

Program szkolenia:

Architektura oprogramowania - ujęcie strategiczne w aspektach kultury organizacyjnej, zarządzania i zorientowania na produkty

Informacje:

Nazwa:	Architektura oprogramowania - ujęcie strategiczne w aspektach kultury organizacyjnej, zarządzania i zorientowania na produkty
Kod:	mngr-arch
Kategoria:	Dla managmenetu
Odbiorcy:	architekci, kierownicy projektów, liderzy zespołów, liderzy techniczni, Scrum Masters, management, Product Owners
Czas trwania:	2 dni
Forma:	warsztaty

W 2012 roku Thoughtworks, wprowadzając mikroserwisy, przyspieszył proces budowania dużych systemów informatycznych o rząd wielkości, przełamując prawo Freda Brooksa i rozwiązując problemy dużych projektów z poprzednich 38 lat.

Dzięki temu firmy takie jak Amazon (AWS) twierdzą, że im bardziej rosną, tym łatwiej jest im dostarczać nowe funkcjonalności użytkownikom końcowym. Aby to osiągnąć, konieczne są zmiany kulturowe i organizacyjne, zupełnie inne podejście do zespołów, architektury i testowania. Istnieje wiele firm, którym udało się wdrożyć to podejście, a jeszcze więcej, które poniosły porażkę.

Warsztat ten jest skierowany do managerów. Omówimy wszystkie najlepsze praktyki umożliwiające tę transformację oraz błędy, które mogą uniemożliwić osiągnięcie korzyści. Wszystko na podstawie rzeczywistych przykładów.

Podczas szkolenia odpowiemy na następujące pytania:

- Jak zakładać nowe produkty i zespoły?
- Jak zarządzać projektami i problemami międzyproduktowymi?
- Jak radzić sobie z architekturą na dużą skalę?
- Jak dokonywać estymacji i planować pracę?
- Jak oceniać zespoły i poszczególne osoby?
- Jak rekrutować nowych pracowników?
- Jak testować tego rodzaju systemy?
- Dlaczego nie można używać ESB i do czego ESB nadal się przydaje?
- Dlaczego nie należy budować generycznych mikroserwisów od razu?
- Dlaczego warto unikać monorepo?
- Skład zespołu: ilu juniorów na zespół?
- Kto to jest senior?
- Czym jest platforma?
- Jakie są typy zespołów (team topologies)?
- Optymalizacja obciążenia poznawczego: jaka powinna być wielkość produktu/mikroserwisu /modułu?
- Jak organizować dyżury i co dzieje się ze wsparciem technicznym?
- Jak mikroserwisy oparte na zdarzeniach obsługują błędy?

- Struktura organizacyjna: ile błędów pojawi się w środowisku produkcyjnym?

Dodatkowe zagadnienia, które omówimy:

- FinSecDevOps
- Ramy decyzyjne Cynefin: typy problemów i podejmowanie decyzji
- Continuous Deployment – jak to działa i co zrobić ze starymi monolitami
- Migracja do mikroserwisów: wzorzec strangler
- Odporność systemów (Resiliency)
- Szablon usług dostosowany do organizacji (Tailored Service Template)
- Typowe błędy w architekturze
- Jak radzić sobie z odmową zespołu produktowego?
- Orkiestracja i choreografia, mapowanie kontekstu

Zalety szkolenia:

Szczegółowy program:

1. Wprowadzenie do mikroservisów i ich wpływu na organizację

1.1. Historia i ewolucja architektury mikroservisowej

1.1.1. Tradycyjne monolity vs. mikroservisy

1.1.2. Przełamanie prawa Freda Brooksa przez Thoughtworks

1.1.3. Jak Amazon i inne firmy skalują mikroservisy

1.2. Wyzwania i pułapki związane z mikroservisami

1.2.1. Typowe błędy organizacyjne i techniczne

1.2.2. Kiedy mikroservisy nie działają?

1.2.3. Koszty wdrożenia i utrzymania

2. Struktura organizacyjna i kultura pracy w modelu mikroservisowym

2.1. Zespoły produktowe i ich organizacja

2.1.1. Struktura zespołów (Team Topologies)

2.1.2. Role i odpowiedzialności w zespołach mikroservisowych

2.1.3. Jaka liczba juniorów i seniorów jest optymalna?

2.2. Zarządzanie zespołami i rekrutacja

2.2.1. Jak rekrutować i szkolić nowych pracowników?

2.2.2. Jak oceniać zespoły i poszczególne osoby?

2.2.3. Jak unikać „bus factor” w zespołach?

2.3. Optymalizacja obciążenia poznawczego

2.3.1. Jaka powinna być wielkość produktu/mikroservisu/modułu?

2.3.2. Jak podzielić system, aby był łatwiejszy do zarządzania?

3. Architektura mikroservisów na dużą skalę

3.1. Kluczowe zasady architektury mikroservisowej

3.1.1. Czym jest platforma i jakie są jej typy?

3.1.2. Jakie podejście do architektury zwiększa skalowalność?

3.1.3. Jakie są typowe błędy w architekturze?

3.2. Podejmowanie decyzji w projektowaniu architektury

3.2.1. Dlaczego warto unikać monorepo?

3.2.2. Kiedy warto stosować wzorzec strangler?

3.2.3. Dlaczego nie budować generycznych mikroserwisów od razu?

3.3. Orkiestracja i choreografia w mikroserwisach

3.3.1. Jak efektywnie zarządzać komunikacją między mikroserwisami?

3.3.2. Context Mapping – kiedy stosować jakie podejście?

4. Zarządzanie procesami i wdrożeniami

4.1. Continuous Deployment w praktyce

4.1.1. Jak wdrażać zmiany szybko i bezpiecznie?

4.1.2. Co zrobić ze starymi monolitami?

4.1.3. Jak testować mikroserwisy na dużą skalę?

4.2. Obsługa błędów i zapewnienie odporności systemu

4.2.1. Jak mikroserwisy obsługują błędy?

4.2.2. Resiliency – jak zabezpieczyć system przed awariami?

4.2.3. Typowe błędy w projektowaniu architektury

4.3. Organizacja wsparcia technicznego

4.3.1. Jak ustawić system dyżurów (on-duty)?

4.3.2. Jak zorganizować wsparcie techniczne zespołów?

4.3.3. Jak zmniejszyć liczbę błędów w produkcji?

5. Zarządzanie projektami i planowanie pracy

5.1. Tworzenie nowych produktów i zespołów

5.1.1. Jak podzielić produkt na zespoły i mikroserwisy?

5.1.2. Jak radzić sobie z projektami międzyproduktowymi?

5.1.3. Jakie są najlepsze praktyki w zarządzaniu backlogiem?

5.2. Estymacje i planowanie pracy

5.2.1. Jak realistycznie oceniać czas wdrożenia?

5.2.2. Jak unikać problemów z nierealistycznymi oczekiwaniami?

5.2.3. Jak podejść do priorytetyzacji zadań?

5.3. Praca z zespołem produktowym

5.3.1. Jak radzić sobie z odmową zespołu produktowego?

5.3.2. Kiedy warto zmieniać priorytety?

5.3.3. Jak budować efektywną współpracę między zespołami?

6. Nowoczesne podejścia do zarządzania mikroserwisami

6.1. FinSecDevOps – integracja bezpieczeństwa i operacji

6.1.1. Jak wdrożyć model DevOps w organizacji?

6.1.2. Jak zapewnić bezpieczeństwo mikroserwisów?

6.1.3. Automatyzacja testów i procesów CI/CD

6.2. Cynefin framework – podejmowanie decyzji w złożonych systemach

6.2.1. Jakie są typy problemów i jak je rozwiązywać?

6.2.2. Jakie modele decyzyjne sprawdzają się w IT?

6.2.3. Jak dostosować zarządzanie do dynamicznie zmieniającego się środowiska?

6.3. Przyszłość mikroserwisów i nowe trendy

6.3.1. Jakie zmiany czekają architekturę systemów IT?

6.3.2. Jakie technologie wspierają mikroserwisy?

6.3.3. Jak przygotować organizację na przyszłe wyzwania?