**Przedmowa**

**Część I. Podstawy**

**Rozdział 1. Czym są mikrousługi?**

- Mikrousługi w skrócie
- Kluczowe pojęcia dotyczące mikrousług
  - Możliwość niezależnego wdrażania
  - Zamodelowane wokół domeny biznesowej
  - Posiadanie własnego stanu
  - Rozmiar
  - Elastyczność
  - Dopasowanie architektury do organizacji zespołów
- Monolit
  - Monolit jednoprocesowy
  - Monolit modułowy
  - Monolit rozproszony
  - Monolity i rywalizacja o dostawy
  - Zalety monolitów
- Technologie pomocnicze
  - Agregacja logów i rozproszone śledzenie
  - Kontenery i Kubernetes
  - Przesyłanie strumieniowe
  - Chmura publiczna i platformy bezserwerowe
- Najważniejsze korzyści
  - Niejednorodność technologii
  - Odporność na błędy
  - Skalowanie
  - Łatwość wdrażania
  - Dopasowanie do organizacji zespołów
  - Komponowalność
- Niedogodności związane z architekturą mikrousług
  - Wrażenia programisty
  - Przeciążenie technologią
  - Koszty
  - Raportowanie
  - Monitorowanie i rozwiązywanie problemów
  - Bezpieczeństwo
  - Testowanie
  - Opóźnienia
  - Spójność danych
- Czy powinienem korzystać z mikrousług?
  - Kiedy mikrousługi mogą się nie sprawdzić?
  - Gdzie mikrousługi działają dobrze?
- Podsumowanie

**Rozdział 2. Jak modelować mikrousługi?**

- Przedstawiamy firmę MusicCorp

- Co decyduje o tym, że granice mikrousługi są dobre?
  - Ukrywanie informacji
  - Spójność
  - Sprzężenia
  - Wzajemne oddziaływanie pomiędzy sprzężeniami a spójnością
- Rodzaje sprzężeń
  - Sprzężenie domen
  - Sprzężenia przelotowe
  - Sprzężenie wspólnych danych
  - Sprzężenia treści
- Wprowadzenie do metodologii projektowania opartego na domenie (DDD)
  - Język wszechobecny
  - Agregat
  - Kontekst ograniczony
  - Mapowanie agregatów i kontekstów ograniczonych na mikrousługi
  - Event Storming
- Projektowanie DDD w kontekście mikrousług
- Alternatywy dla granic domen biznesowych
  - Ulotność
  - Dane
  - Technologia
  - Względy organizacyjne
- Modele mieszane i wyjątki
- Podsumowanie

**Rozdział 3. Dzielenie monolitu**

- Określenie celu
- Migracja przyrostowa
- Monolit rzadko jest Twoim wrogiem
  - Niebezpieczeństwa przedwczesnej dekompozycji
- Co podzielić najpierw?
- Dekompozycja według warstwy
  - Najpierw kod
  - Najpierw dane
- Przydatne wzorce dekompozycji
  - Wzorzec figowca-dusiciela
  - Uruchamianie równoległe
  - Przełącznik funkcji
- Problemy z dekompozycją danych
  - Wydajność
  - Integralność danych
  - Transakcje
  - Narzędzia
  - Bazy danych raportowania
- Podsumowanie

**Rozdział 4. Rodzaje komunikacji mikrousług**

- Od komunikacji wewnątrz procesu do komunikacji między procesami

        - Wydajność
        - Modyfikacje interfejsów
        - Obsługa błędów
    - Technologia komunikacji między procesami: wiele możliwości do wyboru
    - Style komunikacji mikrousług
        - Łącz i dopasowuj
    - Wzorzec komunikacja synchroniczna blokująca
        - Zalety
        - Wady
        - Gdzie stosować wzorzec?
    - Wzorzec komunikacja asynchroniczna nieblokująca
        - Zalety
        - Wady
        - Gdzie stosować wzorzec?
    - Wzorzec komunikacja za pośrednictwem współdzielonych danych
        - Implementacja
        - Zalety
        - Wady
        - Gdzie stosować wzorzec?
    - Wzorzec komunikacja żądanie - odpowiedź
        - Implementacja: komunikacja synchroniczna kontra asynchroniczna
        - Gdzie stosować wzorzec?
    - Wzorzec komunikacja sterowana zdarzeniami
        - Implementacja
        - Co jest wewnątrz zdarzenia?
        - Gdzie stosować wzorzec?
    - Zachowaj ostrożność
    - Podsumowanie

**Część II. Implementacja**

**Rozdział 5. Implementacja komunikacji mikrousług**

- Poszukiwanie idealnej technologii
    - Łatwość zachowania zgodności wstecz
    - Zdefiniuj interfejs w sposób jawny
    - Zachowaj niezależność technologii interfejsów API
    - Spraw, aby Twoja usługa była prosta dla konsumentów
    - Ukryj szczegóły wewnętrznej implementacji
- Wybór technologii
    - Zdalne wywołania procedur
    - REST
    - GraphQL
    - Brokery wiadomości
- Formaty serializacji
    - Formaty tekstowe
    - Formaty binarne
- Schematy
    - Strukturalne i semantyczne naruszenia kontraktu
    - Czy należy używać schematów?

- Obsługa zmian między mikrousługami
- Unikanie zmian naruszających kontrakt
  - Zmiany rozszerzające
  - Tolerancyjny konsument
  - Właściwa technologia
  - Jawny interfejs
  - Wczesne wykrywanie zmian naruszających kontrakt
- Zarządzanie zmianami naruszającymi zgodność wstecz
  - Wdrażanie lockstep
  - Współistnienie niezgodnych ze sobą wersji mikrousług
  - Emulowanie starego interfejsu
  - Jakie podejście preferuję?
  - Umowa społeczna
  - Śledzenie użycia
  - Środki ekstremalne
- Zasada DRY i niebezpieczeństwa wielokrotnego wykorzystywania kodu w świecie mikrousług
  - Udostępnianie kodu za pośrednictwem bibliotek
- Wykrywanie usług
  - DNS
  - Dynamiczne rejestry usług
  - Nie zapomnij o ludziach
- Siatki usług i bramy interfejsów API
  - Bramy API
  - Siatki usług
  - A co z innymi protokołami?
- Dokumentowanie usług
  - Jawne schematy
  - System samoopisujący się
- Podsumowanie

**Rozdział 6. Przepływy pracy**

- Transakcje bazodanowe
  - Transakcje ACID
  - Nadal ACID, ale bez niepodzielności?
- Transakcje rozproszone - dwufazowe zatwierdzanie
- Transakcje rozproszone - po prostu powiedz "nie"
- Sagi
  - Tryby awarii dla sag
  - Implementacja sag
  - Sagi a transakcje rozproszone
- Podsumowanie

**Rozdział 7. Budowanie**

- Krótkie wprowadzenie do ciągłej integracji
  - Czy rzeczywiście stosujesz mechanizmy CI?
  - Modele rozgałęziania
- Potoki budowania a ciągłe dostawy

        - Narzędzia
        - Kompromisy i środowiska
        - Tworzenie artefaktów
    - Mapowanie kodu źródłowego i kompilacji na mikrousługi
        - Jedno gigantyczne repozytorium, jedna gigantyczna kompilacja
        - Wzorzec jedno repozytorium na mikrousługę (tzw. multirepo)
        - Wzorzec monorepo
        - Jakie podejście bym zastosował?
    - Podsumowanie

**Rozdział 8. Wdrażanie**

- Od widoku logicznego do fizycznego
    - Wiele egzemplarzy
    - Baza danych
    - Środowiska
- Zasady wdrażania mikrousług
    - Odizolowane uruchamianie
    - Koncentracja na automatyzacji
    - Infrastruktura jako kod (IaC)
    - Wdrażanie bez przestojów
    - Zarządzanie pożądanym stanem
- Opcje wdrażania
    - Maszyny fizyczne
    - Maszyny wirtualne
    - Kontenery
    - Kontenery aplikacji
    - Platforma jako usługa (PaaS)
    - Funkcja jako usługa (FaaS)
- Która opcja wdrażania jest dla Ciebie odpowiednia?
- Kubernetes i orkiestracja kontenerów
    - Przypadek orkiestracji kontenerów
    - Uproszczony widok pojęć związanych z Kubernetes
    - Wielodostępność i federacja
    - Cloud Native Computing Federation (CNCF)
    - Platformy i przenośność
    - Helm, Operator, CRD. O mój Boże!
    - I jeszcze Knative
    - Przyszłość
    - Czy powinieneś korzystać z Kubernetes?
- Dostawy progresywne
    - Oddzielenie wdrożenia od wydania
    - Na drodze do dostaw progresywnych
    - Przełączniki funkcji
    - Wydania kanarkowe
    - Uruchamianie równoległe
- Podsumowanie

**Rozdział 9. Testowanie**

- Rodzaje testów
- Zakres testów
  - Testy jednostkowe
  - Testy usług
  - Testy od końca do końca
  - Kompromisy
- Implementacja testów usług
  - Mocki czy namiastki usług
  - Inteligentniejsza namiastka usługi
- Kłopotliwe testy od końca do końca
  - Testy kruche i łamliwe
  - Kto pisze testy od końca do końca?
  - Jak długo?
  - Piętrzące się zaległości
  - Metawersje
  - Brak niezależnej testowalności
- Czy należy unikać testów od końca do końca?
  - Testy kontraktu oraz kontrakty konsumenckie
  - Czy należy używać testów od końca do końca?
- Wygoda pracy programistów
- Od fazy przedprodukcyjnej do testowania w produkcji
  - Rodzaje testów w produkcji
  - Bezpieczeństwo testowania w produkcji
  - Średni czas do naprawy kontra średni czas między awariami
- Testy współzależności funkcjonalnych
  - Testy wydajności
  - Testy wytrzymałości
- Podsumowanie

**Rozdział 10. Od monitorowania do obserwowalności**

- Niepokój, panika i zamieszanie
- Jedna usługa, jeden serwer
- Jedna usługa, wiele serwerów
- Wiele usług, wiele serwerów
- Obserwowalność a monitorowanie
  - Filary obserwowalności? Nie tak szybko
- Elementy składowe obserwowalności
  - Agregacja logów
  - Agregacja metryk
  - Rozproszone śledzenie
  - Czy postępujemy właściwie?
  - Ostrzeganie
  - Monitorowanie semantyczne
  - Testowanie w produkcji
- Standaryzacja
- Wybór narzędzi
  - Wybór powinien być demokratyczny
  - Wybieraj narzędzia łatwe do integracji
  - Zapewniaj odpowiedni kontekst

        - Informacje w czasie rzeczywistym
        - Informacje odpowiednie dla Twojej skali
    - Maszynowy ekspert
    - Od czego zacząć?
    - Podsumowanie

**Rozdział 11. Bezpieczeństwo**

- Podstawowe zasady
    - Zasada najmniejszych uprawnień
    - Obrona w głąb
    - Automatyzacja
    - Wbuduj zabezpieczenia w proces dostaw
- Pięć funkcji cyberbezpieczeństwa
    - Identyfikacja
    - Ochrona
    - Wykrywanie
    - Reagowanie
    - Odtwarzanie
- Podstawy zabezpieczeń aplikacji
    - Poświadczenia
    - Łatki bezpieczeństwa
    - Kopie zapasowe
    - Odbudowa
- Zaufanie domyślne kontra zaufanie zerowe
    - Zaufanie domyślne
    - Zaufanie zerowe
    - To jest pasmo
- Zabezpieczanie danych
    - Dane podczas przesyłania
    - Zabezpieczanie danych w spoczynku
- Uwierzytelnianie i autoryzacja
    - Uwierzytelnianie między usługami
    - Uwierzytelnianie użytkowników
    - Popularne implementacje pojedynczego logowania
    - Brama pojedynczego logowania
    - Szczegółowa autoryzacja
    - Problem zdezorientowanego zastępcy
    - Scentralizowana autoryzacja w górze strumienia przetwarzania
    - Autoryzacja zdecentralizowana
    - Tokeny JWT
- Podsumowanie

**Rozdział 12. Niezawodność**

- Co to jest niezawodność?
    - Solidność
    - Zdolność do odtwarzania
    - Rozszerzalność z wdziękiem
    - Trwałe zdolności adaptacyjne

        - Architektura mikrousług
    - Awarie zdarzają się wszędzie
    - Jak wiele to zbyt wiele?
    - Degradowanie funkcjonalności
    - Wzorce stabilności
        - Limity czasu
        - Ponowienia prób
        - Grodzie
        - Bezpieczniki
        - Izolacja
        - Redundancja
        - Middleware
        - Idempotencja
    - Rozłożenie ryzyka
    - Twierdzenie CAP
        - Poświęcenie spójności
        - Poświęcenie dostępności
        - Poświęcenie tolerancji podziału?
        - AP czy CP?
        - To nie jest zasada "wszystko albo nic"
        - Świat rzeczywisty
        - Antykrucha organizacja
    - Inżynieria chaosu
        - Dni ćwiczeń
        - Eksperymenty produkcyjne
        - Wykraczając poza solidność
    - Szukanie winnych
    - Podsumowanie

**Rozdział 13. Skalowanie**

- Cztery osie skalowania
    - Skalowanie pionowe
    - Implementacja
    - Najważniejsze korzyści
    - Ograniczenia
    - Zwielokrotnianie w poziomie
    - Partycjonowanie danych
    - Dekompozycja funkcjonalna
- Łączenie modeli
- Zacznij od małych rozmiarów
- Buforowanie
    - Buforowanie w celu poprawy wydajności
    - Buforowanie w celu skalowania
    - Buforowanie w celu poprawy niezawodności
    - Gdzie buforować
    - Unieważnianie
    - Złota zasada buforowania
    - Aktualność danych a optymalizacja
    - Zatrucie pamięcią podręczną - historia ku przestrodze

        - Własność silna
        - Własność kolektywna
        - Na poziomie zespołu kontra na poziomie organizacji
        - Równoważenie modeli
    - Zespoły wspomagające
        - Społeczności praktyków
        - Platforma
    - Mikrousługi współdzielone
        - Zbyt trudne do rozdzielenia
        - Przekrojowe zmiany
        - Wąskie gardła dostaw
    - Wewnętrzne open source
        - Rola opiekunów
        - Dojrzałość
        - Narzędzia
    - Mikrousługi modułowe
        - Przeglądy zmian
    - Usługa osierocona
    - Studium przypadku: RealEstate.com.au
    - Rozproszenie geograficzne
    - Odwrócone prawo Conwaya
    - Ludzie
    - Podsumowanie

**Rozdział 16. Ewolucyjny architekt**

- Co oznacza ta nazwa?
- Czym jest architektura oprogramowania?
- Umożliwienie wprowadzania zmian
- Ewolucyjna wizja architekta
- Definiowanie granic systemowych
- Konstrukt społeczny
- Warunki do "zamieszkiwania"
- Pryncypialne podejście
    - Cele strategiczne
    - Zasady
    - Praktyki
    - Łączenie zasad i praktyk
    - Praktyczny przykład
- Kierowanie architekturą ewolucyjną
- Architektura w organizacji dostosowanej do strumienia przetwarzania
- Budowanie zespołu
- Wymagane standardy
    - Monitorowanie
    - Interfejsy
    - Bezpieczeństwo architektury
- Zarządzanie i droga utwardzona
    - Przykładowe egzemplarze
    - Spersonalizowany szablon usługi
    - Utwardzona droga na dużą skalę