BO·TT·EGA
IT minds

Program szkolenia:

Domain Driven Design - modelowanie strategiczne i taktyczne

Informacje:

Nazwa:	**Domain Driven Design - modelowanie strategiczne i taktyczne**
Kod:	**ddd-workshop-DDD4**
Kategoria:	Warsztaty eksperckie DDD
Odbiorcy:	analitycy, architekci, developerzy
Czas trwania:	2+2 dni
Forma:	60% wykłady / 40% warsztaty

Szkolenie składa się z 2 sesji, rozdzielonych kilkudniową przerwą, która da Ci czas na refleksje i sformułowanie pytań. W trakcie szkolenia będziemy projektować iteracyjnie system na podstawie wymagań, które *delikatnie* zmieniają się z czasem.

Sesja I - modelowanie strategiczne (2 dni)

- Event Storming - Process Level w celu uchwycenia sedna problemu
- Umiejętności miękkie oparte o gramatykę generatywną Naoma Chomskyego: facylitacja stormingu, techniki zadawania pytań
- Heurystyki odkrywania pod-domen biznesowych
- Destylacja kontekstów w celu znalezienia niezależnych modeli odpowiednich do problemów
- Mapowanie kontekstów - świadome podejmowanie decyzji o zależnościach pomiędzy zespołami
- Archetypy modeli biznesowych jako sprawdzone rozwiązanie generycznych problemów
- Architektura komponentowa - autonomiczne moduły oparte o Single Source of Truth
- Topologie zespołów odpowiednie do dojrzałości domen biznesowych

Sesja II - modelowanie taktyczne (2 dni)

- Event Storming - Tactical Design w celu uchwycenia reguł spójności
- Building Blocks - elementy konstrukcyjne dla modeli
- Style architektoniczne odpowiednia dla klas złożoności problemów
- Zmiany granic kontekstów
- Strategia testowania
- API modułów: proceduralne, restful
- Problemy implementacyjne: ORM, propagacja zdarzeń

Formuła warsztatu to kolejne omawianie technik i ich demonstracja przez trenera, po czym uczestnicy w podgrupach samodzielnie przechodzą tą samą ścieżkę na innych wymaganiach. W zadaniach są ukryte pułapki odzwierciedlające realne problemy projektowe, które omawiamy wspólnie po każdym etapie.

Zalety szkolenia:

- Posiądziesz podstawowe kompetencje miękkie jakie powinien posiadać Modelarz
- Dowiesz się jak dzielić system na niezależne moduły dzięki technice destylacji Bounded Contextów
- Uświadomisz sobie decyzje architektoniczne, jakie musisz podjąć na każdym poziomie z C4 oraz poznasz sposoby zdobywania odpowiedzi na te pytania
- Poznasz różne podejścia do modelowania: obiektowe, funkcyjne, oparte o archetypy

BOTTEGA IT minds

Szczegółowy program:

1. Zadanie wstępne - praca nad systemem, którego granice modułów są niepoprawne

- 1.1. Próba integracji modułów przy pomocy zdarzeń i komend
- 1.2. Analiza problemów
- 1.3. Analiza genezy
- 1.4. Wstęp do wizualizacji architektury systemów
  - 1.4.1. Podejście C4
  - 1.4.2. Drivery architektoniczne na każdym poziomie
  - 1.4.3. Pytania na jakie musimy sobie odpowiedzieć na każdym poziomie

2. Modelowanie Strategiczne

- 2.1. Podejście "od procesu" vs podejście "od domeny" w kontekście modelu Cynefin
- 2.2. Event Storming
  - 2.2.1. Fazy: rozbieżna eksploracja, analiza, synteza i krystalizacja modelu
  - 2.2.2. Facylitacja sesji - Tips and Tricks
  - 2.2.3. Big Picture - otoczenie systemu
    - 2.2.3.1. Notacje wspomagające
  - 2.2.4. Process Level
    - 2.2.4.1. Heurystyki odkrywania pod-domen biznesowych
- 2.3. Techniki destylacji Bounded Contextów
- 2.4. Mapowanie Bounded Contextów
  - 2.4.1. Published Language
  - 2.4.2. Open Host
  - 2.4.3. Anti-corruption Layer
  - 2.4.4. Shared Kernel

BO·TT·EGA
IT minds

2.5. Mapa kontekstów jako artefakt do komunikacji z Product Ownerem

2.6. Wstęp do archetypów modeli biznesowych na przykładzie generycznych pod-domen

2.7. Modularyzacja na podstawie mapy kontekstów

2.7.1. Single Source of Truth

2.7.2. Single Point of Failure

2.8. API komponentów

3. Modelowanie Taktyczne

3.1. Rodzaje logiki

3.2. Event Storming Design Level

3.2.1. Techniki zadawania pytań o granice spójności

3.3. Koncepcja języka Wzorców DDD

3.3.1. Potrzeba większej ilości building blocks niż serwis i encja (procedura i struktura danych)

3.4. Agregaty

3.4.1. Hermetyzacja i otwarcie na rozbudowę

3.4.2. Określanie granic agregatów na podstawie spójności reguł biznesowych

3.4.3. Modelowanie niezmienników

3.4.4. Porównanie kilku modeli - konsekwencje pod względem zmian wymagań

3.4.5. Kiedy nie stosować agregatów

3.5. Value objects

3.5.1. Zwiększenie siły wyrazu

3.5.2. Styl funkcyjny

3.6. Serwisy Domenowe

3.6.1. Model procedur biznesowych

3.7. Repozytoria

3.7.1. Abstrakcja magazynu danych

BOTTEGA IT minds

- 3.7.2. Orientacja na model domenowy zamiast na model danych
- 3.7.3. Hermetyzacja reguł spójności danych
- 3.8. Fabryki
  - 3.8.1. Walidacja
  - 3.8.2. Logika biznesowa podczas składania obiektów
  - 3.8.3. Wsparcie dla testability
- 3.9. Polityki (strategie)
  - 3.9.1. Modelowanie w stylu funkcyjnym
  - 3.9.2. Open Close Principle (SOLID) w praktyce
  - 3.9.3. Podejście Supple Design
  - 3.9.4. Dekorowanie
  - 3.9.5. Umiejscowienie w 4 poziomach modelu
- 3.10. Zdarzenia biznesowe
  - 3.10.1. Wewnątrzmodułowe vs zewnętrzne
  - 3.10.2. Antywzorce
    - 3.10.2.1. State transfer
    - 3.10.2.2. Opresyjne zdarzenia
- 3.11. Specyfikacje
  - 3.11.1. Modelowanie złożonych warunków biznesowych
- 3.12. Praktyczne przykłady modelowania biznesowego z wykorzystaniem Building Blocks
- 3.13. Cztery poziomy modelu
  - 3.13.1. Decision Support
  - 3.13.2. Policy
  - 3.13.3. Operations
  - 3.13.4. Capability

BO·TT·EGA
IT minds

4. Architektura aplikacji - uwspólnienie modelu analitycznego i projektowego

- 4.1. Podział logiki na aplikacyjną i domenową
- 4.2. Podejście warstwowe - rozmieszczenie building blocks na warstwach
  - 4.2.1. Warstwa interfejsów (prezentacji)
  - 4.2.2. Warstwy logiki - rozwarstwienie na dwie warstwy logiki
    - 4.2.2.1. Logika aplikacyjna (API)
      - 4.2.2.1.1. Model tego CO system powinien robić (User Case lub User Story)
      - 4.2.2.1.2. Zapis Domain Story w formie czytelnej prozy
    - 4.2.2.2. Logika domenowa (Building Blocks DDD)
      - 4.2.2.2.1. Model tego JAK i DLACZEGO system powinien się tak zachowywać
  - 4.2.3. Warstwa infrastruktury
    - 4.2.3.1. Dostęp do danych
    - 4.2.3.2. Infrastruktura techniczna
      - 4.2.3.2.1. Problemy ORM
      - 4.2.3.2.2. Problemy propagacji zdarzeń
- 4.3. Przełożenie Use Case/User Story na warstwę aplikacji
- 4.4. Przełożenie modelu biznesowego na building blocks warstwy domenowej

5. Architektura systemu

- 5.1. Dobór strategii integracji na podstawie mapy kontekstów
- 5.2. Zarys architektury integracji modułów
  - 5.2.1. Event Driven Architecture
  - 5.2.2. Microservices