



Microservices od zera do systemu

Kończak Piotr
dla ŁDI 2018



Kim ja jestem?

- Kończak Piotr
- Software Expert w Transition Technologies PSC sp. z o.o.
- 7 lat doświadczenia
- Java, Spring, (NO)SQL
- DDD, Microservices

- <https://github.com/konczak>
- <https://www.linkedin.com/in/konczak-piotr/>
-

- <https://tinyurl.com/ya5jhy7x>

(prezentacja)



Agenda

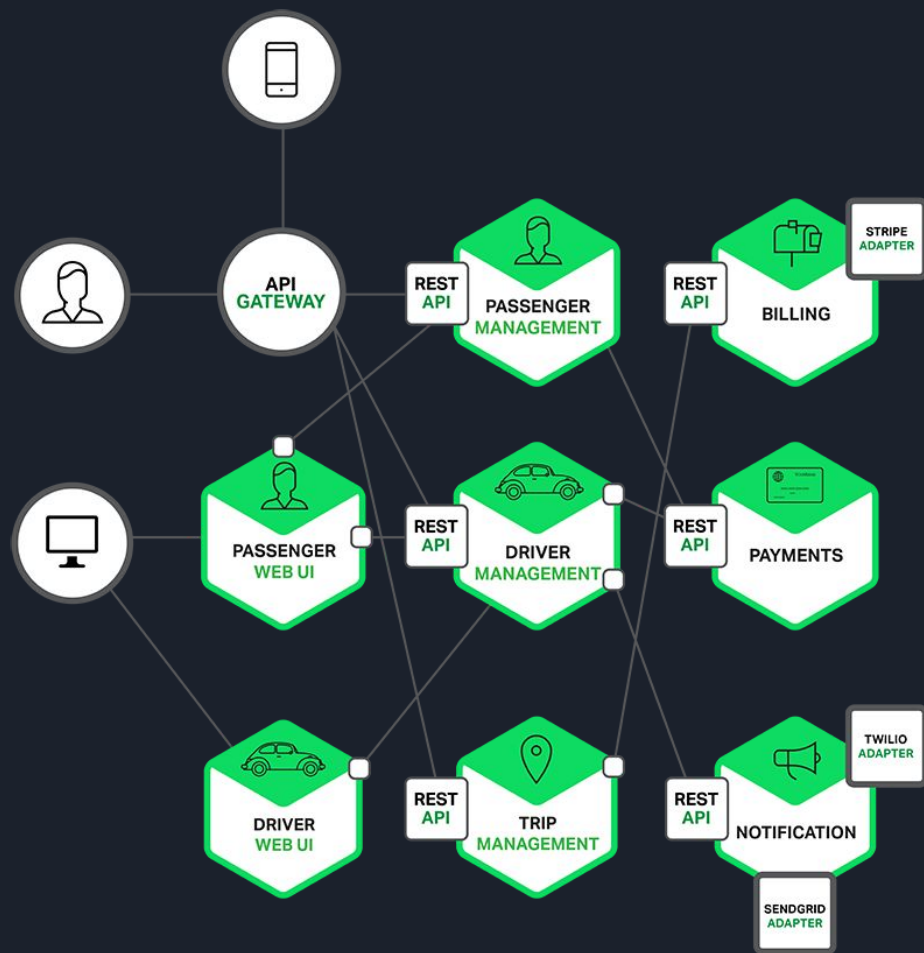
1. Co to są Microservices?
2. Jakie są alternatywy dla Microservices?
3. Jak nie zaczynać projektu w architekturze Microservices,
4. Jak zacząć i dostarczyć projekt oparty o Microservices.

Co to są Microservices?



Co to są Microservices?

Architektura systemu wykorzystująca wiele małych komponentów w celu dostarczenia skomplikowanej usługi biznesowej.



Netflix & Micro-Services





Co to są Microservices?

+ *Zalety*

Stos technologiczny dostosowany do potrzeb biznesowych

Niezależność skalowania

Małe testowalne “black box”

Szybkość wdrożenia (fault tolerant)

- *Wady*

Skomplikowana architektura

Nowe wyzwania dla IT i DevOps

Extra ruch sieciowy

Konieczność stosowania dodatkowych elementów spajających

Jakie są alternatywy dla
Microservices?



Jakie są alternatywy dla Microservices?

Monolith - wszystko w jednym komponencie

Service Oriented Architecture - prekursor Microservices oparty o Enterprise Service Bus

UWAGA: istnieją również inne architektoniczne wzorce projektowe, ale nie są one dokładnymi alternatywami

Jak **nie** zaczynać projektu w architekturze Microservices?



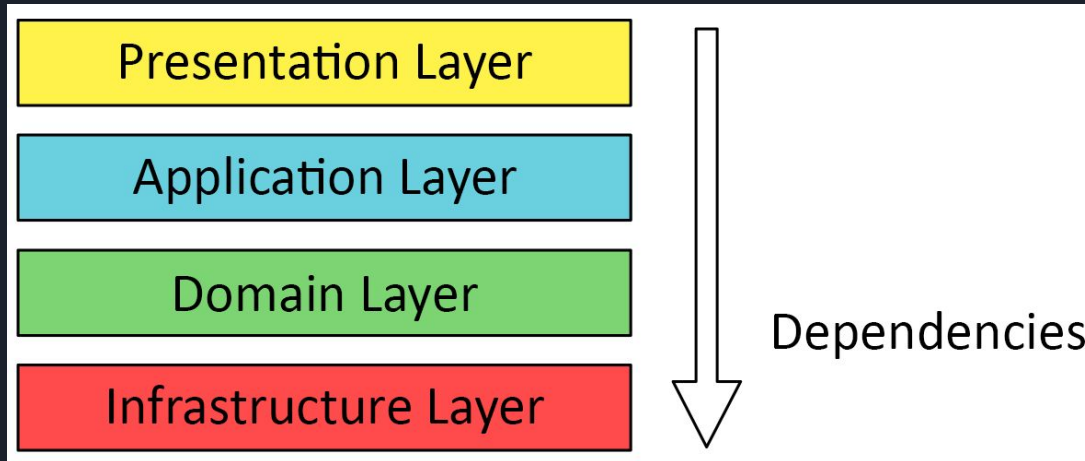
Jak **nie** zaczynać projektu w architekturze Microservices?

- zrobmy projekt Microservices, hura!,
- własna serwerownia i IT niegotowe na Microservices,
- 1 API aka 1 endpoint != Microservice => = nanoservice,
- mamy problemy z wydajnością - dodajmy kolejki zdarzeń (Consumer-Producer),
- każda instancja ma własną VM ale wszystkie mogą być na tym samym serwerze,
- mamy N Microservices, każdy konfiguruje się ręcznie a logi zbiera osobno,

Jak zacząć i dostarczyć
projekt oparty o
Microservices?

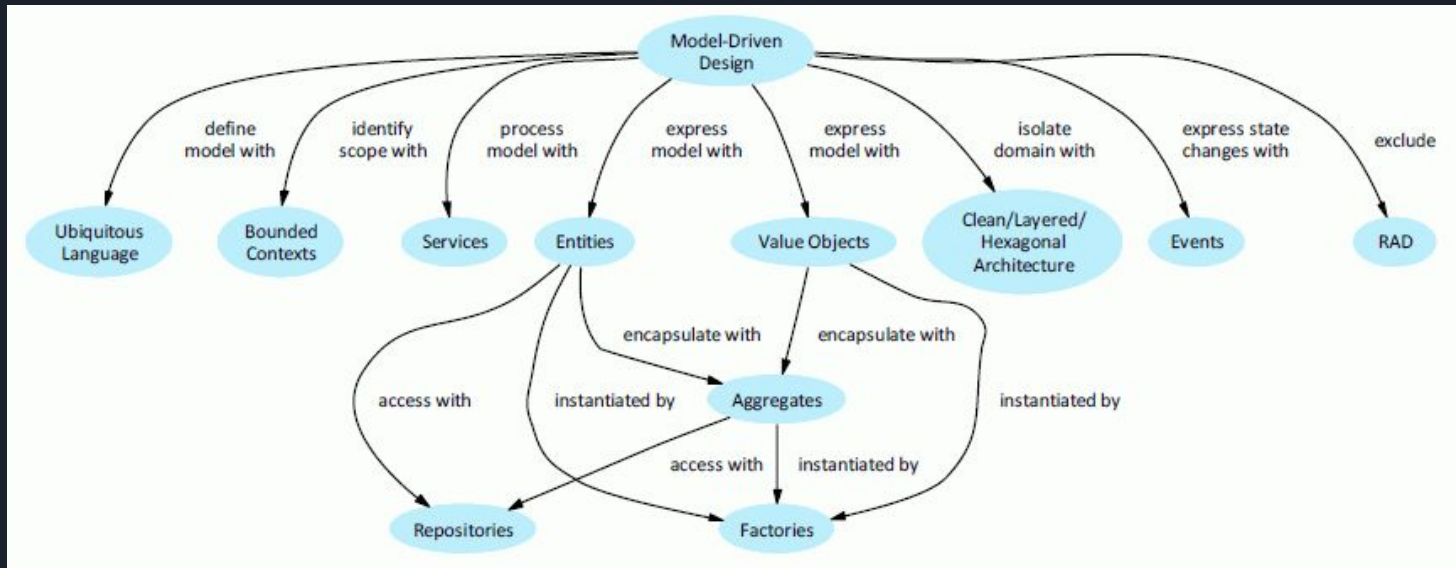
Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
 - a. wewnątrz aplikacji stosujemy architekturę 3 warstwową...



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
 - a. wewnątrz aplikacji stosujemy architekturę 3 warstwową oraz DDD,





Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
 - a. wewnątrz aplikacji stosujemy architekturę 3 warstwową oraz DDD,
 - b. API które jest “stateless”,



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
 - a. wewnątrz aplikacji stosujemy architekturę 3 warstwową oraz DDD,
 - b. API które jest “stateless”,
 - c. wersjonujemy bazę danych (np. Flyway, Liquibase),



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
 - a. wewnątrz aplikacji stosujemy architekturę 3 warstwową oraz DDD,
 - b. API które jest “stateless”,
 - c. wersjonujemy bazę danych,
 - d. scentralizowane wersjonowanie i branching model (np. GitFlow),



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
 - a. wewnątrz aplikacji stosujemy architekturę 3 warstwową oraz DDD,
 - b. API które jest "stateless",
 - c. wersjonujemy bazę danych,
 - d. scentralizowane wersjonowanie i branching model,
 - e. Continuous Integration -> Delivery -> Deployment (Jenkins),



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
 - a. wewnątrz aplikacji stosujemy architekturę 3 warstwową oraz DDD,
 - b. API które jest “stateless”,
 - c. wersjonujemy bazę danych,
 - d. scentralizowane wersjonowanie i branching model,
 - e. Continuous Integration -> Delivery -> Deployment,
 - f. wersjonowane kontenery* i skrypty stawiające środowisko,

*opcjonalne



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
2. Cloud (AWS, MS Azure),
 - a. tańszy na start niż własna serwerownia,
 - b. znani operatorzy mają już rozwiązania na nasze problemy,
 - c. uczę się raz - korzystam wielokrotnie,
 - d. auto-scaling - szczególnie gdy nie wiadomo jakiego ruchu się spodziewać,
 - e. monitoring i metryki: wyjątki, Bad requests, ile osób korzysta z feature, użycie zasobów,
 - f. scentralizowane logi z wyszukiwarką.



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
2. Cloud,
3. automatyzacja testów w Continuous Integration,
 - a. unitowe,
 - b. modułowe (elementy aplikacji współpracują),
 - c. integracyjne (elementy systemu współpracują),
 - d. akceptacyjne (biznesowe),
 - e. regresyjne,
 - f. między wersjami np. stary UI vs nowe API,
 - g. pre-prod (aktualizacja bazy danych)



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
2. Cloud,
3. automatyzacja testów w Continuous Integration,
4. start! obserwujemy, analizujemy i aktualizujemy,



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
2. Cloud,
3. automatyzacja testów w Continuous Integration,
4. start! obserwujemy, analizujemy i aktualizujemy,
5. Gateway jako “dummy” proxy,



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
2. Cloud,
3. automatyzacja testów w Continuous Integration,
4. start! obserwujemy, analizujemy i aktualizujemy,
5. Gateway jako “dummy” proxy,
6. autentykację i autoryzację przenosimy do Gateway,



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
2. Cloud,
3. automatyzacja testów w Continuous Integration,
4. start! obserwujemy, analizujemy i aktualizujemy,
5. Gateway jako “dummy” proxy,
6. autentykację i autoryzację przenosimy do Gateway,
7. dodajemy Service Discovery (np. Eureka, Consul),



Jak zacząć i dostarczyć projekt oparty o Microservices?

1. zaczynamy jako Monolith,
2. Cloud,
3. automatyzacja testów w Continuous Integration,
4. start! obserwujemy, analizujemy i aktualizujemy,
5. Gateway jako “dummy” proxy,
6. autentykację i autoryzację przenosimy do Gateway,
7. dodajemy Service Discovery,
8. wprowadzamy śledzenie przepływu requestów (Sleuth, Zipkin),



Jak zacząć i dostarczyć projekt oparty o Microservices?


1. zaczynamy jako Monolith,
2. Cloud,
3. automatyzacja testów w Continuous Integration,
4. start! obserwujemy, analizujemy i aktualizujemy,
5. Gateway jako “dummy” proxy,
6. autentykację i autoryzację przenosimy do Gateway,
7. dodajemy Service Discovery,
8. wprowadzamy śledzenie przepływu requestów,
9. wdramy microservices wg algorytmu:

```
do {  
    microservice = implementMicroservice()  
    setup(microservice)  
    result = compareCompliance(monolith, microservice)  
    if result == ok => switchToMicroserviceUsage()  
} while (system.containsTooFatService())
```



Ciekawostki

- startup - warto skorzystać z istniejących komponentów i usług (abonament, security np. Okta),
- startup - mniej feature-ów, szybciej na produkcji i zebrać info od użytkowników,
- event oriented system - na początku można użyć wzorca Observer bez kolejki,
- czasami lepiej zamiast usuwania rekordów z bazy użyć flagi boolean deleted,
- zanim system się ustabilizuje warto logi ustawić w DEBUG,
- feature flag, możliwość wyłączenia na produkcji nowej funkcjonalność (np. Togglz),
- warto mieć narzędzie do zarządzania zadaniami, SCRUM, Pull requests i Code review,
- crash control - wyłączamy któryś Microservice i sprawdzamy czy system przetrwał,
- AWS CloudFront może raportować użycie UI, np. kraj użytkownika, użytą przeglądarkę,



Kończak Piotr for ŁDI 2018

<https://github.com/konczak>



Przydatne linki

<https://martinfowler.com/articles/microservices.html>

<https://martinfowler.com/tags/domain%20driven%20design.html>

<https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>

<https://start.spring.io/>

http://cloud.spring.io/spring-cloud-netflix/multi/multi_spring-cloud-netflix.html

https://en.wikipedia.org/wiki/List_of_software_architecture_styles_and_patterns

<https://dzone.com/articles/design-patterns-for-microservices>

<https://dzone.com/articles/9-fundamentals-to-a-successful-microservice-design>

<https://www.oreilly.com/programming/free/files/software-architecture-patterns.pdf>

Linki do obrazków:

<https://i.stack.imgur.com/QgctP.png>

<https://image.slidesharecdn.com/thecaseforchaos-brucewong-dec2014-141219123401-conversion-gate02/95/the-case-for-chaos-12-638.jpg?cb=1418992616>

https://cdn-images-1.medium.com/max/869/1*kOlzID6qMBvz52O2_K-SgA.png

https://covee.com/uploads/img/20170306/201844_035O.png