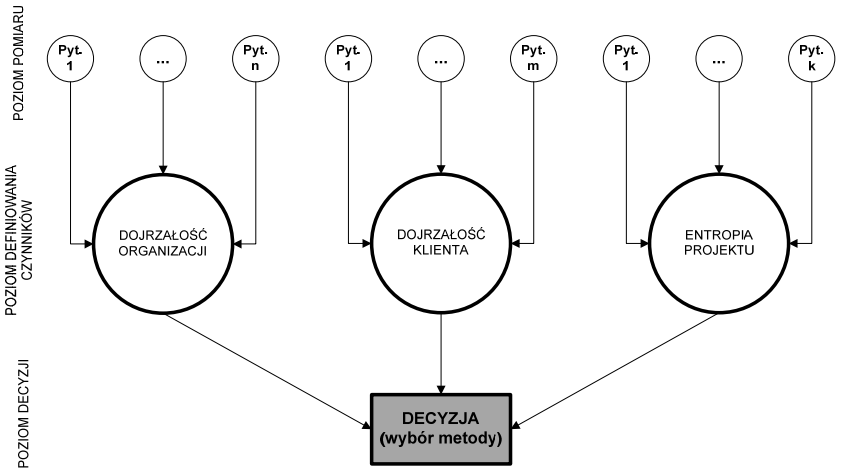
Drugim obszarem procesów decyzyjnych omawianych szeroko we wszystkich metodach zarządzania projektami są procesy zespołowe organizacji dostawcy. To zespół dostawcy pod nadzorem kierownika projektu odpowiada za właściwy przebieg prac, za to jak i kiedy wykonywane są określone prace. Wydaje się zatem słuszne, aby kierownik projektu przed rozpoczęciem realizacji projektu dobrze rozpoznał możliwości realizacji projektu przez zespół, którym kieruje. Rozpoznanie to powinno odbywać się pod kątem kompetencji projektowych, ale również pod kątem kompetencji związanych z pracą grupową. Taka analiza pozwoli kierownikowi projektu na właściwe przydzielenie każdemu pracownikowi odpowiedniej roli w projekcie. Właściwy przydział ról zwiększa wydajność prowadzonych prac. Biorąc pod uwagę również to, że decyzje odnośnie doboru metody zarządzania projektami informatycznymi albo wyboru dobrych praktyk powinny być uzależnione od stanu zespołu dostawcy. Ponadto świadomość własnego poziomu dojrzałości pozwala na lepsze dopasowanie dobrych praktyk związanych z zarządzaniem zespołem. Niezbędne wydaje się więc określenie zbioru możliwych parametrów (czynników) opisujących zespół dzięki czemu kierownik może podejmować trafne decyzje dotyczące mechanizmów motywacji czy przydziału zadań.

4.1.4. Entropia projektu, czyli dążenie do uporządkowania

Trzeci obszar procesów decyzyjnych wiąże się z decyzjami kierowników projektów odniesionych do poziomu informacji w realizowanym projekcie informatycznym, jak również dotyczących oceny jego złożoności, zmienności wymagań oraz niepewności otrzymania końcowego produktu. W przypadku większości projektów informatycznych realizowanych w sposób przyrostowy wymagania zmieniają się w czasie, a ostateczny produkt nie jest dokładnie określony, stąd precyzowanie zadań i procesów wytwarzania jest trudne. Brak wystarczającej wiedzy o stanie projektu implikuje niepewność kierownika, utrudniając podejmowanie decyzji planistycznej (krótko- jak i długofalowych). W związku ze zjawiskiem niepełnego (niewystarczającego) poziomu informacji w projekcie informatycznym kierownicy projektów powinni dopasowywać takie dobre praktyki do bieżącego stanu projektu. Uznano zatem, że problemy natury informacyjnej w projektach mierzone będą miarą ich uporządkowania, co nazwano negentropią projektu [6].

4.1.5. Problematyka dojrzałości w kontekście badań naukowych

Przedstawione w poprzednim rozdziale grupy problemów kierowniczych zostały zidentyfikowane w trakcie badań prowadzonych przez Zakład Zarządzania Technologiami Informatycznymi funkcjonujący w Politechnice Gdańskiej od 2006 roku. Dotychczasowe badania zespołu nad problemami zarządczymi w projektach informatycznych prowadziły do usprawnienia aktywności związanych z doborem i stosowaniem metod zarządzania projektami (takich jak RUP, SCRUM, PRINCE2 etc.) oraz systemowego podejścia do zarządzania projektami z uwzględnieniem powyższych grup problemów (bazując na takich koncepcjach jak CMMI czy architektura korporacyjna).



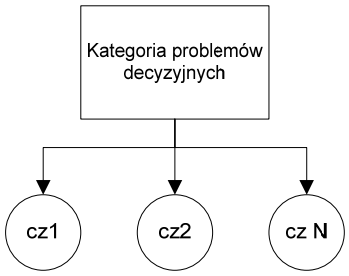
Rys. 4.3. Kierunek badań nad dojrzałością w projektach informatycznych.
Źródło: opracowanie własne.

Dotychczasowy stan badań nad kompletnym rozwiązaniem uwzględniającym problematykę dojrzałości wymusza jednak konieczność uszczegółowienia pewnych jej aspektów. Naturalnym krokiem jest w związku z tym ekstrakcja (dekompozycja) wspomnianych trzech głównych obszarów decyzyjnych (klient, zespół, entropia) do postaci pojedynczych, określonych i nazwanych czynników implikujących kluczowe decyzje w projektach. Należy również określić siłę (wagę) każdej z wyodrębnionych implikacji.

4.2. Wpływ czynników decyzyjnych na katalog decyzji

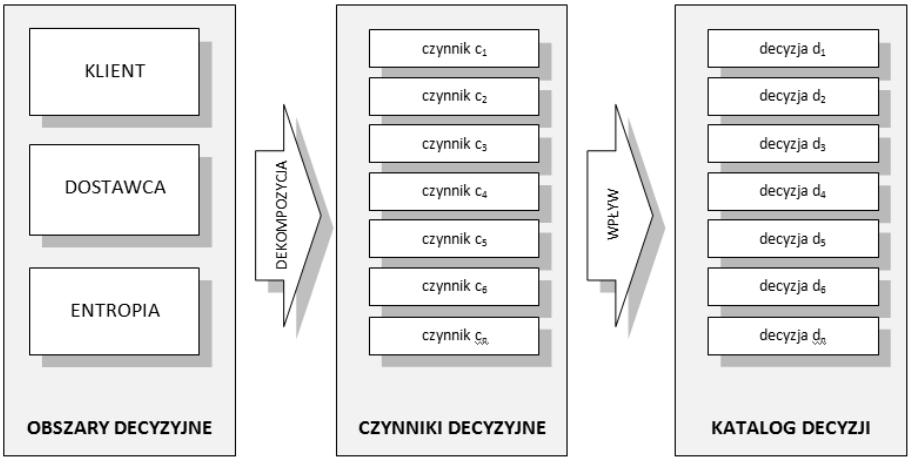
Należy zauważyć, że każdy z opisanych obszarów decyzyjnych składa się z szeregu pomniejszych czynników mających pośredni lub bezpośredni wpływ na późniejsze decyzje projektowe. Stąd też stworzenie kompletnego (pełnego) zbioru czynników decyzyjnych pozwoliłoby na uporządkowanie wiedzy w zakresie podejmowania decyzji projektowych.

Taki zbiór może powstać poprzez dekompozycję (analizę przyczynową) wskazanych problemów decyzyjnych do pomniejszych (pojedynczych) czynników.



Rys. 4.4. Dekompozycja obszarów decyzyjnych do listy czynników decyzyjnych.
Źródło: opracowanie własne.

Zdefiniowanie czynników decyzyjnych oraz korelacji między nimi a decyzjami w projektach a także zbadanie wpływu tych czynników na powodzenie projektów pozwoliłoby przede wszystkim na zgłębienie świadomości o mechanizmach decyzyjnych występujących w projektach oraz pozwoliłoby kierownikom projektów na rozszerzenie wiedzy z zakresu zarządzania projektami [9].



Rys. 4.5. Powiązania między obszarami decyzyjnymi, listą czynników decyzyjnych i ich wpływ na konieczność podjęcia określonych decyzji projektowych.
Źródło: opracowanie własne.

Na podstawie doświadczeń autorów można zatem postawić następujące tezy:

- istnieje skończony zestaw czynników, które są istotne z punktu widzenia trzech wcześniej wymienionych grup problemów decyzyjnych (dojrzałości klienta, dostawcy i entropii).
- podejmowanie decyzji w projektach zależy w różnym stopniu (korelacja) od tych czynników.

Na tym etapie można uznać, że wyniki tej części badań będą stanowiły dwuwymiarową macierz, której przykład przedstawiono w tabeli 4.2.

Tab. 4.2. Przykładowe korelacje między czynnikami decyzyjnymi a zidentyfikowanymi decyzjami projektowymi

	c_1	c_2	c_n
d_1	0,4	0,01	0,03
d_2	0,1	0,9	0
d_3	0	0	0,4

c_n – czynnik decyzyjny, d_n – decyzja projektowa

Wartości w powyższej tabeli zostały wprowadzone podglądowo, gdyż na tym etapie nie ma jeszcze badań pozwalających choćby na szacunkowe określenie siły relacji między decyzjami a czynnikami projektowymi. Mimo to, w oparciu o doświadczenia własne oraz wcześniejsze badania autorów, można wymienić przykładowe czynniki z różnych kategorii (np. osobowe, technologiczne, organizacyjne) oraz decyzje projektowe, by pokazać jak różne one mogą być pod względem wzajemnej siły wpływu.

Przykładowe czynniki (c_n):

- dotyczące klienta i jego dojrzałości:
  - (całkowity/częściowy) brak doświadczeń w realizacji projektów z danym klientem,
  - brak (jakichkolwiek/części) kompetencji technicznych związanych (pośrednio/bezpośrednio) z dziedziną projektu,

- dotyczące dostawcy i jego dojrzałości:

  - pojawienie się nowego członka w zespole (po stronie klienta/wewnątrz zespołu wytwórczego) (przed rozpoczęciem/w trakcie trwania/przed zakończeniem) projektu,
  - (nagłe/narastające) wystąpienie konfliktu w zespole (między dwoma członkami/wieloma), który (dezorganizuje/utrudnia/całkowicie uniemożliwia) prace,
- dotyczące negentropii projektu:
  - zmiana (rozszerzenie/redukcja) wymagań (kluczowych/mniej istotnych) przez klienta,
  - awaria sprzętu (stacji roboczej/serwera).

Przykładowe decyzje projektowe (d_n):

- zatrudnić dodatkowego członka zespołu (w roli analityka/programisty/itd.),
- (skrócić/wydłużyć) cykle pracy,
- zrealizować (rozpoznanie/zakup) narzędzi CASE,
- pozyskać (darmowe/komercyjne) oprogramowanie wspierające zespół w procesach (budowy aplikacji/generowania dokumentacji, pozyskiwania wymagań/itd.),
- dodać (własnymi siłami/kupując usługę zewnętrzną) do istniejącego oprogramowanie nową (niezbędną/przydatną) funkcjonalność,
- zastąpić członka zespołu (zatrudnić nową osobę/dobrać z dostępnych), który (chwilowo/na dłuższy czas/na zawsze) przestał być do dyspozycji (z powodu choroby/zwolnienia/itd.),
- zmienić metodykę prowadzenia bieżącego projektu (na mniej/bardziej elastyczną).

4.3. Projekt zarządzany świadomie

Każdy projekt informatyczny ma swoją indywidualną specyfikę. Doświadczenia uzyskane przez autorów upoważniają jednak do stwierdzenia, iż można określić istotne części wspólne upoważniające do uogólnień. Na bazie wiedzy szczegółowej planuje się uzyskać wyniki badań zasadne dla przeciętnego projektu informatycznego,

Wyniki proponowanego rozwiązania mogą mieć znaczenie dla rozwoju problematyki zarządzania projektami w następujących aspektach:

- Zrealizowanie proponowanego projektu pozwalającego na definicję wszystkich czynników decyzyjnych zwiększa świadomość kierowników projektów o możliwych działaniach (aktywnościach) w trakcie realizowanych projektów.
- Zarazem opracowanie katalogu decyzji umożliwi lepsze planowanie/zarządzanie projektami informatycznymi.
- Zwrócenie uwagi kierowników na aspekty nieformalne (czyli inne niż formalne ustalenia budżetowe, harmonogramowe, zakresowe); są to: dojrzałość klienta czy zespołu realizującego projektu. Tym samym tworzone są nowe warunki brzegowe (nowy trójkąt ograniczeń) w ramach którego funkcjonuje (w szczególności: podejmuje decyzje) kierownik projektu.
- Uzyskanie możliwości diagnozowania przyczyn projektów nieudanych oraz analizowania sukcesów projektów zakończonych zgodnie z założeniami.
- Uzyskanie możliwości prognozowania skutków podjętych przez kierownika działań w danych realiach projektowych.

Proponowane podejście do zarządzania projektami wykracza poza tradycyjne postrzeganie tej dziedziny, wskazuje bowiem na dodatkowe aspekty poza koncentracją na zadaniach, sprawach finansowych oraz harmonogramie.

Podsumowanie

Mimo jakościowych i trudnych do mierzenia aspektów projektu informatycznego autorzy podejmują próbę podejścia w pełni ilościowego – celem jest kwantyfikacja (dobór i nadanie miar) wszystkich istotnych czynników w projekcie IT. Badania takie mają w zamierzeniu rozbudowanie środowisko podejmowania decyzji projektowych poprzez zwiększenie świadomości o czynnikach decyzyjnych i zachodzących między nimi korelacjach. Na tym etapie naturalnie trudno mówić o wymiernych efektach ewentualnego wdrożenia opracowanego rozwiązania. Mimo to można zaryzykować wniosek, iż

wykorzystanie wyników pracy autorów może pośrednio lub bezpośrednio doprowadzić do redukcji ryzyka projektowego, a więc obniży koszty związane z przedsięwzięciami nieudanymi i może stanowić podstawę do zasilania środowisk symulacji projektów informatycznych.

Bibliografia

1. Philips J.: *Zarządzanie projektami IT*, Helion 2007.
2. Schwaber K.: *Sprawne zarządzanie projektami metodą Scrum*, Microsoft, 2004.
3. Sommerville I.: Inżynieria oprogramowania, Wydawnictwa Naukowo-Techniczne, 2003.
4. Chrapko M.: *O zwinnym zarządzaniu projektami*, Helion 2012.
5. Orłowski C., Kowalczuk Z.: *Modelowanie procesów zarządzania technologiami informatycznymi*, PWNT 2012.
6. Orłowski C., Ziółkowski A.: *Supporting Software Project Management Processes Using the Agent System*, Knowledge-Based and Intelligent Information and Engineering Systems: KES 2010.
7. Czarnecki A., Orłowski C., Ziółkowski A., *Validation of an Agent and Ontology-based Information Technology Assessment System,* [w:] Cybernetics and Systems. - Vol. 41, Iss. 1, 2010.
8. Henderson-Sellers B., Gorton I.: *Agent-based Software Development Methodologies. International Conference on Object-Oriented Programming, Systems, Languages and Applications*, OOPSLA, Seattle USA, 2002.
9. Murch R., *Project management. Best practices for IT Professionals.* Prentice Hall PTR, Nowy Jork, USA, 2001.

Rozdział 5

Strategie zarządzania ryzykiem w projektach

Zarządzanie ryzykiem w projektach jest jednym z 9-ciu obszarów aktywności w zarządzaniu projektami, zgodnie z powszechnie akceptowalną metodyką PMI. Działania związane z zarządzaniem ryzykiem mogą mieć charakter aktywny lub reaktywny. Wybór odpowiedniej strategii zależy od wielu elementów i uzależnione jest od wielkości i typu projektu. Podejście aktywne związane jest z ponoszeniem określonych kosztów analizy i zapobiegania ryzyku, które wcale nie musi wystąpić w procesie realizacji. Reaktywne oznacza optymistyczne oczekiwanie niewystąpienia zagrożeń i kosztuje znacznie mniej, chociaż w przypadku wystąpienia zagrożenia skutki mogą być bardzo poważne. W rozdziale przedstawiono uwarunkowania obu podejść do zarządzania ryzykiem.

Zarządzanie zagrożeniami zidentyfikowanymi w projektach może być realizowane na dwa zasadnicze sposoby [11]:

- reaktywnie,
- aktywnie.

Reaktywne reagowanie na zidentyfikowane zagrożenia polega na naprawieniu lub minimalizacji negatywnych skutków zagrożenia, które się zdarzy w trakcie realizacji zadań projektowych. Przy czym założeniem podstawowym jest optymizm polegający na tym, że mimo zidentyfikowania negatywnego zdarzenia zakładamy, że nie powinno ono wystąpić i nie podejmujemy żadnych działań zapobiegawczych, które mogłyby spowodować niewystąpienie lub ograniczenie negatywnego zdarzenia. Realizujemy zadania projektowe ze świadomością możliwości wystąpienia negatywnego zdarzenia i akcje naprawczą podejmujemy dopiero, gdy zdarzenie to wystąpi.

Aktywne podejście do zidentyfikowanych zagrożeń polega na podejmowaniu działań zapobiegawczych mających na celu niewystąpienie negatywnego zdarzenia, które stanowi zagrożenie dla sprawnej realizacji projektu. W tym przypadku założeniem jest nieuchronność wystąpienia negatywnego zdarzenia i podejmowane działania mają na celu ograniczenie prawdopodobieństwa wystąpienia lub ograniczenie negatywnych skutków wystąpienia.

W zależności od wielu różnych uwarunkowań realizacji projektu, cech osobistych kierownika projektu i specyficznych warunków realizacji projektu, podejmowana jest jedna z opisanych strategii zarządzania ryzykiem w projekcie. Każda z tych strategii ma określone wady i zalety i rozważania mające na celu podjecie decyzji, którą z tych strategii przyjąć w konkretnym projekcie, stanowią element zarządzania ryzykiem w projekcie.

5.1. Ryzyko w projekcie

Zgodnie z metodyką PMI[1] jednym z dziewięciu obszarów aktywności w zarządzaniu projektami jest zarządzanie ryzykiem. Ryzyko realizacji przedsięwzięć jest naturalnym uwarunkowaniem projektów wynikającym z niepewności związanej z upływającym czasem i zmianami zachodzącymi zarówno w obszarze działań projektowych jak i szeroko rozumianego otoczenia projektu. Niepewność dotycząca przyszłości jest szczególnie ważnym elementem projektów umiejscowionych w dynamicznie rozwijającej się nowej technologii teleinformatycznej. Szybki rozwój sprzętu i mocy obliczeniowej oraz technologii pracy z wykorzystaniem sprzętu informatycznego, technologie mobilne, technologie chmurowe znacząco zmieniają funkcjonalność rozwiązań biznesowych i powinny być odpowiednio wkomponowane w rozwiązania projektowe[2].

[1] Project Management Institute jest największą, światową organizacją, która zbiera, opracowuje i publikuje informacje o realizacji przedsięwzięć wyznaczając standardy zarządzania projektami. Metodyka PMI definiuje 9 głównych obszarów aktywności, które są podejmowane w zakresie zarządzania projektami [1].

[2] W szczególności może wystąpić przypadek nietrafnego rozwiązania informatycznego, czyli niespełniającego oczekiwań użytkownika lub wadliwego technologicznie, zwane ryzykiem informatycznym [7].

Działania realizowane w ramach aktywności mieszczących się w obszarze zarządzania ryzykiem nie stanowią działań produkcyjnych, mających wpływ na postęp realizacji projektu, ale stanowią istotny element aktywności projektowych, które decydują o końcowym sukcesie projektu i podlegają planowaniu i wpisaniu w harmonogram prac projektowych. Jednym z elementów całkowitego sukcesu projektu jest utrzymanie zdefiniowanych parametrów przedsięwzięcia a w szczególności kosztów prac realizowanych w projekcie w ramach budżetu, który został zdefiniowany w trakcie negocjacji warunków kontraktu na realizację przedsięwzięcia. Dyscyplina budżetowa jest istotna z punktu widzenia całego portfela projektów realizowanych w ramach strategii informatyzacji, gdyż załamanie budżetu jednego projektu może skutkować problemami w realizacji innych przedsięwzięć i realizacji strategii rozwoju firmy [9][4]. Koszty związane z wykonaniem prac projektowych powinny, więc mieścić się w budżecie projektu określonym na etapie podpisywania kontraktu.

Prace związane z szacowaniem budżetu koncentrują się głównie na zadaniach produkcyjnych związanych z wytwarzaniem produktu głównego lub wykonaniu usług związanych z jego powstaniem. Przeciwdziałanie zagrożeniom, które mogą wystąpić na różnych etapach prac projektowych, to domena zarządzania ryzykiem projektu. Koszt prac zmierzających do minimalizacji zidentyfikowanych zagrożeń stanowią też koszt projektu i pochłaniają jego budżet. Należałoby znaleźć odpowiedni balans pomiędzy działaniami produkcyjnymi a działaniami ochronnymi, tak, aby zachować sprawność realizacji projektu, mieć gwarancje sukcesu, ale utrzymać się w zdefiniowanym i wynegocjowanym budżecie projektu. Większość metodyk realizacji przedsięwzięć zwraca uwagę na prawidłową analizę kosztów i korzyści podejmowanych działań projektowych, niezależnie od tego, czego te działania dotyczą [5].

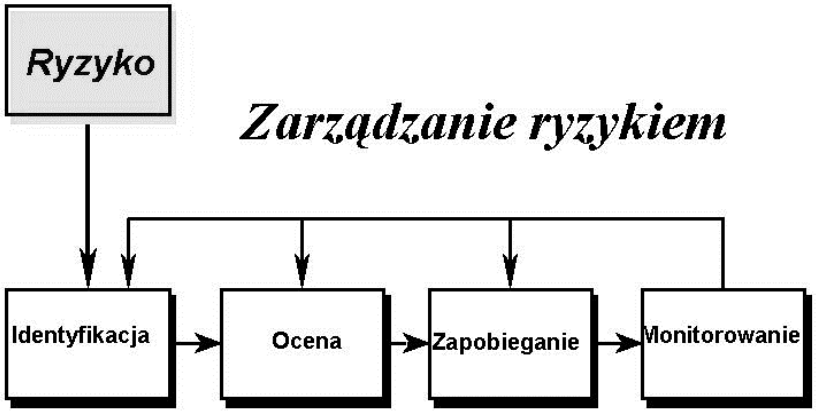
5.2. Zarządzanie ryzykiem w projekcie

Cechą wszystkich projektów jest niepewność, która wynika z nowatorskiego charakteru realizowanych przedsięwzięć. W przypadku projektów informatycznych niepewność jest szczególnie istotnym elementem wynikającym z wykorzystania bardzo szybko zmieniających się technologii, co dodatkowo

komplikuje szczególnie realizację projektów o długim cyklu wytwarzania. Duże projekty o długim cyklu realizacji z założenia są bardziej zagrożone. Szybkość zmian w rozwiązaniach informatycznych i wdrażanie do praktyki nowych rozwiązań jest nieporównywalne z innymi obszarami działań projektowych. Ponadto projekty informatyczne są najczęściej elementem innych złożonych struktur, które też podlegają zmianom w czasie realizacji przedsięwzięcia. Synergia zjawisk zachodzących w obszarze projektowym oraz otoczeniu projektu, stwarza duże ryzyko niepowodzenia całego przedsięwzięcia [6].

W praktyce działalności projektowej niepewność i ryzyko są ściśle ze sobą związane [3]. W literaturze najczęściej cytowane jest stwierdzenie, że "ryzyko jest zobiektywizowaną niepewnością wystąpienia niepożądanego zdarzenia" [12, s. 6]. Nowa, nie rutynowa sytuacja, jaka występuje w przypadku działań projektowych, jest przyczyną niepewności w podejmowaniu decyzji oraz dokonywanych szacunkach różnych elementów koniecznych w trakcie realizacji przedsięwzięcia. Nie rutynowy charakter prac projektowych, oraz wynikający z tego brak zgromadzonych doświadczeń i niemożność odwołania się do wcześniejszych prac, stwarzają przesłanki sprzyjające błędnym decyzjom i pomyłkom w szacowanych wartościach. Te elementy niepewności stanowią uwarunkowania realizacji każdego przedsięwzięcia i każdy kierownik projektu powinien umieć sobie z tym radzić w planowaniu działań projektowych. Zidentyfikowane uwarunkowania i fakty, które mogą stanowić istotne zagrożenie dla prawidłowej realizacji działań projektowych lub celów zdefiniowanych w projekcie, stanowią przedmiot rozważań zarządzania ryzykiem. Rozważania związane z ryzykiem dotyczą nieprzewidywalnej przyszłości. Analizujemy sytuację i zdarzenia, które jeszcze się nie zdarzyły, ale mogą mieć negatywny wpływ na nasze działania projektowe, jeśli zaistnieją w przyszłości. Ryzyko jest więc potencjalnym, niepożądanym zdarzeniem, które może spowodować, że cele realizowanego projektu nie zostaną osiągnięte lub zostanie znacząco zakłócony tok prac wykonawczych.

Proces zarządzania ryzykiem można przedstawić schematycznie tak jak na rys. 5.1.



Rys. 5.1. Model zarządzania ryzykiem [10].

W przedstawionym modelu zarządzanie ryzykiem składa się z cyklicznie powtarzanych działań:

- identyfikowania,
- oceny,
- zapobiegania,
- monitorowania.

Działania identyfikacji ryzyka mają na celu określenie, jakie zdarzenia lub fakty mogą być niepomyślne dla realizowanego projektu. Każde zidentyfikowane zagrożenie należy udokumentować, przedstawiając jego istotę oraz wszystkie warunki jego zaistnienia i przewidywane negatywne konsekwencje dla naszego projektu z określeniem obszarów oddziaływania w projekcie. Stosowane są różne metody identyfikacji, chociaż najczęściej stosowaną metodą są listy kontrolne powstałe w wyniku doświadczeń z innymi projektami, rozbudowane o własne przemyślenia i uwagi dotyczące realizowanego projektu.

Na ocenę zidentyfikowanego zagrożenia składają się:

- oszacowanie prawdopodobieństwa wystąpienia negatywnego zdarzenia,
- oszacowanie potencjalnych negatywnych skutków dla projektu lub strat.

W przypadku stworzonej listy zidentyfikowanych zagrożeń należy określić, które z nich są najważniejsze dla realizacji projektu, co często pro-

wadzi do przedstawienia rankingu zagrożeń branych pod uwagę w procesach zarządzania ryzykiem. Lista zidentyfikowanych zagrożeń, ułożona jest według ocenionej wagi wpływu na projekt. Często jest ona ograniczana w dalszych pracach tylko do pewnej grupy najważniejszych, najgroźniejszych zagrożeń.

Dla każdego branego pod uwagę zagrożenia, pozostającego na liście zagrożeń, należy przygotować odpowiedni plan zapobiegawczy lub zdecydować, że nie podejmujemy takich działań licząc na pomyślność losu i niewystąpienie zdarzenia. Decyzje o niepodejmowaniu działań przeciwdziałania zagrożeniu nie muszą wynikać z oceny prawdopodobieństwa ale powinny być poparte argumentami lub wynikać z przyjętej strategii zarządzania ryzykiem. Plan zapobiegania konkretnemu zagrożeniu jest składową planu zapobiegania ryzyku całego projektu. Działania zapobiegawcze mogą dotyczyć zmniejszeniu prawdopodobieństwa wystąpienia zagrożenia lub minimalizacji negatywnych skutków ich wystąpienia. Idealnym rozwiązaniem jest równoczesne minimalizowanie obu składowych oceny zagrożenia. Sama identyfikacja zagrożeń, ocena każdego z nich oraz zdefiniowanie działania nie powodują minimalizacji zagrożenia a jedynie dają nam świadomość ryzyka, jakie podejmujemy w procesie realizacji projektu.

Realizacja planu zapobiegania ryzyku powinna podlegać cyklicznemu monitorowaniu oraz ocenie skuteczności podejmowanych działań. Wszystkie działania należy odnosić do dokonanych ocen, trzeba też kontrolować zmienność poczynionych założeń i szacunków. Zmiany oceny prawdopodobieństwa wystąpienia niepożądanego zdarzenia czy szacowanych strat mogą wpłynąć na uaktualnienie planu zapobiegania ryzyku lub zmianie decyzji o podjętych lub zaniechanych działaniach. Wszelkie prace związane z realizacja planu zapobiegania ryzyku powinny być dokumentowane. Prace związane z zapobieganiem ryzykiem rozszerzają zakres aktywności podejmowanych przez zespół projektowy i stanowią dodatkową pracochłonność. Działania te powinny zostać wpisane w harmonogram prac projektowych, mimo że stanowią nieprodukcyjną część aktywności projektowych.

Działania związane z zarządzaniem ryzykiem należy powtarzać cyklicznie w całym cyklu życia projektu w ramach planowanych przeglądów ryzyka, zgodnie z opracowanym harmonogramem przedsięwzięcia. W miarę upływu czasu niektóre zagrożenia przestają być aktualne, ale mogą powstać nowe czynniki ryzyka, które zostaną zidentyfikowane w kolejnym cyklicznym prze-

glądzie. Zmieniają się warunki otoczenia, co może mieć wpływ na dokonane oszacowania, a wraz z nimi prawdopodobieństwo wystąpienia negatywnych zdarzeń i wysokość szacowanych strat. Przestrzeganie procedur związanych z zarządzaniem ryzykiem jest często warunkiem sukcesu końcowego realizowanego przedsięwzięcia.

5.3. Budżet projektu

Budżet projektu jest szacowany na podstawie zdefiniowanego zakresu prac projektowych. Szacunek kosztów realizacji poszczególnych zadań stanowi podstawę do wyliczenia kosztów ogólnych projektu. Nie jest to oczywiście prosta suma kosztów wykonania zdefiniowanych zadań, ale powiększona o koszty innych aktywności koniecznych do realizacji w ramach działań projektowych. Jednym z elementów kosztowych, które nie wynikają wprost z szacowania kosztów zadań projektowych, ale z innych aktywności projektowych, są koszty związane z zarządzaniem ryzykiem [8]. Działania związane z zarządzaniem ryzykiem składają się z aktywności zobrazowanych na rysunku 5.1. Każda z tych aktywności wymaga określonej pracochłonności, czyli poświęcenia określonej ilości czasu pracy wybranych pracowników zespołu projektowego, co przeliczane jest na złotówki i obciąża budżet projektu. Aktywności związane z zarządzaniem ryzykiem są, więc działaniami nieprodukcyjnymi, ale koniecznymi w procesie realizacji projektu i obciążającymi budżet.

Planując działania projektowe przewidujemy cyklicznie organizowane zebrania poświęcone zarządzaniu ryzykiem w projekcie. Zebrania te są jedynie spotkaniem mającym na celu identyfikacje zagrożeń oraz analizę stanu wcześniej zidentyfikowanych zagrożeń oraz określenie zakresu i odpowiedzialności za konkretne prace, które należy wykonać w ramach zarządzania ryzykiem. W szczególności w zależności od przyjętej strategii podejścia do zagrożeń są ustalane działania zapobiegawcze lub podejmowana jest decyzja o monitorowaniu zagrożenia ale nie podejmowaniu żadnych działań związanych z zagrożeniem licząc optymistycznie na niewystąpienie tego zagrożenia. Oczywiście podejście reaktywne jest tańsze i w określonym momencie prostsze, gdyż nie wymaga żadnej akcji i kosztów. Ewentualne koszty mogą wystąpić dopiero w momencie wystąpienia negatywnego zdarzenia.

Procedura oceny zidentyfikowanych zagrożeń zależy od ich ilości oraz na czym polega określone zagrożenie [2]. Ocenie podlega prawdopodobieństwo wystąpienia zdarzenia oraz potencjalna strata związana z wystąpieniem negatywnego zdarzenia. Oczywiście wielkości te są szacowane, ale nie w każdym przypadku można łatwo i dobrze oszacować te wartości. W niektórych przypadkach skomplikowanych zagrożeń wymaga to dodatkowych analiz i badań pomocniczych, które angażują czas pracowników na kolejne nieprodukcyjne działania. W zależności od istotności zidentyfikowanego zagrożenia szczegółowość i precyzja szacunków może być różna. W przypadku strategii reaktywnej można bardzo pobieżnie oszacować oba parametry lub zrezygnować z szacowania i zakładać, że zagrożenie to nie wystąpi.

Niezależnie od złożoności działań związanych z oceną poszczególnych zagrożeń, wraz z wzrostem ilości zidentyfikowanych zagrożeń, pracochłonność prac rośnie i koszt wykonania oceny, realizowany w cyklach przewidzianych sesjami zarządzania ryzykiem, uszczupla budżet projektu nie powodując przyrostu produktów będących przedmiotem działań projektowych. Niektóre założenia metodyczne zakładają ograniczenie ilości analizowanych i poddawanych ocenie zagrożeń, ograniczając dalsze prace związane z zarządzaniem ryzykiem tylko do najistotniejszych po wstępnej ocenie [11]. Niezależnie od przyjętej metodyki prac koszt oceny poszczególnych zagrożeń uszczupla budżet projektu. W przypadku stosowania strategii reaktywnej minimalizujemy ten koszt i oszczędzamy w danym momencie budżet projektu. W praktyce sensownym rozwiązaniem wydaje się ograniczenie szacunków do pobieżnej oceny parametrów zagrożenia i pominięcie zagrożeń o małym prawdopodobieństwie wystąpienia lub niskich potencjalnych stratach.

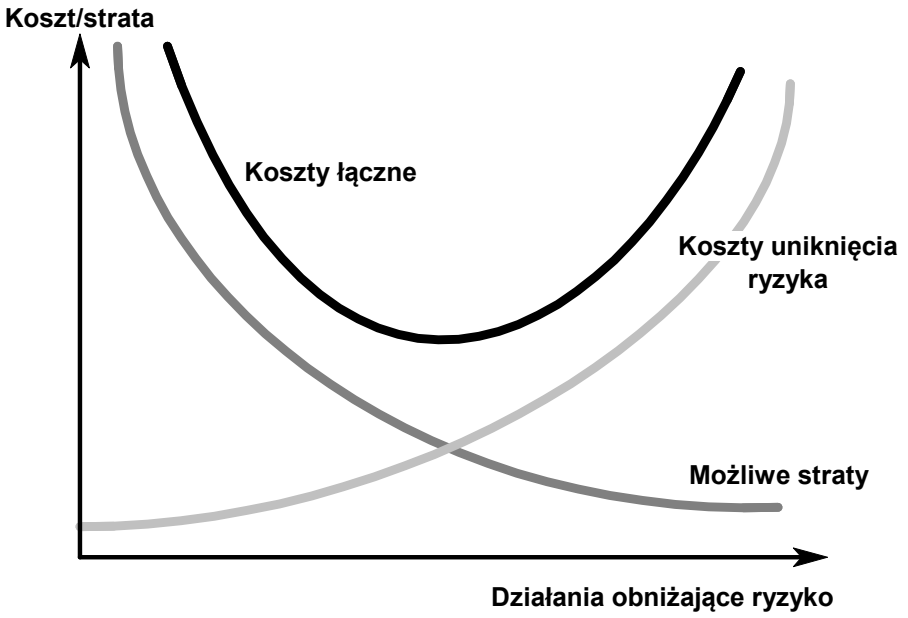
Kolejna faza zarządzania ryzykiem to podjęcie działać zapobiegawczych. Działania zapobiegawcze powinny być adekwatne do zidentyfikowanego zagrożenia i jego oceny. Celem tych działań może być ograniczenie potencjalnych strat w przypadku wystąpienia zagrożenia lub zmniejszenie prawdopodobieństwa wystąpienia negatywnego zdarzenia. W szczególnych przypadkach podejmujemy działania ograniczające negatywny wpływ w obu obszarach. Decyzje o skali podejmowanych działań zapobiegawczych należą do kierownika projektu i zależą od wielu czynników takich jak ocena zidentyfikowanego zagrożenia, wpływ tego zagrożenia na działania projektowe, czy też działania te mogą wynikać z innych uwarunkowań realizacji projektu lub całe-

go portfela projektu opracowanego w ramach strategii firmy. Zaplanowane działania będą jednak obciążały budżet projektu mając na uwadze ewentualne niższe koszty w przypadku zaistnienia danego zagrożenia w przyszłości.

5.4. Działania zapobiegawcze

Działania zapobiegawcze należy wymyśleć zgodnie z tematyką zagrożenia, zaplanować, wpisać w harmonogram działań projektowych oraz zrealizować. Oczywiście dotyczy to tylko przypadku przyjęcia strategii aktywnego reagowania na zidentyfikowane zagrożenia. Cykliczna analiza skuteczności tych działań, realizowana w ramach kolejnych sesji zarządzania ryzykiem, może modyfikować zaplanowane działania lub wprowadzać nowe rozwiązania. Dynamika, skala oraz zakres działań zapobiegawczych powinny być adekwatne do oceny zidentyfikowanego zagrożenia w kontekście negatywnego wpływu na projekt. W szczególności w wyniku analizy możemy zmienić strategie z aktywnej na reaktywną jeśli zaistnieją przesłanki do takiej zmiany. Działania zapobiegawcze mogą mieć charakter działań długofalowych i obejmujących różne obszary aktywności projektowych lub jest to jednorazowa aktywność ukierunkowana na określony efekt. Efektywność podjętych działań zapobiegawczych powinna być oceniona w kolejnym przeglądzie ryzyka projektowego, co stanowi podstawę do modyfikacji tych działań.

Działania zapobiegawcze mają na celu minimalizacje zidentyfikowanego zagrożenia, co możemy osiągnąć podejmując akcje zmierzające do ograniczenia lub wyeliminowania strat powstałych w przypadku zaistnienia niekorzystnego zdarzenia. Innym sposobem działań zapobiegawczych jest wpływanie na minimalizacje prawdopodobieństwa wystąpienia negatywnego zdarzenia. Zarówno jedno rozwiązanie jak i drugie związane jest z ponoszeniem dodatkowych, określonych kosztów podejmowanych działań zapobiegawczych, których celem jest minimalizacja ryzyka. Skala i zakres podejmowanych działań zapobiegawczych mogą być różne zarówno w sensie ilości jak i kosztowności tych działań. Na rysunku 5.2 zobrazowano relacje wskazującą na zmniejszanie się potencjalnych strat związanych z potencjalnym ryzykiem wraz z zwiększeniem nakładów na podejmowane działania zapobiegawcze.



Rys. 5.2. Wpływ kosztów działań zapobiegawczych na straty projektowe. Źródło: [11].

W każdym przypadku istnieje określone minimum kosztów łącznych, które są wypadkową kosztów ponoszonych na działania zapobiegawcze i koszty związane z wystąpieniem ryzyka. Na rysunku obrazuje to krzywa opisana jako koszty łączne. Ponoszone koszty działań zapobiegawczych są pomniejszane o wyliczone zmniejszenie potencjalnych strat w przypadku wystąpienia badanego zagrożenia. Widać z tego, że w zależności od zidentyfikowanego ryzyka należy dopasować do niego działania zapobiegawcze w skali i zakresie adekwatnej do potencjalnych strat wynikających z wystąpienia tego negatywnego zdarzenia. W przypadku strategii reaktywnej rozważania te nie maja zastosowania, gdyż koszty przeciwdziałania są zerowe a potencjalne straty pozostają na poziomie stałym.

Reaktywna strategia zarządzania ryzykiem zakłada przychylność losu niewystąpienie zidentyfikowanego zagrożenia w ciągu prac projektowych. Nie zawsze założenie to jest prawdziwe lub może dotyczyć tylko niektórych zagrożeń. W takim przypadku musimy się liczyć z konsekwencjami wystąpie-

nia zagrożenia i negatywnych skutków dla projektu. Wybór określonej strategii jest kolejnym ryzykiem przed jakim staje zespół projektowy. Odpowiedzialność za przyjęcie strategii ponosi kierownik projektu, ale konsekwencje mogą być w niektórych przypadkach bardzo poważne i dotyczyć całego zespołu lub określonego obszaru przedsięwzięcia.

Podsumowanie

Z przeprowadzonych rozważań wynika, że zarządzanie ryzykiem to nie tylko procesy zobrazowane na rysunku 5.1, ale również problem wyboru odpowiedniej strategii zarządzania ryzykiem w projekcie. Strategia reaktywna optymistycznie zakłada pomyślny zbieg okoliczności i niewystępowanie zdarzeń negatywnych. Podejście takie minimalizuje aktywność zespołu w zakresie zarządzania ryzykiem oraz praktycznie eliminuje koszty związane z profilaktyką. Oczywiście w przypadku niepomyślnego zbiegu okoliczności i wystąpienia zdarzeń negatywnych możemy ponieść bardzo poważne konsekwencje, aż do braku sukcesu projektu lub znaczącego wydłużenia czasu realizacji często łączącego się z istotnym wzrostem kosztów projektu.

Zarządzanie ryzykiem w projektach jest aktywnością nieprodukcyjną posiadającą wysoki stopień niepewności i nieokreśloności w początkowej fazie projektowej, kiedy szacowane są koszty projektu. Podejmowanie działań związanych z zarządzaniem ryzykiem zaczyna generować koszty w momencie rozpoczęcia prac projektowych. Początkowo są to koszty relatywnie niewielkie związane z zwiększona pracochłonnością analiz i ocen. Aktywne zapobieganie ryzyku, związane z podejmowaniem działań zapobiegawczych może generować znacznie większe koszty, które mogą zachwiać budżetem projektu. W związku z tym poszukiwanie optymalnego punktu kosztów łącznych, zgodnie z rysunkiem 5.2 jest istotnym elementem aktywności kierownika projektu z punktu widzenia ekonomiki działań w ramach prac projektowych. W przypadku wyboru strategii aktywnej i poprawnego jej realizowania po zakończeniu projektu nie będziemy wiedzieli czy poniesione koszty na zapobieganie ryzyku były konieczne czy może wystarczyło liczyć na przychylność losu.

Widać z tego wyraźnie, że przyjęcie strategii zarządzania ryzykiem w projekcie, reaktywnej lub aktywnej jest kolejnym ryzykiem, z jakim musi sobie radzić kierownik projektu.

Bibliografia

1. Duncan W. R., *A Guide To The Project Management Body Of Knowledge*, PMI Standards Committee, Project Management Institute, PA 19082 USA.
2. Herkens G. R., *Jak zarządzać projektami*, Wydawnictwo RM, Warszawa, 2003.
3. Kisielnicki J., *Strategia informatyzowania organizacji w świecie ryzyka i niepewności*, w: *Strategia Systemów Informacyjnych 1999*, Akademia Ekonomiczna, Kraków, 1999.
4. Kozarkiewicz A., *Zarządzanie portfelami projektów*, Wydawnictwo Profesjonalne PWN, Warszawa, 2012.
5. Lent B., *Zarządzanie procesami prowadzenia projektów*, Difin, Warszawa, 2005.
6. Pritchard C. L., *Zarządzanie ryzykiem w projektach*, WIG PRESS Warszawa, 2002.
7. Rot A., *Podejście ilościowe i jakościowe w analizie ryzyka informatycznego w małych i średnich przedsiębiorstwach*, w: *Aspekty informatyzacji organizacji*, Prace naukowe Uniwersytetu Ekonomicznego we Wrocławiu nr 55, Wydawnictwo Uniwersytetu Ekonomicznego we Wrocławiu, Wrocław, 2009.
8. Rutkowska J, Rak D., *Ewolucja narzędzi informatycznych wspierających metodę rachunku kosztów działań*, w: *Problemy zarządzania*, Zeszyt naukowy Wydziału Zarządzania Uniwersytet Warszawski, Wydawnictwo Naukowe Uniwersytetu Warszawskiego, Warszawa, 2011 .
9. *Strategiczne zarządzanie projektami*, red. Trocki M., Sońty-Draczkowska E., BIZARRE Sp. z o.o., Warszawa, 2009.
10. Szyjewski Z., *Zarządzanie Projektami Informatycznymi. Metodyka tworzenia systemów Informatycznych. Czynniki sukcesu wymiarowanie projektu*, Placet, Warszawa,2001.

11. Szyjewski Z., *Metodyki zarządzania projektami informatycznymi*, Placet, Warszawa, 2004.
12. Willett A. H., *The Economic Theory of Risk Insurance*, University of Pensylvania Press, Philadelphia, 1951.

Rozdział 6

Analiza procesów workflow jako narzędzie opisu wymagań funkcjonalnych – propozycja metodyki

Narzędzia informatyczne wspierające automatyzację procesów pracy, w szczególności nakierowanej na przetwarzanie informacji i decyzji, nazywane systemami workflow, stają się obecnie najpopularniejszymi obszarami wdrożeń informatycznych w przedsiębiorstwach, również średnich i małych (MŚP). Aby dobrze przygotować się do projektu wdrożeniowego systemu workflow przedsiębiorstwo powinno uporządkować i opisać procesy biznesowe. Opisane procesy są podstawą do zdefiniowania wymagań funkcjonalnych wobec docelowego systemu informatycznego. W niniejszym rozdziale skupiono się na procesie analizy biznesowej realizowanej w przedsiębiorstwach produkcyjnych, dystrybucyjnych oraz usługowych z sektora MŚP. Celem pracy jest propozycja metodyki, która dzięki swej prostocie i wykorzystaniu standardowych, ogólnodostępnych narzędzi mogłaby zostać wykorzystana do analizy wykonanej wewnętrznymi siłami MŚP, których z założenia nie stać na zatrudnianie zewnętrznych ekspertów, drogie metodyki komercyjne czy specjalistyczne narzędzia informatyczne. Główne perspektywy analityczne proponowanej metodyki są zgodne z podejściem strukturalnym (dane i procesy). Metodyka uwzględnia również cztery dodatkowe perspektywy specyficzne dla systemów workflow, a mianowicie: struktury organizacyjnej, struktury lokalizacyjnej, reguł biznesowych oraz pojęć. W rozdziale zaproponowano narzędzia analityczne i informatyczne wykorzystywane w poszczególnych perspektywach. Na zakończenie przedstawiono zastosowanie metodyki w analizie procesów przykładowej organizacji.

Systemy informatyczne typu workflow stają się coraz popularniejszymi narzędziami IT warstwy oprogramowania aplikacyjnego przedsiębiorstw. Zainteresowanie systemami tego typu wynika z popularności i ewolucji procesowego podejścia do zarządzania przedsiębiorstwem oraz ewolucji metody planowania zasobów przedsiębiorstwa i narzędziami informatycznych wspierających tą metodę – systemów klasy ERP. Ustanowienie pojęcia procesu, co miało początek w latach 90-tych XX wieku, jako klucza do zarządzania przedsiębiorstwem, dało sygnał do wdrażania systemów informatycznych wspierających realizację procesów we wszystkich obszarach działalności organizacji gospodarczych. Znacznie wcześniej, od lat 70-tych, przedsiębiorstwa wykorzystywały systemy klasy MRP, które wspierały początkowo planowanie zaopatrzenia materiałowego produkcji, następnie planowanie obciążenia maszyn, pracowników aż wreszcie finansów przedsiębiorstwa. Można stwierdzić, że obecnie procesy operacyjne, w których przetwarzane są materiały, produkty, towary czy wreszcie pieniądze, mają już wsparcie poprzez odpowiednie funkcjonalności systemów ERP. Nadeszła pora rozbudowy narzędzi wspierających procesy, które nie są bezpośrednio związane ze środkami produkcji a procesami przepływu pracy niematerialnej, informacji i decyzji. To procesy informacyjno – decyzyjne, nazywane też administracyjno – biurowymi, np. planowania i kontroli projektów, budżetów, procesy wydawania decyzji operacyjnych np. akceptacji kosztów, parametrów sprzedaży czy procesy realizowane w ramach pracy grupowej np. przygotowywaniem ofert czy umów. Obszar ten wypełniają omawiane systemy typu workflow. Koncepcja systemów workflow, nazywanych również przepływu pracy, sięga rozważań filozoficzno - lingwistycznych z lat 60-tych XX wieku [1]. Ówcześni badacze skupili się na kwestiach języka jako narzędzia komunikacji, przekazywania i koordynacji zadań. Badanie systemów workflow nabrało dynamiki i znaczenia ekonomicznego z chwilą rozwoju komunikacji elektronicznej. Aktualne badania tej dziedziny nakierowane są na modelowanie i projektowanie narzędzi informatycznych wspierających przepływy pracy i informacji w organizacjach. Systemy informatyczne typu workflow najwcześniej zaczęły być wykorzystywane w branży finansowej i ubezpieczeniowej, gdzie wiele procesów ma charakter niematerialny np. obsługa wniosków kredytowych, polis ubezpieczeniowych itd. Jedną z pierwszych aplikacji tego typu był system analizy danych kredytowych na potrzeby banków amerykańskich w latach 70-tych XX wieku [1].

Tradycyjnie, branże te bardzo szybko wdrażają nowe technologie IT, ze względu na wysoką kulturę informatyczną i duże środki inwestowane właśnie w szeroko rozumianą informatykę.

Niniejszy rozdział skupia się na przedsiębiorstwach produkcyjnych, dystrybucyjnych oraz usługowych z sektora firm średnich i małych (MŚP). Zaproponowana metodyka analizy procesów workflow w przedsiębiorstwach MŚP, zakłada wykorzystanie, znanych z teorii projektowania systemów informatycznych, standardowych i ogólnodostępnych metodyk i narzędzi analitycznych. Założeniem doboru narzędzi będzie ich prostota i przystępność, również dla nieinformatyków, tak by przedsiębiorstwa MŚP mogły własnymi siłami, bez drogiego wsparcia zewnętrznego, tworzyć i modelować wizję firmowych procesów workflow. Modele takie można wykorzystać przede wszystkim dla opisu wymagań na wspierające systemy informatyczne, ale również do tworzenia procedur ISO, optymalizacji kosztowych i czasowych procesów czy budowy polityki kontrolingowej organizacji. W rozdziale drugim, zdefiniowano podstawowe pojęcia związane z systemami workflow tj. proces i system workflow. Kolejny rozdział prezentuje perspektywy analizy systemu workflow przyjęte w proponowanej metodyce SPARD. Rozdział czwarty, charakteryzuje, w dużym skrócie, poszczególne narzędzia analityczne wykorzystywane w zaproponowanej metodyce. Ostatni rozdział opisuje analizę wybranych procesów workflow na bazie scenariusza działania przykładowej, nierzeczywistej organizacji. Scenariusz oparty został o założenia projektu analitycznego, w którym brał udział autor, a także wykorzystano w nim proponowaną metodykę.

6.1. Podstawowe pojęcia związane z systemami workflow

Tematyką związaną z systemami workflow zajmuje się obecnie wiele ośrodków akademickich oraz biznesowych. W związku z różnorodnością podejść teoretycznych i praktycznych powstało kilka organizacji standaryzujących zajmujących się stricte workflow lub też częściowo poruszających się w tych obszarach tematycznych. Pośród nich najbardziej znaną jest Workflow Management Coalition (WfMC). WfMC unifikuje pojęcia i tworzy standardy związane z systemami workflow. Bardzo istotnym dokonaniem tej organizacji

jest opublikowanie słownika pojęć i terminów workflow [2]. Dodatkowo WfMC stworzyło dwa standardy notacyjne WF-XML oraz XPDL, służące do opisu i wymiany modeli procesów pomiędzy systemami informatycznymi. Stworzony przez WfMC glosariusz jest jednym z głównych źródeł terminologicznych, na których bazują opracowania naukowe i techniczne. Jak zauważyli Z. Martyniak i M. Ćwiklicki [1,4] w polskich warunkach dodatkową trudnością jest poprawne i jednoznaczne przetłumaczenie zdefiniowanych w nim pojęć. Na potrzeby niniejszego rozdziału przytoczono definicje takich pojęć jak workflow, system workflow oraz definicja procesu workflow. Nie przytaczamy tutaj definicji procesu biznesowego, gdyż nie różni się ona od ogólnie przyjętej. Przykładem trudności tłumaczenia jest m.in. podstawowe pojęcie, a mianowicie „workflow", które nie ma polskiego tłumaczenia „wprost". Spolszczona wersja definicji pojęcia workflow, które najczęściej tłumaczymy jako przepływ pracy, pochodząca źródłowo z glosariusza WfMC, a podana przez Z. Martyniaka [4] brzmi: *„Automatyzacja procesu biznesowego, w całości lub części, podczas którego dokumenty, informacje i zadania są przenoszone od jednego uczestnika do innych dla wykonania działania zgodnie ze zbiorem sformalizowanych zasad"*. Z tego określenia wynika definicja systemu workflow, najczęściej tłumaczonego jako system przepływu pracy. Brzmi ona: *„System workflow jest to system umożliwiający za pomocą oprogramowania tworzenie definicji procesów oraz zarządzanie wykonywaniem instancji procesów uruchomionych na jednym lub wielu silnikach przepływu pracy, który potrafi interpretować definicje procesów, komunikować się z uczestnikami przepływu pracy oraz, tam gdzie jest to wymagane, wywoływać inne aplikacje"* [2,4]. Bardzo istotnym i charakterystycznym elementem systemu informatycznego typu workflow jest jego zdolność do zarządzania procesem biznesowym realizowanym w organizacji według wcześniej zdefiniowanej, przygotowanej czy skonfigurowanej definicji procesu. Zwrócono również uwagę, że zgodnie z ogólnym pojęciem workflow chodzi tutaj o „transportowanie" dokumentów, informacji i zadań pośród uczestników procesu, a więc jednostek organizacyjnych lub poszczególnych osób wchodzących w skład struktury organizacyjnej.

Idąc śladem określenia systemu workflow zacytowano określenie definicji procesu workflow „*jest to taka forma prezentacji procesu biznesowego, która umożliwia zautomatyzowane przetwarzanie, takie jak modelowanie czy*

wykonywanie procesu przez system zarządzania przepływem pracy" [2,3]. Definicja procesu składa się z sieci czynności i powiązań pomiędzy nimi, kryteriów rozpoczęcia oraz zakończenia procesu i informacji na temat poszczególnych czynności, takich jak wykonawcy czynności czy powiązane z czynnościami aplikacje i dane.

6.2. Założenia i narzędzia proponowanej metodyki

Metodyka analityczna zaproponowana w rozdziale czerpie po trosze ze wszystkich najważniejszych podejść do analizy i projektowania systemów informacyjnych. Założeniem wstępnym doboru narzędzi jest ich standardowy i ogólnodostępny charakter oraz przystępność dla biznesowej części zespołów analitycznych. Główne perspektywy analityczne metodyki są zgodne z podejściem strukturalnym. Zgodnie z nim, dla pełnej analizy systemu informacyjnego należy opisać **dane** w nim przetwarzane oraz **procesy**, w ramach których przetwarzanie to następuje [5]. Zwrócono jednak uwagę, że procesy będą opisywane w pełniejszym, biznesowym ujęciu, nie tylko czysto informacyjnym (co jest odmiennością w stosunku do podejścia strukturalnego). Dodatkową perspektywą wykorzystywaną w proponowanej metodyce są **czynności**. Proces biznesowy jest zbiorem czynności, realizowanych w określonym celu. Aby zwiększyć precyzję modelu workflow metodyka proponuje opis każdej czynności wyspecyfikowanej w modelu procesu, na najniższym poziomie, a mianowicie scenariusza jej realizacji. Scenariusz opisuje kroki, które musi wykonać rola (również system, modelowany lub inny) odpowiedzialna za czynność, aby zrealizować jej cel. Aby zamodelować system workflow, metodyka wskazuje cztery dodatkowe perspektywy a mianowicie: **struktury organizacyjnej, struktury lokalizacyjnej, reguł biznesowych oraz pojęć firmowych**. Struktura organizacyjna określa ramy, w których realizowane są procesy workflow. Role, w niej pokazane, odpowiadają na pytanie kto (osoba), lub która jednostka organizacyjna, jest odpowiedzialna za zadania zlecone w workflow. Struktura lokalizacyjna pokazuje rozmieszczenie jednostek organizacyjnych w obszarze geograficznym. Reguły biznesowe określają punkty decyzyjne, w których proces workflow może wykonywać współbieżne lub alternatywne ścieżki realizacji. Pojęcia firmowe są uzupełnieniem słownika danych o terminy istot-

ne dla zrozumienia działania analizowanej organizacji, które nie są bezpośrednio odwzorowane w modelu danych. Proponowana metodyka, w dalszej części pracy, będzie skrótowo określona jako **SPARD** (od pierwszych liter angielskich nazw głównych perspektyw**)**. Do wskazanego zestawu perspektyw wybrano odpowiednie narzędzia analityczne, pełne zestawie prezentuje tabela 6.1.

Tabela 6.1. Perspektywy i narzędzia metodyki SPARD.

Perspektywa	**Narzędzia**
Struktura organizacyjna (ang. organizational **S**tructure)	Diagram struktury organizacyjnej – *Org Chart (OC)*
Struktura lokalizacyjna (ang. localization **s**tructure)	Diagram lokalizacji (na bazie UML-wego diagramu rozlokowania) - *Localization Chart (LC)*
Procesy (ang. **P**rocesses)	Diagram Procesu Biznesowego (zgodny z BPMN) – *Business Process Diagram (BPD)*
Czynności (ang. **A**ctivities)	Scenariusze przypadków użycia *Use-case scenario*
Reguły biznesowe (ang. Business **R**ules)	*Pseudokod*
Dane i pojęcia (ang. **D**ata and concepts structure)	Słownik danych *Data and concepts dictionary (DCD)* Diagram relacji obiektów *Entity Relationship Diagram (ERD)*

Źródło: opracowane własne.

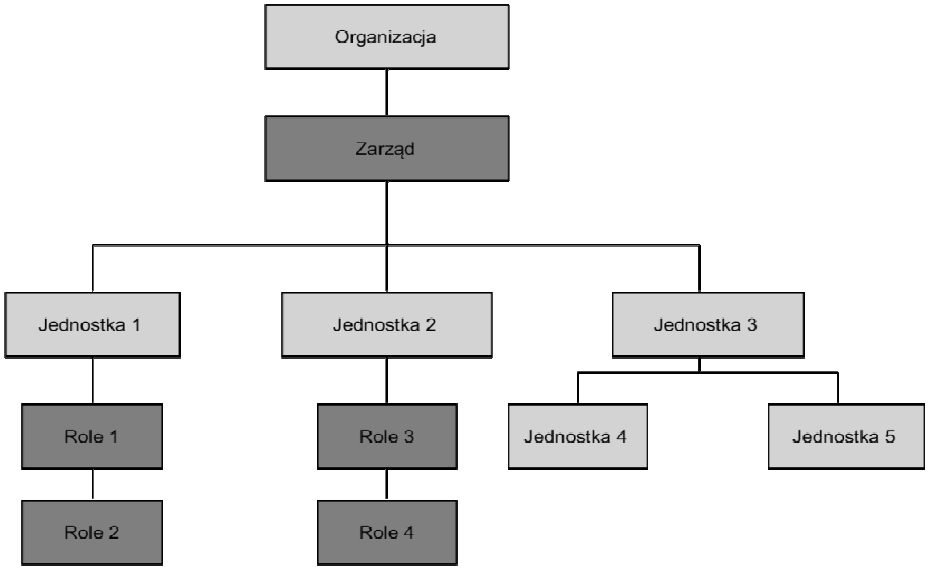
Narzędzia analityczne wykorzystane w metodyce SPARD pochodzą z różnych podejść do analizy systemów informacyjnych. Diagram struktury organizacyjnej nie jest związany z żadnym podejściem informatycznym, jest wykorzystywany w metodach zarządczych i ekonomicznych. Do pokazania układu geograficznego organizacji wykorzystano notację diagramu rozlokowania będącego składową języka UML. Notacja BPMN do modelowania procesów jest obecnie najpopularniejszym narzędziem wykorzystywanym w podejściu procesowym. Przypadki użycia są pojęciem analitycznym, którego pomysłodawcą był Ivar Jacobson, twórca obiektowej metodyki analizy i projektowania OOSE (Object-Oriented System Engineering) [8]. Przypadki uży-

cia są w podejściu obiektowym podstawowym narzędziem specyfikowania wymagań funkcjonalnych wobec systemu informatycznego. W skład UMLa wchodzi narzędzie graficznego - diagram przypadków użycia, jednak pełną specyfikację można wytworzyć poprzez opis scenariusza komunikacji/interakcji użytkownika z systemem. Scenariusz jest narzędziem tekstowym jednak odpowiednio sformalizowanym i jednoznacznym. Pseudokod, słowniki danych oraz diagram związków obiektów (ERD) były szeroko wykorzystywane w podejściu strukturalnym [5]. Zestaw wykorzystanych narzędzi nie jest jednolity pod względem źródła, jednak zgodny z podstawowym założeniem proponowanej metodyki, w której najistotniejszą cechą narzędzia powinna być jego ogólnodostępność oraz prostota ideowa i notacyjna.

6.3. Krótka charakterystyka wybranych narzędzi metodyki SPARD

6.3.1. Perspektywa struktury organizacyjnej - diagramy struktury organizacyjnej

Modelowanie struktury organizacyjnej, w chwili obecnej nie jest uwzględnione w żadnym z głównych podejść do analizy systemów informacyjnych. Bardzo mało jest literatury traktującej o tym temacie. W systemach workflow odpowiedzialność za powierzone zadania i kompetencje do jego wykonania pochodzą wprost ze struktury organizacyjnej, dlatego perspektywa ta została uwzględniona w proponowanej metodyce. Do modelowania tej perspektywy wykorzystany zostanie diagram struktury organizacyjnej ang. org chart (OC). Diagram OC pokazuje podział organizacji na jednostki organizacyjne oraz stanowiska pracy, wraz z ich podległościami. Jednostki organizacyjne stanowią najczęściej grupy pracowników. W przypadku pojedynczych pracowników mówi się o stanowiskach. Na potrzeby modelowania systemów workflow zaproponowano określenie roli w miejsce stanowiska. Rola wskazuje na zdolność i uprawnienia do wykonania jakiejś czynności np. kierownik projektu, magazynier, administrator. Poglądowy diagram struktury organizacyjnej przedstawia rysunek 6.1.



Rys. 6.1. Poglądowy diagram struktury organizacyjnej.
Źródło: Opracowanie własne.

6.3.2. Perspektywa struktury lokalizacyjnej - diagramy struktury lokalizacyjnej

Bardzo istotną informacją dla modelu workflow jest fizyczne rozmieszczenie jednostek organizacyjnych w układzie geograficznym. Aspekt ten jest ważny dla organizacji wielo-oddziałowych lub wielo-firmowych, gdzie wymiana korespondencji najczęściej staje się wąskim gardłem informacyjnym. Dla zobrazowania fizycznej struktury organizacji zaproponowano notację diagramu rozlokowania, wchodzącego w skład języka UML [8]. Symbol węzła w metodyce SPARD oznacza fizyczną, odrębną geograficznie lokalizację, w której realizowana jest działalność modelowanej organizacji. Dodatkowym elementem pokazywanym na diagramie lokalizacji jest programowe wyposażenie poszczególnych lokalizacji. Przykład diagramu struktury lokalizacyjnej zaprezentowano w kolejnym rozdziale.

6.3.3. Perspektywa procesów - diagram procesu biznesowego

Diagram BPD jest podstawowym elementem języka modelowania procesów biznesowych o nazwie Business Process Model and Notation (BPMN),

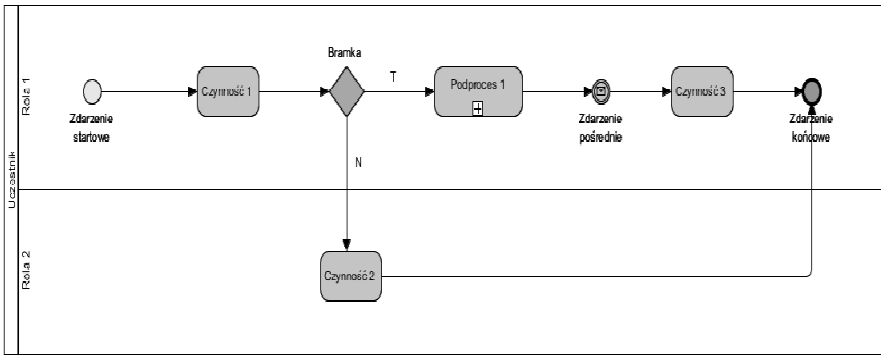
który to w obecnym czasie staje się standardem notacyjnym w podejściu procesowym. Notacja ta powstała w ramach konsorcjum Business Process Management Initiative (BPMI), a obecnie (po połączeniu BPMI z OMG) jest utrzymywana i rozwijana przez konsorcjum OMG [bpmn.org, omg.org]. Podstawowym celem standardu BPMN jest dostarczenie takiej notacji do opisywania procesów biznesowych, która będzie czytelna i zrozumiała zarówno dla biznesowych użytkowników, którzy procesy monitorują i zarządzają nimi, jak i analityków przeprowadzających biznesową analizę oraz programistów odpowiedzialnych za techniczną implementację procesów [6]. Cytowany autor zauważa, że notacja ta powstała z „bezsilności" i trudności w komunikacji pomiędzy biznesowymi interesariuszami systemu a stroną informatyczną odpowiadającą za jego implementację. Opinia ta zgodna jest ze znanym twierdzeniem o złożoności i niedostępności dla nieinformatyków notacji takich jak UML.

Podstawowe kategorie elementów graficznych BPMN:

- miejsca realizacji procesu – uczestnicy oraz role (szczegółowe wyjaśnienie tych pojęć znajduje się poniżej).
- elementy aktywne przepływu (czynności, podprocesy, zdarzenia, bramki)
- połączenia – pokazujące przebieg procesu lub wiadomości przesyłane pomiędzy uczestnikami procesu obiekty danych
- artefakty - to elementy graficzne nie będące elementami przepływu, służące umieszczaniu informacji uzupełniających

Miejsca realizacji procesu to uczestnicy lub role, realizujące czynności bądź podprocesy. Uczestnicy procesu to współpracujące organizacje, firmy czy systemy informatyczne. Role to jednostki organizacyjne lub stanowiska biorące udział w realizacji procesu. Takie podejście jest pochodną koncepcji swimlines (torów pływackich) [7]. Na diagramie BPMN każdy uczestnik i jego składowe role są prezentowane jako prostokątne obszary, przypominające basen pływacki podzielony na tory. Czynności realizowane przez poszczególne role znaczy się odpowiednimi symbolami w obszarze toru danej roli (patrz rysunek 6.2). Właśnie to podejście przemawia za wykorzystaniem BPMN do modelowania workflow. Dzięki niemu niezwykle czytelna staje się kolejność realizacji czynności i odpowiedzialność poszczególnych ról za ich wykonanie. W tym miejscu występuje również konieczność zgodności modelu

struktury organizacyjnej z modelem procesu. Role pokazane na diagramie procesu muszą występować na diagramie struktury organizacyjnej. Opisywanie wszystkich zasad modelowania w BPMN nie jest celem niniejszego rozdziału, warto wspomnieć, iż są już dostępne polskie opracowania poświecone temu tematowi [6,7]. Również uniwersalne i popularne programy graficzne np. Microsoft Visio posiadają zestaw obiektów graficznych zgodnych z BPMN. Poglądowy diagram procesu biznesowego przedstawia rysunek 6.2.



Rys. 6.2. Poglądowy diagram procesu biznesowego w notacji BPMN.
Źródło: opracowanie własne.

6.3.4. Perspektywa danych i pojęć – słownik danych i pojęć

W słowniku danych specyfikuje się wszystkie elementy obiektów danych wykorzystywanych w procesach workflow. Słownik danych to narzędzie tekstowe, ale sformalizowane. Elementy danych pochodzą z obiektów danych na diagramach BPD, reguł biznesowych oraz diagramów ERD. Słownik powinien zawierać również definicje pojęć, które występują w specyficznym języku organizacji. Słownik jest bardzo ważnym elementem komunikacyjnym pomiędzy użytkownikami biznesowymi, analitykami a w następnej kolejności projektantami i programistami. Ci ostatni mogą nie mieć styczności z użytkownikami, jednocześnie muszą jednoznacznie rozumieć działanie organizacji i jej specyficzny język.

W słowniku danych występuję trzy rodzaje elementów:

- dane elementarne – dane nie podlegające dalszemu podziałowi np. cena, wiek, miasto, wzrost. Dla danych elementarnych, opisujemy kontekst (czyli znaczenie dla organizacji) oraz jednostki miary i wartości dopuszczalne.
- pakiety danych – zbiory składający się z danych elementarnych. Pakiety danych definiujemy poprzez kontekst oraz opis z jakich danych elementarnych składa się ów pakiet.
- pojęcia – ponieważ nie są związane z danymi specyfikujemy tylko za pomocą kontekstu. W tym miejscu można wskazać odnośniki do innych materiałów analitycznych lub organizacyjnych np. procedur, regulaminów itd.

Konwencja zapisu pokazana powyżej pochodzi ze strukturalnej metodyki Yourdona [5] (z wyjątkiem pojęć, które zostały dodane przez autora). Przykład słownika danych zaprezentowano w kolejnym rozdziale.

6.4. Przykład zastosowania metodyki SPARD

6.4.1. Ogólna charakterystyka analizowanej organizacji

Podmiotem przykładowego scenariusza będzie firma KPRM – Krakowskie Przedsiębiorstwo Robót Mostowych. Przedsiębiorstwo specjalizuje się w budowie mostów i budowli hydrotechnicznych. KPRM zapewnia całościową obsługę inwestycji od faz projektowych po wykonawstwo i nadzór eksploatacyjny. Siedziba (centrala) firmy znajduje się w Krakowie. Firma jest organizacją wielo-oddziałową, posiadającą oddziały terenowe w Rzeszowie, Kielcach oraz Częstochowie. Każdy z oddziałów posiada własne zasoby i środki wykonawcze. W centrali firmy realizowanych jest jednocześnie kilkanaście kontraktów budowlanych. Oddziały prowadzą po kilka kontraktów budowlanych jednocześnie.

KPRM przygotowuje się do wdrożenia systemu workflow, który wesprze realizacje nowej polityki kontrolingowej firmy. Podstawowym założeniem nowej polityki kontrolingowej jest możliwość bieżącego śledzenia wyników finansowych firmy w szczegółowości do pojedynczego kontraktu. Za poszczególne kontrakty realizowane przez KPRM mają być odpowiedzialne

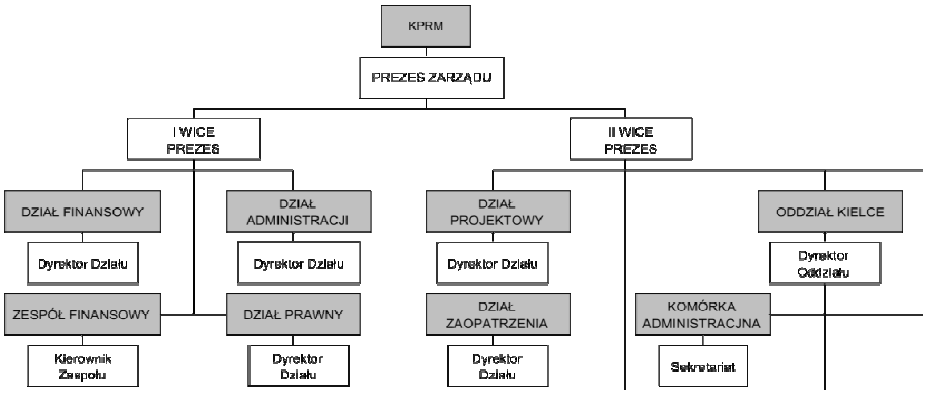
wyznaczone przez Zarząd osoby - kierownicy budów. Za wyniki realizowane w poszczególnych jednostkach organizacyjnych mają bezpośrednio odpowiadać ich menedżerowie. Każdy składnik kosztów i przychodów, który związany jest z kontraktem musi zostać autoryzowany przez kierownika budowy. Operacje realizowane z pośrednictwem działów mają być autoryzowane również przez ich menedżerów (np. koszty transportu ponoszone w całej organizacji muszą zostać autoryzowane przez Dział transportu). Większość operacji ma mieć, w związku z powyższymi założeniami, kilkuetapową autoryzację. Aktualnie podstawowym problemem jest obieg i ewidencja dokumentów kosztowych, w szczególności tych spływających do firmy z zewnątrz oraz trafiających bezpośrednio do zamiejscowych oddziałów firmy. Obieg dokumentów w obecnym podejściu, bazującym na nośnikach papierowych, jest zbyt czasochłonny, wymaga zdublowanych nakładów pracy oraz generuje bardzo wiele pomyłek. W efekcie wyniki finansowe są znane dopiero w okresie deklaracji podatkowej (20-dnia następnego miesiąca). Ze względu na brak wspólnych słowników danych, i wynikające z tego błędy ewidencji, rachunek kosztów został ograniczony do poziomu wymaganego do sprawozdawczości finansowej i podatkowej. W omawianym przykładzie, poddany analizie będzie tylko proces obiegu dokumentów zakupu, który jest ważnym, ale nie jedynym procesem, który należy zoptymalizować i wspomóc systemem workflow, aby efektywnie wdrożyć nową politykę kontrolingową. Proces obiegu dokumentów zakupowych ich akceptacje na wszystkich poziomach oraz finalne księgowanie nazywać będziemy procesem workflow operacji zakupu.

6.4.2. Cykl analizy procesu workflow operacji zakupu

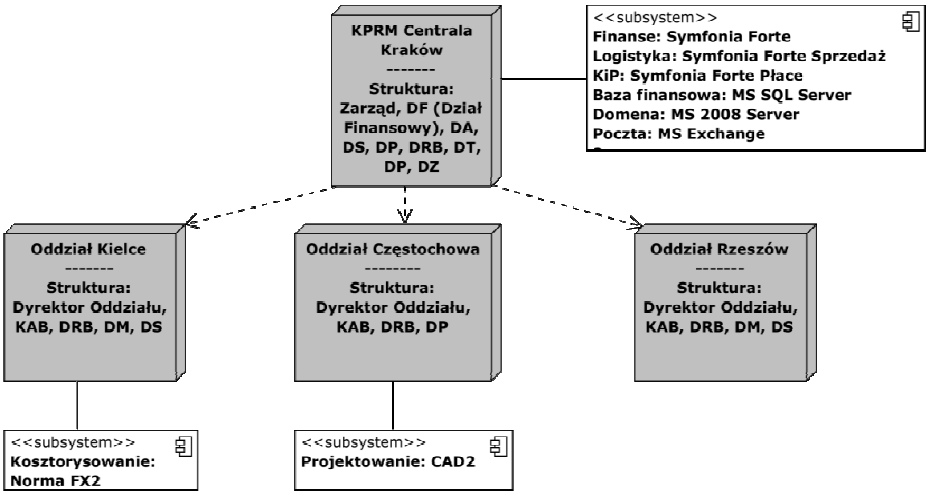
Etap I. Analiza struktury organizacyjnej

Metodyka SPARD, w kontekście kolejnych etapów analizy, zakłada podejście iteracyjne. Pierwszy etapem była analiza struktury organizacyjnej jako szkieletu, po którym przebiegają procesy workflow. Punktem wyjścia była „oficjalna", udokumentowana struktura organizacyjna. Na jej bazie stworzony został układ jednostek organizacyjnych, zaś stanowiska mapowane były na role, istotne dla systemu workflow. Tworzenie diagramu OC nie zakończyło się wraz z końcem tego etapu, zdarzały się sytuacje, gdzie w czasie modelowania procesów odnajdowano dodatkowe role. Fragment diagramu OC dla przedsiębiorstwa KPRM prezentuje rysunek 6.3. Następnie stworzono diagram

struktury lokalizacyjnej (rysunek 6.4). Już na tym etapie rozpoczęto tworzenie słownika danych i pojęć.



Rys. 6.3. Fragment diagramu struktury organizacyjnej przedsiębiorstwa KPRM.
Źródło: opracowanie własne.

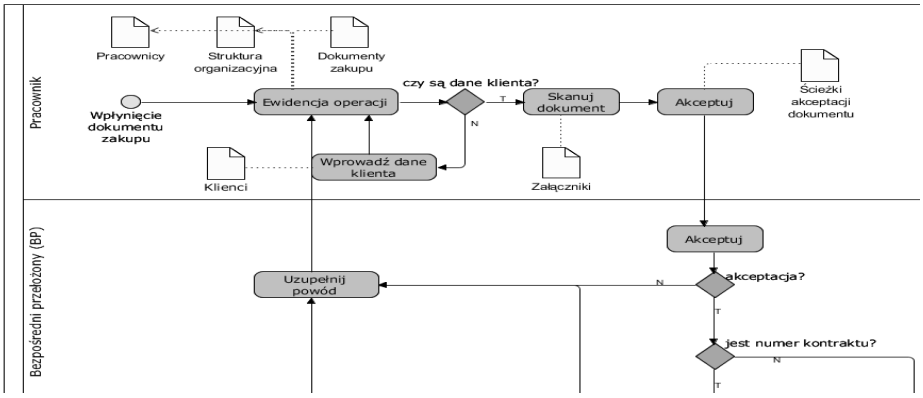


Rys. 6.4. Diagram struktury lokalizacyjnej KPRM.
Źródło: opracowanie własne.

Etap II. Analiza procesu workflow operacji zakupu

W następnej kolejności przeanalizowano proces operacji zakupu, w kontekście ról i czynności wykonywanych w jego ramach. Stworzony został diagram BPD. Etap ten również nie został ostatecznie zamknięty w momencie

przejścia do kolejnych prac. Model procesów może być wielokrotnie poprawiany i modyfikowany, gdyż analizując scenariusze poszczególnych czynności lub reguły biznesowe często zespół analityczny wnosi nowe „pomysły" i poprawki do opisywanych procesów. Diagram BPD dla procesu workflow operacji zakupu przedstawia rysunek 6.5. Diagram został stworzony w komercyjnym narzędziu typu CASE - VP Agilian. Jak wspomniano wcześniej, mógłby być do tego wykorzystany program MS Visio lub darmowy program Bizagi Process Modeler.



Rys. 6.5. Fragment BPD workflow operacji zakupu.
Źródło: opracowanie własne.

Etap III. Analiza reguł biznesowych

Równocześnie z modelem procesu opisuje się reguły biznesowe sterujące jego przebiegiem. Proste reguły można opisać bezpośrednio na modelu, jak w powyższym przypadku (patrz rysunek 6.5). Bardziej złożone powinno się opisać oddzielnie. Najważniejszym zaleceniem jest precyzyjne i jednoznaczne ich zapisanie z wykorzystaniem elementów słownika danych i pojęć. Reguły dla analizowanego procesu workflow operacji zakupu zostały zebrane w tabeli pokazującej ścieżki procesu w zależności od rodzaju zakupu (tabela 6.2). Warunki opisane w tabeli zostały opisane zgodnie z zasadami pseudokodu. W związku dużymi rozmiarami tabeli, reguły akceptacji zostały pokazane tylko

do trzeciego poziomu, źródłowy proces zawierał sześć poziomów. W tabeli ujęto tylko trzy rodzaje operacji, źródłowy proces obsługiwał dziewięć rodzajów.

Tabela 6.2. Reguły ścieżek akceptacji w zależności od rodzaju zakupu (fragment)

Poziomy akceptacji	**0**	**I**		**II**		**III**	
Rodzaje operacji zakupu	***Rola***	***Kiedy***	***Rola***	***Kiedy***	***Rola***	***Kiedy***	***Rola***
materiały podstawowe	Pracownik	zawsze	BP	? kontrakt	KB	zawsze	DZM
materiały biurowe	Pracownik	zawsze	BP	zawsze	DZP	dok_zak_kwota_brutto >2000	Prezes
energia elektryczna i cieplna	Pracownik	zawsze	BP	? kontrakt	KB	dok_zak_kwota_brutto >2000	Wiceprezes

Źródło: opracowanie własne.

Etap IV. Analiza czynności i danych

Kolejnym etapem analizy było rozbicie czynności pokazanych w modelu procesu na kroki jakie musi wykonać rola lub system. Analizując pierwszą czynność procesu workflow „ewidencję operacji", tworzony jest dla niej prosty scenariusz. Scenariusz ten pokazuje podstawową tzw. ścieżkę powodzenia. Scenariusz zakłada powodzenie operacji i nie uwzględnia ewentualnych błędów użytkownika czy systemu. W zapisach scenariusza wykorzystywano słownik danych i pojęć. Przykład fragmentu scenariusza zaprezentowano poniżej:

Proces workflow operacji zakupu
Rodzaj opisu: formalny
Scenariusz podstawowy czynności - *Ewidencja operacji*:
1. Pracownik wybiera [z listy] rodzaj rejestrowanego dokumentu
2. Pracownik wybiera [z listy] klienta z ewidencjonowanego dokumentu
3. Pracownik wprowadza dane nagłówkowe (dok_zak_numer, dok_zak_data_wystawienia, dok_zak_data_ sprzedaży, dok_zak_sposób_płatności [z listy], dok_zak_termin_płatności [z listy]).

Data wpływu, identyfikator pracownika oraz działu (dok_zak_data_wpływu, numer_pracownika, symbol_działu) wypełniają się automatycznie datą systemową oraz danymi zalogowanego użytkownika.

4. Pracownik wypełnia poszczególne linijki dokumentu (dok_zak_opis_pozycji, dok_zak_kwota_netto, dok_zak_stawka, dok_zak_kwota_brutto, rodzaj_zakupu [z listy], numer_kontraktu [z listy])

Scenariusz alternatywny czynności - *Ewidencja operacji*:

2a. Brak klienta na liście, uruchamiana jest czynność - Wprowadź dane klienta

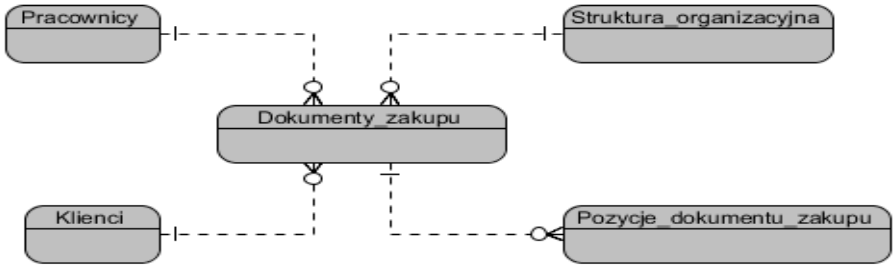
Równocześnie z opisem scenariuszy poszczególnych czynności uszczegóławiano słownik danych i pojęć. Fragment słownika danych i pojęć prezentuje tabela 6.3.

Tabela 6.3. Fragment słownika danych i pojęć

Dane/ pojęcia	Kontekst	Definicja	Wartości dopuszcz.	Jednostka
dok_zak_data	Data sprzedaży widniejąca na dokumencie			*data (dd - mm - yyyy)*
dok_zak_data_wpł	Data wpływu dokumentu do KPRM			*data (dd - mm - yyyy)*
dok_zak_termin_pł	Ilość dni na dokonanie zapłaty		*0-365*	*liczba (4,0)*
dokument_zakupu	dokument zewnętrzny lub wewnętrzny prezentujący operację zakupu dokonaną przez KPRM	dok_zak_numer+symbol_klienta+dok_zak_data_wyst+ dok_zak_data_wpł + id_pracownika + symbol_działu +{dok_zak_opis_pozycji+dok_zak_kwota_netto+dok_zak_stawka+dok_zak_kwota_brutto+rodzaj_zakupu+numer_kontraktu}		
kierownik budowy (KB)	Pracownik wskazany przez Zarząd do nadzoru kontraktu. Zakres obowiązków znajduje się w Zał. X.			

Źródło: opracowanie własne.

Słownik danych i pojęć nie wyczerpuje zagadnienia modelu danych. Po opisaniu poszczególnych obiektów, zamodelowano powiązania pomiędzy nimi. Na tym etapie projektujemy relacyjny model danych. Fragment takiego modelu, dla analizowanego procesu workflow prezentuje diagram ERD na rys. 6.6.



Rys. 6.6. Diagram ERD dla procesu workflow operacji zakupu
Źródło: opracowanie własne.

Podsumowanie

Proponowana w rozdziale metodyka jest pochodną strukturalnego podejścia do analiz systemów informacyjnych. Definiuje się strukturalne perspektywy patrzenia na system informacyjny, a więc procesy i dane. Metodyka SPARD została wzbogacona o perspektywy specyficznie związane z systemem workflow a mianowicie: strukturę organizacyjną, strukturę lokalizacyjną, reguły biznesowe i pojęcia. Do modelowania poszczególnych perspektyw wykorzystano znane i standardowe narzędzia podejścia procesowego, obiektowego oraz strukturalnego. Wszystkie narzędzia, jak pokazuje analiza przypadku zastosowania, posiadają prostą, intuicyjną notację a do jej tworzenie można wykorzystać niedrogie, uniwersalne narzędzia graficzne lub open-sourcowe narzędzia typu CASE. Cechy te, zdaniem autora, świadczą o możliwości zastosowania metodyki w przedsiębiorstwach MŚP, gdzie wewnętrzne zespoły analityczne, po krótkim szkoleniu wstępnym, mogłyby samodzielnie modelować procesy workflow. Większego wkładu merytorycznego wymagałbym etap integracji modeli i stworzenia wspólnego dokumentu analitycznego.

Stworzony, zgodnie z metodyką SPARD, model może być punktem wyjścia do zbierania wymagań na narzędzie informatyczne wspierające workflow, jak również do rozpoczęcia symulacji i optymalizacji procesów, opisu procedur ISO czy budowania polityki kontrolingowej przedsiębiorstwa.

Bibliografia

1. Ćwiklicki M., *Podstawy systemów workflow*, Wydawnictwo Akademii Ekonomicznej w Krakowie, 2006.
2. WfMC, *Workflow Management Coalition, Terminology & Glossary*, Document Number WFMC-TC-1011, Issue 3.0, http://www.wfmc.org/ index.php ?option = com_docman&Itemid=74, (stan na 2012-12-27).
3. Renk R., Knapik R., Hołubowicz W. , *Standardy workflow przy budowie systemu informatycznego*, materiały badawcze projektu IST VISP http://www.visp-project.org/docs/publications/ renk_artykul_KSTiT_2006_workflow.pdf, (stan na 2012-12-27).
4. Martyniak Z., *Teoretyczne podstawy systemów workflow*, Informatyka, Warszawa, nr 3/2000.
5. Yourdon E., Współczesna analiza strukturalna, WNT, 1996, .
6. Drejewicz S., Zrozumieć BPMN. Modelowanie procesów biznesowych, Helion, 2011.
7. Piotrowski M., Notacja modelowania procesów biznesowych BPMN. Podstawy, BTC, 2007.
8. Fowler M., UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3rd ed., Addison-Wesley, 2003.

Rozdział 7

Zestawienie metryk oprogramowania obiektowego opartych na statycznej analizie kodu źródłowego

W rozdziale dokonano zestawienia metryk oprogramowania obiektowego, których wyniki możliwe są do uzyskania przy zastosowaniu metod statycznej analizy kodu źródłowego. Krótko przedstawiono historię i najważniejsze dokonania w tej dziedzinie. Wyjaśniono cel tworzenia oraz korzyści wynikające z zastosowania metryk. Opisane są najpopularniejsze metryki i wymienione jest kilka z pozostałych metryk. Przytoczone są również przykłady przeprowadzonych badań oraz przedstawiona jest autorska propozycja dalszego rozwoju metryk jako jeden z czynników potrzebnych do oceny jakości zaimplementowanych wzorców projektowych.

Programowanie ma nieco dłuższą historię niż pierwsze mikroprocesory. Jednak już od lat 60 [18, p. 1] osoby związane z tą dziedziną zastanawiały się jak zmierzyć artefakty powstające w procesie programowania. Pierwszą stosowaną miarą pozwalającą zmierzyć wielkość oprogramowania była liczba wierszy kodu LOC (ang. *line of code*), oraz jednostka odpowiadająca jednemu tysiącu wierszy kodu KLOC (ang. *kilo line of code*). W kolejnych dziesięcioleciach pojawiały się inne miary, bardziej abstrakcyjne niż KLOC, takie jak liczba punktów funkcyjnych (polegająca na zliczaniu wejść, wyjść i plików) [18, p. 4] czy złożoność cyklomatyczna [9, pp. 339-340] wprowadzona w 1976 roku przez McCabe'a i wyznaczająca ilość możliwych ścieżek do wykonania w badanym fragmencie programu. Przełomowym wydarzeniem w 1977 roku było zaproponowane przez Halsteada oddzielenie nauki dotyczącej komputerów - informatyki od nauki zajmującej się oprogramowaniem [9, pp. 337-339]. Nauka ta utożsamiana jest z metodami Halsteada zakładającymi podział kodu programu na operatory i operandy. Halstead na podstawie zaproponowanych

wielkości opracował kilka równań pozwalających na wyznaczenie długości, wielkości, złożoności, trudności i kilku innych parametrów kodu źródłowego.

Powstałe miary nazywane są również metrykami oprogramowania i w ujęciu inżynierii oprogramowania wyjaśnione jako funkcja, która odwzorowuje pewną jednostkę oprogramowania w wartość liczbową [3, p. 111]. Samo znaczenie metryk oprogramowania przetrwało do teraźniejszości bez większych zmian. Natomiast rosnąca popularność paradygmatu programowania obiektowego spowodowała, że metryki oprogramowania, które powstały w drugiej połowie XX wieku dla oprogramowania strukturalnego, stały się nieefektywne [15, p. 1]. Odpowiedzią na to było powstawanie metryk oprogramowania obiektowego (równolegle do rozwoju samego paradygmatu obiektowości), które uwzględniały takie aspekty jak: dziedziczenie, hermetyzację, polimorfizm oraz abstrakcję. Wyjątek stanowiła złożoność cyklomatyczna, która jest wciąż wykorzystywana w niektórych metrykach obiektowych.

7.1. Cel stosowania metryk

Można wymienić wiele celów, dla których stosuje się metryki oprogramowania. Różnorodność celów wynika z roli jaką pełni zainteresowana osoba w procesie wytwarzania oprogramowania. Jednym z najpopularniejszych celów jest określenie jakości oprogramowania. Pierwszymi miarami jakości oprogramowania były ilość błędów na KLOC, czy zgodność ze specyfikacją [6, p. 103].

Ważnym celem metryk oprogramowania jest wczesne wykrywanie kodu, na który należy zwrócić szczególną uwagę, lub kodu, który w przyszłości będzie wymagał refaktoryzacji [15, p. 2]. Dużą uwagę poświęcono na predykcję błędów w oprogramowaniu w [8, pp. 1-2], z wykorzystaniem metod sztucznej inteligencji [15, p. 3], np. sztucznych sieci neuronowych [7, p. 2].

7.2. Przegląd wybranych najpopularniejszych metryk

Przez lata badań i prac nad oprogramowaniem powstało wiele różnych metryk. W [2, p. 1] podane jest, że ich ilość może sięgać nawet 500 różnych metryk. Jednak w przypadku oprogramowania obiektowego za najpopularniej-

sze [2, p. 1] [17, p. 1] uznane są metryki, które zostały opisane w trzech kolejnych podrozdziałach.

7.2.1. Zestaw metryk Chidambera i Kemerera

Pierwsze wzmianki dotyczące działań Shyama R. Chidambera i Chrisa F. Kemerera pojawiły się w 1991 roku [15, p. 3], a w 1994 roku [3, p. 46] zaproponowali oni 6 metryk mierzących aspekty związane z obiektowością. Metryki te zostały nazwane zestawem CK, od pierwszych liter nazwisk ich twórców, i składają się z:

- WMC (od ang. *Weighted Methods per Class*) oznacza sumę ważoną metod w klasie. Metryka ta wyznacza złożoność klasy na podstawie sumy ważonej jej metod. Wagi wyznaczane są poprzez złożoność cyklomatyczną każdej z metod. Przy założeniu, że wszystkie metody charakteryzują się tą samą złożonością to wartością metryki jest ilość metod. Wysoka wartość oznacza zbyt dużą złożoność klasy i taka klasa powinna zostać zdekomponowana na mniejsze klasy. Niska wartość nie niesie za sobą żadnego ryzyka. Wartość zalecona w [9, p. 362] to mniej niż 20.

- DIT (od ang. *Depth of Inheritance Tree*) oznacza głębokość drzewa dziedziczenia. Wartość tej metryki jest określona przez najdłuższą ścieżkę dziedziczenia, tj. maksymalną liczbę poziomów nadklas, z których dziedziczy badana klasa. Wysoka wartość oznacza bardzo wyspecjalizowaną klasę, wysoki stopień powtórnego użycia kodu oraz niesie za sobą ryzyko zbyt dużej złożoności. Niska wartość oznacza klasę abstrakcyjną. Wartość zalecona w [9, p. 362] to mniej niż 6. Warto zwrócić uwagę, że w nowoczesnych językach programowania (np. Java, C#) każda klasa niejawnie dziedziczy z abstrakcyjnej klasy bazowej (np. object).

- NOC (od ang. *Number of Children of Class*) oznacza liczbę potomków klasy. Wartość tej metryki określa liczbę bezpośrednich potomków w hierarchii dla badanej klasy. Zbyt wysokie wartości wskazują na złe wykorzystanie dziedziczenia. Wartość zalecona w [4, p. 2] to mniej niż 6.

- CBO (od ang. *Coupling Between Object*) oznacza powiązania pomiędzy obiektami. Wartość tej metryki jest sumą wszystkich klas powiązanych z badaną klasą, w inny sposób niż dziedziczenie. Niska wartość wskazuje na dokładne określenie funkcji pełnionych przez klasę oraz sposobu komunikowania się z innymi klasami. Sprzyja to późniejszym zmianom, ponieważ mo-

dyfikacja klasy o niskim powiązaniu będzie wymagała zmian w małej liczbie innych klas. Wartość zalecona w [4, p. 2] to mniej niż 8.

- RFC (od ang. *Response for a Class*) oznacza odpowiedź klasy. Metryka ta zlicza ilość metod wykonywanych przez badaną klasę oraz metod innych klas, z których badana klasa korzysta. Wysoka wartość metryki oznacza większą funkcjonalność i złożoność klasy, jak też dużą odpowiedzialność klasy. Zalecana wartość w zależności od źródła to zakres od 0 do 35 [4, p. 2] lub zakres od 20 do 100 [15, p. 3].

- LCOM (od ang. *Lack of Cohesion on Methods*) oznacza brak spójności metod. Metryka ta wskazuje stopień powiązania metod z atrybutami. W odróżnieniu od pozostałych metryk CK większa wartość metryki oznacza słabszą cechę, tj. słabsze powiązanie. Wartością metryki jest różnica pomiędzy liczbą par metod odwołujących się do przynajmniej jednego wspólnego atrybutu a liczbą par metod odwołujących się do różnych atrybutów. Wysoka wartość oznacza zbyt dużą ilość funkcji, które zostały przydzielone badanej klasie. Wartość zalecona w [4, p. 2] to maksymalnie 1.

7.2.2. Zestaw metryk Metrics of Object Oriented Design

Pierwsze wzmianki dotyczące działań Fernando B. e Abreu i Miguela Goulao pojawiły się w 1994 roku [3, p. 41], zaproponowali wtedy zestaw 6 metryk mierzących aspekty związane z obiektowością. Rok później przedstawili w [1, pp. 3-8] poprawiony zestaw wspominanych metryk. Zestaw metryk skrótowo nazywany jest MOOD i składa się z:

- AHF (od ang. *Attribute Hiding Factor*) oznacza współczynnik ukrycia atrybutów. Wartość tej metryki określona jest przez stopień hermetyzacji klasy, wyrażony poprzez stosunek liczby publicznych atrybutów do łącznej liczby wszystkich atrybutów zdefiniowanych w klasie. Zalecana wartość jest jak najwyższa (bliska 100%), co jest spójne z podstawowymi założeniami paradygmatu obiektowości dotyczącymi hermetyzacji.

- MHF (od ang. *Method Hiding Factor*) oznacza współczynnik ukrycia metod. Metryka ta jest analogiczna do AHF z tą różnicą, że zliczany stosunek dotyczy metod w klasie, tj. stosunek liczby publicznych metod do łącznej liczby wszystkich metod zdefiniowanych w badanej klasie. Następna różnica względem AHF to zalecana wartość, MHF nie powinno być bliskie wartości 100%, ponieważ klasa będzie wtedy zamknięta dla innych klas i jej funkcje w

systemie będą budzić wątpliwości. Wartość zalecona w [17, p. 3] należy do przedziału od 10% do 40%.

- AIF (od ang. *Attribute Inheritance Factor*) oznacza współczynnik dziedziczenia atrybutów. Wartość tej metryki określa wykorzystanie mechanizmu dziedziczenia. Jest wyrażona poprzez liczbę odziedziczonych atrybutów w stosunku do sumy wszystkich atrybutów zdefiniowanych w klasie. Niska wartość może wskazywać na mały stopień powtórnego wykorzystania kodu, czego efektem może być jego powielanie. Wartość zalecona w [17, p. 3] to przedział od 40% do 80%.

- MIF (od ang. *Method Inheritance Factor*) oznacza współczynnik dziedziczenia metod. Metryka ta jest analogiczna do AIF z tą różnicą, że zliczany stosunek dotyczy metod w klasie, tj. liczba odziedziczonych metod w stosunku do sumy wszystkich metod. Niska wartość wskazuje na niski poziom wykorzystania dziedziczenia, wysoka wartość wskazuje na zbyt dużą odpowiedzialność klasy lub niepoprawne wykorzystanie mechanizmu dziedziczenia. Wartość zalecona w [17, p. 4] to przedział od 60% do 80%.

- PF (od ang. *Polymorphism Factor*) oznacza współczynnik polimorficzności. Wartość tej metryki określona jest przez stopień pokrycia metod w podklasach. Jest wyrażona poprzez liczbę metod, które przeładowują dziedziczone metody w stosunku do liczby wszystkich możliwych relacji polimorficznych. Niska wartość wskazuje na niewykorzystanie paradygmatu obiektowości w badanym kodzie, w takim przypadku mogą wystąpić wyniki nieokreślone [2, p. 6]. Wysoka wartość wskazuje na zbyt skomplikowane hierarchie dziedziczenia. Wartość zalecona w [15, p. 6] to około 10%.

- CF (od ang. *Coupling Factor*) oznacza współczynnik powiązań. Wartość tej metryki określona jest przez stopień powiązań pomiędzy klasami w sposób inny niż dziedziczenie, podobnie jak miało to miejsce w CBO z zestawu metryk CK. Wartość jest wyrażona poprzez sumę powiązań pomiędzy parą klas w stosunku do maksymalnej liczby powiązań nie będących dziedziczeniem. Wysoka wartość oznacza zbyt wiele zależności pomiędzy klasami. Wartość zalecona w [17, p. 4] to przedział od 5% do 20%.

7.2.3. Zestaw metryk Roberta C. Martina

W 1994 roku [11, p. 1] Robert C. Martin zaproponował 5 metryk mierzących aspekty związane z obiektowością, szczególnie skupiające się na ba-

daniu zależności pomiędzy pakietami (za pakiet może zostać uznana również przestrzeń nazw lub cała biblioteka). Zestaw ten nazywany jest zestawem Martina i składa się z:

- Ca (od ang. *Afferent coupling*) oznacza powiązania do wewnątrz. Wartość tej metryki to liczba klas spoza badanego pakietu, które zależą od klas w badanym pakiecie. Wysoka wartość oznacza dużą odpowiedzialność badanego pakietu wobec innych. Zaleca się aby wartość ta była jak najmniejsza.

- Ce (od ang. *Efferent coupling*) oznacza powiązania na zewnątrz. Metrykę można uznać za odwrotność metryki Ca. Wartość tej metryki to liczba klas z badanego pakietu, które zależą od klas znajdujących się w innych pakietach. Wysoka wartość wskazuje na dużą zależność badanego pakietu od innych, co przekłada się na niestabilność badanego pakietu. Zaleca się aby wartość była jak najmniejsza.

- A (od ang. *Abstractness*) oznacza abstrakcję. Wartość tej metryki to ilość abstrakcyjnych elementów w badanym pakiecie w stosunku do wszystkich elementów pakietu. Skrajnie wysokie wartości (bliskie 100%) uznawane są za stabilne, ponieważ są wysoce abstrakcyjne. Skrajnie niskie wartości (bliskie 0%) uznawane są za niestabilne i są to konkretne klasy, od których nie powinny w dużym stopniu zależeć inne. Zalecane wartości to skrajne 0% lub 100%.

- I (od ang. *Instability*) oznacza niestabilność. Wartość tej metryki jest wyznaczana na podstawie innych metryk zestawu Martina i wyliczana jest jako wartość Ce w stosunku do sumy Ce + Ca. Jej wartość oznacza podatność na zmiany co przekłada się na stabilność pakietu. Skrajnie niska wartość (bliska 0%) oznacza dużą odpowiedzialność wobec innych pakietów. Przeciwna, skrajnie wysoka wartość oznacza pakiet, który w dużej mierze zależy od innych pakietów. Zalecane wartości to skrajne 0% lub 100%.

- Dn (od ang. *Normalized distance from main sequence*) oznacza znormalizowaną odległość od ciągu głównego. Robert C. Martin do swojego zestawu metryk wprowadził pojęcie ciągu głównego. Ciąg główny to linia łącząca punkty (1, 0) i (0, 1) w układzie współrzędnych metryk A i I. Zalecane położenie klas wskazane na podstawie metryk A i I to jeden z końców ciągu głównego, co oznacza, że są abstrakcyjne i stabilne lub niestabilne i konkretne. Jeśli nie znajdują się na żadnym z końców ciągu to zaleca się aby znajdowały się jak najbliżej linii ciągu głównego [3, p. 55].

7.2.4. Podsumowanie przedstawionych metryk

Wymienione trzy zestawy metryk uzupełniają się wzajemnie: metryki CK dotyczą miar na poziomie klasy; metryki MOOD dotyczą miar na poziomie systemu; metryki Martina dotyczą miar na poziomie pakietów. Jak też bezpośrednio metryki CK i MOOD uzupełniają się w pomiarze spójności (dla CK) oraz hermetyzacji i polimorfizmu (dla MOOD). Występuje również częściowe pokrywanie się tych dwóch zestawów na poziomie poszczególnych metryk. Warto zaznaczyć, że zestawy metryk CK i MOOD powstały w oparciu o podstawy teoretyczne, a zestaw metryk Martina na podstawie doświadczenia ich twórcy [10, p. 17].

Na przestrzeni lat została podjęta niejedna próba weryfikacji wspominanych zestawów metryk. Najważniejsze z nich zostały wymienione w [3, pp. 44-45, 51-53], [10, pp. 17-35] oraz [9, pp. 363-365]. Żadna z tych prób nie wskazała jedynej słuszności któregoś z wymienionych zestawów metryk. Wyjątkiem są badania przedstawione w [12, p. 7] oraz [13, pp. 5-9], gdzie zestaw metryk CK spotyka się z dużą krytyką a zestaw MOOD zostaje uznany jako jeden z lepszych [10, p. 35].

7.3. Przegląd wybranych pozostałych metryk

O ciągłej popularności wymienionych wyżej metryk może świadczyć opisywanie ich w [14, pp. 2-3] oraz w [5, pp. 32-33], jak też obie wymienione publikacje proponują własne metryki, które w pewnym stopniu bazują na innych metrykach (w obu przypadkach na zestawie CK). Podobnie jest z innymi metrykami, które w znacznym stopniu nawiązują do swoich poprzedników.

W przypadku metryki LCOM, B. Henderson-Sellers, L. Constantine oraz Graham [3, p. 49] zaproponowali nowe wersje tej metryki, nazywając je kolejno LCOM2 i LCOM3. Wartość LCOM2 określona jest przez sumę metod, które nie mają dostępu do poszczególnych atrybutów klasy. Wartość LCOM3 określona jest przez liczbę metod, które nie odwołują się do poszczególnych atrybutów klasy. W. Li i S. Henry [3, p. 49] zaproponowali kolejną wersję LCOM4, która następnie w [3, p. 50] doczekała się rozwinięcia przez M. Hintz'a oraz B. Montazerii.

W przypadku zestawu metryk MOOD, jej autorzy w 2001 roku [15, p. 5] zaproponowali 4 nowe metryki wyznaczające współczynnik: efektywności ukrycia atrybutów; efektywności ukrycia metod; dziedziczenia wewnętrznego; polimorfizmu parametrycznego.

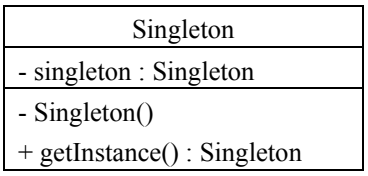
Zestaw metryk Lorenza i Kidda (nazywane metrykami LK) reprezentuje nieco szersze podejście niż wymienione wcześniej metryki, ponieważ metryki te dotyczą całego systemu oraz pojedynczych klas [10, p. 14]. Metryki LK składają się z dwóch grup: projektowych oraz projektowania. Sześć metryk projektowych mierzy ogólne aspekty systemu. Natomiast 27 metryk projektowania odpowiedzialnych jest za dostarczenie informacji na poziomie metod, klas, czy też całego modelu.

7.4. Propozycja kierunków dalszego rozwoju metryk

Wymienione w trzecim punkcie niniejszego rozdziału zalecane wartości dotyczyły ogólnego kodu programu, bez specyfikowania jaką funkcję pełni w systemie bądź z jakich komponentów składa się. Wartości te muszą cechować się szeroką uniwersalnością, przez co w szczególnych przypadkach będą znacząco różne od otrzymywanych wartości z pomiaru metryk. Zmniejszając wspomnianą uniwersalność można uzyskać zalecane wartości metryk, które będą w większym stopniu pokrywały się z wynikami pomiarów nawet w skrajnych przypadkach.

Proponowanym tu kierunkiem rozwoju i wykorzystania metryk jest użycie ich jako jeden z czynników do wyznaczenia jakości zaimplementowanych wzorców projektowych, co zostało opisane w dalszej części rozdziału i jest wstępem do dalszych badań. Wzorce projektowe wykorzystywane w oprogramowaniu to specyficzne struktury kodu programu, które często nie wynikają z paradygmatu programowania obiektowego a najczęściej z dobrych praktyk wielu programistów. Wyznaczenie jakości zaimplementowanych wzorców należy podzielić na przynajmniej dwa etapy: pierwszy odpowiedzialny za wyszukanie wystąpień wzorców; drugi odpowiedzialny za ocenę jego implementacji. W ramach drugiego etapu została podjęta próba wyznaczenia wartości najpopularniejszych metryk, które to będą zalecane dla wzorców projektowych. Wyznaczenie wartości metryk to jeden z czynników po-

trzebnych do końcowej oceny jakości. Należy jednak pamiętać, że każdy wzorzec to unikatowa struktura i dla każdego z osobna należy je wyznaczyć. Dla uproszczenia wyniki zostały przedstawione dla wzorca Singleton, uznawanego za jeden z najprostszych wzorców projektowych. Wykorzystana została najprostsza modelowa implementacja tego wzorca pełniąca podstawową funkcję udostępniania tylko jednej instancji obiektu, rysunek 7.1 przedstawia modelową implementację w notacji UML. Najprostsza implementacja oznacza również podatność na ogólne wady tego wzorca.



Rys. 7.1. Model wzorca Singleton w notacji UML.
Źródło: oprac. własne.

Dla zestawu metryk CK zgodnie z definicją przedstawioną w rozdziale trzecim otrzymano następujące zalecane wartości:

- WMC i RFC - możliwie jak najmniejsza;
- DIT, NOC - brak dziedziczenia w obu przypadkach;
- CBO - zgodnie z zaleceniami dla ogólnego kodu programu, tj. przedstawionymi w rozdziale trzecim;
- LCOM - metryka nie występuje w najprostszej implementacji wzorca Singleton (miałaby znaczenie w przypadku wymogu konfiguracji samej instancji Singletonu).

Wyznaczenie zalecanych wartości dla zestawu metryk MOOD wymagało nadinterpretacji ich definicji, czego wynikiem nie będą współczynniki a po prostu ogólne zalecenia. Nadinterpretacja dotyczy metryk, w których należy odwołać się do więcej niż jednej klasy, tu odwołanie to dotyczy bezpośrednio samego wzorca Singleton oraz klas, które go wykorzystują. Zalecane wartości to:

- AHF - 100% ukrycia atrybutów;
- MHF - tylko jedna metoda publiczna, udostępniająca instancję Singletonu;

- AIF, MIF, PF - zgodnie z zalecanymi wartości dla metryk DIT i NOC z zestawu CK tu również zalecany jest brak dziedziczenia;

- CF - metryka nie występuje w najprostszej implementacji wzorca.

Dla zestawu metryk Martina również konieczna była nadinterpretacja definicji poszczególnych metryk. Nadinterpretacja polega na uznaniu klasy zawierającej wzorzec Singleton za badany pakiet. Zalecane wyniki to:

- Ca – dopuszczalna jest wysoka wartość, niska wartość podważa sens wykorzystania wzorca Singleton;

- Ce, A - możliwie jak najniższa;

- I - możliwie jak najniższa, co wynika z wartości metryk potrzebnych do wyznaczenia I;

- Dn - położenie bliskie początku układu współrzędnych.

Warto zaznaczyć, że zalecane wartości w kilku przypadkach są przeciwne względem zalecanych wartości dla ogólnego kodu programu, wynika to ze specyfiki wzorca Singleton i potwierdza konieczność wyznaczenia zalecanych wartości także dla innych wzorców projektowych. Dla podkreślenia różnic wynikających z zalecanych wartości została przygotowana tabela 7.1 przedstawiająca porównanie metryk w zależności od wybranych mierzonych cech. Podobne badania zostały przedstawione w [16, pp. 8-10] gdzie autorzy dokonali nadinterpretacji wybranych metryk CK, tworząc własne metryki, dla których przedstawili zalecane wartości. W przypadku wzorca Singleton uzyskali podobne zalecane wartości do wartości wspomnianych wcześniej w rozdziale, dla przykładu wartość metryki MAXDIT (pochodna DIT) w obu przypadkach wyniosła zero, co mimo różnej definicji obu metryk oznacza spełnienie tej samej cechy tj. braku dziedziczenia we wzorcu Singleton.

Tabela 7.1. Porównanie metryk w zależności od wybranych mierzonych cech.

Zastosowanie do pomiaru: / Metryka	Dziedziczenie	Hermetyzacja	Polimorfizm	Abstrakcja	Zależności	Złożoność	Ilość elementów
WMC	N; N;	N; N;	N; N;	N; N;	N; N;	T; T;	T; T;
DIT	N; T;	N; N;	N; T;	N; N;	N; T;	N; T;	N; T;
NOC	N; T;	N; N;	N; T;	N; N;	N; T;	N; T;	N; T;
CBO	N; N;	N; N;	N; N;	N; N;	T; T;	T; T;	N; N;
RFC	N; T;	N; T;	N; N;	N; N;	N; T;	N; T;	N; T;
LCOM	N; T;	N; T;	N; N;	N; N;	N; T;	N; T;	N; T;
AHF	N; N;	T; T;	N; N;	N; N;	N; N;	N; N;	N; N;
MHF	N; N;	T; T;	N; N;	N; N;	N; N;	N; N;	N; N;
AIF	N; T;	N; N;	N; N;	N; N;	N; N;	N; N;	N; N;
MIF	N; T;	N; N;	N; N;	N; N;	N; N;	N; N;	N; N;
PF	N; N;	N; T;	N; T;	N; T;	N; N;	N; N;	N; N;
CF	N; N;	N; N;	N; N;	N; N;	N; T;	N; N;	N; N;
Ca	N; N;	N; N;	N; N;	N; N;	T; T;	N; N;	T; T;
Ce	N; N;	N; N;	N; N;	N; N;	T; T;	N; N;	T; T;
A	N; N;	N; N;	N; N;	N; T;	N; T;	N; N;	N; N;
I	N; N;	N; N;	N; N;	N; N;	N; T;	N; N;	N; N;
Dn	N; N;	N; N;	N; N;	N; T;	T; T;	N; N;	N; N;

Pierwszy znak przedstawia ogólną możliwość zastosowania danej metryki do pomiaru wybranej cechy w wzorcu Singleton (na podstawie zalecanych wartości), drugi znak możliwość zastosowania do ogólnego kodu programu. Dopuszczalne znaki to “tak” (T), “nie” (N).

Podsumowanie

Wielu programistów nawet nie wie, że istnieje coś takiego jak metryki oprogramowania. Wśród pozostałej części wielu nie wie jak z nich korzystać i jakie wnioski można z nich wyciągnąć. Metryki nie są nieomylne, dlatego nie mogą być jedyną słuszną wyrocznią do pomiaru oprogramowania. W związku z tym każdy kto sięga po metryki powinien świadomie z nich korzystać i z

dużą odpowiedzialnością interpretować ich wyniki. Każda taka osoba powinna też przed skorzystaniem z metryk poznać ich założenia, aby lepiej zrozumieć wyniki.

Wciąż powstają nowe metryki, ale mimo to popularność wymienionych trzech zestawów metryk nie maleje. Zatem można uznać je za dobre, jednak wciąż należy pamiętać, że nie istnieją uniwersalne metryki oprogramowania.

Bibliografia

1. Abreu F. B., Goulao M., Esteves R.: *Toward the Design Quality Evaluation of Object-Oriented Software Systems*, Proceedings of the 5th International Conference on Software Quality, Austin, Texas, USA, 1995.
2. Bluemke I., Zając P.: *Metryki MOOD w Rational Rose*, e-Informatica, Wrocław, 2003.
3. Bodziechowski B.: *Analiza jakości kodu przy użyciu metryk oprogramowania na wybranych przykładach*, Akademia Górniczo-Hutnicza, Kraków, 2012.
4. Chandra E., Linda E.: *Class Break Point Determination Using CK Metrics Thresholds*, Global Journal of Computer Science and Technology, Cambridge (Massachusetts), USA, 2010.
5. Goel B., M., Bhatia P., K.: *Analysis of Reusability of Object-Oriented System using CK Metrics*, International Journal of Computer Applications, New York, USA, 2012.
6. Jayaswal B. K., Patton P.C.: *Oprogramowanie godne zaufania*, Helion, Gliwice, 2009.
7. Jureczko M.: *Ocena jakości obiektowo zorientowanego projektu programistyczne na podstawie metryk oprogramowania*, PWN, Warszawa, 2008.
8. Jureczko M., Madeyski L.: *Predykcja defektów na podstawie metryk oprogramowania - identyfikacja klas projektów*, PWNT Gdańsk, 2010.
9. Kan S. H.: *Metryki i modele w inżynierii jakości oprogramowania*, PWN SA, Warszawa, 2006.

10. Kielar M.: *Metryki kodu dla programowania obiektowego*, Uniwersytet Warszawski, Warszawa, 2011.
11. Martin R.: *OO Design Quality Metrics An Analysis of Dependencies*, Proc. of Workshop Pragmatic and Theoretical Directions in Object-Oriented Software Metrics, OOPSLA'94, 1994.
12. Mayer T., Hall T.: *A Critical Analysis of Current OO Design Metrics*, South Bank University, London, UK, 1999 .
13. Mayer T., Hall T.: *Measuring OO Systems: A Critical Analysis of the MOOD Metrics*, South Bank University, London, UK, 1999.
14. Michura J., Capretz M. A. M., Wang S.: *Extension of Object-Oriented Metrics Suite for Software Maintenance*, Hindawi Publishing Corporation, Cairo, Egypt, 2013.
15. Podgórski W.: *Metryki obiektowe i ich interpretacja*, Politechnika Wrocławska, Wrocław, 2009.
16. Vernazza T., Granatella G., Succi G., Benedicenti L., Mintchev M.: *Defining metrics for software components*, Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics, Orlando, USA, 2000.
17. Walter B.: *Metryki obiektowe jako wskaźniki jakości kodu i projektu*, Politechnika Poznańska, InMost, Poznań, 2006.
18. Weichbroth P., Orłowski C.: *Przegląd miar oceny oprogramowania*, Politechnika Gdańska, Gdańsk, 2009.

Rozdział 8

Analiza wymagań w metodologiach zarządzania projektami informatycznymi

Niniejszy rozdział koncentruje się na roli analizy wymagań w początkowych fazach procesu planowania prac projektowych projektów informatycznych i formalizacji analizy wymagań w metodolagiach zarządzania projektami. Uwzględnione zostały zarówno metodyki zarządcze (np. PRINCE2, PMBoK), metodyki wytwórcze, a także zwinne metodyki wytwarzania oparte na manifeście agilowym (Agile Manifesto). Analizie zostały poddane poszczególne fazy wybranych metodologii zarządzania projektami informatycznymi w kontekście zbierania i definiowania wymagań klienta.

Abraham Lincoln twierdził, że pierwsze cztery godziny z sześciu przeznaczonych na ścięcie drzewa powinny być poświęcone na ostrzenie siekiery! Ta maksyma niesie przesłanie, iż odpowiednie przygotowanie czyni pracę nie tylko łatwiejszą i sprawniejszą, ale pozwala także na zmniejszenie ryzyka niepowodzeń. Fred Brooks, amerykański informatyk, laureat nagrody Turinga z 1999 roku, wyraża opinię, iż najtrudniejszym elementem wszystkich projektów programistycznych jest decydowanie, co ma stanowić efekt projektu. Najważniejszą funkcją, jaką pełni twórca oprogramowania w relacji z klientem, mającą ogromny wpływ na wynik projektu, jest iteracyjne zbieranie i precyzowanie wymagań [5, s. 67]. Profesor Z. Szyjewski natomiast zwraca uwagę na istotność precyzyjnego określenia celów i założeń w pierwszej fazie realizacji projektu informatycznego, gdyż prawidłowa realizacja tej fazy warunkuje podejmowane decyzje i w istotny sposób wpływa na sukces całego projektu [18, s. 60].

8.1. Wybrane metodologie zarządzania projektami informatycznymi

W literaturze poświęconej zagadnieniom zarządzania projektami można zidentyfikować co najmniej kilkanaście zdefiniowanych metodyk zarządzania opracowanych przez dominujące firmy lub organizacje informatyczne. Poniższa tabela zawiera przegląd najczęściej stosowanych metodologii zarządzania projektami informatycznymi. Z uwagi na różny charakter projektów informatycznych (produkcyjne lub wdrożeniowe) i różne potrzeby zarządcze, niżej wymienione metodyki zostały sklasyfikowane w czterech kategoriach: zarządcze, wytwórcze, adaptacyjne i organizacyjne [10]. Metodyki te charakteryzują się także różnym spojrzeniem na projekt, inną perspektywą, stopniem szczegółowości wymagań dokumentacyjnych i definicji procesów, a także zróżnicowanym wskazaniem na istotność poszczególnych etapów i zadań menedżerskich. Poniższa tabela 8.1 przedstawia wybrane metodologie zarządczych, wytwórczych i adaptacyjnych metodologii zarządzania projektami informatycznymi.

Tabela 8.1. Charakterystyka wybranych metodologii zarządzania projektami informatycznymi

Nazwa skrócona	Pełna nazwa i charakterystyka metodyki	Kategoria metodologii
PRINCE2	PRojects IN a Controlled Environment 2 (*tłum. ty w sterowanym środowisku*). Metodyka mająca sowanie w zarządzaniu różnego rodzaju projektami (nie tylko informatycznymi). Przeznaczona jest dla wsparcia wybranych aspektów zarządzania projektem. Charakteryzuje się ściśle zdefiniowanym modelem procesów podzielonym na trzy obszary: zarządzania strategicznego, zarządzania operacyjnego oraz czania produktów [15]. Zastosowanie tej metodyki daje możliwość standaryzacji na wysokim poziomie, gółową dokumentację i kontrolę jakości przebiegu poszczególnych etapów projektu. Przeciwnicy tej todologii jako jej słabość wskazują zbyt duży nacisk na formalizację i dokumentację działań (nadmierna	Zarządcze

Nazwa skrócona	Pełna nazwa i charakterystyka metodyki	Kategoria metodologii
	kratyzację), co wiąże się z brakiem „zwinności" metodyki.	
PMBoK	Project Management Body of Knowledge nazywana jest też metodyką PMI (Project Management Institute). PMBoK stanowi alternatywę dla PRINCE2, jednak obejmuje pewne obszary, których brakuje w PRINCE2 (np. zarządzanie zasobami ludzkimi i zaopatrzeniem [10, s. 31]. PMBoK to kompendium wiedzy zbudowane w celu dostarczania kierownikowi projektu reguł i metod działania w całym cyklu życia projektu. Wyodrębniono w nim 42 procesy podzielone na pięć grup: inicjowania, planowania, monitorowania i kontroli, wykonawcze oraz zamknięcia. Metodyka zawiera dziewięć obszarów wiedzy związanej z zarządzaniem projektami (zarządzanie integralnością projektu, zakresem, czasem, kosztami, jakością, zasobami ludzkimi, komunikacją, ryzykiem i zaopatrzeniem), a do każdego obszaru przypisane są	
RUP/Eclipse	Rational Unified Process jest procesem iteracyjnego, przyrostowego sposobu wytwarzania oprogramowania (wywodzi się z modelu spiralnego). Twórcą tej dyki jest firma Rational Software Corporation (przejęta w 2003 roku przez IBM). Metodyka ta charakteryzuje się dużą elastycznością i możliwością łączenia z mi metodykami zarządczymi (np. PRINCE2). RUP jest bezpośrednio powiązany z językiem UML[3], a główną ideą RUP jest inicjacja prac na trzech poziomach: wysokim poziomie abstrakcji, niskim poziomie trakcji i prac wytwórczych. Eclipse jest natomiast bardzo rozbudowanym środowiskiem umożliwiającym zarówno planowanie, dokumentację jak i wytwarzanie	Wytwórcze

[3] UML – ang. Unified Modelling Language

Nazwa skrócona	Pełna nazwa i charakterystyka metodyki	Kategoria metodologii
	różnego rodzaju aplikacji i systemów. W przypadku RUP i narzędzi Eclipse na szczególną uwagę ją: Ecplise Process Framework Composer oraz standard OpenUP (Open Unified Process).	
MSF	Metodologia Microsoft Framework Solution powstała w 1994 i jest wynikiem wieloletnich doświadczeń pracowników pionu Microsoft Consulting Services. Sam Microsoft (na stronach MSDN[4]) definiuje dologię MSF jako adaptacyjne podejście ce pomyślne dostarczanie rozwiązań technologicznych. MSF koncentruje się napięciu głównych aspektach zarządzania projektami, tj.: dopasowaniu celów sowych i technologicznych, zdefiniowaniu jasnych celów projektu, ról i zakresu obowiązków, implementacji iteracyjnego procesu sterowanego kamieniami milowymi/punktami kontrolnymi oraz proaktywnym zarządzaniu ryzykiem i skutecznym reagowaniu na zmiany. Obecna wersja MSF 4.0 została opracowana w 2005 roku i podobnie jak wyżej opisane metodyki jest oparta na modelu procesów.	
XP	eXtreme Programming, powstał na początku lat 90-tych XX wieku. Kent Beck, uznawany za twórcę tej metodyki, zdefiniował cztery główne wartości: komunikację, prostotę, pętlę zwrotną oraz śmiałość. Dodatkowo metodykę tą wzbogacono o tzw. Stand-up-meetings, czyli szybkie spotkania na stojaka, mające na celu omówienie bieżących efektów projektu. Brak dokładnej specyfikacji dla tej metodyki oraz odejście od zarządzania procesowego daje powody uznawać XP bardziej za zbiór dobrych praktyk, niż pełną metodykę zarządzania projektami.	Adaptacyjne (zwinne, lekkie, ang. Agile)

[4] Źródło: http://msdn.microsoft.com/pl-pl/library/jj161047.aspx

Nazwa skrócona	Pełna nazwa i charakterystyka metodyki	Kategoria metodologii
Scrum	Twórcami Scrum są Ken Schwaber i Jeff Sutherland. Scrum jest „metodą, przy użyciu której ludzie mogą z powodzeniem rozwiązywać złożone problemy adaptacyjne, by w sposób produktywny i kreatywny wytwarzać produkty o najwyższej możliwej wartości" [17]. Scrum nie jest procesem, a opisuje ogólne sposoby postępowania, w obrębie których możliwe jest stosowanie różnego rodzaju procesów i technik. Scrum zakłada utworzenie zespołów scrumowych, powiązanych z nim ról, zdarzeń artefaktów i reguł postępowania. Wszystkie elementy są niezbędne do osiągnięcia sukcesu.	
DSDM	Dynamic Systems Development Method została ponowana przez brytyjskie konsorcjum DSDM. dologia ta jest związana z RAD (Rapid Application Development) co oznacza szybkie wytwarzania cji. Sednem DSDM jest korzystanie z dużej ilości towych komponentów i wykorzystaniu technik typowania. W DSDM zdefiniowano 15 ról dla ków zespołu projektowego, wyznaczając każdej roli odpowiednia zadania i zakres odpowiedzialności. Jedną z podstawowych technik DSMD jest Timeboxing[5] (zasada 80%/20%) oraz MoSCow[6].	
FDD	Feature Driven Development należy również do tzw. zwinnych metodyk inżynierii oprogramowania. FDD jest przykładem procesu skoncentrowanego na	

[5] Timeboxing – jest techniką zarządzania, polegająca na ustaleniu konkretnej daty zakończenia projektu i podzielenia planu na okresy czasowe (Timeboxes), z reguły trwające od 2-6 tygodni. Każdy z okresów ma wyznaczone własne cele, budżety i terminy (deadline) [12].

[6] MoSCoW – jest metodą priorytetyzacji opartą na czterech kategoriach (Must have – funkcjonalność obowiązkowa; Should have – powinno mieć; Could have – funkcjonalność potrzebna, ale warunkowo; Won't have – możliwe do implementacji w przyszłości, nie teraz) [1].

Nazwa skrócona	Pełna nazwa i charakterystyka metodyki	Kategoria metodologii
	cjonalności. Priorytetem w tej metodyce jest wytwarzanie funkcjonalnego oprogramowania w powtarzalny sposób. Jedną z częściej wykorzystywanych technik w FDD jest programowanie eksptremalne.	

Źródło: Opracowanie własne na podstawie studiów literaturowych.

Wybór i wdrożenie odpowiedniej metodologii zarzadzania projektami informatycznymi nie jest łatwy i w zasadzie także jest projektem. Każda z metodologii w inny sposób podchodzi o kluczowych zagadnień i koncentruje się na innych priorytetach, w różny sposób definiuje struktury, procesy, role i zadania. Do czynników, które mogą pomóc w wyborze i wdrożeniu metodologii mogą pomóc m. in.: zrozumienie celów projektu, korzyści i wpływu efektów na firmę/organizację, silne wsparcie ze strony decydentów, dobra znajomość najważniejszych metodologii i podjęcie próby wdrożenia ich z odpowiednim dostosowaniem do warunków otoczenia, odpowiednie szkolenie dla pracowników mających pracować zgodnie z wdrażaną metodologią. Nie bez znaczenia jest także zbudowanie odpowiedniego i kompetentne zespołu, w tym także ustanowienie lidera-„właściciela" metodologii, który będzie odpowiedzialny za jej stosowanie.

8.2. Zarządzanie wymaganiami i klasyfikacja wymagań

Problem analizy wymagań klienta jest elementem lub etapem praktycznie każdej metodyki zarządzania projektami. Jednak podejście do tego zagadnienia w jest zróżnicowane. W niniejszym podrozdziale zostaną przybliżone najczęściej stosowane metodyki zarządzania projektami ze szczególnym uwzględnieniem etapu analizy wymagań.

Jednym z najbardziej ogólnych, a jednocześnie najbardziej rozpowszechnionym w branży informatycznej modelem klasyfikacji wymagań jest opracowany przez Roberta Grady i upowszechniony głownie przez firmę Helwett-Packard model FURPS [8]. Poza wymaganiami funkcjonalnymi (czyli

tym co dane oprogramowanie powinno robić), wyszczególnia się szereg ważnych wymagań pozafunkcjonalnych, do których należą: użyteczność, niezawodność, wydajność i przystosowalność. Wymagania niefunkcjonalne mogą być rozszerzone o kolejne elementy, które tworzą model FURPS+, a zawierają dodatkowo:

- wymagania dotyczące wyglądu (ang. Design requirements),
- wymagania dotyczące implementacji (ang. Implementation requirements),
- wymagania dotyczące interfejsu (ang. Interface requirements),
- wymagania fizyczne (ang. Phisical requirements).

Na rysunku 8.1. przedstawiono model FURPS+, zawierający klasyfikacyjny kanon elementów analizy wymagań rozszerzony o dodatkowe wymagania pozafunkcjonalne.

Wymagania FURPS+	
Wymagania funkcjonalne **F** (ang. functionality) + **D** - wymagania dotyczące wyglądu (ang. Design requirements), **I** - wymagania dotyczące implementacji (ang. Implementation requirements), **I** - wymagania dotyczące interfejsu (ang. Interface requirements), **P** - wymagania fizyczne (ang. Phisical requirements)	**Wymagania pozafunkcjonalne** **U** - użyteczność (ang. usability) **R** - niezawodność (ang. reliability) **P** - wydajność (ang. performance) **S** - przystosowalność (ang. supportability)

Rys. 8.1. Klasyfikacja wymagań – model FURPS+
Źródło: Opracowanie własne na podstawie: [19, s. 20] oraz [6]

Według IBM podczas weryfikacji wymagań funkcjonalnych zazwyczaj należy brać pod uwagę następujące element [6]:

- Audytowanie (ang. Auditing) - mające zapewnić odpowiednie ścieżki audytu wykonywania systemu.

- Licencjonowanie (ang Licencing) – mające zapewnić śledzenie, instalowanie i monitorowanie użycia licencji.
- Lokalizacja (ang. Localization) – mające zapewnić obsługę wielojęzyczności systemu.
- Wiadomości (ang. Mail) – mające zapewnić funkcję umożliwiającą wysyłanie i odbieranie wiadomości.
- Pomoc online (ang. Online help) – mające zapewnić możliwość wsparcia przez Internet.
- Drukowanie (ang. Pronting) - mające zapewnić możliwość drukowania.
- Raportowanie (ang. Reporting) – mające zapewnić możliwość raportowania (np. błędów, stanu systemu itp.)
- Bezpieczeństwo (ang. Security) – mające zapewnić usługi zabezpieczające dostęp do zasobów lub informacji.
- Zarządzanie systemem (ang. System management) – mające zapewnić funkcjonalność zarządzania aplikacją w środowisku dystrybucyjnym.

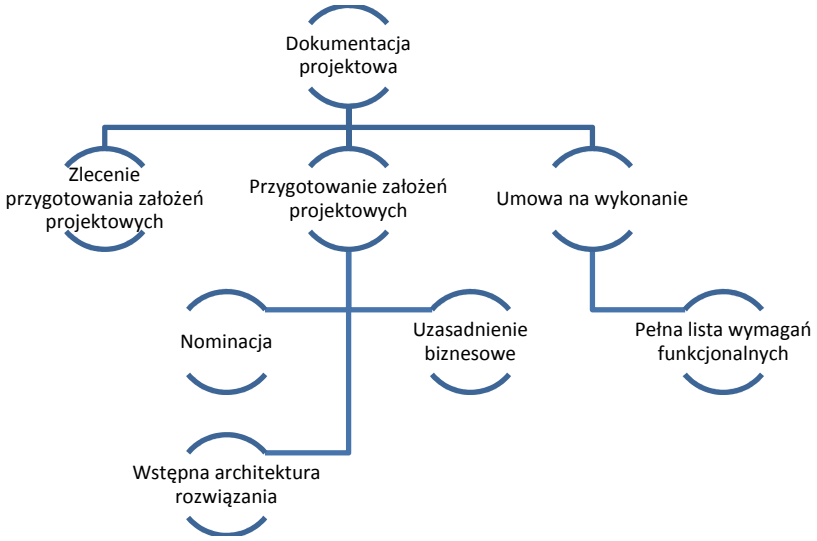
Przepływ pracy (ang. Workflow) – mający zapewnić wsparcie dla obiegu dokumentów i innych zadań, włączając w to proces recenzowania i zatwierdzania dokumentów.

8.3. Wybrane metodyki zarządzania projektami informatycznymi - charakterystyka i podejście do analizy wymagań

Podejście do analizy wymagań w metodyce PRINCE2

PRINCE2 (ang. PRoject IN Controlled Environment) w praktyce oznacza właściwe udokumentowanie powodów uruchomienia projektu, jego przebiegu oraz zamknięcia [10, s. 13]. Metodyka ta nie jest dedykowana jedynie dla projektów IT i może być z powodzeniem stosowana w innego rodzaju przedsięwzięciach. PRINCE2 koncentruje się na procesach, a w szczególności na sposobach podejmowania decyzji, kwestiach zarządczych i właściwy dostosowaniu metodologii do potrzeb chwili. Składa się z ośmiu procesów, ośmiu komponentów i trzech technik. Pierwszym procesem jest „Uruchomienie Projektu/Przygotowania Założeń Projektu". W tym procesie znajduje się odpo-

wiedni dla analizy wymagań etap „Przygotowanie Założeń Projektowych PZP". Efektem tego podprocesu powinna być „Pełna lista wymagań funkcjonalnych", opracowana w formie spójnego dokumentu. Etap przygotowania złożeń projektowych przedstawia poniższy schemat (rys. 8.2.):



Rys. 8.2. Fragment struktury podziału pracy stworzonej wg metodyki PRINCE2
Źródło: Opracowanie własne na podstawie: [10, s. 22].

Na dalszym etapie realizacji projektu należy opracować specyfikację techniczną oraz plan zarządzania zmianą (jak i wiele innych czynników), które mogą mieć wpływ na zmianę wymagań lub doprecyzowanie wymagań (zwłaszcza tych pozafunkcjonalnych).

Więcej na temat metodyki PRINCE2 można znaleźć w opracowaniach [10] lub „PRINCE2® Skuteczne Zarządzanie Projektami" polskojęzyczna wersja podręcznika opracowanego przez Office of Government Commerce (OGC).

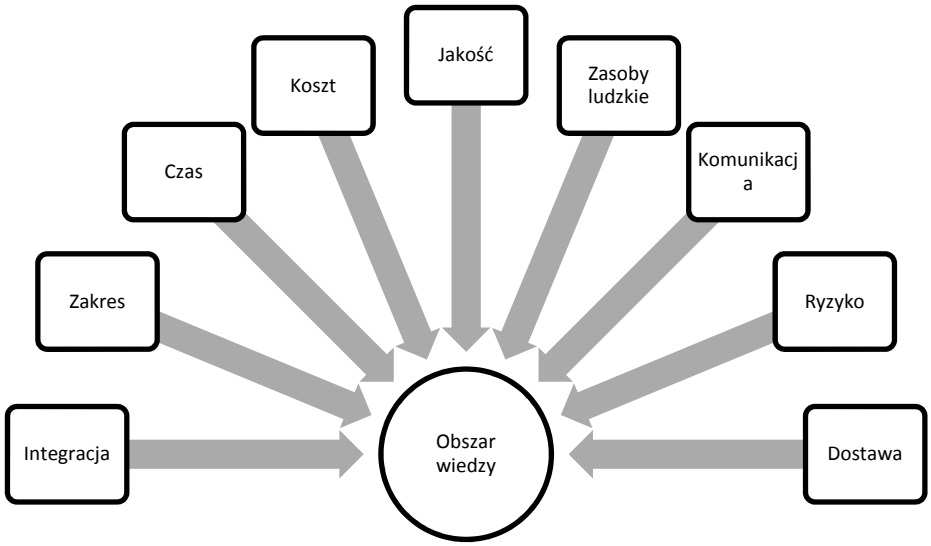
Podejście do analizy wymagań w metodyce PMBoK

PMBoK (ang. Project Management Body of Knowledge) jest podejściem konkurencyjnym w stosunku do PRINCE2. Koncentruje się na zbieraniu dobrych praktyk i zaleceń związanych z zarządzaniem projektami w ramach

zdefiniowanych obszarów wiedzy. Wkracza tym samym w obszary „nie obsługiwane" przez PRINCE2 tj. zarządzanie zasobami ludzkimi i zaopatrzeniem. Obecna czwarta wersja PMBoK charakteryzuje się, podobnie jak PRINCE2, podejściem procesowym i definiuje 5 grup procesów:

1. rozpoczęcia,
2. planowania,
3. realizacji,
4. kontroli,
5. zakończenia.

Zdefiniowano także dziewięć obszarów wiedzy (przedstawionych na rysunku poniżej) oraz przyporządkowano każdy z pięciu procesów do jednego z obszarów wiedzy.



Rys. 8.3. Obszary wiedzy w PMBoK
Źródło: Opracowanie własne.

Zbieranie wymagań interesariuszy i ich udokumentowanie jest jednym z 44 zdefiniowanych procesów PMBoK (wersja 4) i znajduje się w grupie procesów planistycznych, obszar wiedzy: zakres. Dokładna nazwa procesu to 5.1 Zbieranie wymagań (ang. 5.1 Collect Requirements). PMBoK definuje tutaj

także techniki (zalecenia co do stosowanych metod) związane z realizacją tego etapu zarządzania projektem. Są nimi:

- Wywiad – spotkania, których celem jest poznanie potrzeb interesariuszy,
- Grupy specjalistyczne – ustalenia zakresu, faktów, potrzeb w toku dyskusji ekspertów,
- Warsztaty – uzgodnienie kwestii spornych, wypracowanie kompromisu, zacieśnianie współpracy pomiędzy zespołem projektowym, a interesariuszami,
- Techniki grupowej analizy kreatywnej – np. burza mózgów, mapa myśli/pomysłów, diagramy pokrewieństwa,
- Techniki podejmowania decyzji grupowych – jednomyślności, większości, demokracji, dyktatu,
- Ankiety,
- Obserwacja,
- Prototypowanie.

Efektem tego procesu są dokumenty zawierające wymagania (ang. Requirements Documentation), plan zarządzania wymaganiami (ang. Requirements Management Plan) oraz matryca śledzenia wymagań (ang. Requirements Traceability Matrix). Ten ostatni element jest tabelą łączącą wymaganie z potrzebą biznesową lub inną przyczyną jej powstania.

Kolejnym procesem istotnym z punktu widzenia analizy wymagań jest proces oznaczony numerem 5.2 „Definiowanie zakres projektu" (ang. Define Scope). Oczywiście ten proces może zostać uruchomiony po zakończeniu etapu zbierania wymagań (często na wejściu procesu wymagany jest dokument zawierający wszystkie wymagania), jednak może on być traktowany także jako tzw. lista życzeń. To kierownik projektu definiuje jakie funkcjonalności, w jakiej formie i jakości, można zrealizować uwzględniając ograniczenia czasowe, budżetowe i posiadanych zasobów.

Szerzej na ten temat napisano w pracach: [14], [10] oraz w serwisach internetowych http://zarzadzanieprojekt.pl i http://www.pmi.org.pl/wiedza/biblioteka-pm/.

Analiza wymagań w Rational Unified Process

RUP jest bezpośrednio powiązany z językiem modelowania UML (ang. Unified Modelling Language) i jest szablonem przyrostowego sposobu wytwarzania oprogramowania (model spiralny). W tym podejściu zakładamy, że jest ustalona pewna wstępna wizja projektu i zezwolenie na analizę biznesową uzasadniającą potrzebę jej realizacji. RUP jest powszechnym standardem wykorzystywanym przez tysiące firm na całym świecie. Jego niekwestionowaną zaletą jest zestaw gotowych do użycia rozwiązań wspomagających ustalenie kluczowych kamieni milowych projektu, a także zasobów potrzebnych do jego realizacji. Główną ideą metodyki jest uruchomienie prac z uwzględnieniem podziału na dziewięć dyscyplin projektowych i cztery fazy (przedstawione w tabeli 8.2), z których każda może zawierać jedną bądź wiele iteracji.

W metodyce RUP analiza wymagań związana jest z fazą opracowania, a jej głównymi produktami są: wszystkie istotne przypadki użycia (około 80%), architektura całego rozwiązania, plan projektu, plan jakości i lista ryzyk. Wymagania są tu traktowane jako lista stanowiąca niejako pomost pomiędzy kwestiami biznesowymi a technicznymi. Ich zapis powinien być zrozumiały dla przyszłych twórców rozwiązania, w tym szczególnie programistów.

Rozwinięcie metodyki RUP jest obecnie bezpośrednio powiązane z open source'owym projektem Eclipse (Eclipse Process Framework – EPF), który ma dwa cele: opracowanie rozszerzalnego *frameworka* i narzędzi wykonawczych dla procesów inżynierii oprogramowania oraz zapewnienie przykładowego i rozszerzalnego zbioru procesów wspierających iteracyjne, zwinne i przyrostowe tworzenie oprogramowania.

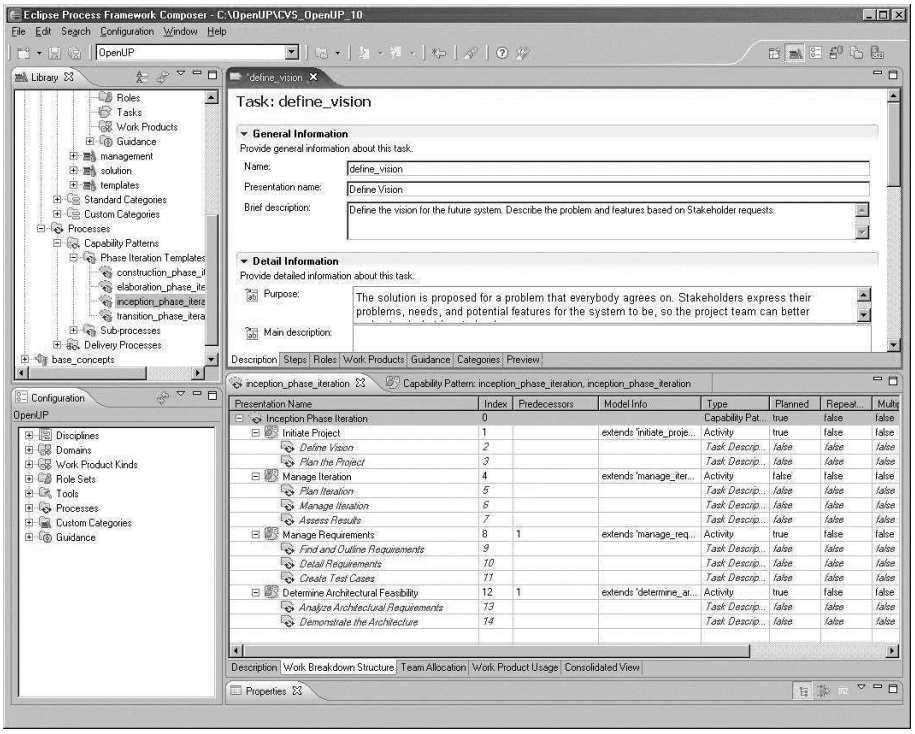
W tym celu opracowano narzędzia: Exemplary tool: EPF Composer oraz OpenUP (ang. Open Unified Process). EPF Composer jest narzędziem służącym do budowy własnych procesów deweloperskich oprogramowania poprzez użycie predefiniowanych schematów (rys. 8.4. przedstawia przykładowy ekran aplikacji Ecplise Process Framework Composer). OpenUP natomiast jest implementacją iteracyjnego, przyrostowego, ustrukturyzowanego cyklu życia aplikacji.

Dokładne opisanie każdego procesu oraz wsparcie narzędziowe pozwala zespołom pracującym zgodnie z ta metodykę koncentrować się na odpowiednich zadaniach.

Tabela 8.2. Architektura RUP.

		Fazy (Phases)							
	Dyscypliny (Dysciplines)	Rozpoczęcie (Inception)	Opracowanie (Elaboration)		Konstrukcja (Construction)			Przekazanie (Transition)	
Elementy metody RUP (Method Content)	Modelowanie biznesowe (Business Modeling)								
	Wymagania (Requirements)								
	Analiza i Projektowanie Analysis & Design)								
	Implementacja (Implementation)								
	Testy (Test)								
	Wdrożenie (Deployment)								
	Konfiguracja i zarządzanie zmianą (Configuration and change management)								
	Zarządzanie projektem (Project Management)								
	Środowisko (Environment)								
		Początek (Initial)	Etap 1 (E1)	Etap 2 (E2)	C 1	C 2	C N	T1	T2
		Iteracje (Iterations)							
		Proces (Process)							

Źródło: Opracowanie własne na podstawie: Help – Eclipse Process Framework Composer.



Rys. 8.4. Ecplise Process Framework Composer
Źródło: http://www.eclipse.org/epf/images/epf_composer.jpg

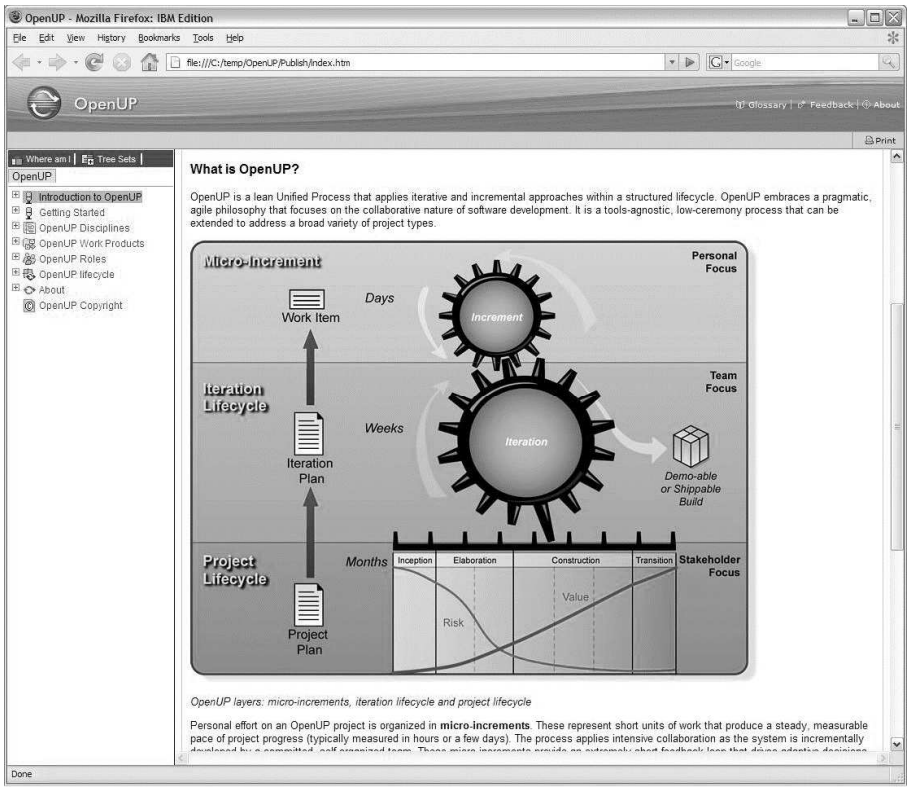
Gotowe wzorce opisu procesów związanych z analizą wymagań (przedstawione na rysunku 8.4) oraz procesów związanych z realizacją całego projektu (zdefiniowane w standardzie OpenUP – rysunek 8.5) dają duże możliwości adaptacyjne niemal dla każdego rodzaju projektów informatycznych.

Z punktu widzenia analizy wymagań w metodyce RUP na szczególną uwagę zasługują predefiniowane w narzędziu Ecplise Process Framework Composer wzorce procesów z grupy Inception Phase Iteration, do których należą m.in.:

- Proces inicjacji projektu (Initiate Project),
  - Wizja projektu,
  - Plan projektu,
- Zarządzanie iteracjami (Manage Iteration),
  - Plan iteracji,
  - Zarządzanie iteracjami,

  - Ocena rezultatów,
- Zarządzanie wymaganiami (Manage Reqiurements),
  - Znajdowanie i określanie wymagań,
  - Uszczegółowienie wymagań,
  - Tworzenie przypadków testowych,
- Ustalanie wykonalności architektonicznej (Determine Architectural Feasibility)
  - Analiza wymagań architektonicznych,
  - Zademonstrowanie architektury.

Szczegółowy opis wszystkich procesów, zaleceń, narzędzi, zaleceń i dobrych praktyk znajduje się m. in. w standardzie OpenUP (dostępny na stronie www.eclipse.org), którego dokumentację przedstawia rysunek 8.5.



Rys. 8.5. OpenUP
Źródło: http://www.eclipse.org/epf/images/openup.jpg

Analiza wymagań w Microsoft Solution Framework

Microsoft Solution Framework jest metodyką rozwijaną od 1994 roku i stosowaną przez Microsoft. Jest nazywane często schematem („framework") wskazując odrębność od pełnej metodologii. Od tego czasu MSF ewoluowało bazując na sukcesach i wieloletnim doświadczeniu swoich pracowników. Obecnie struktura MSF jest zarządzana i rozwijana przez grupę profesjonalistów oraz wspomagana i weryfikowana przez międzynarodowa grupę doradczą [3]. Należy nadmienić, iż nie jest jedną metodyką polecaną i stosowaną przez giganta z Redmond. Na oficjalnych stronach bazy wiedzy Microsoft http://msdn.microsoft.com/ można znaleźć również opisy metodologii „odchudzonych" Scrum i innych. MSF koncentruje się na pięciu zasadniczych elementach [20]:

- Dopasowanie celów biznesowych i technologicznych,
- Ustanowienie jasnych celów projektu, ról i obowiązków,
- Implementowanie iteracyjnego procesu sterowanego kamieniami milowymi/punktami kontrolnymi,
- Proaktywne zarządzanie ryzykiem,
- Reagowanie na zmiany skutecznie.

Rys. 8.6. Model MSF. Źródło: [11]

Pełny model MSF przedstawia rysunek 8.6 (opisy oryginalne, anglojęzyczne).

Filozofia MSF oparta jest także na modelu zespołu MSF, który dzieli zespół na odpowiednie segmenty. Każdy segment ma zdefiniowane cele i zakresy odpowiedzialności związane z realizacją projektu. Poniżej wypunktowano najważniejsze role w zespole MSF, które można łączyć w sytuacjach dotyczących małych zespołów i rozszerzyć w sytuacjach dotyczących dużego zespołu. Role te nie narzucają żadnego schematu organizacyjnego lub propozycji stanowisk i często są zróżnicowane w poszczególnych zespołach i firmach.
Role zespołów projektowych MSF są następujące [20]:

Rola: Menedżer produktu
Cele:

- Zapewnienie, że rozwiązanie dostarcza odpowiednią wartość biznesową,
- Definicja rozwiązań w ramach ograniczeń projektu,
- Zapewnienie, że potrzeby i oczekiwania klientów będą spełnione.

Rola: Menedżer programu
Cele:

- Dostarczanie rozwiązań w ramach ograniczeń projektu,
- Konfigurowanie środków, dzięki którym potrzeby i oczekiwania sponsora zostaną spełnione,

Rola: Architekt
Cele:

- Projektowanie rozwiązania spełniającego cele biznesowe w ramach ograniczeń projektu.

Rola: Deweloper
Cel:

- Budowanie rozwiązań zgodnych ze specyfikacjami.

Rola: Doświadczenie użytkownika
Cele:

- Maksymalizowanie użyteczności rozwiązania,

- Zwiększanie gotowości i skuteczności użytkowników,
- Zapewnienie, że potrzeby i oczekiwania użytkowników będą spełnione.

Rola: Tester
Cel:

- Zatwierdzanie rozwiązań do publikacji (tylko po upewnieniu się, że wszystkie aspekty rozwiązania spełniają lub przekraczają ich odpowiednie zdefiniowane poziomy jakośc).

Rola: Wydanie/Operacje
Cele:

- Sprawne wdrażanie i przejście do wdrożeń,
- Zapewnienie, że potrzeby i oczekiwania działu informatycznego/operacji biznesowych są spełnione.

Analiza wymagań w tej metodologii związana jest przede wszystkim z rolą „Zarządzanie produktem". W skład tej roli wchodzą takie obszary funkcjonalne jak: marketing/komunikacja korporacyjna, analiza biznesowa, planowanie produktu. Bazy Microsoft Technet podają przykłady zastosowania MSF do projektów informatycznych, gdzie elementy analizy wymagań metodologii MSF sprowadzają się do dwóch faz projektu [19]: fazę opracowania wizji (ang. Envisioning Phase) oraz fazę planowania (ang. Planning Phase).

Faza opracowania wizji obejmuje między innymi zadania:

1. Wprowadzenie i cele /w tym biznesowe oraz dotyczące wyglądu/ (ang. Introduction and goals /business and design/),
2. Utwórz wykaz problemów (ang. Create the problem statement),
3. Utwórz Wizji (ang Create the vision statement),
4. Zdefiniuj profile użytkowników (ang. Define end user profiles),
5. Oceń bieżącą sytuację /na ogólnym/wysokim poziomie/ (ang. Assess the current situation /high level/),
6. Zidentyfikuj wymagania „wysokiego" poziomu (ang. Identify high level requirements);
7. Zdefiniuj zakres projektu (ang. Define the project scope),
8. Zdefiniuj koncepcję rozwiązania (ang. Define the solution concept),

9. Stwórz zespół (ang. Set up a team),
10. Określ strukturę projektu (ang. Define the project structure),
11. Oceń ryzyko (ang. Assess risk),

z których pierwsze 9 zadań związanych jest z opracowaniem tzw. dokumentu „Wizji i zakresu projektu" (ang. Vision/sciope document). Osobami (rolami) odpowiedzialnymi za te zadania są menedżer produktu oraz menedżer programu.

Wykaz problemów (ang. problem statement) jest zazwyczaj krótkim dokumentem opisującym specyfikę przypadków, wymagań i życzeń biznesowych klientów w stosunku do całego projektu. Im bardziej precyzyjnie zostanie on zdefiniowany i opisany, tym łatwiej będzie mierzyć sukces projektu.

Zalecenia dotyczące „wizji" projektu podkreślają, aby wszelkie ustalenia charakteryzowały się podstawowymi cechami SMART, czyli wizja powinna być: specyficzna (ang. Specific), mierzalna (ang. Measurable), osiągalna (ang. Achievable), odpowiednia (ang. Relevat) i osadzona w czasie (ang. Time-based).

Definiowanie profili użytkowników natomiast pozwala zrozumieć co jest krytycznym czynnikiem sukcesu projektu z punktu widzenia użytkowników, dla których dany projekt jest tworzony. W celu zebrania jasnych i zrozumiałych opisów od każdego z użytkowników zespół projektowy odpowiedzialny za to zadanie tworzy odpowiednie klasy użytkowników. Proces profilowania użytkowników rozwija zestaw wymagań użytkowników. Połączenie tych wymagań z wymaganiami biznesowymi i wymaganiami dotyczącymi wyglądu (design) pomoże zdefiniować późniejszy zakres projektu, będący efektem fazy opracowania wizji projektu (ang. Envisioning Phase).

Faza planowania (ang. Planning Phase) ma za zadanie dostarczyć następujących elementów:

- Specyfikacji funkcjonalnej.
- Głównego planu projektu.
- Głównego harmonogramu projektu.
- Zaktualizowanego głównego dokumentu szacowania ryzyka.

W metodyce MSF bezpośrednio z analizą wymagań związana jest specyfikacja funkcjonalna. Jest ona pewnego rodzaju repozytorium projektu (najczęściej wirtualnym) i powiązanych z nim artefaktów wyglądu opracowanych

podczas fazy planowania. Artefakty są przede wszystkim wynikiem działań projektowych podczas projektowania koncepcyjnego, projektowania logicznego i fizycznego procesów fazy planowania. Artefakty mogą zawierać modele, takie jak diagramy przypadków, scenariuszy użycia, lista funkcji, zrzuty ekranu interfejsu użytkownika, projektowanie bazy danych lub inne. Kluczowe cele specyfikacji funkcjonalnej to: konsolidacja wspólnego rozumienia wymagań biznesowych, dot. wyglądu i wymagań użytkowników, podzielenie problemu (projektu) i logiczne zmodularyzowanie rozwiązań, wprowadzenie planu ramowego, zaplanowanie i zbudowanie rozwiązania. Specyfikacja funkcjonalna ma również posłużyć jako element umowy między zespołem projektowym a klientem i innymi interesariuszami.

Specyfikacja funkcjonalna MSF powstaje w trakcie realizacji zadań:

- Kompletowania szczegółowych ocen bieżącego środowiska (ang. Complete detailed assessment of the current environment).
- Opracowanie projektu rozwiązania i architektury (ang. Develop the solution design and architecture).
- Walidacja technologii (ang. Validate the technology).

Analiza wymagań w eXtreme programming

Adaptacyjny sposób wytwarzania oprogramowania to przede wszystkim zbiór praktyk przedkładających tzw. „miękkie" techniki zarządcze nad z góry zdefiniowanymi i obowiązkowymi procesami i narzędziami. Oznacza to między innymi zrezygnowanie z centralnego zarządzania projektami na rzecz trybu koordynacyjnego (przekazanie uprawnień w zakresie podejmowania decyzji dla koordynatorów). Metodyki tego typu stawiają na samoorganizujące się zespoły, które aktywnie realizują cele przedsięwzięcia. Daje to możliwość szybszej reakcji na pojawiające się błędy, defekty, czy niewłaściwe decyzje i w konsekwencji skuteczniej dostarczać konkretną wartość biznesową (osiągać cele).

eXtreme Programming nie doczekało się jeszcze precyzyjnej specyfikacji. Zastosowanie tej metodologii jest uzależnione od kontekstu konkretnej organizacji. Główne wartości tego sposobu zarządzania ujęto w cztery filary:

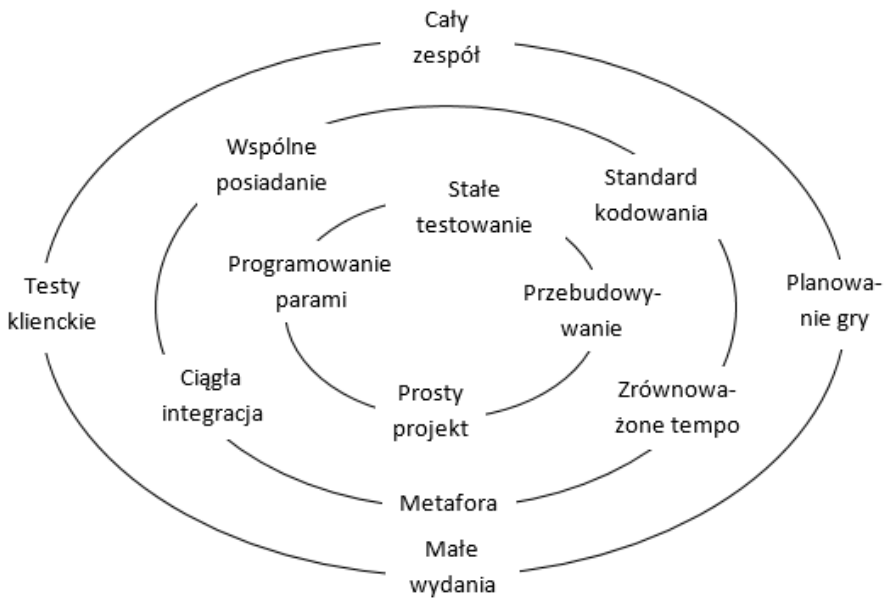
- komunikację,
- prostotę,
- informację zwrotną,

- śmiałość.

Ostatnio metodykę eXtreme Programming wzbogacono w realizowane codziennie tzw. „spotkania na stojaka". XP przewiduje tylko dwie role:

1. Klienci – definiujący listę życzeń (wymagań) i weryfikujący efekt (produkt)
2. Programiści – odpowiedzialni za odpowiednie zaprojektowanie, wykonanie i przetestowanie tworzonego rozwiązania.

Dobre praktyki XP można przedstawić za pomocą 12 reguł/technik związanych z realizacją projektu informatycznego (rys. 8.7).



Rys. 8.7. "Dwunastka" eXtreme Programming
Źródło: Opracowanie własne na podstawie:
http://xprogramming.com/images/circles.jpg

W skrócie dobre praktyki eXtreme Programming można opisać jako:

1. **Cały zespół** – każdy współpracownik w projekcie jest integralną częścią zespołu. Zespół formułuje się także z udziałem przedstawicieli biznesowych, zwanych klientami, którzy biorą czynny udział w pracach projektowych.

2. **Planowanie gry**:
    a. Przekazanie listy wymagań przez klienta – klienci mogą je przekazać np. w postaci „opowieści użytkownika" (ang. user stories), zawierających krótkie zobrazowanie wymaganych funkcjonalności.
    b. Pobieżne oszacowanie kosztów realizacji projektu, podział „projektu" na elementy i iteracje, czyli ustalenie co i w jakiej kolejności może być zrobione.
    c. Podjęcie decyzji przez klienta – uzgodnienie harmonogramu prac.
3. **Małe wydania** – programiści jak najszybciej przekazują część rozwiązania w celu weryfikacji kierunku rozwoju oprogramowania i potwierdzenia słuszności założeń. Termin przekazania pierwszej części może się wahać (zasadniczo) od tygodnia do trzech miesięcy w zależności o wielkości projektu.
4. **Metafora systemu** – ustalenie wspólnego języka komunikacji pomiędzy klientami biznesowymi a programistami.
5. **Prosty projekt** i zasada YAGNI (ang. You Are Not Gonna Need It – tłumaczenie: „nie będziesz tego potrzebował") – projekt powinien być realizowany w możliwie najprostszy sposób, bez „wybiegania w przyszłość" i tworzenia czegoś co w danym momencie nie jest wymagane.
6. **Stałe testowanie.** Testy jednostkowe – sugerowanie jest zastosowanie odpowiednich narzędzi i zastosowanie programowania kierowanego testami, które zakłada wykonanie testów przed opracowaniem funkcjonalności. Należy pamiętać też o testach akceptacyjnych będących weryfikacją wykonanego fragmentu projektu z przekazaną wcześniej listą wymagań („opowieści użytkownika").
7. **Przebudowywanie.** Kod aplikacji jest stale sprawdzany, a niepotrzebne czy powtarzające się fragmenty usuwane, co pozwala eliminować błędy integracyjne. Ten etap również jest wspomagany narzędziami automatycznych testów.
8. **Programowanie parami**. Zespół projektowy znajduje się w jednej wspólnej przestrzeni. Programiści piszą kod parami, co zwiększa poprawność i optymalność kodu. Decyzje podejmowane są wspólnie.
9. **Kolektywne posiadanie kodu**. Żadna z osób (programistów) nie posiada kodu na wyłączność. Nie ma sytuacji, że jedna osoba pracuje nad

osobnym modułem systemu. Każdy programista może pracować nad dowolnym fragmentem kodu.

10. **Stała integracja**. Wszystkie zmiany są integrowane w ramach codziennej kompilacji.
11. **Testy klienckie –** zapewniają stały kontakt z klientem. Zespół programistów ma zapewniony stały dostęp do klienta, dla którego jest tworzony system. W przypadku tworzenia rozwiązania dla wielu klientów powołuje się menedżera produktu.
12. **Standardy kodowania**. Wszyscy członkowie zespołu projektowego tworzą kod w takim samym standardzie. Programista nie powinien dać się zidentyfikować na podstawie stylu pisania kodu.

Adaptacyjny sposób realizacji projektu wg metodyki XP niejako ogranicza analizę wymagań do tzw. „user stories", będących „opowieścią" o tym jakie są życzenia użytkownika-zamawiającego w zakresie funkcjonalności zamówionego produktu. Wymagania są modyfikowane i uaktualniane w trakcie konsultacji z klientem-użytkownikiem, podczas etapu „małe wydania". Małą uwagę przywiązuje się w tej metodologii do starannego procesu tworzenia rozbudowanej dokumentacji, koncentrując się na jak najszybszym tworzeniu nowych wersji produktu.

Podsumowanie

Celem tego rozdziału było zwrócenie uwagi na różne podejście do problemu analizy wymagań w najczęściej stosowanych metodologiach zarządzania projektami informatycznymi. Przytoczone przykłady metodologii: PRINCE2 (PRojects IN a Controlled Environment 2), PMBoK (Project Management Body of Knowledge), RUP (Rational Unified Process), MSF (Microsoft Solution Framework) oraz XP (eXtreme Programming) wskazują nie tylko na różne formalne podejście do zbierania wymagań, ale także różne restrykcje w zakresie ich dokumentacji. Niemal każda z metodyk jest obecnie silnie wspierana przez zaawansowane narzędzia informatyczne pozwalające nie tylko realizować zadania krok po kroku, ale także kontrolować cały proces realizacji projektu, włącznie z tworzeniem odpowiedniej dokumentacji. Bardzo dobrym przykładem jest tu otwarty standard OpenUP i narzędzia Eclipse. Także

Microsoft wyposażył członków zespołu używających metodologii MSF w szereg narzędzi informatycznych bezpośrednio powiązanych ze środowiskiem programistycznym Visual Studio 2012 (obecna wersja). Współczesny cykl życia produktów informatycznych, duża konkurencja i presja czasu to tylko kilka z czynników, które skłaniają do stosowania lekkich, adaptacyjnych metodyk wytwarzania oprogramowania/zarządzania projektami. W tych metodykach cechami dominującymi w zakresie analizy wymagań są: minimalizacja dokumentacji analitycznej i stosowalności różnorodnych technik analitycznych z ukierunkowaniem na tzw. „opowiadania użytkowników" (ang. user stories) oraz przeniesienie części zadań związanych z analizą wymagań na fazy „udoskonalania" prototypów przedstawianych do oceny użytkownikom/klientom.

Bibliografia

1. Agile Academy. Agile Practices Help Sheet. MoSCow. http://www.agileacademy.com.au/agile/sites/default/files/aboutpdfs/AgilePracticesMoSCoWAgile%20Academy.pdf
2. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) - Fourth Edition. Project Management Institute 2008.
3. Apostoluk M., Wprowadzenie do Microsoft Solution Framework. Politechnika Wrocławska. 2006. Źródło: http://www.haksior.com/artykuly/wprowadzeniedomsf.pdf
4. Bartyzel M.: Oprogramowanie szyte na miarę. Helion, Gliwice 2012.
5. Berkun S.: Sztuka zarządzania projektami. Helion, Gliwice 2006.
6. Eeles Peter (Senior IT Architect, IBM): Capturing Architectural Requirements. http://www.ibm.com/developerworks/ rational/library/4706.html
7. Fitsilis P.: Comparing PMBOK and Agile Project Management software development processes. In: Advances in Computer and Information Sciences and Engineering, 2008, pp 378-383. http://link.springer.com/chapter/10.1007/978-1-4020-8741-7_68
8. Grady R., Practical Software Metrics for Project Management and Process Improvement. Prentice-Hall, 1992

9. Graessle P., Baumann H., Baumann P.: UML 2.0 w akcji. Przewodnik oparty na projektach. Helion, Gliwice 2006.
10. Koszlajda A.: „Zarządzanie projektami IT. Przewodnik po metodykach." HELION, Gliwice 2010
11. Microsoft Technet. Chapter 1 - Introduction to the Microsoft Solutions Framework. http://technet.microsoft.com/en-us/library/bb497060.aspx
12. Pankaj Jalote: The Timeboxing Process Model for Iterative Software Development. http://www.sciencedirect.com/science/article/pii/S0065245803620024
13. Phillips J.: Zarządzanie projektami IT. Helion, Gliwice 2005.
14. PMBOK® Guide and Standards. Źródło: http://www.pmi.org/PMBOK-Guide-and-Standards.aspx
15. PRINCE2 Process Model: http://download.ilxgroup.com/docs/ downloads/P2_ProcessMap_PL.pdf
16. Prywata M.: Zastosowanie metodyk zarzadzania projektami w tworzeniu aplikacji internetowych. http://www.web.gov.pl/g2/big/2010_05/892b33cf3170847c71f3f21a7f07d873.pdf (dostępne: 2013-06-30)
17. Schwaber K., Sutherland J.: The Scrum Guide. Wersja polska. Źródło: https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum%20Guide%20-%20PL.pdf#zoom=100
18. Szyjewski Z.: Zarządzanie projektami informatycznymi. Metodyka tworzenia systemów informatycznych. Placet, Warszawa 2001.
19. Technet: Migrating Oracle on UNIX to SQL Server on Windows. CaseStudy. http://technet.microsoft.com/en-us/library/bb656285.aspx.
20. Visual Studio 2012. Microsoft Solutions Framework (MSF), http://msdn.microsoft.com/pl-pl/library/jj161047.aspx.
21. Wrycza S., Marcinkowski B.: Język inżynierii systemów SysML. Architektura i zastosowania. Helion. Gliwice 2010.

Autorzy i afiliacje

WSTĘP
prof. dr hab. Zdzisław Szyjewski
Instytut Informatyki w Zarządzaniu, Wydział Nauk Ekonomicznych i Zarządzania, Uniwersytet Szczeciński
zszyjew@wneiz.pl
dr Karolina Muszyńska
Instytut Informatyki w Zarządzaniu, Wydział Nauk Ekonomicznych i Zarządzania, Uniwersytet Szczeciński
karolina.muszynska@wneiz.pl

ROZDZIAŁY 1 ORAZ 2
mgr inż. Tomasz Protasowicki
Instytut Systemów Informatycznych, Wydział Cybernetyki, Wojskowa Akademia Techniczna
tomasz.protasowicki@wat.edu.pl
dr inż. Jerzy Stanik
Instytut Systemów Informatycznych, Wydział Cybernetyki, Wojskowa Akademia Techniczna
jstanik@wat.edu.pl

ROZDZIAŁ 3
dr inż. Robert Waszkowski
Instytut Systemów Informatycznych, Wydział Cybernetyki, Wojskowa Akademia Techniczna
robert.waszkowski@wat.edu.pl

ROZDZIAŁ 4
dr inż. Artur Ziółkowski
Katedra Zastosowań Informatyki w Zarządzaniu, Wydział Zarządzania i Ekonomii, Politechnika Gdańska
aziolko@zie.pg.gda.pl
dr inż. Tomasz Sitek
Katedra Zastosowań Informatyki w Zarządzaniu, Wydział Zarządzania i Ekonomii, Politechnika Gdańska
tsitek@zie.pg.gda.pl

ROZDZIAŁ 5
prof. dr hab. Zdzisław Szyjewski
Instytut Informatyki w Zarządzaniu, Wydział Nauk Ekonomicznych i Zarządzania, Uniwersytet Szczeciński
zszyjew@wneiz.pl

ROZDZIAŁ 6
dr Jan Trąbka
Katedra Informatyki, Uniwersytet Ekonomiczny w Krakowie
jan.trabka@uek.krakow.pl

ROZDZIAŁ 7
mgr inż. Rafał Wojszczyk
Katedra Inżynierii Komputerowej, Wydział Elektroniki i Informatyki, Politechnika Koszalińska
rafal.wojszczyk@tu.koszalin.pl

ROZDZIAŁ 8
mgr Tomasz Komorowski
Instytut Informatyki w Zarządzaniu, Wydział Nauk Ekonomicznych i Zarządzania, Uniwersytet Szczeciński
tomasz.komorowski@wneiz.pl