BO·TT·EGA
IT minds

# Program szkolenia:

# Projektowanie systemów modularnych, rozproszonych i Event Driven: podejście praktyczne

## Informacje:

| | |
|---|---|
| **Nazwa:** | **Projektowanie systemów modularnych, rozproszonych i Event Driven: podejście praktyczne** |
| **Kod:** | **Arch-practical** |
| **Kategoria:** | Architektura systemów i aplikacji |
| **Odbiorcy:** | developerzy, DevOps, architekci |
| **Czas trwania:** | 2-3 dni |
| **Forma:** | 50% wykłady / 50% warsztaty |

Od architekta oczekuje się, że na podstawie celu biznesowego i oczekiwanych funkcjonalności dostarczy implementowalny projekt systemu. Od wysokopoziomowych diagramów, przez narzędzia i niskopoziomowe rozwiązania techniczne, tak aby klient dostał dokładnie to czego potrzebuje, otrzymał niezbędne charakterystyki jakościowe, jak na przykład niskie koszty utrzymania i by jednocześnie nie przepłacił za całość.
Tymczasem większość zespołów deweloperskich podchodzi do architektury systemów beztrosko. Nie wie na jakie cechy niefunkcjonalne optymalizować swoją architekturę, nie bierze pod uwagę ograniczeń ludzkich i organizacji, zbyt szeroko definiuje odpowiedzialności modułów i mikroserwisów, nie wie jaki styl architektoniczny rozwiązania przyjąć i dlaczego, lub wprowadza kosztowne podejścia i narzędzia, które nie zwracają nakładu pracy na nie poniesionego. Wiele zespołów nie potrafi wizualizować, weryfikować i analizować możliwych architektur.
Źle przygotowana architektura jest w stanie pogrzebać każdy projekt, niezależnie od poziomu operowania językiem, bibliotekami i frameworkami.

Na tym szkoleniu nauczysz się wszystkiego co niezbędne, by świadomie tworzyć rozwiązania w jednym z trzech najpopularniejszych stylów architektonicznych: modularnym monolicie, rozproszonych synchronicznych mikroserwisach, i mikroserwisach asynchronicznych, podejściu inaczej zwanym Event Driven Architecture.

W ramach szkolenia uczestnicy ćwiczą trzy style projektując nowy system. Omawiamy typowe błędy i analizujemy rozwiązania. Uczymy jak świadomie wybrać styl adekwatny do sytuacji, jak tworzyć części systemu z wysokim SLA, jak zapewnić wysoką wydajność, jak podzielić moduły i mikroserwisy.
Pokazujemy najnowsze rozwiązania takie jak rozproszone event logi (Apache Kafka) i doradzamy jak ich możliwości wpływają na projekt. Analizujemy jak struktura organizacyjna i zakładane cechy niefunkcjonalne wpływają na decyzje projektowe. Wszystko to podajemy na przykładach z życia, z projektów w których braliśmy sami udział.

Po szkoleniu uczestnicy potrafią

- świadomie i profesjonalnie zdecydować na co optymalizują architekturę i które obostrzenia organizacyjno-techniczne muszą wziąć pod uwagę
- stworzyć profesjonalny projekt architektoniczny w modelu C4, zweryfikować jego poprawność, przedstawić go inwestorowi oraz współpracować z innymi architektami nad projektem większego systemu

- świadomie zdecydować czy projekt wymaga modularnego monolitu, mikroserwisów synchronicznych, czy architektury event-driven i w jakim zakresie
- zapewnić wysokie SLA ścieżek krytycznych, i odpowiednią ich wydajność
- zapewnić elastyczność systemu i gotowość na nowe wymagania przy minimalnym koszcie rozwoju
- i w końcu stworzyć architekturę która będzie w pełni implementowalna

# Zalety szkolenia:

- praktyczne podejście
- stworzenie profesjonalnego projektu architektonicznego w modelu C4
- szkolenie jest agnostyczne względem języka programowania: nie ma znaczenia czy używasz na co dzień .NET, JVM, Ruby czy PHP. Będziemy tworzyć diagramy w Javie, ale nie trzeba znać samego języka

BOTTEGA IT minds

# Szczegółowy program:

**1. Jak organizować architekturę w firmie, jak organizować architektów, jak być odpowiedzialnym architektem i jak pracować z innymi**

**2. Narzędzia i notacje (model C4)**

**3. Niefunkcjonalne wymagania, ograniczenia organizacyjne, ludzkie, formalne, ich wpływ na architekturę**

**4. Świadomy wybór kierunku optymalizacji architektury i jej konsekwencje (koszty utrzymania, wydajność, SLA, odporność na awarie, nakład inicjalny pracy, możliwości rozwoju i elastyczność, dostępność)**

**5. Trzy style architektury współczesnej: modularny monolit**

- 5.1. skąd się wziął monolit niemodularny, czemu wiele projektów tak kończy
- 5.2. jakie są zalety, wady, kiedy świadomie używać
- 5.3. ryzyko wdrożenia z przykładami; jakie są sposoby minimalizacji problemu
- 5.4. jak dzielić na moduły: czym jest prawdziwy moduł i zmiana paradygmatu dzielenia (zachowania, nie dane)
- 5.5. ćwiczenia w budowaniu architektury modularnego monolitu

**6. Trzy style architektury współczesnej: system rozproszony synchroniczny**

- 6.1. jakie są zalety, wady, kiedy świadomie używać
- 6.2. jak wyznaczać granice, moduły vs mikroserwisy
- 6.3. jak testować systemy rozproszone
- 6.4. metody komunikacji i kontroli: orkiestracja vs choreografia
- 6.5. jak obsłużyć błędy komunikacji i brak transakcyjności
- 6.6. jak zapewnić wydajność przy narzucie komunikacji sieciowej
- 6.7. ćwiczenia w budowaniu architektury rozproszonej synchronicznej

**7. Trzy style architektury współczesnej: system Event Driven (rozproszony asynchroniczny)**

- 7.1. jakie są zalety, wady, kiedy świadomie używać
- 7.2. procesowanie strumieni, łączenie zdarzeń z różnych strumieni i zapewnienie następstwa czasowego

BOTTEGA IT minds

- 7.3. serwisy stanowe a bezstanowe, read model
- 7.4. Sagi i procesy długotrwające
- 7.5. obsługa błędów w systemie asynchronicznym
- 7.6. nowoczesne narzędzia do messagingu na przykładzie Apache Kafka
  - 7.6.1. liniowa skalowalność i persystencja zdarzeń
  - 7.6.2. gwarancje kolejności
  - 7.6.3. partycjonowanie
  - 7.6.4. grupy konsumentów
  - 7.6.5. grupy konsumentów
  - 7.6.6. compacted topic i automatyczny read model
  - 7.6.7. transakcyjność publikacji
- 7.7. ćwiczenia w budowaniu architektury Event Driven

**8. Świadomy wybór stylów architektonicznych obecnych w projekcie**

- 8.1. kiedy, które i jak wybrać
- 8.2. Jak i kiedy mieszać
- 8.3. Jak wielkość organizacji i projektu wpływa na zmianę podejścia

**9. Typowe błędy projektów architektonicznych na przykładach**

**10. Optymalizacja na niskie koszty rozwoju**

**11. Optymalizacja na wysokie SLA**

**12. Narzędzia do weryfikacji architektury w monolitach i systemach rozproszonych**

- 12.1. archunit
- 12.2. consumer driven contract