

Wprowadzenie

- Dla kogo jest ta książka? (20)
- Wiemy, co sobie myślisz (21)
- Metapoznanie: myślenie o myśleniu (23)
- Zmusz swój mózg do posłuszeństwa (25)
- Ważne uwagi (26)
- Recenzenci techniczni (28)
- Podziękowania (29)

Rozdział 1. Tu zaczyna się wspaniałe oprogramowanie

- Rock-and-roll jest wieczny! (31)
- Nowa elegancka aplikacja Ryśka... (33)
- Co przede wszystkim zmieniłbyś w aplikacji Ryśka? (38)
- Doskonałe oprogramowanie... ma więcej niż jedną z wymienionych już cech (40)
- Wspaniałe oprogramowanie w trzech prostych krokach (43)
- W pierwszej kolejności skoncentruj się na funkcjonalności (48)
- Test (53)
- Szukamy problemów (55)
- Analiza metody search() (56)
- Stosuj proste zasady projektowania obiektowego (61)
- Projekt po raz pierwszy, projekt po raz drugi (66)
- Jak łatwo można wprowadzać zmiany w Twojej aplikacji? (68)
- Poddawaj hermetyzacji to, co się zmienia (71)
- Delegowanie (73)
- Nareszcie doskonałe oprogramowanie (jak na razie) (76)
- OOA&D ma na celu tworzenie wspaniałego oprogramowania, a nie dodanie Ci papierkowej roboty (79)
- Kluczowe zagadnienia (80)

Rozdział 2. Daj im to, czego chcą

- Nadszedł czas na kolejny pokaz Twych programistycznych umiejętności (84)
- Test programu (87)
- Nieprawidłowe zastosowanie (coś w tym stylu) (89)
- Czym jest wymaganie? (90)
- Tworzenie listy wymagań (92)
- Zaplanuj, co może się popsuć w systemie (96)
- Problemy w działaniu systemu są obsługiwane przez ścieżki alternatywne (98)
- (Ponowne) przedstawienie przypadku użycia (100)
- Jeden przypadek użycia, trzy części (102)
- Porównaj wymagania z przypadkami użycia (106)
- Twój system musi działać w praktyce (113)
- Poznajemy Szczęśliwą Ścieżkę (120)
- Przybornik projektanta (134)

Rozdział 3. Kocham cię, jesteś doskonały... A teraz - zmień się

- Jesteś bohaterem! (138)

- Jesteś patałachem! (139)
- Jedyne pewniki analizy i projektowania obiektowego (141)
- Ścieżka oryginalna? Ścieżka alternatywna? Kto to wie? (146)
- Przypadki użycia muszą być zrozumiałe przede wszystkim dla Ciebie (148)
- Od startu do mety: jeden scenariusz (150)
- Wyznanie Ścieżki Alternatywnej (152)
- Uzupełnienie listy wymagań (156)
- Powielanie kodu jest bardzo złym pomysłem (164)
- Ostateczny test drzwiczek (166)
- Napisz swoją własną zasadę projektową! (167)
- Przybornik projektanta (168)

Rozdział 4. Zaczynamy używać naszych aplikacji w rzeczywistym świecie

- Jeden pies, dwa psy, trzy psy, cztery... (170)
- Twoje oprogramowanie ma kontekst (171)
- Określ przyczynę problemu (172)
- Zaplanuj rozwiązanie (173)
- Opowieść o dwóch programistach (180)
- Delegowanie w kodzie Szymka - analiza szczegółowa (184)
- Potęga aplikacji, których elementy są ze sobą luźno powiązane (186)
- Zwracaj uwagę na rzeczowniki występujące w przypadku użycia (191)
- Od dobrej analizy do dobrych klas... (204)
- Diagramy klas bez tajemnic (206)
- Diagramy klas to nie wszystko (211)
- Kluczowe zagadnienia (215)

Rozdział 5. (część 1.) Nic nie pozostaje wiecznie takie samo

- Firma Gitary/Instrumenty Strunowe Ryśka rozwija się (222)
- Klasy abstrakcyjne (225)
- Diagramy klas bez tajemnic (ponownie) (230)
- Ściągawka z UML-a (231)
- Porady dotyczące problemów projektowych (237)
- Trzy kroki tworzenia wspaniałego oprogramowania (po raz kolejny) (239)

Rozdział 5. (część 2.) Zabierz swoje oprogramowanie na 30-minutowy trening

- Wróćmy do aplikacji wyszukiwawczej Ryśka (258)
- Dokładniejsza analiza metody search() (261)
- Korzyści, jakie dała nam analiza (262)
- Dokładniejsza analiza klas instrumentów (265)
- Śmierć projektu (decyzja) (270)
- Zmieńmy złe decyzje projektowe na dobre (271)
- Zastosowanie "podwójnej hermetyzacji" w aplikacji Ryśka (273)
- Nigdy nie obawiaj się wprowadzania zmian (279)
- Elastyczna aplikacja Ryśka (282)
- Testowanie dobrze zaprojektowanej aplikacji Ryśka (285)
- Jak łatwo można zmodyfikować aplikację Ryśka? (289)
- Wielki konkurs łatwości modyfikacji (290)

- Spójna klasa realizuje jedną operację naprawdę dobrze (293)
- Przegląd zmian wprowadzanych w oprogramowaniu dla Ryśka (296)
- Doskonałe oprogramowanie to zazwyczaj takie, które jest "wystarczająco dobre" (298)
- Przybornik projektanta (300)

Rozdział 6. "Nazywam się Art Vandelay... jestem Architektem"

- Rozwiązywanie dużych problemów (302)
- Wszystko zależy od sposobu spojrzenia na duży problem (303)
- Wymagania i przypadki użycia to dobry punkt wyjściowy... (308)
- Potrzebujemy znacznie więcej informacji (309)
- Określanie możliwości (312)
- Możliwość czy wymaganie (314)
- Przypadki użycia nie zawsze pomagają ujrzeć ogólny obraz tworzonego oprogramowania (316)
- Diagramy przypadków użycia (318)
- Mały aktor (323)
- Aktorzy to także ludzie (no dobrze... nie zawsze) (324)
- A zatem zabawmy się w analizę dziedziny! (329)
- Dziel i rządź (331)
- Nie zapominaj, kim tak naprawdę jest klient (335)
- Czym jest wzorzec projektowy? (337)
- Potęga OOA&D (i trochę zdrowego rozsądku) (340)
- Przybornik projektanta (342)

Rozdział 7. Porządkowanie chaosu

- Czy czujesz się nieco przytłoczony? (344)
- Potrzebujemy architektury (346)
- Zaczniemy od funkcjonalności (349)
- Co ma znaczenie dla architektury (351)
- Trzy P dotyczące architektury (352)
- Wszystko sprowadza się do problemu ryzyka (358)
- Scenariusze pomagają zredukować ryzyko (361)
- Koncentruj się na jednej możliwości w danej chwili (369)
- Architektura jest strukturą Twojego projektu (371)
- Podobieństwa po raz kolejny (375)
- Analiza podobieństw: ścieżka do elastycznego oprogramowania (381)
- Co to znaczy? Zapytaj klienta (386)
- Zmniejszanie ryzyka pomaga pisać wspaniałe oprogramowanie (391)
- Kluczowe zagadnienia (392)

Rozdział 8. Oryginalność jest przereklamowana

- Zasada projektowania - w skrócie (396)
- Zasada otwarte-zamknięte (397)
- OCP, krok po kroku (399)
- Zasada nie powtarzaj się (402)
- Zasada DRY dotyczy obsługi jednego wymagania w jednym miejscu (404)

- Zasada jednej odpowiedzialności (410)
- Wykrywanie wielu odpowiedzialności (412)
- Przechodzenie od wielu do jednej odpowiedzialności (415)
- Zasada podstawienia Liskov (420)
- Studium błędnego sposobu korzystania z dziedziczenia (421)
- LSP ujawnia ukryte problemy związane ze strukturą dziedziczenia (422)
- Musi istnieć możliwość zastąpienia typu bazowego jego typem pochodnym (423)
- Naruszenia LSP sprawiają, że powstający kod staje się mylący (424)
- Deleguj funkcjonalność do innej klasy (426)
- Użyj kompozycji, by zebrać niezbędne zachowania z kilku innych klas (428)
- Agregacja - kompozycja bez nagłego zakończenia (432)
- Agregacja a kompozycja (433)
- Dziedziczenie jest jedynie jedną z możliwości (545)
- Kluczowe zagadnienia (437)
- Przybornik projektanta (438)

Rozdział 9. Oprogramowanie jest wciąż przeznaczone dla klienta

- Twój przybornik narzędziowy powoli się wypełnia (442)
- Wspaniałe oprogramowanie tworzy się iteracyjnie (444)
- Schodzenie w głąb: dwie proste opcje (445)
- Programowanie w oparciu o możliwości (446)
- Programowanie w oparciu o przypadki użycia (447)
- Dwa podejścia do tworzenia oprogramowania (448)
- Analiza możliwości (452)
- Pisanie scenariuszy testowych (455)
- Programowanie w oparciu o testy (458)
- Podobieństwa po raz wtóry (460)
- Kładziemy nacisk na podobieństwa (464)
- Hermetyzujemy wszystko (466)
- Dopasuj testy do projektu (470)
- Testy bez tajemnic... (472)
- Udowodnij klientowi, że wszystko idzie dobrze (478)
- Jak dotąd używaliśmy programowania w oparciu o kontrakt (480)
- Tak naprawdę programowanie w oparciu o kontrakt dotyczy zaufania (481)
- Programowanie defensywne (482)
- Podziel swoją aplikację na mniejsze fragmenty funkcjonalności (491)
- Kluczowe zagadnienia (493)
- Przybornik projektanta (496)

Rozdział 10. Scalając to wszystko w jedno

- Tworzenie oprogramowania, w stylu obiektowym (500)
- Trans-Obiektów (504)
- Mapa metra w Obiekcie (506)
- Lista możliwości (509)
- Przypadki użycia odpowiadają zastosowaniu, możliwości odpowiadają funkcjonalności (515)
- A teraz zacznij powtarzać te same czynności (519)
- Dokładniejsza analiza sposobu reprezentacji sieci metra (521)

- Używać klasy Line czy też nie używać... oto jest pytanie (530)
- Najważniejsze sprawy związane z klasą Subway (536)
- Ochrona własnych klas (539)
- Czas na przerwę (547)
- Wróćmy znowu do etapu określania wymagań (549)
- Koncentruj się na kodzie, a potem na klientach (551)
- Powtarzanie sprawia, że problemy stają się łatwiejsze (555)
- Jak wygląda trasa? (560)
- Samemu sprawdź Przewodnik Komunikacyjny po Obiekcie (564)
- Ktoś chętny na trzeci cykl prac? (567)
- Podróż jeszcze nie dobiegła końca... (569)

Dodatek A Dziesięć najważniejszych tematów (których nie poruszyliśmy)

- Nr 1. JEST i MA (572)
- Nr 2. Sposoby zapisu przypadków użycia (574)
- Nr 3. Antywzorce (577)
- Nr 4. Karty CRC (578)
- Nr 5. Metryki (580)
- Nr 6. Diagramy sekwencji (581)
- Nr 7. Diagramy stanu (582)
- Nr 8. Testowania jednostkowe (584)
- Nr 9. Standardy kodowania i czytelny kod (586)
- Nr 10. Refaktoryzacja (588)

Dodatek B Stosowanie języka obiektowego

- UML i diagramy klas (591)
- Dziedziczenie (593)
- Polimorfizm (595)
- Hermetyzacja (596)
- Kluczowe zagadnienia (600)

Skorowidz (603)