

Wstęp	9
Wprowadzenie	17

Część I. Budowa architektury wspierającej modelowanie domeny

1. Modelowanie domeny	25
Czym jest model domeny	25
Język domeny	28
Testy jednostkowe modeli domeny	28
Klasy danych są idealne dla obiektów wartości	33
Obiekty wartości a jednostki	35
Nie wszystko musi być obiektem — funkcja usługi domeny	37
Magiczne metody Pythona umożliwiają posługiwanie się modelami w standardowy sposób	38
Wyjątki także mogą wyrażać koncepcje domeny	38
2. Wzorzec Repozytorium	41
Zapisywanie modelu domeny	42
Trochę pseudokodu — czego będziemy potrzebować?	42
Zastosowanie zasady odwrócenia zależności do dostępu do danych	43
Przypomnienie — nasz model	43
„Normalny” sposób — model zależy od ORM	44
Odwrócenie zależności — ORM zależy od modelu	45
Wprowadzenie do wzorca Repozytorium	48
Abstrakcja repozytorium	49
Gdzie tkwi haczyk	50

Budowa imitacji repozytorium na potrzeby testów nie jest łatwa	53
Czym są porty i adaptery w Pythonie	54
Podsumowanie	54
3. Interludium na temat powiązań i abstrakcji	57
Abstrakcja stanu wspomaga testowanie	58
Wybór właściwych abstrakcji	61
Implementacja wybranych abstrakcji	62
Testowanie od brzegu do brzegu z imitacjami i wstrzykiwaniem zależności	64
Czemu by nie użyć biblioteki łańcuchów?	65
Podsumowanie	68
4. Pierwszy przypadek użycia — API Flask i warstwa usług	69
Łączenie naszej aplikacji z prawdziwym światem	71
Pierwszy test kompleksowy	71
Prosta implementacja	72
Błędy wymagające sprawdzenia bazy danych	73
Wprowadzenie warstwy usług i testowanie jej za pomocą FakeRepository	74
Typowa funkcja usługowa	76
Dlaczego wszystko nazywa się usługą	78
Rozmieszczenie plików w folderach, aby uzyskać przejrzysty obraz struktury	79
Podsumowanie	80
Zasada odwrócenia zależności w praktyce	80
5. TDD na wysokich i niskich obrotach	83
Jak wygląda nasza piramida testów?	83
Czy przenieść testy warstwy domeny do warstwy usługowej?	84
Wybór rodzaju testów do napisania	85
Wysokie i niskie obroty	86
Całkowite oddzielenie testów warstwy usługowej od domeny	86
Rozwiązanie — przeniesienie wszystkich zależności domeny do konfiguracji testów	87
Dodawanie brakującej usługi	87
Ulepszanie testów kompleksowych	89
Podsumowanie	90
6. Wzorzec Jednostka Pracy	91
Jednostka pracy współpracuje z repozytorium	91
Testy integracyjne jednostki pracy	93

Jednostka pracy i jej menedżer kontekstu	94
Prawdziwa jednostka pracy używa sesji SQLAlchemy	95
Imitacja jednostki pracy do testów	96
Używanie jednostki pracy w warstwie usługowej	97
Testy zatwierdzania i wycofywania zmian	98
Zatwierdzenia jawne i niejawne	98
Przykłady — użycie jednostki pracy do grupowania operacji w jednostkę atomową	99
Przykład 1. Realokacja	99
Przykład 2. Zmiana liczebności partii	100
Porządkowanie testów integracyjnych	100
Podsumowanie	101
7. Agregaty i granice spójności	103
Czemu nie wykonać wszystkiego w arkuszu kalkulacyjnym?	104
Niezmienniki, ograniczenia i spójność	104
Niezmienniki, współbieżność i blokady	104
Czym jest agregat	105
Wybór agregatu	106
Jeden agregat = jedno repozytorium	109
Kwestia wydajności	110
Optymistyczna współbieżność a numery wersji	111
Opcje implementacji numerów wersji	113
Sprawdzanie zgodności z regułami integralności danych	114
Wymuszanie przestrzegania zasad dotyczących współbieżności za pomocą poziomów izolacji bazy danych	115
Przykład pesymistycznej kontroli współbieżności — SELECT FOR UPDATE	116
Podsumowanie	117
Część I — podsumowanie	118

Część II. Architektura sterowana zdarzeniami

8. Zdarzenia i szyna wiadomości	123
Jak nie narobić bałaganu	124
Pilnujemy porządku w kontrolerach sieciowych	124
Dbajmy też o porządek w modelu	125
To może warstwa usługowa	125
Zasada pojedynczej odpowiedzialności	126

Wielkie wejście szyny wiadomości	127
Model rejestruje zdarzenia	127
Zdarzenia to proste klasy danych	127
Model zgłasza zdarzenia	127
Szyna wiadomości wiąże zdarzenia z procedurami obsługi	128
Opcja 1. Warstwa usługowa odbiera zdarzenia z modelu i umieszcza je w szynie wiadomości	129
Opcja 2. Warstwa usług sama zgłasza zdarzenia	130
Opcja 3. Jednostka pracy publikuje zdarzenia w szynie wiadomości	131
Podsumowanie	134
9. Szyna wiadomości w pełnej krasie	137
Nowy wymóg prowadzi do opracowania nowej architektury	138
Zmiana architektury — wszystko będzie procedurą obsługi zdarzeń	139
Zamiana funkcji usługowych na procedury obsługi wiadomości	140
Teraz szyna wiadomości odbiera zdarzenia od jednostki pracy	142
Wszystkie testy także napiszemy pod kątem zdarzeń	143
Tymczasowa brzydka sztuczka — szyna wiadomości musi zwracać wyniki	144
Modyfikacja API, aby działał ze zdarzeniami	144
Implementacja nowego wymogu	145
Nasze nowe zdarzenie	146
Próba nowej procedury obsługi	146
Implementacja	147
Nowa metoda modelu domeny	148
Opcja — testy jednostkowe procedur obsługi zdarzeń w izolacji przy użyciu imitacji szyny wiadomości	149
Podsumowanie	151
Co osiągnęliśmy	151
Po co to wszystko	151
10. Polecenia i procedury obsługi poleceń	153
Polecenia i zdarzenia	153
Różnice w zakresie obsługi wyjątków	154
Zdarzenia, polecenia i obsługa błędów	156
Synchroniczne wychodzenie z błędów	159
Podsumowanie	160

11. Architektura oparta na zdarzeniach — integracja mikrousług za pomocą zdarzeń	163
Rozproszona kula błota i myślenie rzeczownikami	163
Obsługa błędów w systemach rozproszonych	166
Alternatywa — rozprężenie pod względem czasowym przy użyciu wiadomości asynchronicznych	167
Użycie kanału publikacji-subskrypcji Redis do integracji	168
Test kompleksowy, który to wszystko sprawdzi	169
Redis to kolejny cienki adapter wokół naszej szyny wiadomości	170
Nowe zdarzenie wyjściowe	171
Zdarzenia wewnętrzne i zewnętrzne	172
Podsumowanie	172
 12. Wzorzec podziału odpowiedzialności między polecenia i zapytania (CQRS)	175
Modele domeny służą do zapisu	175
Większość klientów nie kupi waszych mebli	177
Post-Redirect-Get i CQS	178
Trzymajcie się mocno	180
Testowanie widoków CQRS	180
Opcja „oczywista” — użycie istniejącego repozytorium	181
Twój model domeny nie jest zoptymalizowany pod kątem operacji odczytu	182
Oczywista opcja 2 — użycie ORM	182
SELECT N+1 i inne sprawy związane z wydajnością	183
Czas całkiem obniżyć loty	183
Aktualizacja tabeli modelu odczytu za pomocą procedury obsługi zdarzeń	184
Zmiana implementacji modelu odczytu jest łatwa	186
Podsumowanie	187
 13. Wstrzykiwanie zależności (i bootstrapping)	189
Zależności jawne i niejawne	189
Czy jawne zależności nie są dziwne i nie pachną Javą?	192
Przygotowywanie procedur obsługi — ręczne wstrzykiwanie zależności przy użyciu domknięć i funkcji częściowych	194
Alternatywa z użyciem klas	195
Skrypt rozruchowy	195
Przekazywanie procedur obsługowych do szyny wiadomości w czasie działania programu	198
Użycie funkcji rozruchowej w punktach wejścia	199
Inicjalizacja DI w testach	200

Prawidłowe tworzenie adaptera — działający przykład	201
Definicja implementacji abstrakcyjnej i konkretnej	201
Utworzenie fałszywej wersji dla testów	202
Prawdziwy test integracyjny	203
Podsumowanie	204
Epilog	207
A Podsumowanie — schemat i tabela	223
B Szablon struktury projektu	225
C Wymiana infrastruktury — wszystko za pomocą CSV	233
D Repozytorium i Jednostka Pracy w Django	239
E Walidacja	247