

# Specyfikacja oprogramowania

## Inżynieria wymagań

Wydanie III



Karl Wieggers, Joy Beatty

Helion



Tytuł oryginału: Software Requirements 3

Tłumaczenie: Ireneusz Jakóbiak (wstęp, rozdz. 1 – 32, dod. B – C)  
Radosław Lesisz (dod. A)

ISBN: 978-83-246-9166-1

© 2014 Helion S.A.

Authorized Polish translation of the English edition of Software Requirements, 3rd Edition ISBN 9780735679665 © 2013 Karl Wieggers and Seilevel.

This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Wydawnictwo HELION  
ul. Kościuszki 1c, 44-100 GLIWICE  
tel. 32 231 22 19, 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!  
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres  
<http://helion.pl/user/opinie/speop3>  
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

# Spis treści

Wstęp .....	19
Podziękowania .....	25

## CZĘŚĆ I WYMAGANIA DOTYCZĄCE OPROGRAMOWANIA. CO, DLACZEGO I KTO?

---

<b>Rozdział 1</b>	<b>Najważniejsze wymaganie dotyczące oprogramowania .....</b>	<b>29</b>
	Definicja wymagań dotyczących oprogramowania .....	31
	Niekłóre interpretacje słowa „wymaganie” .....	32
	Poziomy i rodzaje wymagań .....	33
	Praca na trzech poziomach .....	37
	Wymagania produktu a wymagania projektu .....	40
	Opracowywanie wymagań i zarządzanie nimi .....	41
	Opracowywanie wymagań .....	41
	Zarządzanie wymaganiami .....	43
	W każdym projekcie istnieją wymagania .....	43
	Gdy złe wymagania trafiają na dobrych ludzi .....	45
	Niewystarczające zaangażowanie użytkownika .....	45
	Niedokładne planowanie .....	46
	Pełzające wymagania użytkowników .....	46
	Niejednoznaczne wymagania .....	46
	Złocenie .....	47
	Przeoczeni interesariusze .....	47
	Korzyści płynące z wysokiej jakości procesu dotyczącego wymagań .....	47
<b>Rozdział 2</b>	<b>Wymagania z punktu widzenia użytkownika .....</b>	<b>49</b>
	Luka oczekiwań .....	50
	Kim jest klient? .....	51
	Partnerstwo klient-twórca oprogramowania .....	53
	Wymaganiowa karta praw klienta oprogramowania .....	55
	Wymaganiowa karta obowiązków klienta oprogramowania .....	57

	Tworzenie kultury poszanowania wymagań .....	60
	Identyfikowanie osób decyzyjnych .....	62
	Osiąganie porozumienia co do wymagań .....	62
	Baza dla wymagań .....	63
	Co zrobić, jeśli nie osiągnięto porozumienia? .....	64
	Zgoda co do wymagań w projektach zwinnych .....	65
<b>Rozdział 3</b>	<b>Dobre praktyki w inżynierii wymagań .....</b>	<b>67</b>
	Struktura procesu opracowywania wymagań .....	70
	Dobre praktyki. Pozyskiwanie wymagań .....	72
	Dobre praktyki. Analizowanie wymagań .....	74
	Dobre praktyki. Specyfikowanie wymagań .....	76
	Dobre praktyki. Walidacja wymagań .....	77
	Dobre praktyki. Zarządzanie wymaganiami .....	77
	Dobre praktyki. Wiedza .....	79
	Dobre praktyki. Zarządzanie projektem .....	80
	Wdrażanie nowych praktyk .....	82
<b>Rozdział 4</b>	<b>Analityk biznesowy .....</b>	<b>85</b>
	Rola analityka biznesowego .....	86
	Zadania analityka biznesowego .....	87
	Najważniejsze umiejętności analityka .....	88
	Najważniejsza wiedza analityka .....	92
	Jak zostać analitykiem biznesowym? .....	92
	Były użytkownik .....	92
	Były programista albo tester .....	93
	Były (lub jednoczesny) menedżer projektu .....	94
	Specjalista w swojej dziedzinie .....	94
	Żółtodziób .....	94
	Rola analityka w projektach zwinnych .....	95
	Rozwijanie współpracy w obrębie zespołu .....	96

## CZĘŚĆ II OPRACOWYWANIE WYMAGAŃ

---

<b>Rozdział 5</b>	<b>Określanie wymagań biznesowych .....</b>	<b>101</b>
	Definiowanie wymagań biznesowych .....	102
	Identyfikowanie pożądaných korzyści biznesowych .....	102
	Wizja produktu i zakres projektu .....	102
	Sprzeczne wymagania biznesowe .....	104
	Dokument wizji i zakresu .....	105
	1. Wymagania biznesowe .....	107
	2. Zakres i ograniczenia .....	112
	3. Kontekst biznesowy .....	114

Techniki przedstawiania zakresu .....	116
Diagram kontekstowy .....	116
Mapa ekosystemu .....	117
Drzewo funkcjonalności .....	118
Lista zdarzeń .....	119
Skupienie na zakresie .....	120
Korzystanie z celów biznesowych podczas podejmowania decyzji dotyczących zakresu ...	121
Ocena wpływu zmian zakresu .....	121
Wizja i zakres w projektach zwinnych .....	122
Korzystanie z celów biznesowych, aby określić koniec projektu .....	123
<b>Rozdział 6 Słuchanie głosu użytkownika .....</b>	<b>125</b>
Klasy użytkowników .....	126
Klasyfikowanie użytkowników .....	126
Identyfikowanie klas użytkowników .....	129
Personifikacje użytkowników .....	131
Nawiązywanie kontaktu z przedstawicielami użytkowników .....	132
Mistrz produktu .....	133
Zewnętrzni mistrzowie produktu .....	134
Oczekiwania wobec mistrza produktu .....	135
Wielu mistrzów produktu .....	136
Informowanie o potrzebie zaangażowania mistrza produktu .....	137
Pułapki, na które należy uważać .....	138
Przedstawiciele użytkowników w projektach zwinnych .....	138
Godzenie sprzecznych wymagań .....	140
<b>Rozdział 7 Pozyskiwanie wymagań .....</b>	<b>143</b>
Techniki pozyskiwania wymagań .....	145
Wywiady .....	145
Warsztaty .....	146
Grupy fokusowe .....	148
Obserwacje .....	149
Kwestionariusze .....	150
Analiza interfejsów systemu .....	151
Analiza interfejsu użytkownika .....	152
Analiza dokumentów .....	152
Planowanie pozyskiwania wymagań .....	153
Przygotowania do pozyskiwania wymagań .....	154
Czynności związane z pozyskiwaniem wymagań .....	156
Czynności po zebraniu wymagań .....	158
Organizowanie i udostępnianie notatek .....	158
Dokumentowanie kwestii otwartych .....	158
Klasyfikowanie informacji uzyskanych od użytkownika .....	159
Skąd wiedzieć, że to już wszystko? .....	162

	Na co uważać podczas pozyskiwania wymagań? .....	163
	Wymagania oczywiste oraz pochodne .....	163
	Odnajdowanie pominiętych wymagań .....	165
<b>Rozdział 8</b>	<b>Zrozumieć wymagania użytkowników .....</b>	<b>167</b>
	Przypadki użycia oraz opowieści użytkowników .....	169
	Podejście bazujące na przypadkach użycia .....	172
	Przypadki użycia i scenariusze użytkownika .....	173
	Identyfikowanie przypadków użycia .....	181
	Badanie przypadków użycia .....	182
	Walidacja przypadków użycia .....	184
	Przypadki użycia i wymagania funkcjonalne .....	185
	Związane z przypadkami użycia pułapki, na które należy uważać .....	186
	Korzyści płynące z wymagań zorientowanych na użytkownika .....	187
<b>Rozdział 9</b>	<b>Gra według reguł .....</b>	<b>189</b>
	Systematyka reguł biznesowych .....	191
	Fakty .....	192
	Ograniczenia .....	192
	Wyzwalacze działań .....	194
	Wnioski .....	195
	Obliczenia .....	195
	Niepodzielne reguły biznesowe .....	196
	Dokumentowanie reguł biznesowych .....	196
	Odkrywanie reguł biznesowych .....	198
	Reguły biznesowe i wymagania .....	200
	Wiązanie wszystkiego w całość .....	201
<b>Rozdział 10</b>	<b>Dokumentowanie wymagań .....</b>	<b>203</b>
	Specyfikacja wymagań dotyczących oprogramowania .....	205
	Wymagania dotyczące etykiet .....	208
	Postępowanie z brakami .....	210
	Interfejs użytkownika i SRS .....	210
	Szablon wymagań dotyczących oprogramowania .....	212
	1. Wstęp .....	213
	2. Opis ogólny .....	214
	3. Funkcjonalności systemu .....	215
	4. Wymagania dotyczące danych .....	216
	5. Wymagania interfejsów zewnętrznych .....	217
	6. Atrybuty jakościowe .....	218
	7. Wymagania międzynarodowe i lokalizacyjne .....	219
	8. Pozostałe wymagania .....	219
	Dodatek A. Glosariusz .....	220
	Dodatek B. Modele analityczne .....	220
	Specyfikacja wymagań w projektach zwinnych .....	220

<b>Rozdział 11</b>	<b>Pisanie doskonałych wymagań .....</b>	<b>223</b>
	Cechy doskonałych wymagań .....	224
	Cechy wymagań .....	224
	Cechy zbiorów wymagań .....	226
	Wskazówki dotyczące pisania wymagań .....	227
	Perspektywa systemu czy perspektywa użytkownika .....	227
	Styl pisania wymagań .....	228
	Poziom szczegółowości .....	231
	Techniki przedstawiania wymagań .....	232
	Unikanie wieloznaczności .....	233
	Unikanie niekompletności .....	236
	Przykładowe wymagania — przed i po .....	237
<b>Rozdział 12</b>	<b>Jeden obraz wart jest 1024 słowa .....</b>	<b>241</b>
	Modelowanie wymagań .....	242
	Od głosu użytkownika do modeli analitycznych .....	243
	Wybór właściwej reprezentacji .....	244
	Diagram przepływu danych .....	246
	Diagram torowy .....	250
	Diagram przejść stanów i tabela stanów .....	251
	Mapa dialogu .....	254
	Tabele decyzyjne i drzewa decyzyjne .....	257
	Tabele zdarzenie-reakcja .....	259
	Kilka słów o diagramach UML .....	261
	Modelowanie w projektach zwinnych .....	262
	Ostatnie przypomnienie .....	262
<b>Rozdział 13</b>	<b>Specyfikowanie wymagań danych .....</b>	<b>265</b>
	Modelowanie relacji między danymi .....	265
	Słownik danych .....	268
	Analiza danych .....	271
	Specyfikowanie raportów .....	272
	Pozyskiwanie wymagań dotyczących raportów .....	272
	Co należy wziąć pod uwagę podczas specyfikowania raportów? .....	273
	Szablon specyfikacji raportu .....	274
	Kokpit zarządzania .....	277
<b>Rozdział 14</b>	<b>Wykraczanie poza funkcjonalność .....</b>	<b>279</b>
	Atrybuty jakościowe oprogramowania .....	280
	Odkrywanie atrybutów jakościowych .....	282
	Definiowanie atrybutów jakościowych .....	285
	Zewnętrzne atrybuty jakościowe .....	286
	Wewnętrzne atrybuty jakościowe .....	299
	Specyfikowanie wymagań jakościowych w języku Planguage .....	304
	Kompromisy związane z atrybutami jakościowymi .....	305

Implementowanie wymagań dotyczących atrybutów jakościowych .....	307
Ograniczenia .....	308
Atrybuty jakościowe w projektach zwinnych .....	310
<b>Rozdział 15 Ograniczanie ryzyka z wykorzystaniem prototypowania .....</b>	<b>313</b>
Prototypowanie. Co i dlaczego? .....	314
Makiety i dowody koncepcji .....	315
Prototypy ewolucyjne i do wyrzucenia .....	316
Prototypy papierowe i elektroniczne .....	319
Praca z prototypami .....	321
Ocenianie prototypów .....	323
Ryzyka prototypowania .....	325
Presja skonstruowania prototypu .....	325
Rozproszenie szczegółami .....	326
Nierealne oczekiwania co do wydajności .....	326
Nadmierne nakłady ponoszone na prototypy .....	327
Czynniki decydujące o powodzeniu prototypowania .....	327
<b>Rozdział 16 Najpierw to, co najważniejsze — określanie priorytetów wymagań .....</b>	<b>329</b>
Dlaczego wymaganiom należy nadawać priorytety? .....	330
Praktyczne podejście do nadawania priorytetów .....	331
Gierki z wymaganiami .....	332
Niektóre techniki określania priorytetów .....	333
Wchodzi czy odpada? .....	334
Porównywanie parami i szeregowanie rangowe .....	334
Skala trzypoziomowa .....	334
Metoda MoSCoW .....	336
100 złotych .....	337
Nadawanie priorytetów na podstawie wartości, kosztu i ryzyka .....	338
<b>Rozdział 17 Walidacja wymagań .....</b>	<b>343</b>
Walidacja i weryfikacja .....	345
Przeglądy wymagań .....	345
Inspekcja .....	347
Lista kontrolna defektów .....	351
Wskazówki dotyczące oceniania wymagań .....	352
Wyzwania związane z ocenianiem wymagań .....	353
Prototypowanie wymagań .....	355
Testowanie wymagań .....	355
Walidacja wymagań z wykorzystaniem kryteriów akceptacji .....	359
Kryteria akceptacji .....	359
Testy akceptacyjne .....	361



<b>Rozdział 18</b>	<b>Ponowne wykorzystanie wymagań .....</b>	<b>363</b>
	Dlaczego powtórnie korzystać z wymagań? .....	364
	Aspekty wielokrotnego korzystania z wymagań .....	364
	Skala ponownego użycia .....	365
	Zakres modyfikacji .....	366
	Mechanizm ponownego użycia .....	366
	Rodzaje informacji o wymaganiach, które można poddać powtórnemu użyciu .....	368
	Najczęściej spotykane scenariusze wielokrotnego użycia .....	369
	Linia oprogramowania .....	369
	Reengineering i zastępowanie systemów .....	369
	Inne okazje do wielokrotnego użycia .....	370
	Wzorce wymagań .....	371
	Narzędzia wspomagające wielokrotne użycie .....	372
	Przystosowanie wymagań do wielokrotnego użycia .....	372
	Przeszkody i czynniki sukcesu wielokrotnego użycia .....	374
	Przeszkody .....	374
	Czynniki sukcesu .....	376
<b>Rozdział 19</b>	<b>Więcej niż tylko opracowywanie wymagań .....</b>	<b>379</b>
	Szacowanie nakładów na wymagania .....	380
	Od wymagań do planów projektu .....	383
	Szacowanie wielkości projektu i niezbędnych nakładów na podstawie wymagań .....	383
	Wymagania a harmonogram .....	385
	Od wymagań do konstrukcji i kodu .....	386
	Architektura i alokacja .....	387
	Konstrukcja oprogramowania .....	388
	Konstrukcja interfejsu użytkownika .....	389
	Od wymagań do testów .....	391
	Od wymagań do sukcesu .....	393

### CZĘŚĆ III WYMAGANIA W RÓŻNYCH KLASACH PROJEKTÓW

---

<b>Rozdział 20</b>	<b>Projekty zwinne .....</b>	<b>397</b>
	Ograniczenia procesu kaskadowego .....	398
	Zwinne podejście do programowania .....	399
	Najważniejsze aspekty zwinnego podejścia do opracowywania wymagań .....	399
	Zaangażowanie klienta .....	399
	Szczegółowość dokumentacji .....	400
	Rejestr wymagań i priorytety .....	400
	Właściwy czas .....	401
	Epiki, opowieści użytkowników i funkcjonalności. O rany! .....	402
	Spodziewaj się zmian .....	403
	Dostosowywanie praktyk związanych z opracowywaniem wymagań do projektów zwinnych ...	403
	Przejsie na metodyki zwinne. I co teraz? .....	404

<b>Rozdział 21</b>	<b>Projekty ulepszające i zastępujące</b>	<b>407</b>
	Spodziewane trudności	408
	Techniki pracy nad wymaganiami, gdy system już istnieje	408
	Nadawanie priorytetów przy wykorzystaniu celów biznesowych	410
	Uwaga na lukę	411
	Zachowanie poziomu wydajności	411
	Kiedy stare wymagania nie istnieją	412
	Które wymagania specyfikować?	412
	Jak odkrywać wymagania w istniejących systemach?	414
	Przekonywanie do przyjęcia nowego systemu	415
	Czy możemy iterować?	416
<b>Rozdział 22</b>	<b>Projekty bazujące na gotowych rozwiązaniach</b>	<b>419</b>
	Wymagania dotyczące wyboru produktów gotowych	420
	Opracowywanie wymagań użytkowników	420
	Rozpatrywanie reguł biznesowych	421
	Identyfikowanie potrzeb związanych z danymi	421
	Definiowanie wymagań jakościowych	421
	Ocenianie rozwiązań	422
	Wymagania dotyczące implementacji gotowych produktów	424
	Wymagania dotyczące konfiguracji	425
	Wymagania dotyczące integracji	425
	Wymagania dotyczące rozszerzeń	426
	Wymagania dotyczące danych	426
	Zmiany w procesach biznesowych	426
	Najczęściej spotykane problemy mające związek z gotowymi rozwiązaniami	427
<b>Rozdział 23</b>	<b>Projekty zlecane na zewnątrz</b>	<b>429</b>
	Odpowiedni stopień szczegółowości wymagań	430
	Interakcje na linii zleceniodawca – wykonawca	431
	Zarządzanie zmianami	433
	Kryteria akceptacji	433
<b>Rozdział 24</b>	<b>Projekty automatyzacji procesów biznesowych</b>	<b>435</b>
	Modelowanie procesów biznesowych	436
	Korzystanie z bieżących procesów w celu opracowania wymagań	437
	Najpierw przyszłe procesy	438
	Modelowanie biznesowych miar wydajności	438
	Dobre praktyki w projektach automatyzacji procesów biznesowych	440
<b>Rozdział 25</b>	<b>Projekty analityki biznesowej</b>	<b>441</b>
	Przegląd projektów analityki biznesowej	441
	Opracowywanie wymagań w projektach analityki biznesowej	443
	Priorytetyzacja prac przy użyciu decyzji	444
	Definiowanie sposobów korzystania z informacji	445

Specyfikowanie potrzeb danych .....	446
Definiowanie analiz przekształcających dane .....	449
Ewolucyjny charakter analizy .....	450
<b>Rozdział 26 Projekty systemów wbudowanych oraz innych systemów czasu rzeczywistego ...</b>	<b>453</b>
Wymagania, architektura oraz alokacja systemu .....	454
Modelowanie systemów czasu rzeczywistego .....	455
Diagram kontekstowy .....	456
Diagram przejść stanów .....	456
Tabela zdarzenie-reakcja .....	457
Diagram architektury .....	459
Prototypowanie .....	460
Interfejsy .....	460
Wymagania czasowe .....	461
Atrybuty jakościowe dotyczące systemów wbudowanych .....	462
Wyzwania związane z systemami wbudowanymi .....	467

## **CZĘŚĆ IV ZARZĄDZANIE WYMAGANIAMI**

---

<b>Rozdział 27 Praktyki zarządzania wymaganiami .....</b>	<b>471</b>
Proces zarządzania wymaganiami .....	472
Baza dla wymagań .....	473
Kontrolowanie wersji wymagań .....	474
Atrybuty wymagań .....	476
Śledzenie statusów wymagań .....	477
Rozwiązywanie problemów związanych z wymaganiami .....	479
Mierzenie nakładów ponoszonych na wymagania .....	480
Zarządzania wymaganiami w projektach zwinnych .....	482
Po co zarządzać wymaganiami? .....	483
<b>Rozdział 28 Zmiany się zdarzają .....</b>	<b>485</b>
Po co zarządzać zmianami? .....	485
Kontrolowanie pełzania zakresu .....	486
Polityka kontrolowania zmian .....	487
Podstawowe pojęcia związane z procesem kontrolowania zmian .....	488
Opis procesu kontrolowania zmian .....	489
1. Cel i zakres .....	489
2. Role i odpowiedzialności .....	489
3. Stany wnioskowanych zmian .....	490
4. Kryteria początkowe .....	490
5. Zadania .....	490
6. Kryteria końcowe .....	492
7. Raportowanie statusu zmiany .....	492
Dodatek. Atrybuty zapisywane dla każdego wniosku o zmianę .....	492

Rada kontroli zmian .....	493
Skład rady .....	494
Statut rady .....	494
Renegocjowanie zobowiązań .....	495
Narzędzia do kontrolowania zmian .....	495
Pomiar aktywności dotyczącej zmian .....	496
Analiza wpływu zmiany .....	497
Procedura analizy wpływu .....	498
Szablon analizy wpływu .....	501
Zarządzanie zmianami w projektach zwinnych .....	501
<b>Rozdział 29 Ogniuwa w łańcuchu wymagań .....</b>	<b>505</b>
Śledzenie wymagań .....	505
Argumenty przemawiające za śledzeniem wymagań .....	508
Macierz śledzenia wymagań .....	509
Narzędzie służące do śledzenia wymagań .....	512
Procedura dotycząca śledzenia wymagań .....	513
Czy śledzenie wymagań jest wykonalne? Czy jest konieczne? .....	514
<b>Rozdział 30 Narzędzia inżynierii wymagań .....</b>	<b>517</b>
Narzędzia do opracowywania wymagań .....	519
Narzędzia wspomagające pozyskiwanie wymagań .....	519
Narzędzia do prototypowania .....	519
Narzędzia do modelowania .....	520
Narzędzia do zarządzania wymaganiami .....	520
Korzyści płynące ze stosowania narzędzi do zarządzania wymaganiami .....	520
Możliwości narzędzi do zarządzania wymaganiami .....	522
Wybór oraz implementacja narzędzia do pracy z wymaganiami .....	524
Wybór narzędzia .....	525
Konfiguracja narzędzia i procesów .....	525
Wspieranie adaptacji użytkowników .....	527

## CZĘŚĆ V IMPLEMENTACJA INŻYNIERII WYMAGAŃ

---

<b>Rozdział 31 Ulepszanie procesów inżynierii wymagań .....</b>	<b>531</b>
Związek wymagań z innymi procesami projektu .....	532
Wymagania i różne grupy interesariuszy .....	534
Zachęcanie do angażowania się w zmiany .....	534
Podstawy usprawniania procesu programistycznego .....	536
Analiza przyczyn źródłowych .....	538
Cykl usprawniania procesu .....	539
Ocena bieżących praktyk .....	540
Planowanie działań ulepszających .....	540

Tworzenie, pilotowanie i wdrażanie procesów .....	542
Ocenianie wyników .....	542
Elementy procesu inżynierii wymagań .....	543
Elementy procesu opracowywania wymagań .....	545
Elementy procesu zarządzania wymaganiami .....	546
Czy jesteśmy już na miejscu? .....	546
Tworzenie planu usprawnienia procesu pracy z wymaganiami .....	548
<b>Rozdział 32 Wymagania dotyczące oprogramowania a zarządzanie ryzykiem .....</b>	<b>551</b>
Podstawy zarządzania ryzykiem w oprogramowaniu .....	552
Elementy zarządzania ryzykiem .....	552
Dokumentowanie ryzyka grożącego projektowi .....	553
Planowanie zarządzania ryzykiem .....	556
Ryzyko związane z wymaganiami .....	556
Pozyskiwanie wymagań .....	557
Analiza wymagań .....	558
Specyfikacja wymagań .....	558
Walidacja wymagań .....	559
Zarządzanie wymaganiami .....	559
Zarządzanie ryzykiem to Twój przyjaciel .....	560
<b>Epilog .....</b>	<b>561</b>

## DODATKI

---

<b>Dodatek A Samoocena bieżących praktyk dotyczących wymagań .....</b>	<b>565</b>
<b>Dodatek B Poradnik usuwania problemów z wymaganiami .....</b>	<b>571</b>
<b>Dodatek C Przykładowe dokumenty wymagań .....</b>	<b>591</b>
Słowniczek .....	613
Bibliografia .....	621
Skorowidz .....	633



## ROZDZIAŁ 2

# Wymagania z punktu widzenia użytkownika

*Gerard, menedżer wyższego szczebla w Contoso Pharmaceuticals, ma umówione spotkanie z Celiną, kierowniczką Działu IT.*

— *Musimy mieć system informacyjny, który umożliwi śledzenie odczynników chemicznych — zaczął Gerard. — System powinien śledzić wszystkie pojemniki z chemikaliami, które mamy w magazynie i w laboratoriach. Chemicy, zamiast każdorazowo kupować nową skrzynkę odczynników, będą mogli pójść po nie do kogoś po drugiej stronie korytarza. Dzięki temu zaoszczędzimy mnóstwo pieniędzy. Poza tym dział BHP powinien mieć możliwość tworzenia urzędowych raportów na temat zużycia i likwidacji chemikaliów przy mniejszym nakładzie pracy, niż to jest teraz. Czy uda ci się zbudować taki system do czasu przeprowadzenia audytu zgodności za pięć miesięcy?*

— *Rozumiem, dlaczego ten projekt jest tak ważny, Gerardzie — odpowiedziała Celina. — Ale zanim zobowiązę się do jakiegoś terminu, będziemy musieli ustalić wymagania dla tego systemu.*

— *Co masz na myśli? Przecież właśnie opowiedziałem ci o moich wymaganiach — Gerard był zdezorientowany.*

— *W zasadzie to tylko podałeś mi kilka ogólnych celów biznesowych tego projektu — wyjaśniła Celina. — To za mało informacji, abym mogła wiedzieć, jaki program napisać i jak długo może to potrwać. Chciałabym, żeby jeden z naszych analityków biznesowych popracował z kilkoma przyszłymi użytkownikami. Dzięki temu lepiej zrozumiemy, jakiego systemu potrzebują.*

— *Chemicy są bardzo zajęci — zaprotestował Gerard. — Nie mają czasu na grzebanie się we wszystkich szczegółach, zanim zaczniesz pisać program. Czy twoi ludzie nie mogą domyślić się, co trzeba zrobić?*

— *Jeśli będziemy zgadywać, co użytkownicy muszą robić w systemie, nie wykonamy porządnie naszej pracy — odpowiedziała Celina. — Jesteśmy twórcami oprogramowania, a nie chemikami. Zdążyłam już się przekonać, że jeśli nie poświęcimy czasu na zrozumienie problemu, nikt nie będzie zadowolony z wyników naszej pracy.*

— *Nie mamy na to wszystko czasu — upierał się Gerard. — Dostałaś już moje wymagania. A teraz proszę, żebyś zbudowała system. Informuj mnie o postępach.*

Tego typu rozmowy są prowadzone regularnie w świecie twórców oprogramowania. Klienci domagający się nowych systemów często nie rozumieją, jak ważne są informacje pochodzące od użytkowników przyszłego systemu oraz od pozostałych interesariuszy. Marketingowcy ze wspaniałymi wizjami produktu są przekonani, że potrafią właściwie reprezentować interesy jego potencjalnych nabywców. Nic jednak nie zastąpi pozyskiwania wymagań bezpośrednio od osób, które będą korzystać z produktu. Niektóre z metod zwinnego tworzenia oprogramowania zalecają, aby przedstawiciel klienta, czasami

nazywany właścicielem produktu, blisko współpracował z zespołem programistów. Jak napisano w jednej z książek poświęconych programowaniu zwinnemu, „projekt jest **kierowany** w stronę sukcesu przez klienta i programistów, pracujących zgodnie ze sobą” (Jeffries, Anderson i Hendrickson, 2001).

Część z problemów związanych z wymaganiami wynika z nieporozumień co do różnych poziomów wymagań, które zostały opisane w rozdziale 1., „Najważniejsze wymaganie dotyczące oprogramowania” — biznesowych, użytkownika i funkcjonalnych. Gerard określił niektóre z celów biznesowych, tj. korzyści, jakie odniesie Contoso po wdrożeniu nowego systemu śledzenia odczynników. Cele biznesowe są głównym elementem wymagań biznesowych. Gerard nie potrafił jednak całościowo omówić wymagań użytkownika, ponieważ nie jest docelowym użytkownikiem systemu. Użytkownicy z kolei mogą opisać zadania, jakie powinni wykonywać w systemie, ale nie potrafią określić wszystkich wymagań funkcjonalnych, które muszą zaimplementować programiści, aby realizowanie tych zadań było możliwe. Analitycy biznesowi powinni współpracować z użytkownikami w celu głębszego zrozumienia ich potrzeb.

W rozdziale tym omówiono relację na linii klient-twórca oprogramowania, która odgrywa krytyczną rolę w powodzeniu projektu. Zaproponujemy wymaganiową kartę praw klienta oprogramowania oraz odpowiadającą jej wymaganiową kartę obowiązków klienta oprogramowania. Karty te podkreślają istotę zaangażowania klienta, a w szczególności użytkownika końcowego w opracowywanie wymagań. W niniejszym rozdziale omówiono także kluczową kwestię osiągnięcia porozumienia co do zbioru wymagań zaplanowanych do realizacji w konkretnym wydaniu produktu lub iteracji. W rozdziale 6., „Słuchanie głosu użytkownika”, opisano różne typy klientów oraz użytkowników, a także omówiono sposoby na zaangażowanie odpowiednich przedstawicieli użytkownika w proces pozyskiwania wymagań.



### Odrzucony produkt końcowy

Kiedyś, gdy odwiedzałem dział IT pewnej firmy, usłyszałem smutną historię. Programiści właśnie zbudowali nowy system informacyjny, przeznaczony do wewnętrznego użytku w firmie. Od samego początku wkład ze strony jego użytkowników był znikomy. W dniu, w którym z dumą pokazali swój nowy system, użytkownicy odrzucili go jako zupełnie nieprzydatny. Taka reakcja poruszyła programistów, ponieważ ciężko pracowali, aby sprostać temu, co w ich mniemaniu było wymaganiami użytkowników. Co wobec tego zrobili w tej sytuacji? Naprawili system. Firmy zawsze naprawiają systemy, gdy programiści błędnie zrozumieją wymagania, chociaż koszty takiego działania zawsze są wyższe, niż byłyby w sytuacji, gdyby przedstawiciele użytkowników od samego początku zaangażowali się w projekt.

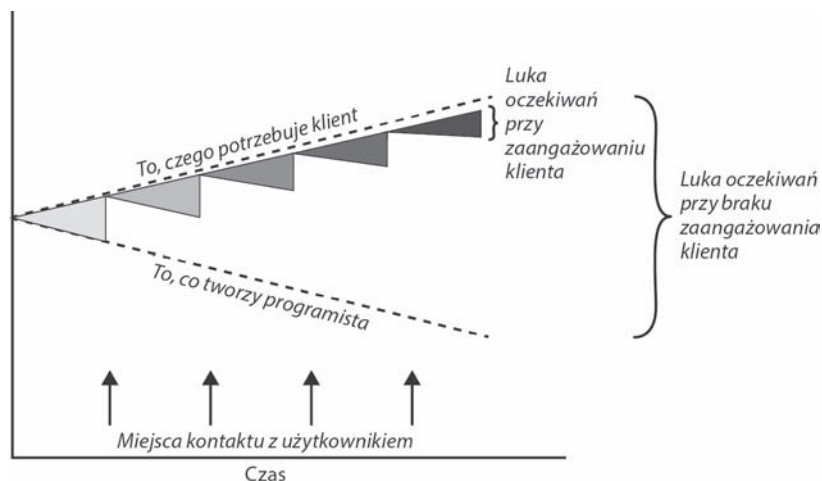
Programiści, rzecz jasna, nie planowali, że będą musieli poświęcić czas na wprowadzanie poprawek w wadliwym systemie, w związku z czym ich następny projekt musiał zaczekać w kolejce. W takiej sytuacji przegrana stała się udziałem wszystkich. Programiści byli przygnębieni, użytkownicy zmartwieni, ponieważ ich nowy system nie był gotowy na czas, a zarząd niezadowolony z powodu wydania sporych pieniędzy oraz kosztów utraconych korzyści spowodowanych opóźnieniem w pracach nad innymi produktami. Głębokie i trwające od samego początku zaangażowanie klienta w projekt mogłoby nie doprowadzić do tego niefortunnego, chociaż wcale nie tak rzadko spotykanego końca.

## Luka oczekiwań

Bez wystarczającego zaangażowania klienta nie ma możliwości uniknięcia sytuacji, w której pod koniec projektu powstanie luka oczekiwań — różnica między tym, czego w rzeczywistości potrzebują klienci, a tym, co utworzyli programiści, bazując na informacjach, jakie uzyskali na początku prac nad projektem



(Wiegiers, 1996), co pokazano na rysunku 2.1 w postaci przerywanych linii. Tak samo jak we wcześniejszej opowieści, luka oczekiwań pojawia się niczym nieprzyjemna niespodzianka, która czekała wszystkich interesariuszy. Z naszych doświadczeń wynika, że niespodzianki w oprogramowaniu nigdy nie niosą ze sobą dobrych wieści. Ze względu na zmiany zachodzące w świecie biznesu wymagania stają się nieaktualne, w związku z czym konieczna jest ciągła współpraca z klientami.



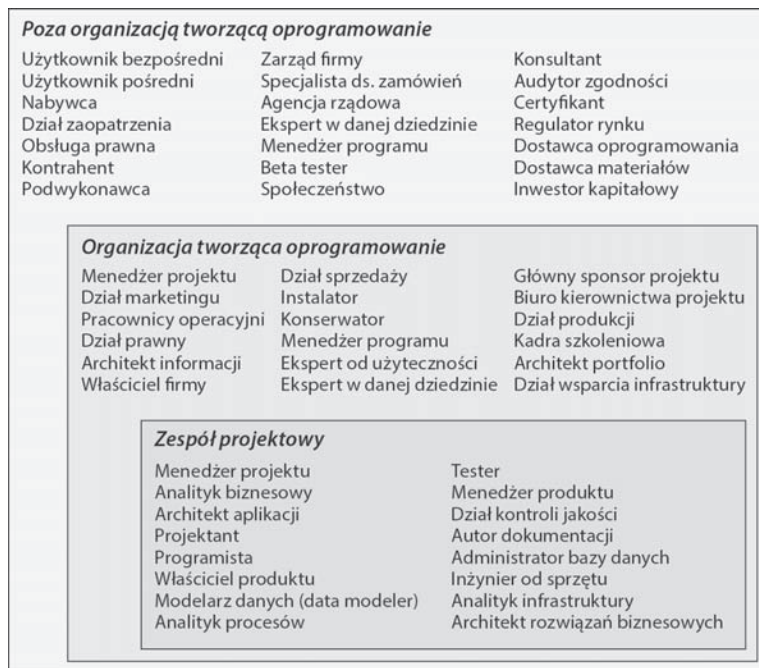
RYSUNEK 2.1. Często angażowanie się klienta zmniejsza lukę oczekiwań

Najlepszym sposobem na ograniczenie luki oczekiwań jest aranżowanie częstych kontaktów z odpowiednimi przedstawicielami użytkowników. Kontakty te mogą przyjmować formę wywiadów, rozmów, przeglądania wymagań, testowania interfejsu użytkownika, oceniania prototypów oraz — w przypadku programowania zwinnego — informacji zwrotnych otrzymywanych od użytkownika na temat kolejnych przyrostów działającego oprogramowania. Każde spotkanie daje możliwość likwidowania luki oczekiwań, ponieważ to, co utworzy programista, będzie w większym stopniu zbieżne z potrzebami klienta.

Oczywiście zaraz po każdym kontakcie z klientem luka oczekiwań zacznie rosnąć, gdy tylko ponownie ruszy proces programowania. Im częstsze jednak będą kontakty, tym łatwiej będzie pozostać na właściwej ścieżce rozwoju produktu. Jak pokazują coraz bardziej kurczące się szare trójkąty na rysunku 2.1, cykl spotkań doprowadzi pod koniec projektu do znacznie mniejszej luki oczekiwań oraz uzyskania rozwiązania, które będzie o wiele bliższe rzeczywistym potrzebom klienta. To właśnie dlatego jedną z głównych zasad programowania zwinnego jest prowadzenie częstych rozmów między programistami a klientami. Reguła ta znakomicie sprawdza się w dowolnym projekcie.

## Kim jest klient?

Zanim porozmawiamy o klientach, powinniśmy zająć się interesariuszami. **Interesariusz** to osoba, grupa albo organizacja, która aktywnie uczestniczy w projekcie, na które mają wpływ procesy odbywające się w projekcie lub wynik realizacji tego projektu albo które mogą mieć wpływ na proces lub jego wynik. Interesariusze mogą pozostawać wewnątrz lub na zewnątrz zespołu projektowego oraz organizacji tworzącej produkt. Na rysunku 2.2 zidentyfikowano wielu potencjalnych interesariuszy należących do tych kategorii. Rzecz jasna, nie wszyscy z nich będą występować we wszystkich projektach i sytuacjach.



**RYСУNEK 2.2.** Potencjalni interesariusze w zespole projektowym, w organizacji tworzącej oprogramowania oraz poza nią

Analiza interesariuszy jest ważną częścią opracowywania wymagań (Smith, 2000; Wiegiers, 2007; IIBA, 2009). Podczas identyfikowania potencjalnych interesariuszy w danym projekcie rzucić szeroką sieć, abyś mógł uniknąć przeoczenia jakiegś ważnej ich grupy. W następnej kolejności możesz zredukować listę kandydatów na interesariuszy do osób, których opinia istotnie będzie potrzebna, dzięki czemu zrozumiesz wszystkie wymagania oraz ograniczenia projektu, a Twój zespół dostarczy właściwe rozwiązanie.

Klienci stanowią podzbiór interesariuszy. **Klient** to osoba lub organizacja, która osiąga bezpośrednie lub pośrednie korzyści z użytkowania produktu. Klienci oprogramowania mogą domagać się określonych danych wyjściowych generowanych przez produkt, płacić za nie, wybierać je lub określać, korzystać z nich albo je uzyskiwać. Do klientów pokazanych na rysunku 2.2 należą użytkownicy bezpośredni i pośredni, główny sponsor projektu, nabywca i dział zaopatrzenia. Niektórzy interesariusze nie są klientami, jak na przykład dział prawny, audytorzy zgodności, dostawcy, kontrahenci oraz inwestorzy kapitałowi. Gerard, menedżer, którego już poznaliśmy, jest reprezentantem głównego sponsora projektu, który płaci za produkt. Klienci tacy jak on przedstawiają wymagania biznesowe, które tworzą bazę projektu oraz stanowią biznesowe uzasadnienie jego rozpoczęcia. Jak to zostanie omówione w rozdziale 5., „Określanie wymagań biznesowych”, wymagania biznesowe dokumentują cele biznesowe, jakie chcieliby zrealizować klienci, firma oraz inni interesariusze. Wszystkie pozostałe wymagania dotyczące produktu muszą być zgodne z tymi celami.

Wymagania użytkowników powinny pochodzić od osób, które rzeczywiście będą używać — bezpośrednio bądź pośrednio — danego produktu. Użytkownicy ci (często nazywani **użytkownikami końcowymi**) stanowią podzbiór grupy użytkowników. Bezpośredni użytkownicy będą osobiście korzystać z produktu. Z kolei użytkownicy pośredni mogą korzystać z systemu bez styczności z nim, jak na przykład kierownik magazynu, który za pośrednictwem poczty elektronicznej otrzymuje automatyczne raporty na temat dziennych ruchów magazynowych. Użytkownicy mogą opisywać zadania, które muszą wykonywać za pomocą produktu, określać potrzebne im dane wyjściowe oraz charakterystyki jakościowe, jakich oczekują od produktu.



## Przypadek brakującego interesariusza

Znam projekt, w którym już prawie skończono pozyskiwać wymagania, gdy podczas przeglądania przepływu procesów analityk biznesowy zapytał interesariusza:

- Czy jesteś pewien, że etap obliczania podatku w tym przepływie jest poprawny?
- Nie wiem — odpowiedział interesariusz. — Naliczanie podatków nie należy do mnie.

Od tego jest dział podatków.

Podczas trwających kilka miesięcy prac nad projektem nikt z zespołu programistycznego nigdy nie rozmawiał z żadną osobą z działu podatków. Nikt nawet nie wiedział, że taki dział w ogóle **istnieje**. Jak tylko analitycy biznesowi spotkali się z przedstawicielem działu podatków, okazało się, że można sporządzić długą listę brakujących wymagań związanych z prawnymi następstwami przyjętego sposobu implementacji funkcji odpowiedzialnych za naliczanie podatków. W rezultacie projekt opóźnił się o kilka miesięcy. Aby uniknąć tego rodzaju nieprzyjemnych sytuacji, można skorzystać ze schematu organizacyjnego przedsiębiorstwa w celu odnalezienia wszystkich interesariuszy, na których będzie mieć wpływ nowy system.

Klienci przekazujący wymagania biznesowe czasami twierdzą, że wypowiadają się w imieniu rzeczywistych użytkowników, chociaż są zbyt oddaleni od ich pracy, aby mogli opisać ich faktyczne wymagania. W przypadku firmowych systemów informatycznych, zakontraktowanych zleceń oraz rozwijania aplikacji na zamówienie wymagania biznesowe powinny pochodzić od osoby, która ponosi ostateczną odpowiedzialność za wartość biznesową, jaka jest oczekiwana od produktu. Wymagania użytkowników powinny pochodzić od osób, które będą naciskać klawisze, dotykać ekranu albo korzystać z danych wyjściowych. Jeśli między nabywcą, który płaci za produkt, a końcowymi użytkownikami istnieje przepaść, można mieć pewność, że pojawią się kłopoty.

Inna sytuacja ma miejsce podczas prac nad oprogramowaniem komercyjnym, kiedy klient oraz użytkownik często są tą samą osobą. Osoby wcielające się w klientów, takie jak personel działu marketingu albo menedżer produktu, zwykle starają się przewidzieć, co będzie potrzebne klientowi. Jednak nawet w przypadku oprogramowania komercyjnego powinieneś dążyć do zaangażowania użytkowników w proces zbierania wymagań, co opisano w rozdziale 7., „Pozyskiwanie wymagań”. Jeśli tego nie zrobisz, bądź gotowy na czytanie recenzji wytykających niedociągnięcia w produkcie, których można byłoby uniknąć, gdyby użytkownicy wnieśli swój wkład w jego kształtowanie.

Wśród interesariuszy projektu mogą pojawiać się konflikty. Wymagania biznesowe czasami odzwierciedlają strategię organizacyjną albo ograniczenia budżetowe, które nie są oczywiste dla użytkowników. Użytkownicy niepokodzeni z koniecznością korzystania z nowego systemu informatycznego — który został im narzucony siłą przez zarząd — mogą odmawiać współpracy z twórcami oprogramowania, których postrzegają jako zwiastun niechcianej przyszłości. Takie osoby często nazywane są „grupami przegranych” (Gause i Weinberg, 1989). Aby zapobiegać tego typu potencjalnym konfliktom, wypróbuj strategii komunikacyjnych dotyczących celów i ograniczeń projektu, które pomogą budować wzajemną zgodę oraz powstrzymać spory i urazy.

## Partnerstwo klient-twórca oprogramowania

Doskonałe oprogramowanie jest wynikiem dobrze opracowanego projektu bazującego na doskonałych wymaganiach. Doskonałe wymagania są wynikiem efektywnej współpracy między twórcami oprogramowania a klientami (w szczególności rzeczywistymi użytkownikami), innymi słowy — partnerstwa. Wspólny wysiłek może być skuteczny tylko wtedy, gdy wszystkie zaangażowane strony wiedzą, czego potrzebują, aby odnieść sukces, oraz rozumieją i szanują to, co jest potrzebne do odniesienia sukcesu przez ich partnerów. Gdy w trakcie realizacji projektu narasta presja, łatwo jest

zapomnieć, że wszyscy interesariusze mają wspólny cel. Jest nim wytworzenie produktu, który dla każdego z interesariuszy przedstawia odpowiednią wartość biznesową oraz go wynagradza. Osobą, która powinna podtrzymywać taką zbiorową współpracę, jest zazwyczaj analityk biznesowy.

Wymaganiowa karta praw klienta oprogramowania, zawarta w tabeli 2.1, wymienia 10 praw, z których mogą zasadnie korzystać klienci, dotyczących współpracy z analitykami biznesowymi oraz programistami podczas wykonywania czynności mających związek z opracowywaniem wymagań. Każde z tych praw implikuje odpowiadający mu obowiązek po stronie analityków biznesowych lub programistów. „Twoje” prawa i odpowiedzialności, wymienione na obu listach, dotyczą klienta oprogramowania, które jest tworzone w ramach projektu.

**TABELA 2.1.** *Wymaganiowa karta praw klienta oprogramowania*

<b>Masz prawo</b>
Oczekiwać, że analityk biznesowy będzie mówił Twoim językiem.
Oczekiwać, że analityk biznesowy zapozna się z Twoimi zadaniami oraz celami.
Oczekiwać, że analityk biznesowy w odpowiedniej formie zarejestruje Twoje wymagania.
Otrzymywać wyjaśnienia dotyczące praktyk związanych z pozyskiwaniem wymagań oraz na temat wymagań docelowych.
Zmieniać swoje wymagania.
Oczekiwać atmosfery wzajemnego poszanowania.
Poznawać nowe pomysły, a także alternatywne wymagania oraz ich rozwiązania.
Opisywać cechy, które sprawią, że produkt będzie prosty w użyciu.
Poznawać sposoby na takie dostosowanie wymagań, aby poprzez ich wielokrotne użycie przyspieszyć proces rozwoju oprogramowania.
Otrzymać system, który spełnia Twoje potrzeby funkcjonalne i oczekiwania jakościowe.

Ponieważ przeciwieństwem prawa jest obowiązek, w tabeli 2.2 wymieniono 10 obowiązków, które w ramach procesu opracowywania wymagań ma klient wobec analityków biznesowych i programistów. Jeśli wolisz, możesz uważać je za kartę praw twórców oprogramowania. Jeśli obu tych list nie da się w pełni zastosować w Twojej organizacji, możesz je zmodyfikować, aby lepiej pasowały do Twoich lokalnych realiów.

**TABELA 2.2.** *Wymaganiowa karta obowiązków klienta oprogramowania*

<b>Masz obowiązek</b>
Informować analityka biznesowego oraz programistów o swoich zadaniach.
Poświęcać czas na przekazywanie oraz wyjaśnianie swoich wymagań.
Konkretnie i szczegółowo przedstawiać informacje na temat wymagań.
Na prośbę analityka podejmować w porę decyzje.
Respektować oszacowania programistów dotyczące kosztu oraz możliwości realizacji wymagań.
We współpracy z programistami określać realne priorytety wymagań.
Przeglądać wymagania oraz oceniać prototypy.
Ustanawiać kryteria akceptacji.
Bezwzględnie zgłaszać zmiany wymagań.
Uszanować proces opracowywania wymagań.

Powyższe prawa oraz obowiązki mają zastosowanie w stosunku do rzeczywistych klientów, gdy oprogramowanie jest tworzone na wewnętrzne potrzeby firmy, w ramach kontraktu lub dla określonej grupy głównych odbiorców. W przypadku produkcji oprogramowania na szeroki rynek powyższe prawa i obowiązki odnoszą się w większym stopniu do osób wcielających się w klientów, takich jak menedżer produktu.

W ramach planowania projektu główny klient oraz interesariusze powinni zapoznać się z tymi listami i wynegocjować poszczególne punkty w celu osiągnięcia obopólnej zgody. Należy upewnić się, że osoby uczestniczące w opracowywaniu wymagań rozumieją i akceptują swoje odpowiedzialności. Takie zrozumienie może ograniczyć późniejsze tarcia, gdy jedna ze stron będzie oczekiwać czegoś, czego nie chce lub nie może spełnić druga strona.

**Pułapka.** Nie zakładaj, że osoby biorące udział w projekcie będą instynktownie wiedzieć, jak współpracować ze sobą podczas opracowywania wymagań. Poświęć czas na wyjaśnienie, jak mogą one efektywnie pracować razem. Dobrym pomysłem będzie zapisanie decyzji dotyczących podejścia do rozwiązywania problemów związanych z wymaganiami, jakie mogą pojawić się podczas prac nad projektem. Taka informacja będzie cennym narzędziem ułatwiającym komunikację podczas wszystkich prac nad projektem.

## Wymaganiowa karta praw klienta oprogramowania

Poniżej omówiono 10 praw, z których mogą korzystać klienci podczas prac nad wymaganiami.

### Prawo nr 1. Oczekiwać, że analityk biznesowy będzie mówił Twoim językiem

Omawianie wymagań powinno być skoncentrowane na Twoich potrzebach biznesowych oraz zadaniach do wykonania, przy czym powinieneś posługiwać się słownictwem biznesowym. Zastanów się nad możliwością przekazania analitykowi biznesowemu swojej branżowej nomenklatury razem ze słownikiem terminów. Podczas rozmowy z analitykiem nie powinieneś stosować w nadmiarze żargonu technicznego.

### Prawo nr 2. Oczekiwać, że analityk biznesowy zapozna się z Twoimi zadaniami oraz celami

Współpracując z Tobą podczas pozyskiwania wymagań, analityk biznesowy może lepiej zrozumieć Twoje zadania biznesowe oraz dowiedzieć się, jak wpasować nowy system w Twój świat. Dzięki temu programiści utworzą rozwiązanie, które spełni Twoje oczekiwania. Poproś analityków biznesowych oraz programistów, aby zobaczyli, czym zajmujecie się Ty i Twoi koledzy w pracy. Jeśli nowy system ma zastąpić stary, analityk powinien popracować na starym systemie, tak jak Ty to robisz. W ten sposób zobaczy on miejsce obecnego systemu w przepływie pracy oraz pozna obszary, w których można go ulepszyć. Nie oczekuj, że analityk biznesowy będzie wiedzieć wszystko o Twoich czynnościach biznesowych oraz znać całą potrzebną terminologię (patrz obowiązek nr 1).

### Prawo nr 3. Oczekiwać, że analityk biznesowy w odpowiedniej formie zarejestruje Twoje wymagania

Analityk biznesowy zapozna się ze wszystkimi informacjami, jakich udzielił mu interesariusze, i zada dodatkowe pytania, które umożliwią mu oddzielenie wymagań użytkowników od ról biznesowych, wymagań funkcjonalnych, celów jakościowych i pozostałych elementów. Ostatecznym celem tej analizy jest uzyskanie dopracowanego zestawu wymagań zapisanych w odpowiedniej postaci, jaką

jest na przykład dokument specyfikacji wymagań oprogramowania albo narzędzie do zarządzania wymaganiami. Taki zbiór wymagań dokumentuje osiągnięcie porozumienia między interesariuszami, które dotyczy funkcji, jakości oraz ograniczeń przyszłego produktu. Wymagania powinny być zapisane oraz zorganizowane w formie, która ułatwi ich zrozumienie. Zweryfikowanie tej specyfikacji oraz pozostałych wymagań, a także przedstawienie ich w innej postaci, takiej jak graficzne modele analityczne, pomogą zagwarantować, że dokładnie odzwierciedlają one Twoje potrzeby.

#### **Prawo nr 4. Otrzymywać wyjaśnienia dotyczące praktyk związanych z pozyskiwaniem wymagań oraz na temat wymagań docelowych**

Różne praktyki mogą przyczynić się do tego, że proces pozyskiwania wymagań oraz zarządzania nimi będzie zarówno skuteczny, jak i wydajny, a wiedzę na temat wymagań będzie można przedstawić w różnych postaciach. Analityk biznesowy powinien wytłumaczyć praktyki, które zaleca do stosowania, oraz wyjaśnić, jakie informacje znajdują się w końcowych wersjach poszczególnych dokumentów. Może on na przykład utworzyć graficzne schematy uzupełniające wymagania sformułowane pisemnie. Diagramy te mogą być dla Ciebie obce i dość skomplikowane, ale użyta w nich notacja nie powinna być trudna do zrozumienia. Analityk biznesowy powinien wyjaśnić cel każdego diagramu, znaczenie użytych w nim symboli oraz zweryfikować go pod względem poprawności. Jeżeli nie przedstawi on takich wyjaśnień, nie zawahaj się go o to poprosić.

#### **Prawo nr 5. Zmieniać swoje wymagania**

Analitycy biznesowi oraz programiści nie powinni oczekiwać, że z miejsca podasz wszystkie swoje wymagania, ani że wymagania te pozostaną niezmiennie przez cały okres prac nad produktem. Masz prawo dokonywania zmian w wymaganiach w miarę rozwijania się działalności firmy, otrzymywania nowych informacji od interesariuszy albo gdy głębiej zastanowisz się nad swoimi potrzebami. Zmiana jednak zawsze ma swoją cenę. Czasami dodanie nowej funkcjonalności będzie wymagać zrezygnowania z innej funkcji albo zrewidowania budżetu lub harmonogramu prac. Do ważnych zadań analityka biznesowego należy ocena konsekwencji, jakie niosą ze sobą zmiany, zarządzanie nimi oraz informowanie o nich. Podejmij z nim współpracę w celu osiągnięcia porozumienia dotyczącego wypracowania prostego, ale jednocześnie skutecznego procesu obchodzenia się ze zmianami.

#### **Prawo nr 6. Oczekiwać atmosfery wzajemnego poszanowania**

Relacje między klientami a programistami czasami stają się napięte. Dyskusje na temat wymagań mogą prowadzić do frustracji, jeśli ich uczestnicy nie rozumieją się nawzajem. Wspólna praca może otworzyć oczy członków zespołu na problemy, z którymi boryka się każda z grup. Klienci biorący udział w opracowywaniu wymagań mają prawo oczekiwać, że analitycy biznesowi i programiści będą traktować ich z szacunkiem oraz docenią czas, jaki inwestują w sukces projektu. Podobnie klienci powinni okazywać szacunek członkom zespołu programistycznego, gdyż wszyscy razem współpracują po to, aby osiągnąć wspólny cel, jakim jest zakończony sukcesem projekt. Tutaj wszyscy znajdują się po tej samej stronie.

#### **Prawo nr 7. Poznawać nowe pomysły, a także alternatywne wymagania oraz ich rozwiązania**

Poinformuj analityka biznesowego o sytuacjach, w jakich Twój obecny system nie sprawdza się w procesie biznesowym, aby zyskać pewność, że nowy system nie zautomatyzuje mało efektywnych lub przestarzałych procesów. Nie chciałbyś także mieć do czynienia z doraźnymi rozwiązaniami. Analityk biznesowy może zasugerować wprowadzenie poprawek w Twoich procesach biznesowych. Kreatywny analityk dodaje także nowe wartości, proponując nowe funkcje, których użytkownicy nawet sobie nie wyobrażali.

## Prawo nr 8. Opisywać cechy, które sprawią, że produkt będzie prosty w użyciu

Możesz spodziewać się, że analityk biznesowy zada Ci pytania o cechy oprogramowania, które wykraczają poza Twoje potrzeby funkcjonalne. Cechy te, innymi słowy, atrybuty jakościowe, sprawiają, że oprogramowanie jest łatwiejsze lub przyjemniejsze w użyciu i umożliwia użytkownikom efektywniejsze realizowanie ich celów. Użytkownicy czasami domagają się, żeby produkt był **przyjazny** albo **odporny**, chociaż tego typu określenia są zbyt subiektywne, aby mogli się nimi kierować programiści. Analityk biznesowy powinien zapytać Cię o to, co rozumiesz przez pojęcia „przyjazny” albo „odporny”. Opowiedz mu, które aspekty Twojego obecnego systemu wydają Ci się „przyjazne”, a które nie. Jeżeli nie omówisz tego aspektu z analitykiem, będziesz mieć duże szczęście, jeśli gotowy produkt spełni Twoje oczekiwania.

## Prawo nr 9. Poznać sposoby na takie dostosowanie wymagań, aby poprzez ich wielokrotne użycie przyspieszyć proces rozwoju oprogramowania

Wymagania czasami mogą być w pewnym stopniu elastyczne. Analityk biznesowy może znać istniejące składniki oprogramowania albo inne wymagania, które w znacznej mierze pokrywają się z opisanymi przez Ciebie potrzebami. W takich przypadkach powinien on zasugerować Ci wprowadzenie zmian w wymaganiach lub zrezygnowanie z nadmiernego dostosowywania systemu do Twoich potrzeb, dzięki czemu programiści będą mogli skorzystać z tych składników. Korekta wymagań, gdy nadarza się praktyczna okazja do skorzystania z gotowych rozwiązań, pozwala zaoszczędzić czas i pieniądze. Czasami elastyczne podejście do wymagań jest niezbędne, jeśli chcesz zintegrować ze swoim systemem komercyjne produkty z półki, gdyż rzadko zdarza się, aby miały one dokładnie takie charakterystyki, jakie są Ci potrzebne.

## Prawo nr 10. Otrzymać system, który spełnia Twoje potrzeby funkcjonalne i oczekiwania jakościowe

Jest to najważniejsze prawo użytkownika, **ale** może ono zostać zrealizowane tylko wtedy, gdy wyraźnie przekazesz wszystkie informacje umożliwiające programistom zbudowanie właściwego produktu, gdy programiści poinformują Cię o dostępnych opcjach i ograniczeniach oraz gdy wszystkie strony osiągną porozumienie. Nie zapomnij o wyrażeniu wszystkich swoich oczekiwań; w przeciwnym razie programiści nie będą się mogli do nich prawidłowo odnieść. Klienci czasami nie zgłaszają potrzeb, co do których są przekonani, że są oczywiste. Potwierdzenie wspólnego punktu widzenia w zespole pracującym nad projektem jest jednak równie ważne, jak zaproponowanie czegoś nowego.

## Wymaganiowa karta obowiązków klienta oprogramowania

Ponieważ przeciwieństwem prawa jest obowiązek, poniżej wymieniono 10 obowiązków, które mają przedstawiciele klienta podczas definiowania wymagań i zarządzania nimi w ramach projektu.

### Obowiązek nr 1. Informować analityka biznesowego oraz programistów o swoich zadaniach

W kwestii zdobycia informacji na temat pojęć związanych z Twoimi zadaniami oraz poznania branżowego słownictwa zespół programistyczny jest zdany na Ciebie. Twoim celem nie będzie przekształcenie analityków biznesowych w ekspertów w Twojej dziedzinie, ale udzielenie im pomocy w zrozumieniu Twoich problemów i celów. Z dużym prawdopodobieństwem analitycy nie dysponują wiedzą, którą Ty i Twoi koledzy uważacie za oczywistą.

## Obowiązek nr 2. Poświęcać czas na przekazywanie oraz wyjaśnianie swoich wymagań

Klienci nie mają zbyt dużo czasu do dyspozycji. Ci z nich, którzy są zaangażowani w pracę nad wymaganiami, często należą do najbardziej zajętych osób w firmie. Niemniej Twoim obowiązkiem jest poświęcenie czasu na warsztaty, rozmowy i inne aktywności związane z pozyskiwaniem i walidacją wymagań. Czasami analityk biznesowy może sądzić, że zrozumiał Twoje wyjaśnienia, ale później dojdzie do wniosku, że potrzebuje dalszych informacji. Należy okazać mu cierpliwość, gdyż w taki właśnie iteracyjny sposób zbiera on i precyzuje wymagania — na tym polega złożona natura komunikacji międzyludzkiej oraz taki jest klucz do osiągnięcia sukcesu w tworzeniu oprogramowania. Jeśli wysiłek związany z wyjaśnianiem wymagań zostanie skupiony w kilku nieprzerwanych godzinach, praca nad wymaganiami zajmie mniej czasu, niż gdyby ten sam wysiłek był rozproszony na przestrzeni paru tygodni.

## Obowiązek nr 3. Konkretnie i szczegółowo przedstawiać informacje na temat wymagań

Kuszące jest pozostawienie wymagań w postaci mało szczegółowej i niejasnej, gdyż precyzowanie wymagań jest żmudne i wymaga czasu (lub gdy ktoś chce uniknąć odpowiedzialności związanej z podejmowaniem decyzji). W końcu jednak ktoś będzie musiał rozwiązać wątpliwości i wyjaśnić niejednoznaczności. Najbardziej kompetentną osobą do podejmowania takich decyzji jesteś Ty. W przeciwnym wypadku będziesz musiał zaufać, że analityk biznesowy oraz programiści trafnie odgadną Twoje wymagania. Nie ma przeszkód, aby tymczasowo oznaczać wymagania jako **do wyjaśnienia (TBD)** w celu wskazania, że potrzebne są dodatkowe ustalenia albo informacje. Czasami adnotacja TBD jest używana, ponieważ określone wymaganie jest trudne do określenia i nikt nie chce się nim zająć. Postaraj się wyjaśnić cel istnienia każdego z wymagań, dzięki czemu analityk biznesowy będzie mógł je poprawnie sformułować. Jest to najlepszy sposób na zagwarantowanie, że produkt spełni Twoje oczekiwania.

## Obowiązek nr 4. Na prośbę analityka podejmować w porę decyzje

Podobnie jak wykonawca budujący Twój wymarzony dom, analityk biznesowy będzie Cię prosić o podjęcie wielu decyzji, wśród których można wymienić rozwiewanie wątpliwości wynikających ze sprzecznych prośb otrzymanych od różnych klientów, dokonywanie wyborów spośród niezgodnych ze sobą atrybutów jakościowych oraz ocenianie dokładności informacji. Klienci upoważnieni do podejmowania takich decyzji powinni bezzwłocznie rozstrzygać wątpliwości analityka, gdy zostaną o to poproszeni. Programiści nie będą mogli sprawnie kontynuować pracy, jeśli nie podejmiesz stosownych decyzji, a czas spędzony przez nich na czekaniu na Twoją odpowiedź może opóźnić rozwój produktu. Jeśli prośby o Twój czas staną się uciążliwe, pamiętaj, że system jest tworzony dla Ciebie. Analitycy biznesowi często potrafią pomagać innym osobom w podejmowaniu decyzji, jeśli więc utkniesz w miejscu, będziesz mógł poprosić ich o pomoc.

## Obowiązek nr 5. Respektować oszacowania programistów dotyczące kosztu oraz możliwości realizacji wymagań

Wszystkie funkcje oprogramowania mają swój koszt. Najbardziej kompetentnymi osobami do szacowania tego kosztu są programiści. Implementacja niektórych funkcji może być niewykonalna technicznie lub zaskakująco kosztowna. Niektóre z wymagań mogą wymagać nieosiągalnej w danym środowisku wydajności albo korzystania z danych, które nie są dostępne w systemie. Programista może odgrywać rolę posłańca złych wieści na temat wykonalności albo kosztów. Powinieneś szanować jego ocenę, nawet jeśli oznacza ona, że nie otrzymasz tego, o co prosiłeś w dokładnie takiej postaci, jaką sobie wyobrażałeś. Czasami będziesz mógł zmienić swoje wymagania w taki sposób, aby ich realizacja stała się możliwa albo tańsza. Na przykład spełnienie prośby, aby jakaś reakcja następowała „natychmiastowo”, może być niewykonalne, ale precyzyjniejsze określenie ram czasowych („do 50 milisekund”) może sprawić, że osiągnięcie założonego celu będzie możliwe.



## Obowiązek nr 6. We współpracy z programistami określać realne priorytety wymagań

Tylko w przypadku nielicznych projektów można wygospodarować wystarczające zasoby oraz czas, aby zaimplementować wszystkie funkcjonalności, które chcą mieć do dyspozycji klienci. Określenie, które funkcje są niezbędne, które przydatne, a bez których klienci mogą się obyć, stanowi ważną część analizy wymagań. W określaniu priorytetów wymagań to Ty odgrywasz wiodącą rolę. Programiści mogą udostępniać informacje o kosztach i ryzyku związanym z każdym wymaganiem lub opowieści użytkowników pomagające ustalić ostateczne priorytety. Jeśli priorytety, jakie ustanowisz, będą realne, pomożesz programistom wytworzyć maksymalną wartość przy najniższym koszcie i we właściwym czasie. Wspólne ustalanie priorytetów jest kluczem do sukcesu w projektach zwinnych, dzięki czemu programiści mogą zacząć oddawać użyteczne oprogramowanie tak szybko, jak to jest tylko możliwe.

Powinieneś uszanować oszacowanie zespołu programistycznego określające, jaka część żądanej funkcjonalności może zostać oddana w zadanym czasie i przy istniejących ograniczeniach zasobów. Jeśli wszystko, co jest Ci potrzebne, nie mieści się w zakresie projektu, osoby podejmujące decyzje będą musiały w oparciu o priorytety ograniczyć zakres projektu, wydłużyć harmonogram albo wygospodarować dodatkowe fundusze lub ludzi. Nadawanie każdemu wymaganiu wysokiego priorytetu nie jest rozsądne ani nie świadczy o dobrej współpracy.

## Obowiązek nr 7. Przeglądać wymagania oraz oceniać prototypy

Jak przekonasz się w rozdziale 17., „Walidacja wymagań”, przeglądy koleżeńskie należą do najwydajniejszych aktywności związanych z produkcją oprogramowania, jakie są dostępne. Dopuszczenie klientów do uczestnictwa w przeglądach stanowi klucz do stwierdzenia, czy wymagania wykazują pożądane charakterystyki dotyczące ich kompletności, poprawności oraz wymagalności. Przegląd daje także okazję przedstawicielom klienta do oceny, w jakim stopniu praca wykonywana przez analityka biznesowego zaspokaja potrzeby projektu. Zapracowani klienci często niechętnie poświęcają czas na weryfikowanie wymagań, ale ich zaangażowanie zwróci się z nawiązką. Analityk biznesowy powinien Ci przekazywać wymagania do weryfikacji w ciągu całego procesu ich pozyskiwania w możliwych do ogarnięcia fragmentach. Nie powinien rzucić na Twoje biurko grubego tomu z dokumentacją, gdy gromadzenie wymagań zostanie już przez niego „zakończony”.

Trudno wyobrazić sobie, jak będzie działać oprogramowanie wyłącznie na podstawie wymagań sformułowanych na piśmie. Aby lepiej zrozumieć Twoje potrzeby i poznać sposoby na ich zaspokojenie, analityk biznesowy albo programiści budują czasami prototypy przyszłego produktu. Twoja opinia na temat tych wstępnych, częściowych albo próbnych implementacji stanowi dla programistów bardzo cenną informację.

## Obowiązek nr 8. Ustanawiać kryteria akceptacji

Skąd programiści wiedzą, że skończyli? Skąd wiedzą, że oprogramowanie, które utworzyli, spełnia wymagania różnych klientów? Ponieważ jesteś klientem, jednym z Twoich zadań będzie określenie kryteriów akceptacji, czyli wstępnie zdefiniowanych warunków, które powinien spełnić produkt, aby mógł zostać uznany za gotowy do odbioru. Do kryteriów tych należą testy akceptacji, weryfikujące, czy produkt pozwala użytkownikom prawidłowo wykonywać pewne ważne czynności biznesowe. Inne kryteria akceptacji mogą dotyczyć nieprzekroczenia szacowanego poziomu defektów, osiągnięcia założonej wydajności przeprowadzania określonych operacji w środowisku roboczym albo możliwości spełnienia wymagań stawianych przez certyfikaty zewnętrzne. W celu wyłonienia szczegółów zawartych w opowieściach użytkowników projekty zwinne w znacznie większym stopniu bazują na testach akceptacji niż na wymaganiach podanych w formie pisemnej. Testerzy są w stanie ocenić, czy dane wymaganie zostało prawidłowo zaimplementowane, ale nie zawsze będą dokładnie wiedzieć, co **Twoim** zdaniem jest możliwym do przyjęcia efektem pracy oprogramowania.

## Obowiązek nr 9. Bezwłocznie zgłaszać zmiany wymagań

Nieustannie zmieniające się wymagania w poważnym stopniu zagrażają terminowemu oddaniu przez zespół programistyczny wysokiej jakości produktu. Zmian nie da się uniknąć i często są one cenne, ale im później w procesie tworzenia oprogramowania nastąpi ich wdrożenie, tym większe będzie ich oddziaływanie na projekt. Gdy tylko zdasz sobie sprawę z tego, że musisz zmienić wymaganie, poinformuj o tym analityka biznesowego. Aby ograniczyć negatywny wpływ zmian, kieruj się zdefiniowanym dla projektu procesem kontrolowania zmian. Dzięki temu zagwarantujesz, że zmiany nie zostaną pominięte, wpływ każdej z nich zostanie przeanalizowany, a wszystkie proponowane zmiany będą konsekwentnie rozpatrywane. W rezultacie interesariusze będą mogli podejmować słuszne decyzje dotyczące wprowadzania określonych zmian na właściwych etapach prac nad projektem.

## Obowiązek nr 10. Uszanować proces opracowywania wymagań

Pozyskiwanie oraz określanie wymagań należą do największych wyzwań związanych z tworzeniem oprogramowania. Podejście analityka biznesowego do prac związanych z wymaganiami ma swoje racjonalne uzasadnienie. Chociaż możesz przy tym odczuwać frustrację, czas, jaki poświęcisz na zrozumienie wymagań, jest doskonałą inwestycją. Cały proces stanie się mniej uciążliwy, jeżeli uszanujesz techniki stosowane przez analityka biznesowego. Nie wahaj się prosić go o wyjaśnienie, do czego potrzebne mu są określone informacje, albo dlaczego chce, abyś brał udział w czynnościach związanych z pozyskiwaniem wymagań. Wzajemne zrozumienie oraz szacunek dla sposobu pracy oraz potrzeb drugiej osoby w znacznym stopniu pomagają w ustanowieniu skutecznej współpracy, być może nawet dającej satysfakcję.

## Tworzenie kultury poszanowania wymagań



Szefowa firmowego działu odpowiedzialnego za wymagania zwierzyła się kiedyś z następującego problemu:

— Mam kłopot z uzyskaniem od niektórych z naszych programistów zgody na ich uczestnictwo w pracach nad wymaganiami. W jaki sposób mogę wytłumaczyć im, jaką wartość ma ich wkład?

W innej organizacji analityk biznesowy był świadkiem konfliktu między programistami zbierającymi szczegółowe informacje na temat danych wprowadzanych do systemu księgowego a szefem działu IT, który w kwestii wymagań chciał przeprowadzić burzę mózgów bez uciekania się do jakiegokolwiek techniki pozyskiwania wymagań. Analityk ten zapytał mnie, czy czytelnicy mojej książki są narażeni na podobny konflikt kulturowy.

Problemy te egzemplifikują wyzwania, z jakimi można się spotkać podczas prób zaangażowania analityków biznesowych, programistów oraz klientów do wspólnej pracy nad wymaganiami. Być może sądzisz, że dla użytkownika oczywisty jest fakt, iż z większym prawdopodobieństwem otrzyma to, czego potrzebuje, jeśli przedstawi swoje wymagania. Programiści powinni zdawać sobie sprawę z tego, że ich udział w procesie pozyskiwania wymagań ułatwi im życie i że nie będą później dostawać po głowach nadlatującymi ze wszystkich stron dokumentami zawierającymi wymagania. Oczywiście nie wszyscy są aż tak entuzjastycznie nastawieni do pracy nad wymaganiami, jak Ty. Gdyby tak było, prawdopodobnie każdy z nich zostałby analitykiem biznesowym!

Konflikty kulturowe często pojawiają się wtedy, gdy zespoły programistyczne pracują nad wymaganiami. Istnieją osoby, które znają ryzyko związane z próbami tworzenia oprogramowania w oparciu o minimalne lub komunikowane telepatycznie wymagania. Istnieją też osoby uważające, że wymagania są zbędne. Nawiązanie współpracy przy takich przedsięwzięciach, jak wymiana istniejącego systemu, może być trudne, jeśli użytkownicy są przekonani, że dane przedsięwzięcie nie ma związku z ich obowiązkami służbowymi i nie warto poświęcać na nie czasu. Zrozumienie,

dlatego ludzie nie chcą uczestniczyć w opracowywaniu wymagań, stanowi pierwszy krok w kierunku rozwiązania tego problemu.

Być może niechętnym osobom nie przedstawiono konkretnych praktyk związanych z określaniem wymagań, a może wyniosły one złe doświadczenia ze złe przeprowadzonego procesu implementacji wymagań, uczestnicząc w działaniach, których efektem była obszerna, niekompletna i pomijająca rzeczywiste wymagania specyfikacja. Takie wspomnienia u każdego wywoływałyby niesmak. Prawdopodobnie osoby te nie rozumieją i nie doceniają wartości prawidłowo realizowanych praktyk. Mogą też nie zdawać sobie sprawy z ceny, jaką musiały płacić, pracując w przeszłości w prowizorycznym i źle ustrukturyzowanym środowisku. Ceną tą jest najczęściej konieczność wprowadzania poprawek skutkujących opóźnieniami w oddawaniu finalnych wersji produktu oraz słabej jakości oprogramowanie. Tego typu poprawki są dokonywane przez osoby uczestniczące w projekcie w ramach ich codziennych zadań, w związku z czym nikt nie dostrzega w nich przejawu nieefektywnej pracy.

Jeśli próbujesz zaangażować programistów, menedżerów oraz klientów, upewnij się, że wszyscy z nich rozumieją, iż niedogodności, z jakimi wcześniej miała do czynienia organizacja oraz jej klienci, wynikały z niewłaściwie zdefiniowanych wymagań. Osobom, które osobiście nie doświadczyły żadnych problemów, pokaż przykłady niekorzystnego wpływu, jaki wywierają niewłaściwe wymagania. Wyraż ich koszt w jednostkach, które będą miały sens w danej organizacji; mogą to być pieniądze, czas, niezadowolony klient albo utracone korzyści biznesowe. Menedżerowie ds. rozwoju zwykle nie zdają sobie sprawy, jak bardzo niedociągnięcia w opracowaniu wymagań ograniczają produktywność ich zespołów. Pokaż im zatem, w jakim stopniu niewłaściwe wymagania spowalniają prace nad projektami oraz prowadzą do obszernych i drogich korekt kursu.

Programiści pełnią w projekcie rolę interesariuszy, ale czasami ich wkład nie jest uwzględniany i stają się oni „ofiarami” wymagań, które im narzucono. Programiści powinni zatem odnieść korzyści z udzielania informacji, które sprawią, że dokumentacja wymagań będzie przydatna i zrozumiała w możliwie najwyższym stopniu. Osobiście wolę weryfikować wymagania programistów w miarę ich ewoluowania. Dzięki temu programiści będą wiedzieć, z czym mają do czynienia i zauważą kwestie wymagające dodatkowych wyjaśnień. Wkład programisty jest również potrzebny podczas specyfikowania wewnętrznych atrybutów jakościowych, które nie są widoczne dla użytkowników. Programiści mogą zgłaszać sugestie, o których nikt wcześniej nie pomyślał — prostsze sposoby na realizowanie pewnych czynności, funkcjonalności zbyt czasochłonne do zaimplementowania, niepotrzebnie narzucone ograniczenia, brakujące wymagania (takie jak metody obsługiwanie wyjątków), a także okazje do skorzystania z przewagi, jaką dają nowe technologie.

Zespół kontroli jakości oraz testerzy także wnoszą cenny wkład w opracowywanie doskonałych wymagań. Zamiast czekać na ukończenie produktu, zaangażuj te spostrzegawcze osoby na wczesnym etapie iteracyjnych przeglądów wymagań. Z dużym prawdopodobieństwem podczas opracowywania przypadków użycia i scenariuszy znajdą one wiele niejednoznaczności, konfliktów oraz wątpliwości związanych z wymaganiami. Testerzy mogą także udzielić informacji na temat specyfikowania weryfikowalnych wymagań dotyczących atrybutów jakościowych.

Opór wobec zmian w procesie albo kulturze może wskazywać na obawy, brak pewności lub niedostateczną wiedzę. Jeśli zdołasz określić przyczyny tego oporu, będziesz mógł go załagodzić za pomocą wsparcia, wyjaśnień i edukacji. Pokaż tym osobom, że ich wkład nie leży wyłącznie w ich własnym interesie, ale także przyczynia się do lepszych rezultatów osiągniętych przez cały zespół.

Kierownictwo organizacji musi zrozumieć potrzebę prowadzenia skutecznej analizy biznesowej oraz inżynierii wymagań w ramach strategicznych działań firmy. Chociaż ukierunkowane na dany projekt działania oraz oddolne inicjatywy odgrywają ważną rolę, żadne z osiągniętych ulepszeń ani korzyści prawdopodobnie nie przetrwają zakończenia prac nad projektem ani reorganizacji firmy, jeśli kierownictwo także nie będzie zaangażowane w pracę nad wymaganiami.

## Identyfikowanie osób decyzyjnych



W przypadku projektów programistycznych mogą być podejmowane setki decyzji. Bardzo często mają one krytyczne znaczenie dla możliwości kontynuowania dalszych prac. Może pojawić się potrzeba rozwiązania jakiegoś konfliktu, przyjęcia (albo odrzucenia) zaproponowanej zmiany lub zatwierdzenia zestawu wymagań odnoszących się do określonego wydania produktu. Już na początku prac nad projektem określ, kto będzie podejmować decyzje i w jaki sposób będzie to robić. Mój znajomy, Chris, który jest doświadczonym menedżerem projektu, powiedział mi kiedyś: „Stwierdziłem, że zazwyczaj w projekcie jest jedna osoba podejmująca decyzje. Może to być główny sponsor w ramach danej organizacji. Nie poddaję się, dopóki nie zidentyfikuję tej osoby, po czym pilnuję, aby zawsze była ona informowana o postępach prac nad projektem”. Nie ma definitywnej odpowiedzi na pytanie, kto powinien podejmować kluczowe decyzje. Zwykle najlepiej sprawdza się niewielka grupa odpowiedzialna za główne obszary systemu — mogą to być zarząd, klienci, analitycy biznesowi, programiści albo dział marketingu. W rozdziale 28., „Zmiany się zdarzają”, opisano radę kontroli zmian, która podejmuje decyzje dotyczące wcielania w życie proponowanych zmian w wymaganiach.

Grupa odpowiedzialna za podejmowanie decyzji powinna wybrać **lidera ds. decyzji** oraz określić **regułę decyzyjną**, opisującą sposób podejmowania decyzji. Istnieje wiele różnych reguł decyzyjnych, z których można wybierać. Należą do nich między innymi następujące reguły (Gottesdiener, 2001):

- Wybory podejmuje lider ds. decyzji i omawia je (lub nie) z pozostałymi osobami w grupie.
- Grupa głosuje i podejmowana jest decyzja większości.
- Grupa głosuje, ale do zatwierdzenia decyzji potrzebna jest jednomyślność.
- Grupa prowadzi dyskusję i negocjuje w celu osiągnięcia porozumienia. Wszyscy muszą zaakceptować decyzję i popierać ją.
- Lider ds. decyzji może przekazać odpowiedzialność za podjęcie decyzji określonej osobie.
- Decyzję podejmuje grupa, ale istnieje możliwość zgłoszenia weta do tej decyzji.

Nie istnieje uniwersalnie słuszna ani najlepsza reguła decyzyjna. Dana reguła nie sprawdzi się w każdej sytuacji, w związku z czym grupa powinna przyjąć wytyczne, dzięki czemu będzie wiadomo, kiedy głosować, kiedy dążyć do porozumienia, kiedy przekazywać odpowiedzialność itd. Osoby, które będą podejmować decyzje związane z wymaganiami, powinny wybrać regułę decyzyjną, zanim jeszcze skonfrontują się ze swoją pierwszą poważną decyzją.

## Osiągnięcie porozumienia co do wymagań

Osiągnięcie porozumienia co do wymagań produktu lub jego fragmentu, jaki ma zostać utworzony, leży u podstaw partnerstwa klient-twórca oprogramowania. W osiągnięciu tej zgody zaangażowanych jest wiele stron:

- Klienci zgadzają się, że wymagania zaspokajają ich potrzeby.
- Programiści zgadzają się, że zrozumieli wymagania i że są one wykonalne.
- Testerzy zgadzają się, że wymagania są weryfikowalne.
- Zarząd zgadza się, że wymagania pozwolą osiągnąć jego cele biznesowe.

W wielu organizacjach stosowane jest kwitowanie wymagań na znak ich akceptacji przez interesariusza. Wszyscy uczestnicy procesu zatwierdzania wymagań powinni zdawać sobie sprawę z tego, co takie kwitowanie oznacza i jakie problemy mogą z tego wyniknąć. Jednym z takich zagrożeń jest traktowanie przez przedstawiciela klientów lub menedżera projektu podpisywania wymagań jako pozbawionego znaczenia rytuału: „Dostałem jakąś kartkę papieru z moim nazwiskiem, więc ją podpisałem, bo inaczej programiści nie mogliby zacząć pisać kodu”. Takie podejście może prowadzić do wystąpienia w przyszłości problemów, gdy osoba, która podpisała dokument, jest zaskoczona tym, co zostało wyprodukowane: „No pewnie, że pokwitowałem wymagania, ale nie miałem czasu, żeby je wszystkie czytać. Zaufałem wam, chłopaki, a wy mnie zawiedliście”.

Równie problematyczne jest traktowanie przez menedżera ds. rozwoju kwitowania jako sposobu na zamrożenie wymagań. Gdy tylko nastąpi zmiana wymagań, zaprotestuje on: „Przecież pokwitowaliście wymagania, tak więc właśnie to będziemy budować. Skoro chcieliście czegoś innego, trzeba było powiedzieć”.

Obie te postawy ignorują fakt, że nie da się poznać wszystkich wymagań na wczesnym etapie projektu i że wymagania bez wątplenia ulegną zmianie wraz z upływem czasu. Zatwierdzanie zestawu wymagań jest słusznym działaniem, które zamyka pewien etap prac nad wymaganiami. Wszyscy uczestnicy projektu powinni jednak dokładnie zdawać sobie sprawę, co oznacza ich podpis złożony na dokumencie.



**Ważne.** Nie stosuj kwitowania jako broni. Uważaj je za kamień milowy; za jasne wyrażenie wspólnego porozumienia co do czynności, które doprowadziły do złożenia podpisów na dokumencie, oraz skutków, jakie będą one wywierać na przyszłe zmiany. Jeśli osoby decyzyjne nie muszą czytać każdego słowa w specyfikacji wymagań, wybierz jakąś technikę komunikacji, taką jak pokaz slajdów, która pozwoli podsumować najważniejsze elementy i ułatwi szybkie osiągnięcie porozumienia.

## Baza dla wymagań

O wiele ważniejsze od rytuału kwitowania jest ustanowienie **bazy odniesienia** dla porozumienia dotyczącego wymagań, tj. obrazu chwili, który reprezentuje obecne porozumienie (Wiegers, 2006). Baza odniesienia jest zweryfikowanym i zatwierdzonym zbiorem wymagań, które służą jako podstawa do dalszego rozwijania projektu. Obojętne, czy Twój zespół stosuje formalny proces kwitowania, czy też jakiś inny sposób na osiąganie porozumienia w kwestii wymagań, tekst takiego porozumienia powinien brzmieć mniej więcej następująco:

*Potwierdzam, że powyższy zbiór wymagań przedstawia nasze rozumienie wymagań, które dotyczą kolejnego etapu prac nad projektem, oraz że opisane tu rozwiązania spełnią nasze potrzeby, jak je w obecnej chwili rozumiemy. Wyrażam zgodę na wprowadzanie w niniejszej bazie przyszłych zmian dotyczących wymagań, zgodnie ze zdefiniowanym dla projektu procesem zmian. Zdaję sobie sprawę z faktu, że zmiany mogą wymagać renegotjowania zobowiązań dotyczących kosztów, zasobów oraz harmonogramu.*

W niektórych organizacjach podobny do powyższego zapis jest umieszczany na stronie z podpisami, dzięki czemu osoby parafujące wymagania będą dokładnie wiedzieć, co kwitowanie oznacza w ich sytuacji.

Porozumienie, które jest osiągnięte w podobny sposób, pomaga zredukować konflikty, które mogą się pojawić, gdy zostaną ujawnione przeoczone wymagania lub kiedy w czasie prac nad projektem zmieniają się wymagania rynkowe albo biznesowe. Dobrze zdefiniowany proces tworzenia bazy odniesienia pozwala zachować wszystkim głównym interesariuszom pewność co do następujących sytuacji:

- Dział kontaktu z klientem albo dział marketingu jest pewny, że zakres projektu nie wymknie się spod kontroli, ponieważ klienci weryfikują decyzje dotyczące zmiany zakresu projektu.
- Przedstawiciele użytkowników są pewni, że zespół programistyczny będzie z nimi współpracować w celu znalezienia właściwego rozwiązania, nawet jeśli w chwili rozpoczęcia prac nie wzięto pod uwagę wszystkich wymagań.
- Kierownictwo zespołu programistycznego może być spokojne, gdyż zespół programistyczny ma partnera biznesowego, który dopilnuje, aby praca nad projektem była skupiona na osiągnięciu założonych celów, i który współpracuje z zespołem w celu zrównoważenia harmonogramu, kosztów, funkcjonalności oraz jakości.
- Analitycy biznesowi oraz menedżerowie projektu są pewni, że są w stanie zapanować nad zmianami w sposób, który ograniczy do minimum chaos.
- Zespół kontroli jakości oraz testerzy mogą bez obaw opracowywać skrypty testowe i przygotowywać się do wykonywania swoich zadań w ramach projektu.

Gdy osoby decyzyjne zdefiniują już bazę odniesienia, analityk biznesowy powinien zacząć kontrolować zmiany, jakim są poddawane wymagania. Dzięki temu zespół będzie mógł w razie konieczności zmienić zakres projektu w kontrolowany sposób, który uwzględni badanie wpływu proponowanych zmian na harmonogram prac oraz bierze pod uwagę pozostałe czynniki mające wpływ na osiągnięcie sukcesu. Potwierdzenie jednoznaczną umową wstępnych czynności związanych z opracowywaniem wymagań pozwala kształtować współpracę klient-twórca oprogramowania na drodze zmierzającej do osiągnięcia sukcesu w projekcie.

## Co zrobić, jeśli nie osiągnięto porozumienia?

Zdobycie pokwitowań od wszystkich istotnych interesariuszy może być trudne. Przeszkody mogą wynikać z przyczyn logistycznych, napiętych terminarzy, a także z niechęci niektórych osób do zaciągania zobowiązań, z których będą później rozliczani. Jeśli interesariusze obawiają się, że po zatwierdzeniu wymagań nie będą mogli ich zmieniać, mogą zwlekać z ich pokwitowaniem. Taka sytuacja może spowodować wpadnięcie w straszną pułapkę paraliżu analitycznego. Wiele zespołów próbowało wysyłać wiadomości e-mail o następującej treści: „Jeśli do najbliższego piątku nie zadeklarujesz się co do swoich zmian albo ich nie pokwitujesz, uznamy, że akceptujesz wymagania w ich obecnej postaci”. Jest to jedna z dostępnych opcji, ale tak naprawdę **nie** jest ona równoznaczna z osiągnięciem porozumienia. Opcja ta wiąże się także z ryzykiem wytworzenia napiętych relacji z tymi interesariuszami, co do których założyłeś ich milczącą zgodę. Postaraj się zrozumieć, dlaczego nie czują się oni komfortowo, musząc kwitować wymagania, i postępuj stosownie do sytuacji.

W takich przypadkach postępuj lepiej, ruszając ostrożnie do przodu przy jednoczesnym założeniu, że nie uzyskałeś potwierdzeń od wszystkich niesubordynowanych interesariuszy. Odnotuj na swojej liście zagrożeń fakt, że pewni interesariusze nie pokwitowali wymagań, łącznie z przewidywanym skutkiem, jaki brak części wymagań lub ich nieprawidłowe określenie wywrze na projekt. Skontaktuj się z tymi osobami w ramach czynności związanych z zarządzaniem ryzykiem. W pozytywny sposób powiedz im, że zdajesz sobie sprawę, iż nie zatwierdzili oni jeszcze wymagań, ale że prace nad projektem posuwają się do przodu z obecnymi wymaganiami stanowiącymi bazę odniesienia po to, aby nie opóźnić projektu. Poinformuj ich, że jeśli chcą wprowadzić zmiany, istnieje służący do tego celu proces. Powinieneś postępować w taki sposób, jakby interesariusz istotnie zatwierdził wymagania, a Ty zajmujesz się tylko nawiązaniem z nim kontaktu.

## Zgoda co do wymagań w projektach zwinnych

W projektach zwinnych nie ma miejsca na formalne pokwitowania. W tego typu projektach zarządzanie wymaganiami przyjmuje zwykle formę opowieści użytkowników w rejestrze wymagań. Właściciel produktu oraz zespół osiągają podczas sesji planistycznych porozumienie co do opowieści, które zostaną zaimplementowane w kolejnej iteracji. Zestaw opowieści jest dobierany na podstawie ich priorytetów oraz sprawności (wydajności) zespołu. Po wybraniu i zatwierdzeniu zestawu opowieści w nim zawarte zostają zamrożone. Zgłoszone zmiany w wymaganiach zostaną rozpatrzone w przyszłych iteracjach. W projektach zwinnych nie próbuje się z góry uzyskać zatwierdzenia pełnego zakresu wymagań przez interesariusza. W takich projektach pełny zbiór funkcjonalności wyłania się wraz z upływem czasu, chociaż wizja oraz pozostałe wymagania biznesowe powinny zostać zdefiniowane już na początku. W rozdziale 20., „Projekty zwinne”, omówiono, jak obsługiwane są wymagania w programowaniu zwinnym.



Pracowałem kiedyś z klientem, który domagał się kwitowania wymagań, chociaż projekt był tworzony zgodnie z metodyką zwinną. Nasz zespół musiał wykazać się kreatywnością, aby dawać sobie radę w kontekście, w którym zwykle nie stosuje się kwitowania. Zespół analityków biznesowych współpracował blisko z użytkownikami w celu pozyskiwania wymagań w postaci opowieści użytkowników oraz innych modeli, takich jak przepływy procesów oraz tabele stanów. Prosiłiśmy użytkowników, aby „kwitowali” te dokumenty w odpowiednich momentach, gdy nie było żadnych brakujących wymagań, **o których mogliby oni wiedzieć**, a także gdy nie było żadnych zastrzeżeń co do spisanych przez nas wymagań, **które mogliby oni zgłosić**. Ponieważ użytkownicy brali udział w czynnościach związanych z określaniem wymagań, programiści nie pracowali nad rozwiązaniem, które daleko odbiegałoby od przyjętych założeń. Taka definicja „kwitowania” daje również użytkownikom prawo do późniejszego stwierdzenia, że potrzebne im jest nowe rozwiązanie albo że coś zdefiniowali źle.

W odróżnieniu od historycznego znaczenia kwitowania jako „zatwierdzania i zamrażania wszystkich wymagań na wstępie” tego typu podejście nie spycha nikogo w róg, gdzie można się poczuć, jakby się całym swoim życiem poświadczало ogromny dokument, który z trudem można zrozumieć. Klienci także nie są zmuszani do potwierdzenia, że wymagania są bliskie ideałowi i że wszystkie kwestie zostały uwzględnione już na samym początku. Taki sposób kwitowania pozwala zachować istotę metod zwinnych. Podobnie jak to jest w przypadku opisanego wcześniej procesu kwitowania, jego istotą jest osiągnięcie porozumienia co do pewnego zestawu wymagań — bazy odniesienia — który zostanie zaimplementowany w następnym cyklu wytwórczym; porozumienia dającego jasny obraz oraz wspólne rozumienie tego, co tak naprawdę ono oznacza.

Zazwyczaj w projektach zwinnych to właściciel produktu jawnie zatwierdza lub odrzuca wymagania do wdrożenia w danej iteracji. Wymagania te składają się ze zbioru opowieści wraz z towarzyszącymi im kryteriami akceptacji oraz testami akceptacyjnymi. Ostatecznym „pokwitowaniem” jest przyjęcie działającego i przetestowanego oprogramowania, które zostało otrzymane w danej iteracji.

Jak sformułowała to konsultantka Nanette Brown: „Nawet w środowisku zwinnym koncepcja kwitowania może pełnić ważną funkcję. Proces zwinny nakazuje nam »objąć zmianę«, ale pojęcie zmiany funkcjonuje wyłącznie z punktem odniesienia. Nawet w zespołach, w których istnieje dobra komunikacja, poszczególne osoby mogą różnie interpretować bieżące plany oraz aktualny status projektu. Dla jednej osoby »zmiana« może oznaczać to, co inna osoba uważa za coś już zatwierdzonego. Nie ma jednak przeszkód, abyś uważał kwitowanie za błahą czynność, potwierdzającą, że »jesteśmy tutaj«. Dzisiejsze »jesteśmy tutaj« nie oznacza, że jutro nie możemy być gdzie indziej, ale przynajmniej dzięki temu wiemy, że osiągnęliśmy porozumienie oraz wspólny punkt odniesienia”.



## Następne kroki

- Zidentyfikuj klientów, w tym użytkowników końcowych, którzy są odpowiedzialni za definiowanie w Twoim projekcie wymagań biznesowych oraz wymagań użytkowników. Które z zapisów zawartych w karcie praw oraz karcie obowiązków stosują oni w praktyce? Których nie stosują?
- Przedyskutuj kartę praw ze swoimi głównymi klientami, aby dowiedzieć się, czy ich zdaniem korzystają oni ze swoich praw. Omów kartę obowiązków, aby osiągnąć porozumienie co do obowiązków, których mogą oni przestrzegać. Zmień kartę praw oraz kartę obowiązków w taki sposób, aby wszystkie strony zgodziły się co do zasad opisujących Waszą współpracę. Sprawdź, czy wszyscy interesariusze zachowują równowagę między przysługującymi im prawami a obowiązkami.
- Jeśli jesteś klientem biorącym udział w projekcie programistycznym i uważasz, że Twoje prawa nie są przestrzegane w zadowalającym stopniu, omów kartę praw z menedżerem projektu albo analitykiem biznesowym. Ze swojej strony zadeklaruj chęć przestrzegania karty obowiązków w dążeniu do relacji opartej w większym stopniu na współpracy.
- Jeśli w Twojej organizacji istnieje formalny proces kwitowania, zastanów się, jaką rolę odgrywa on obecnie. Podejmij współpracę z menedżerem programistów albo szefem działu kontaktu z klientem (albo marketingu), aby osiągnąć porozumienie dotyczące faktycznego znaczenia kwitowania w ramach procesu zatwierdzania wymagań.
- W bieżącym projekcie lub w jednym z poprzednich projektów poszukaj przykładu na niedostateczne zaangażowanie klienta. Zastanów się, jaki wpływ miało to na projekt. Sprawdź, czy możesz oszacować ryzyko pod względem liczby późniejszych zmian w wymaganiach, czasu przeznaczonego na wprowadzanie poprawek w produkcie już po jego oddaniu lub straconych możliwości biznesowych. Na podstawie zdobytego doświadczenia wyciągnij wnioski na przyszłość, a także przekonaj inne osoby, że zaangażowanie klienta odgrywa ważną rolę.



# Skorowidz

## A

adaptacja użytkowników, 527  
aktor, 172  
    drugorzędny, 172  
    główny, 172  
alokacja, 387  
alokacja systemu, 454  
analityk biznesowy, 79, 85, 141, 534  
    umiejętności, 88  
    wiedza, 92  
    zadania, 87  
analiza, 68, 449, 450, 581  
    danych, 271  
    dokumentów, 73, 152  
    interfejsów, 75  
    interfejsu systemu, 151  
    interfejsu użytkownika, 152  
    luk, 411  
    przyczyn źródłowych, 538  
    raportów, 74  
    wpływu zmiany, 497  
    wykonalności wymagań, 75  
    wymagań, 42, 74, 558  
    zmian, 78  
aplikacja, 30  
architektura, 387  
architektura systemu, 454  
aspekty modyfikowalności, 300  
atribut jakościowy, 33, 160, 218, 280–285, 305, 462, 610  
    wewnętrzny, 281, 299  
    efektywność, 299  
    modyfikowalność, 299

    powtórne użycie, 301  
    przenośność, 301  
    skalowalność, 302  
    weryfikowalność, 303  
zewnętrzny, 281, 286  
    bezpieczeństwo, 295  
    dostępność, 286  
    instalowalność, 287  
    integralność, 288  
    interoperacyjność, 289  
    koszt jakości, 286  
    niezawodność, 292  
    ochrona, 294  
    użyteczność, 296  
    wydajność, 290  
    wytrzymałość, 293

    atributy, 266  
    wniosków o zmianę, 493  
    wymagań, 476  
automatyzacja procesów, 437  
    biznesowych, 440  
awarie, 465

## B

badanie przypadków użycia, 182  
baza  
    dla wymagań, 63, 473  
    odniesienia, 78  
bazowy plan wymagań, 81  
bezpieczeństwo, 219, 295  
bezpieczeństwo systemów  
    wbudowanych, 466  
biznesowe miary wydajności, 438  
braki, 210

## C

cechy wymagań  
    jednoznaczność, 225  
    kompletność, 224  
    niezbędność, 225  
    odpowiednia szczegółowość, 231  
    poprawność, 224  
    priorytet, 225  
    weryfikowalność, 225  
    wykonalność, 224  
cechy zbiorów wymagań  
    kompletność, 226  
    modyfikowalność, 226  
    możliwość śledzenia, 226  
    spójność, 226  
cel, 489  
    biznesowy, 104, 121, 410  
    pozyskiwania, 153  
    usprawnienia, 547  
charakterystyki użytkowników, 214  
chmura, 419  
cykl  
    usprawniania procesu, 539  
    życia projektu, 80, 344

## D

decyzje, 444  
definiowanie  
    atributów jakościowych, 285  
    kryteriów akceptacji, 77  
    wymagań biznesowych, 102

## definiowanie

- wymagań jakościowych, 421
- zakresu, 120

- deklaracja wizji, 110

- dekompozycja opowieści, 402

## diagram

- aktywności, 246
- architektury, 459
- czynności, 262
- klas, 245, 261, 267
- kontekstowy, 74, 116, 245, 456
- przejsć stanów, 245, 251, 456
- przepływu danych, 245–248
- przyczyn i skutków, 539
- przypadków użycia, 172, 245, 262
- relacji encji, 267
- stanów, 262
- torowy, 245, 250
- związków encji, 245, 266

- diagramy UML, 261

## dobre praktyki, 72

- analizowanie wymagań, 74
- pozyskiwanie wymagań, 72
- specyfikowanie wymagań, 76
- w inżynierii wymagań, 68
- walidacja wymagań, 77
- wiedza, 79
- zarządzanie projektem, 80
- zarządzanie wymaganiami, 48, 77

## dokumentowanie

- reguł biznesowych, 196
- ryzyka, 553
- wizji i zakresu, 35, 105, 591
- wymagań, 203, 353, 591

- doświadczenie, 82

- dowód koncepcji, 316

## drzewo

- decyzyjne, 245, 257
- funkcjonalności, 37, 118

- duże dane, 447

- dystrybucja kwestionariuszy, 73

**E**

- efektywność, 299

## elementy

- modeli analitycznych, 244
- procesu
  - inżynierii wymagań, 543
  - zarządzania wymaganiami, 545
- zarządzania ryzykiem, 552

- elipsa, 34

- encje, 266

- epika, 402

- etapy procesu inspekcji, 349

- etykiety, 208

**F**

- facylitacja warsztatów

- pozyskiwania wymagań, 73

- fakty, 192

- fakty pochodne, 195

- format warunek-konsekwencje, 554

- funkcjonalności, 33, 37

- główne, 113

- systemu, 215, 603

**G**

- godzenie sprzeczności, 140

## gotowe

- produkty, 426, 427
  - rozwiązania, 419

## grupa

- fokusowa, 73, 148
  - interesariusza, 534
  - powtarzalna, 270

**H**

- harmonogram, 385

- hierarchia interesariuszy, 127

- hierarchiczne znaczniki

- tekstowe, 209

- historia zmian, 78

**I**

- identyfikowanie

- klas użytkowników, 129
  - korzyści biznesowych, 102
  - osób decyzyjnych, 62
  - potrzeb, 421
  - przypadków użycia, 181
  - wymagań użytkowników, 73
  - zdarzeń, 73
  - źródeł wymagań, 76

## implementacja

- dobrych praktyk, 83
  - gotowych produktów, 424
  - inżynierii wymagań, 529
  - wymagań, 307

## informacje

- na temat wymagań, 34, 368
  - od użytkownika, 159

## informowanie

- interesariuszy, 79
  - programistów, 80

## inspekcja dokumentacji

- wymagań, 346
  - etapy, 349
  - kryteria końcowe, 351
  - kryteria początkowe, 348
  - role, 348
  - uczestnicy, 347

- instalowalność systemu, 288

- integralność, 288

- integralność danych, 216

- interesariusz, 51

## interfejs, 36

- komunikacyjny, 218, 610
  - oprogramowania, 218, 609
  - sprzętowy, 218, 609
  - systemu, 151
  - użytkownika, 74, 152, 210, 217, 390, 609
  - zewnętrzny, 609

- interoperacyjność, 289

- inżynieria wymagań, 67, 80, 517, 529, 543

- iteracja, 397, 401

## J

język Planguage, 304, 310

## K

kardynalność, 267

karta

- obowiązków klienta, 54, 57
- praw klienta, 54, 55

katalog reguł biznesowych, 197

klasy

- projektów, 395
- użytkowników, 72, 126, 130, 214

klient, 52

kokpit zarządzania, 277

komunikacja, 86, 577

konfiguracja narzędzia, 525

konflikty kulturowe, 60

koniec projektu, 123

konstrukcja

- interfejsu użytkownika, 389
- oprogramowania, 388

kontekst biznesowy, 114, 595

kontrolowanie

- wersji wymagań, 474
- zmian, 487, 488, 489

korzystanie z informacji, 445, 446

korzyści biznesowe, 102

koszt jakości, 286

krotność, 267

kryteria

- akceptacji, 77, 359, 433
- dopasowania, 344
- końcowe, 492
- początkowe, 490

kwestionariusze, 150

## L

lider ds. decyzji, 62

linia oprogramowania, 369

lista

- kontrolna defektów, 351
- zdarzeń, 119

logiczny model danych, 216

logika biznesowa, 189

luka oczekiwania, 50

luki, 411

## Ł

łańcuch śledzenia, 511

łatwości

- uczenia się, 297
- użytkowania, 297

łącza śledzenia, 510, 512

## M

macierz

- CRUD, 271
- priorytetów, 339
- śledzenia wymagań, 79, 509
- z ocenami produktu, 423

magazyn danych, 247

makieta, 315

mapa

- dialogu, 246, 254, 322
- przypadku użycia, 358
- ekosystemu, 117

maszyna stanów, 251

materiały uzupełniające, 214

menedżer projektu, 94, 534

metoda

- MoSCoW, 336
- stu złotych, 337

miary sukcesu, 108

mistrz produktu, 72, 133

- angażowanie, 137
- zadania, 135
- zewnętrzny, 134

model

- celów biznesowych, 110
- kluczowych wskaźników wydajności, 439
- systemu, 136
- V, 344

modele analityczne, 243, 244

modelowanie

- problemów, 249
- procesów biznesowych, 436
- relacji, 265
- systemów czasu rzeczywistego, 455
- środowiska aplikacji, 74
- wymagań, 75, 242

modyfikowalność, 299

możliwość biznesowa, 107

## N

najmniejsza zbywalna

funkcjonalność, 402

nakłady na wymagania, 380, 480

narzędzia do

- kontrolowania zmian, 495
- modelowania, 520
- pozyskiwania wymagań, 519
- prototypowania, 519
- śledzenia wymagań, 512
- zarządzania wymaganiami, 79, 517–524

nearshoring, 429

niedokładne planowanie, 46

niekompletność, 236

niepodzielne reguły biznesowe, 196

nierealne oczekiwania, 326

niezawodność, 292

numerowanie hierarchiczne, 208

## O

obliczenia, 195

obowiązki klienta, 57

obserwacje, 149

obserwowanie użytkowników, 73

ocenianie

- bieżących praktyk, 540
- produktu, 422
- prototypów, 323
- stabilności wymagań, 496
- wieloetapowe, 424
- wymagań, 352, 353
- wyników, 542

ochrona, 294

odkrywanie

- atrybutów jakościowych, 282
- reguł biznesowych, 198

odpowiedzialności, 489

offshoring, 429

ograniczające reguły biznesowe, 193

ograniczanie ryzyka, 313

ograniczenia, 33, 36, 161, 192, 308, 593

- procesu kaskadowego, 398
- projektu, 112, 214

- określanie priorytetów wymagań, 329, 333, 338, 410
- opis
  - ogólny, 214, 601
  - procesu kontrolowania zmian, 489
- opowieści użytkowników, 170, 246, 402
- opracowywanie wymagań, 38, 44, 70, 99, 379, 420, 519, 545
- oprogramowanie
  - dedykowane, 426
  - jako usługa, 419
- outsourcing, 429
- oznaczanie
  - przypadków użycia, 175
  - wymagań, 76
- P**
- paraliż analityczny, 165
- partnerstwo klient-twórca
  - oprogramowania, 53
- pełzające wymagania
  - użytkowników, 46
- pełzanie zakresu, 112, 486
- personalizacje użytkowników, 131
- perspektywa
  - produktu, 214
  - systemu, 227
  - użytkownika, 227
- pilotowanie procesów, 542
- pisanie wymagań, 227, 228
- plan
  - projektu, 383
  - usprawnienia, 549
  - wymagań, 81
- planowanie, 575
  - działań ulepszających, 540
  - pozyskiwania wymagań, 153
  - zarządzania ryzykiem, 556
- porozumienie dotyczące
  - wymagań, 63
- poszanowanie wymagań, 60
- powtórne użycie, 301
- poziom
  - jakości usług, SLA, 286
  - szczegółowości, 231
- poziomych wymagań, 33, 37
- pozyskiwanie wymagań, 37, 41, 47, 72, 143–165, 519, 557
  - analiza dokumentów, 152
  - analiza interfejsów, 151
  - cele, 153
  - czynności, 156, 158
  - dokumenty, 153
  - grupy fokusowe, 148
  - harmonogram, 153
  - klasyfikowanie informacji, 159
  - kwestionariusze, 150
  - obserwacje, 149
  - planowanie, 153
  - przygotowania, 154
  - pytania bezkontekstowe, 164
  - ryzyko, 153
  - spodziewane efekty, 153
  - strategia, 153
  - warsztaty, 146
  - wymagania brakujące, 165
  - wywiady, 145
- praktyki inżynierii wymagań, 565
- prawa klienta, 55
- priorytety
  - projektu, 115
  - wymagań, 329–335
- priorytetyzacja
  - atributów jakościowych, 283
  - prac, 444
  - wieloprzebiegowa, 336
  - wymagań, 75
- problemy
  - z analizą, 581
  - z komunikacją, 577
  - z planowaniem, 575
  - z pozyskiwaniem, 578
  - z procesami, 573
  - z produktem, 574
  - z walidacją, 586
  - z wymaganiami, 479, 571
  - z zarządzaniem
    - wymaganiami, 587
    - z zarządzaniem zmianami, 587
  - z specyfikowaniem, 585
- proces, 190, 573
  - biznesowy
    - analiza, 436
    - doskonalenie, 436
    - model, 436
    - modelowanie, 437
    - notacja, 436
    - reengineering, 436
    - zarządzanie, 436
  - kaskadowy, 398
  - kontroli zmian, 78
  - kontrolowania zmian, 488
  - opracowywania wymagań, 70, 71
  - projektu, 532
  - przyszłe, 436
  - zarządzania wymaganiami, 472
  - zastany, 436
- produkt, 30, 574
- profil operacyjny, 423
- profile interesariuszy, 114
- programista, 93
- programowanie zwinne, 397
- projekty
  - analityki biznesowej, 441
  - automatyzacji
    - procesów biznesowych, 435
  - bazujące
    - na gotowych rozwiązaniach, 419
  - systemów
    - czasu rzeczywistego, 453
    - wbudowanych, 453
  - ulepszające, 407
  - zastępujące, 407
  - zlecane na zewnątrz, 429
  - zwinne, 397–405
    - atributy jakościowe, 310
    - dokumentacja, 400
    - modelowanie, 262
    - opracowywanie wymagań, 399, 403
    - przedstawiciele
      - użytkowników, 138
    - rejestr wymagań, 400
    - rola analityka, 95
    - specyfikacja wymagań, 220
    - wizja, 122
    - wymagania, 65
    - zaangażowanie klienta, 399
    - zakres, 122
    - zarządzanie wymaganiami, 482
    - zarządzanie zmianami, 501

propozycje rozwiązań, 161  
 prostokąt, 34  
 prototyp, 314  
   do wyrzucenia, 316, 318, 321  
   ewolucyjny, 318  
 prototypowanie, 519  
   nakłady, 327  
   oprogramowania, 313  
   powodzenie, 327  
   rozproszenie szczegółami, 326  
   ryzyka, 325  
   wydajność, 326  
   wymagań, 355  
 prototypy  
   elektroniczne, 319  
   ewolucyjne, 316, 317  
   papierowe, 319  
   pionowe, 316  
   techniczne, 74  
 przedstawiciele użytkowników,  
 132, 138  
 przeglądy  
   formalne, 346  
   koleżeńskie, 345  
   nieformalne, 346  
   wymagań, 77, 345  
 przenośność, 301  
 przepływ  
   alternatywny, 176  
   interfejsu użytkownika, 254  
   normalny, 176  
 przeróbki, 45  
 przetwarzanie danych, 449  
 przydzielanie wymagań, 75  
 przypadki użycia, 169, 173,  
 180–186, 322, 597  
 przystosowanie wymagań, 372  
 przywracalność, 293  
 pytania bezkontekstowe, 164

## R

rada kontroli zmian  
   renegocjowanie zobowiązań,  
   495  
   skład, 494  
   statut, 494  
 ramy czasowe, 122  
 raportowanie statusu zmiany, 492

raporty, 216, 608  
 recenzenci, 354  
 reengineering, 369  
 reguły  
   biznesowe, 33, 36, 76, 160,  
   180, 189–193, 421  
   dokumentacja, 196  
   fakty, 192  
   niepodzielne, 196  
   obliczenia, 195  
   odkrywanie, 199  
   ograniczenia, 192  
   systematyka, 191  
   wnioski, 195  
   wyzwalacze działań, 194  
 decyzyjne, 62  
 minimalnych odległości, 198  
 rejestr wymagań, 138, 400  
 relacja, 267  
   typu extend, 179  
   typu include, 179  
 relacje  
   między atrybutami, 305  
   między danymi, 265  
 renegocjowanie zobowiązań  
   projektowych, 81  
 reprezentacja, 245  
 rodzaje wymagań, 33, 190  
 rola, 489  
   analitka, 95  
   analitka biznesowego, 86  
 rozkład nakładów pracy, 71  
 rozwiązanie, 30  
 ryzyka, 81, 313, 551  
   biznesowe, 111  
   prototypowania, 325  
   związane z wymaganiami,  
   556

## S

SaaS, Software as a Service, 419  
 scenariusze użytkownika, 173  
 schematy blokowe, 246  
 sekwencja czynności, 321  
 skalowalność, 302  
 SLA, Service Level Agreement, 286  
 słownik danych, 75, 80, 216, 245,  
 268, 605

słowo kluczowe  
   GOAL, 310  
   METER, 310  
   SCALE, 310  
   STRETCH, 310  
   WISH, 310  
 sortowanie wymagań, 330  
 specyfikacja  
   funkcjonalna, 205  
   techniczna, 307  
   SRS, 35, 210  
 specyfikowanie, 68  
   potrzeb danych, 446  
   raportów, 272, 273  
   wymagań oprogramowania,  
   35, 42, 76, 205, 220, 233,  
   412, 558, 585, 600  
   danych, 265  
   jakościowych, 304  
   pozafunkcjonalnych, 76  
 spotkanie inspekcyjne, 350  
 SRS, software requirements  
   specification, 35, 210  
 stany wnioskowanych zmian,  
 490  
 status wymagań, 78, 477  
 stopień swobody, 115  
 struktura  
   analitki biznesowej, 442  
   danych, 270  
 strzałka, 34  
 stymulator, 115  
 symulowanie wymagań, 77  
 system, 30, 453  
 systematyka reguł biznesowych, 191  
 systemy czasu rzeczywistego,  
 453, 455  
   interfejsy, 460  
   modelowanie, 455  
   wymagania czasowe, 461  
 systemy wbudowane, 453  
   atomyty jakościowe, 462  
   bezpieczeństwo, 466  
   efektywność, 463  
   niezawodność, 464  
   ochrona, 465  
   użyteczność, 466  
   wydajność, 463  
   wytrzymałość, 464

szablon, 106  
 analizy wpływu, 501  
 opisu procesu kontrolowania  
 zmian, 489  
 specyfikacji raportu, 274  
 specyfikacji wymagań,  
 212–219  
 SRS, 526  
 śledzenia elementów ryzyka,  
 554  
 wymagań, 212  
 szablony dokumentów  
 wymagań, 76  
 szacowanie  
 nakładów, 81, 383, 500  
 priorytetów, 334  
 ryzyka, 553  
 szczegółowość  
 dokumentacji, 400, 413  
 wymagań, 430  
 szkielec, 317, 323  
 szkolenie analityków  
 biznesowych, 79

## Ś

śledzenie  
 elementów ryzyka, 554  
 nakładów, 82  
 problemów, 78  
 statusów wymagań, 477  
 wymagań, 78, 414, 505–514

## T

tabela  
 decyzyjna, 257  
 stanów, 245, 251  
 śledzenia, 509  
 zdarzenie-reakcja, 245, 259,  
 457  
 TBD, to be determined, 210  
 techniki  
 określania priorytetów, 333  
 pozyskiwania wymagań, 145,  
 154  
 przedstawiania wymagań, 232  
 przedstawiania zakresu, 116  
 reprezentacji, 245

tester, 93  
 testowanie wymagań, 77, 355, 391  
 testy, 186  
 testy akceptacyjne, 310, 361  
 tolerancja na błędy, 293  
 tworzenie  
 harmonogramu, 385  
 interfejsu użytkownika, 74  
 planu bazowego, 207  
 procesów, 542  
 słownika, 80  
 typ danych WORD, 309

## U

ulepszanie procesów, 531, 536, 538  
 umiejętności analityka, 88  
 UML, 261, 455  
 unikanie  
 niekompletności, 236  
 ryzyka, 553  
 wieloznaczności, 233  
 usuwanie  
 danych, 216  
 problemów, 572  
 wieloznaczności, 234, 235  
 uszczegółowianie przypadków  
 użycia, 178  
 uzgadnianie wymagań, 62  
 użyteczność, 218, 296  
 użytkownik, 92  
 klasyfikacja, 126, 129  
 personifikacja, 131  
 użytkownik końcowy, 52

## W

walidacja, 68  
 przypadków użycia, 184  
 wymagań, 42, 77, 343, 359,  
 559, 586  
 warsztaty, 146  
 warunki  
 końcowe, 175, 180  
 początkowe, 175, 180  
 wdrażanie, 115  
 nowych praktyk, 82  
 procesów, 542  
 rozwiązań, 572

wersje wymagań, 474  
 weryfikowalność, 303  
 wiedza, 69, 79  
 analityka, 92  
 wywnioskowana, 195  
 wielkość projektu, 383  
 wielokrotne użycie wymagań,  
*Patrz* wykorzystywanie  
 wymagań  
 wieloznaczność, 233–235  
 wizja produktu, 72, 102, 111  
 właściciel produktu, 95  
 wnioski, 195  
 wskaźniki efektywności, 439, 547  
 współpraca, 96  
 wybór  
 mistrza produktu, 72  
 produktów gotowych, 420  
 wydajność, 219, 290, 411, 438  
 wydajność prototypu, 326  
 wyjątki, 176  
 wykonawca, 431  
 wykorzystywanie wymagań, 363  
 czynniki sukcesu, 376  
 mechanizm, 366  
 narzędzia, 372  
 przeszkody, 374  
 scenariusze, 369  
 skala, 365  
 zakres modyfikacji, 366  
 wymagania  
 analiza, 558  
 atrybuty, 476  
 bazujące na danych, 447  
 biznesowe, 33, 53, 101–123,  
 159, 190, 591  
 sprzeczne, 104  
 szablon, 106  
 czasowe, 461  
 danych, 161  
 dokumenty, 591  
 doskonałe, 53, 223  
 dotyczące  
 atrybutów jakościowych,  
 307  
 bezpieczeństwa, 610  
 danych, 216, 426, 605  
 dostępności, 611  
 etykiet, 208

implementacji gotowych produktów, 424  
 integracji, 425  
 interfejsów zewnętrznych, 609  
 konfiguracji, 425  
 ochrony, 610  
 oprogramowania, 29, 31, 212, 551  
 przetwarzania dokumentów, 232  
 raportów, 272  
 rozszerzeń, 426  
 użyteczności, 610  
 wyboru produktów gotowych, 420  
 wydajności, 610  
 wytrzymałości, 611  
 funkcjonalne, 33, 160, 185, 208, 215, 245, 415  
 interfejsu, 33  
 interfejsów zewnętrznych, 217  
 jakościowe, 279, 304, 421  
 lokalizacyjne, 219  
 międzynarodowe, 219  
 nakłady, 480  
 negatywne, 236  
 niejednoznaczne, 46  
 oczywiste, 164  
 określanie priorytetów, 329  
 pochodne, 164  
 ponowne wykorzystanie, 363, 364  
 pominięte, 165  
 porównywanie parami, 334  
 pozafunkcjonalne, 33, 36, 76, 510  
 pozyskiwanie, 557  
 problemy, 479  
 produktu, 40  
 projektu, 40  
 prototypowanie, 355  
 przejściowe, 40  
 przykładowe, 237  
 ryzyko, 556  
 specyfikacja, 558  
 sprzeczne, 140  
 statusy, 477

stopień szczegółowości, 430  
 systemowe, 33, 36, 454  
 szacowanie nakładów, 380  
 szacowanie priorytetów, 335  
 szeregowanie rangowe, 334  
 tabela śledzenia, 509  
 testowanie, 355  
 użytecznościowe, 218  
 użytkowników, 35, 46–50, 160, 167, 420  
 usuwanie problemów, 571  
 w projektach analityki biznesowej, 443  
 ulepszających, 409  
 zastępujących, 409  
 zwinnych, 65  
 walidacja, 359, 559  
 wersje, 474  
 wydajnościowe, 219  
 zarządzanie, 472, 483  
 zewnętrznych interfejsów, 160  
 zorientowane na użytkownika, 187  
 wymaganiowa karta obowiązków, 57  
 praw, 55  
 wytrzymałość, 293  
 wywiady, 73, 145  
 wyzwalacze działań, 194  
 wzorce wymagań, 371

## Z

zaangażowanie interesariuszy, 535  
 klienta, 51  
 użytkownika, 45  
 zadania, 490  
 mistrza produktu, 135  
 analityka biznesowego, 87  
 zagrożenia, 219  
 zakres, 489, 593  
 pierwszego wydania, 113  
 ponownego użycia, 368  
 projektu, 72, 102, 112, 121, 213  
 w projektach zwinnych, 122  
 wydania wstępnego, 594

zakresy kolejnych wydań, 113  
 komplementarne, 104  
 zależności, 215  
 zależności biznesowe, 112  
 założenia biznesowe, 112  
 założenie, 215  
 zarządzanie projektem, 69, 80  
 ryzykiem, 551–553, 556, 560  
 wymaganiami, 41–44, 69, 77, 469, 482, 520–524, 546, 559, 587  
 zakresem, 122  
 zmianami, 433, 485, 501, 587  
 zastępowanie systemów, 369, 417  
 zdarzenie biznesowe, 259  
 czasowe, 73, 259  
 sygnałowe, 73, 259  
 zespoły inspekcyjne, 354  
 zewnętrzne interfejsy, 36  
 wymaganie interfejsu, 33  
 zleceniodawca, 431  
 złożenie, 47  
 zmiany zakresu, 121

## Ź

źródła wymagań, 76





# PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW  
w działający bankomat!

**Dowiedz się więcej i dołącz już dzisiaj!**

<http://program-partnerski.helion.pl>

GRUPA WYDAWNICZA

 **Helion SA**

# Lektura obowiązkowa każdego analityka i osób odpowiedzialnych za wymagania!



Zebranie i opracowanie wymagań dotyczących tworzonego oprogramowania to jeden z fundamentów udanego projektu. Znajomość zakresu prac jest kluczową informacją dla wszystkich osób prowadzących projekt oraz bezcennym źródłem wiedzy dla deweloperów tworzących kod. Brzmi prosto, ale wcale tak nie jest! Identyfikacja interesariuszy, dokumentacja wymagań, określanie ich wartości biznesowej — to tylko niektóre z wyzwań stojących przed analitykami i ich zespołami!

Sięgnij po tę książkę, by uniknąć typowych problemów i pułapek. W kolejnych rozdziałach znajdziesz kluczowe informacje na temat wymagań dotyczących oprogramowania, roli analityka biznesowego oraz dobrych praktyk w inżynierii wymagań. Część II tej książki została poświęcona opracowywaniu wymagań. Dowiedz się, jak określać wymagania biznesowe, rozmawiać z użytkownikami oraz dokumentować i walidować wymagania. W prawdziwym świecie spotkasz się z różnymi typami projektów. W zależności od ich charakteru trzeba będzie na bieżąco dostosowywać poznane techniki. Projekty zwinne, projekty systemów wbudowanych, automatyzacja procesów biznesowych to tylko część z omawianych obszarów. Książka ta jest klasycznym podręcznikiem, obowiązkową lekturą każdego analityka oraz osób odpowiedzialnych za wymagania.

## Dzięki tej książce:

- nauczysz się identyfikować interesariuszy oraz rozmawiać z klientami
- poznasz dobre praktyki w inżynierii wymagań
- zrozumiesz zadania analityka biznesowego
- ograniczysz ryzyko dzięki prototypowaniu
- poznasz projekty różnego typu
- zrozumiesz proces zarządzania wymaganiami

**helion.pl**  
księgarnia  
internetowa

Nr katalogowy: 25286



Księgarnia internetowa  
<http://helion.pl>



Zamówienia telefoniczne:  
**0 801 339900**



**0 601 339900**

Microsoft Press



**Helion**

Sprawdź najnowsze promocje:  
• <http://helion.pl/promocje>  
Książki najchętniej czytane:  
• <http://helion.pl/bestsellery>  
Zamów informacje o nowościach:  
• <http://helion.pl/nowosci>

Helion SA  
ul. Kościuszki 1c, 44-100 Gliwice  
tel.: 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
<http://helion.pl>



ISBN 978-83-246-9166-1



9 788324 691661

cena: 99,00 zł

Informatyka w najlepszym wydaniu