

Spis treści

PRZEDMOWA	11
PODZIĘKOWANIA	13
WPROWADZENIE	15
1	
NEGATYWNY WPŁYW ZŁOŻONOŚCI NA PRODUKTYWNOŚĆ	21
Czym jest złożoność?	24
Złożoność cyklu życiowego projektu	25
Planowanie	25
Definiowanie	26
Projektowanie	26
Budowanie	27
Testowanie	27
Wdrażanie	29
Złożoność w teorii oprogramowania i algorytmów	30
Złożoność w nauce	35
Złożoność w procesach	37
Złożoność w życiu codziennym, czyli kara tysiąca cięć	38
Podsumowanie	39
2	
ZASADA 80/20	40
Podstawy zasady 80/20	40
Optymalizacja oprogramowania	41
Produktywność	43
Wskaźniki sukcesu	44
Skupienie i rozkład Pareta	47
Implikacje dla programistów	48
Wskaźniki sukcesu programisty	49
Rozkład Pareta w rzeczywistości	50

Rozkład Pareta jest fraktalny	54
Wskazówki praktyczne dotyczące zasady 80/20	56
Zasoby	58

3

TWORZENIE PRODUKTU O MINIMALNEJ NIEZBĘDNEJ FUNKCJONALNOŚCI60

Problem	61
Utrata motywacji	62
Rozproszenie uwagi	62
Praca na przestrzeni czasu	63
Brak reakcji	63
Błędne założenia	64
Niepotrzebna złożoność	64
Tworzenie produktu o minimalnej niezbędnej funkcjonalności	66
Cztery filary konieczne podczas tworzenia produktu o minimalnej niezbędnej funkcjonalności	69
Zalety produktu o minimalnej niezbędnej funkcjonalności	71
Tryb tajny kontra produkt o minimalnej niezbędnej funkcjonalności	71
Podsumowanie	72

4

TWORZENIE CZYSTEGO I PROSTEGO KODU73

Dlaczego należy tworzyć czysty kod?	73
Tworzenie czystego kodu — zasady	75
1. Spójrz z szerszej perspektywy	76
2. Stań na ramionach olbrzymów	77
3. Twórz kod dla ludzi, nie dla urządzeń	78
4. Używaj odpowiednich nazw	79
5. Zachowaj spójność i zgodność ze standardami	81
6. Używaj komentarzy	82
7. Unikaj niepotrzebnych komentarzy	85
8. Zasada najmniejszego zaskoczenia	86
9. Nie powtarzaj się	87
10. Zasada pojedynczego celu	89
11. Testuj kod	91
12. Małe jest piękne	93
13. Prawo Demeter	94
14. Nie potrzebujesz tego	98
15. Nie używaj zbyt wielu poziomów wcięcia	99
16. Używaj wskaźników	101
17. Zasada harcerza i refaktoryzacja	101
Podsumowanie	102

5		
PRZEDWCZESNA OPTIMALIZACJA JEST ŹRÓDŁEM WSZELKIEGO ZŁA		104
Sześć typów przedwczesnej optymalizacji		104
Optymalizacja funkcji kodu		105
Optymalizacja funkcjonalności		105
Optymalizacja planowania		106
Optymalizacja skalowalności		106
Optymalizacja projektu testów		106
Optymalizacja kodu w podejściu zorientowanym obiektowo		107
Przedwczesna optymalizacja — przykład		107
Sześć podpowiedzi dotyczących poprawy wydajności działania		111
Najpierw pomiar, później usprawnienia		112
Pareto jest królem		112
Korzyści z optymalizacji algorytmicznej		114
Bufor ponad wszystko		115
Mniej znaczy więcej		117
Wiedzieć, kiedy skończyć		118
Podsumowanie		118
6		
PRZEPŁYW		120
Czym jest przepływ?		120
Jak osiągnąć przepływ?		122
Jasno zdefiniowane cele		122
Mechanizm informacji zwrotnych		122
Równowaga między wyzwaniami i umiejętnościami		123
Wskazówki dla programistów		124
Podsumowanie		126
Zasoby		126
7		
„DOBRE WYKONUJ JEDNO ZADANIE” I INNE ZASADY SYSTEMU UNIX		128
Powstanie systemu UNIX		129
Ogólne omówienie filozofii systemu UNIX		129
15 użytecznych zasad systemu UNIX		131
1. Każda funkcja powinna dobrze wykonywać jedno zadanie		131
2. Proste jest lepsze niż złożone		134
3. Małe jest piękne		135
4. Prototyp powinien być tworzony jak najwcześniej		136
5. Ważna jest przenośność, a nie efektywność		137
6. Dane należy przechowywać w jednorodnych plikach tekstowych		139
7. Należy wykorzystywać zalety dźwigni w oprogramowaniu		141
8. Należy unikać wewnętrznych interfejsów użytkownika		142
9. Każdy program powinien mieć postać filtra		145
10. Gorsze jest lepsze		147

11. Czysty kod jest lepszy od sprytniej działającego kodu	148
12. Programy powinny mieć możliwość łączenia się	149
13. Należy zapewnić niezawodność kodu	149
14. Należy naprawiać co się da oraz pozwalać na wczesne i widoczne awarie	150
15. Jeśli można, to należy opracowywać programy przeznaczone do tworzenia programów	152
Podsumowanie	153
Zasoby	153

8

W PROJEKTOWANIU MNIEJ ZNACZY WIĘCEJ	154
Minimalizm w ewolucji telefonów komórkowych	155
Minimalizm w wyszukiwaniu	156
Material Design	157
Jak przygotować projekt minimalistyczny?	158
Używaj pustej przestrzeni	159
Usunięcie elementów projektowych	160
Usuwanie funkcjonalności	161
Ograniczenie wariantów czcionek i liczby kolorów	163
Zachowaj spójność	163
Podsumowanie	164
Zasoby	164

9

SKUPIENIE	165
Broń przeciwko złożoności	165
Zjednoczenie zasad	168
Podsumowanie	170

LIST OD AUTORA	173
-----------------------------	------------