VMPL

SOFTWARE HOUSE

MICROSERVICES

WYZNACZANIE GRANIC PODZIAŁU

SAVOIR VIVRE

- Włączcie Mute
- Kamerę możecie mieć włączoną lub wyłączoną wasz wybór
- W razie pytań, napiszcie na czacie „pytanie” i od razu wyłączcie Mute i zadajcie pytajcie głosowo
- Ja będę zadawał sporo pytań, w pierwszej kolejności odpowiadacie na czacie
- Przed przerwą zawsze podaję godzinę restartu szkolenia po powrocie kolejno odlicz swoją obecność na czacie 1, 2, …, 10

Michał Michaluk

Java Developer
Trainer / Consultant @ BO·TT·EGA IT minds

DDD / Craftsmanship / Architecture
Legacy Code Refactoring

https://github.com/michal-michaluk/

@michal_michaluk

Michał Michaluk

AGENDA

- Architektura systemu
  - Wyzwania i typowe błędy w architekturze Microserwisów
  - Motywacja dla Mikroserwisów
  - Mikroserwisy vs. Monolith
  - Ćwiczenie: Integracja modułów

AGENDA

- Określanie i ocena granic serwisów
  - Notacja Event Storming
  - Data Flow Diagram
- Techniki / heurystyki projektowania podziału
  - Bounded contexts
  - Defensywna modularyzacja

AGENDA

- Stereotypy Mikrousług:
  - Bounded Context
  - Backend for Frontend
  - Invariant
  - Long running process
  - Read Model
  - Algorithm
  - CRUD

PRZERWY

- 15 min co ok. 1,5 h
- Przerwa obiadowa 30 min ok 12:00


VM.PL
4F


kubernetes

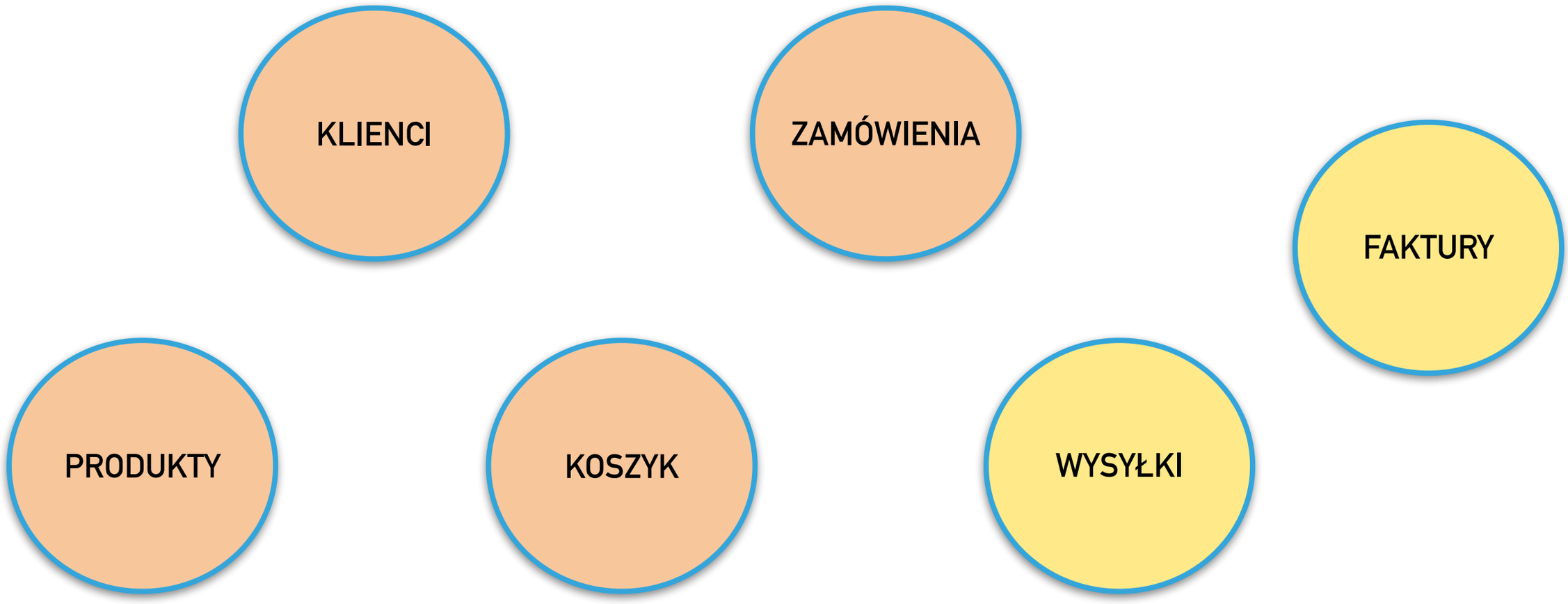

APACHE
kafka


redis


MICRONAUT


elastic

PODZIAŁ NA MODUŁY / MIKROSERWISY



PODZIAŁ NA MODUŁY / MIKROSERWISY

tabelka z klientami
jako micro

KLIENCI

Składowanie bieżących
i historycznych zamówień
jako micro

ZAMÓWIENIA

Integracji z systemem
księgowym
jako micro

FAKTURY

PRODUKTY

dokument store
z opisem produktu
jako micro

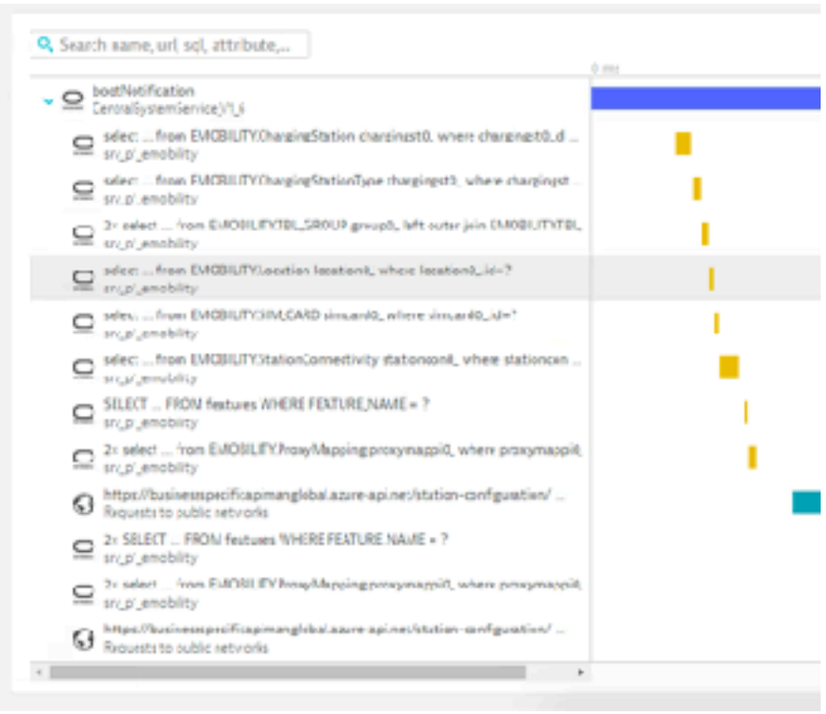
KOSZYK

tabelka łącząca
klient -> produkt + ilość
jako micro

WYSYŁKI

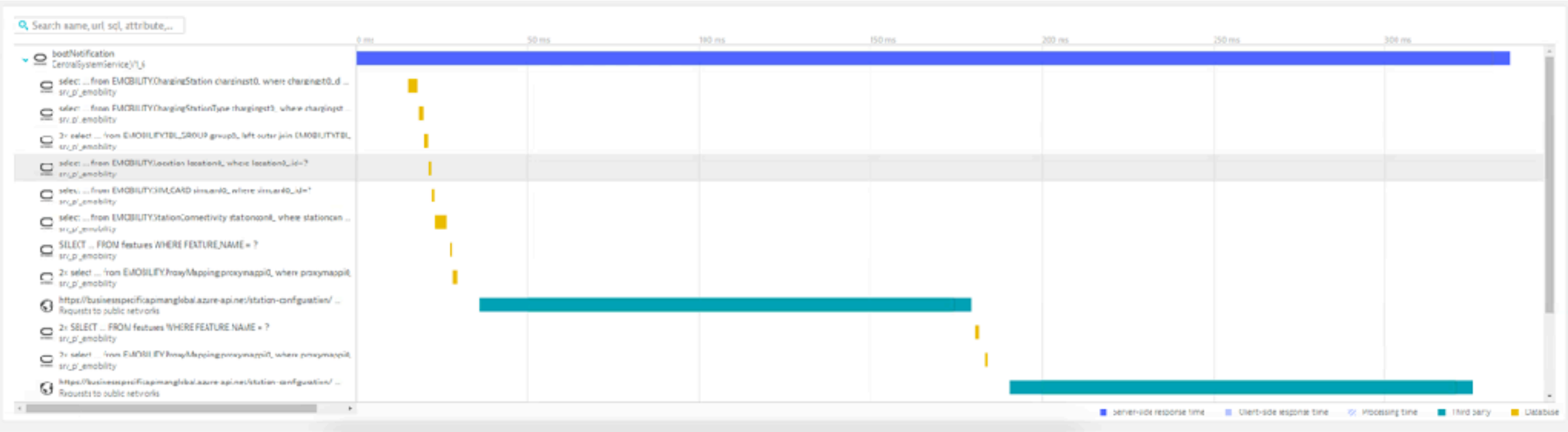
Integracji z systemami
spedytorów
jako micro

PROBLEM

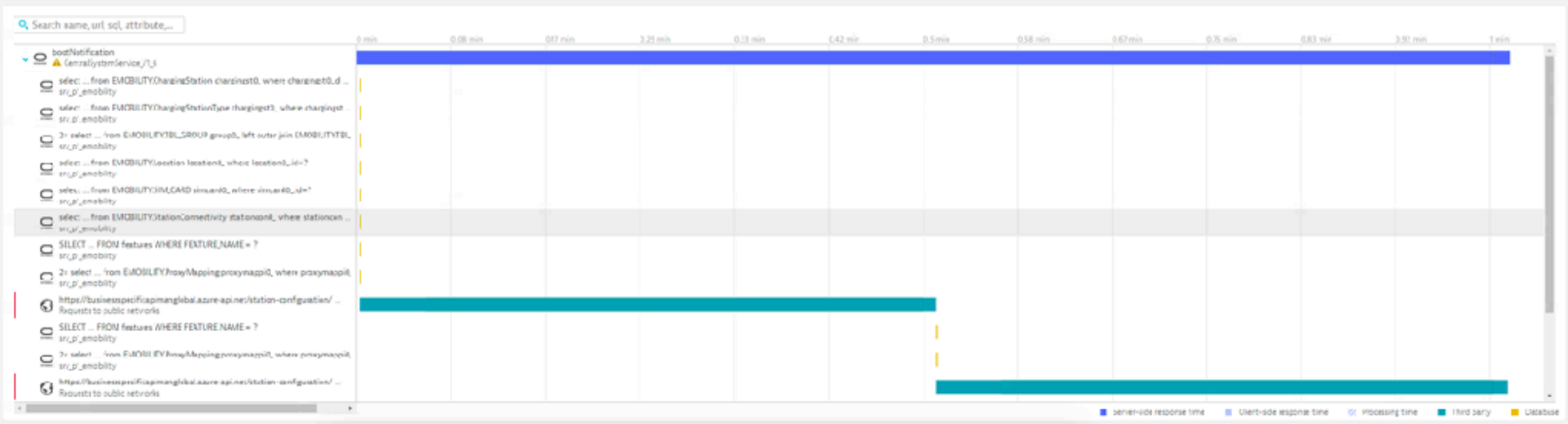


- Podczas przetwarzania żądania w systemie **Monolitycznym**
  - Co zabiera najwięcej czasu?
  - Co jest najbardziej zawodne?

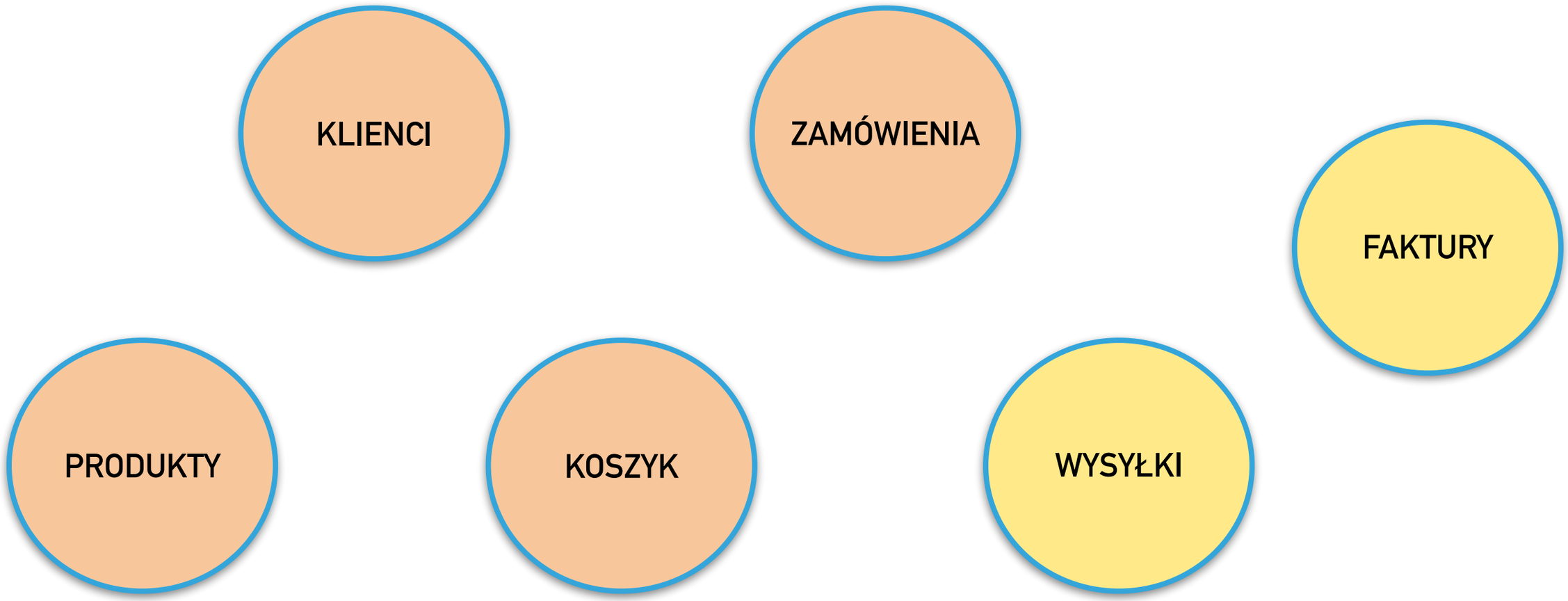
PROBLEM MIKROSERWISÓW



WIELKI PROBLEM



PODZIAŁ NA MODUŁY / MIKROSERWISY



PODZIAŁ NA MODUŁY / MIKROSERWISY

Wszystkie informacje o kliencie
łącznie z dokonywanymi zakupami

CRM

KATALOG PRODUKTÓW

Wyszukiwarka zawierająca
pełen opis potrzebny dla klientów

MAGAZYN / ERP

Stany magazynowe
Wprowadzanie produktów

Procesy wspierające realizację zamówienia

REALIZACJA ZAMÓWIEŃ

KOSZYK I CHECKOUT

Wyliczanie na podstawie koszyka:
- rabatów
- dostępnych form płatności
- dostępnych form dostawy

System księgowy

KSIĘGOWOŚĆ

MOTYWACJA MIKROSERWISY

- Po co stosować Mikroserwisy?
- Czy istnieje **dobra** alternatywa?

MOTYWACJA MIKROSERWISY

- Prawidłowa motywacja dla mikro usług:
  - Niezależne skalowanie
  - Niezależny deployment
  - Niezależna technologia
- Kosztem:
  - Problemy systemów rozproszonych
  - Zwiększone koszty operacyjne

PRZERWA

- wracamy punktualnie o:

10:00

ĆWICZENIE ZAPOZNAWCZE

Treść zadania:

https://github.com/**michal-michaluk/vm-public**/blob/main/**01-zadanie-vm.pdf**

ĆWICZENIE: INTEGRACJA MODUŁÓW W SYSTEMIE

Narysuj integrację pomiędzy komponentami systemu wykorzystując 3 style komunikacji:

- command – rozkaz wykonania pracy, który skutkuje zmianą stanu (tryb rozkazujący)
- event – zgłoszenie zmiany stanu (tryb oznajmiający, czas przeszły, dokonany)
- query – odpytanie o stan bez zmiany stanu (tryb pytający)

SALES

find(criteria):List<OrderDTO>
applyDiscount(orderId,
percentage)

PAYMENT

pay(amount, cause, payerId)

MESSAGE
BROKER/BUS

CRM

promote(clientId, status)

NOTIFIER

???

ODPOWIEDZIALNOŚĆ (API) MODUŁÓW:

- **SALES**
  - potrafi wyszukać zamówienia (w postaci DTO) na podstawie dowolnych kryteriów
  - potrafi nadać rabat na wskazane zamówienie o wskazanej wartości procentowej
- **NOTIFIER**
  - potrafi wysłać maila lub list papierowy o wskazanej treści pod wskazany adres
- **CRM**
  - zarządza danymi klientów i treściami umów
  - potrafi emitować zdarzenia oznajmujące fakty dokonania zmian na klientach
- **PAYMENT**
  - obsługuje płatności, gdzie klient może płacić w kilku transzach
  - potrafi emitować zdarzenia oznajmujące fakty dotyczące etapów płatności:
    - Zlecenie płatności (całej kwoty, zaliczki, transzy)
    - Odrzucenie zlecenia z powodu blokady płatnika
    - Wykonanie płatności (całej kwoty, zaliczki, transzy)

ZAPROPONUJ ROZWIĄZANIE SCENARIUSZA 1.

- Jeżeli klient zostanie awansowany w module CRM do rangi VIP to wówczas:
  - dostaje powiadomienie mailowe z podsystemu mailingu
  - otrzymuje list z treścią nowej umowy z modułu wysyłki, którą musi podpisać aby zacząć współpracę na nowych warunkach
  - otrzymuje 10% rabatu na wszystkie swe zamówienia, które są w statusie DRAFT

ZAPROPONUJ ROZWIĄZANIE SCENARIUSZA 2.

- Integracja modułu SALES z modułem CRM i PAYMENT na przykładzie wymagania:

  Jeżeli klient dokona w ciągu roku zakupów spełniających pewne kryteria (suma wydatków lub ilość sztuk, itd.) oraz posługuje się X razy kartą kredytową YYY to wówczas zostaje mu nadany status GOLD CLIENT.

ĆWICZENIE

- Praca w grupie
  zostaniecie „rozrzuceni” po pokojach
- Nie będę was słyszał w waszych pokojach ćwiczeniowych
- Po kilku minutach zacznę krążyć po pokojach
  omawiać / pomagać z zadaniem
- Po określonym czasie wrócimy z pokojów ćwiczeniowych
  i omówimy ćwiczenie

SALES

find(criteria):List<OrderDTO>
applyDiscount(orderId,
percentage)

PAYMENT

pay(amount, cause, payerId)

Read model

MESSAGE
BROKER/BUS

Event

CRM

promote(clientId, status)

Command

NOTIFIER

???

PRZERWA

- wracamy punktualnie o:

12:30

MIKROSERWISY CZYNNIKI SUKCESU

- Dobry podział = Dobre API
- Sprawność operacyjna (Dev-Ops)
- Dobra jakość pojedynczych usług

VM PL
SOFTWARE HOUSE

PYTANIA

OKREŚLANIE I OCENA GRANIC SERWISÓW

- Notacja Event Storming
- Data Flow Diagram

EVENT STORMING BIG PICTURE - SKUPIENIE SIĘ NA ZACHOWANIACH

VM PL
SOFTWARE HOUSE

EVENT STORMING

EVENT STORMING NOTATION

KLIENT

OFERTA

TERMIN
POKÓJ
CENA

REZERWUJ

OFERTA
USER

MICRO REZERWACJE

BUSINESS RULE

REZERWUJEMY TYLKO POKOJE WOLNE WE WSKAZANYM TERMINIE

BUSINESS RULE

OBOWIĄZUJE CENA PRZEDSTAWIONA WCZEŚNIEJ KLIENTOWI

BOOKING.COM

ZAREZERWOWANO

DFD - DATA FLOW DIAGRAM

Wszystkie informacje o kliencie
łącznie z dokonywanymi zakupami

CRM

KATALOG PRODUKTÓW

Wyszukiwarka zawierająca
pełen opis potrzebny dla klientów

MAGAZYN / ERP

Stany magazynowe
Wprowadzanie produktów

Procesy wspierające realizację zamówienia

REALIZACJA ZAMÓWIEŃ

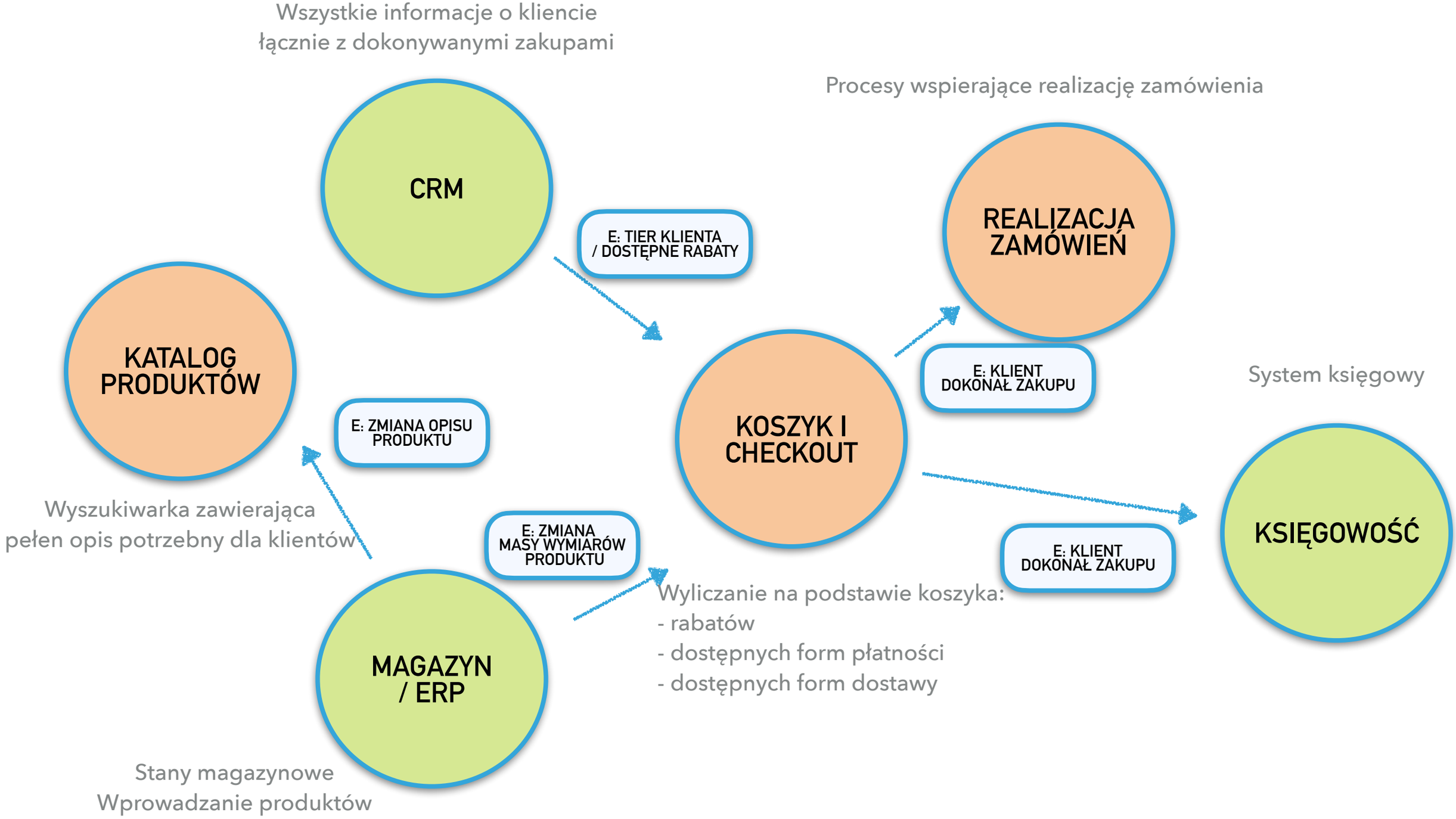
KOSZYK I CHECKOUT

Wyliczanie na podstawie koszyka:
- rabatów
- dostępnych form płatności
- dostępnych form dostawy

System księgowy

KSIĘGOWOŚĆ

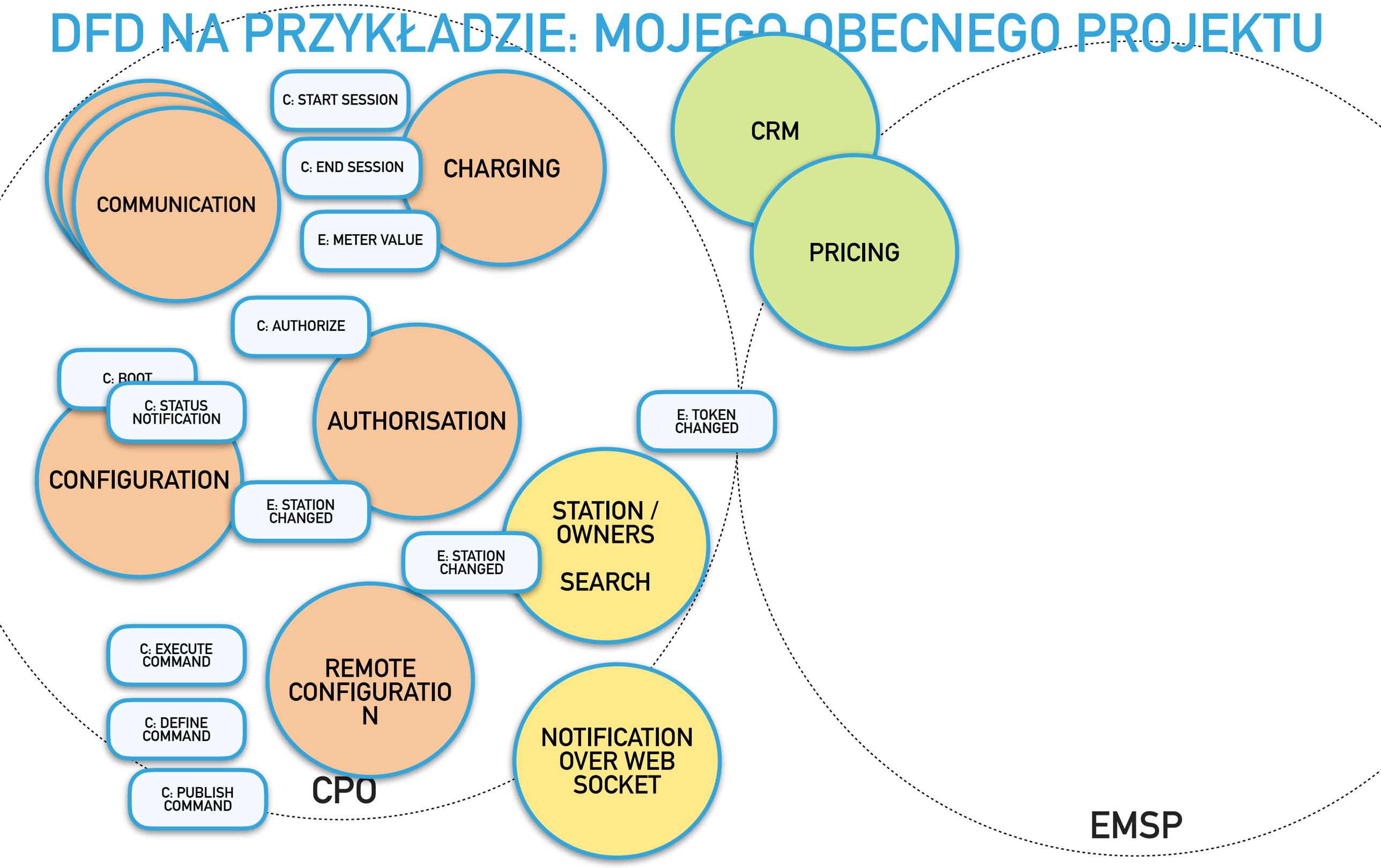
DFD - DATA FLOW DIAGRAM



OCENA PODZIAŁU NA MIKROSERWISY

- Kryteria oceny
  - Spójność reguł biznesowych
    (np. można rezerwować tylko dostępne apartamenty)
  - Dostępność aplikacji (częstość i typ interakcji)
  - Ilość informacji wymienianej
- Techniki oceny
  - Event Storming Design Level
  - Data Flow Diagram

DFD NA PRZYKŁADZIE: MOJEGO OBECNEGO PROJEKTU



PRZERWA

- wracamy punktualnie o:

14:10

TECHNIKI / HEURYSTYKI PROJEKTOWANIA PODZIAŁU

- Bounded contexts
- Defensywna modularyzacja

STRATEGIC DESIGN – BOUNDED CONTEXT

- Bounded Context
  - Obszar biznesu
  - Dedykowane oprogramowanie odizolowane od „obcych wpływów”
- Ludzie którzy nad nim pracują (rotacja psuje BC)
- Współpracujący ludzie z biznesu
- Użytkownicy
- Testerzy

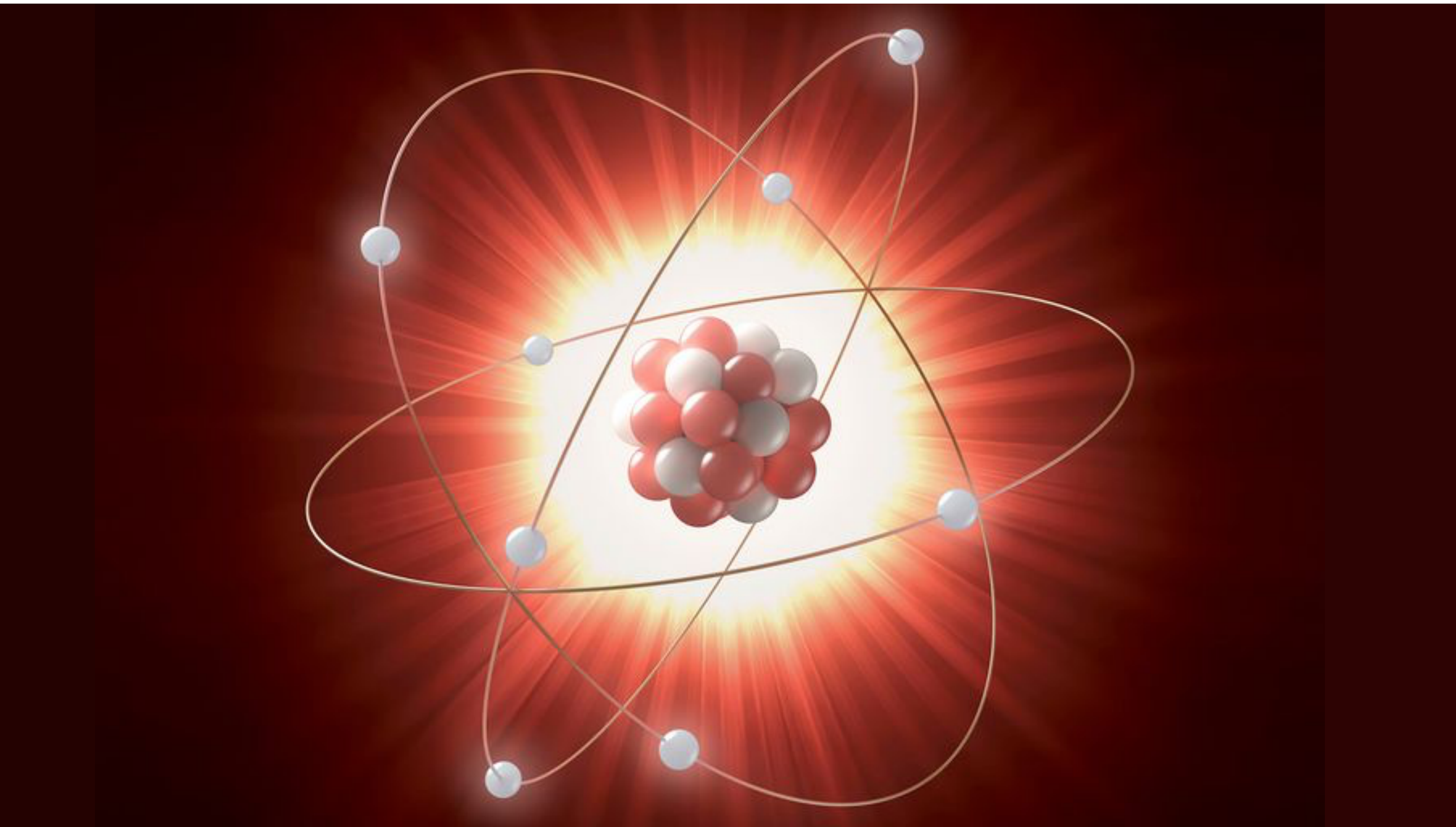
VM PL
SOFTWARE HOUSE

STRATEGIC DESIGN – BOUNDED CONTEXT

- Bounded Context
  - Obszar biznesu
  - Dedykowane oprogramowanie odizolowane od „obcych wpływów”
- Ludzie którzy nad nim pracują (rotacja psuje BC)
- Współpracujący ludzie z biznesu
- Użytkownicy
- Testerzy

TOŻSAME ZROZUMIENIE ZŁOŻONYCH POJĘĆ BIZNESOWYCH
ZBIEŻNE WARTOŚCI BIZNESOWE
TAK ZWANE UBIQUITOUS LANGUAGE

BOUNDED CONTEXT, MODULAR MONOLITH & MICROSERVICES



BOUNDED CONTEXTS

Wszystkie informacje o kliencie
łącznie z dokonywanymi zakupami

CRM

KATALOG PRODUKTÓW

Wyszukiwarka zawierająca
pełen opis potrzebny dla klientów

MAGAZYN / ERP

Stany magazynowe
Wprowadzanie produktów

Procesy wspierające realizację zamówienia

REALIZACJA ZAMÓWIEŃ

KOSZYK I CHECKOUT

Wyliczanie na podstawie koszyka:
- rabatów
- dostępnych form płatności
- dostępnych form dostawy

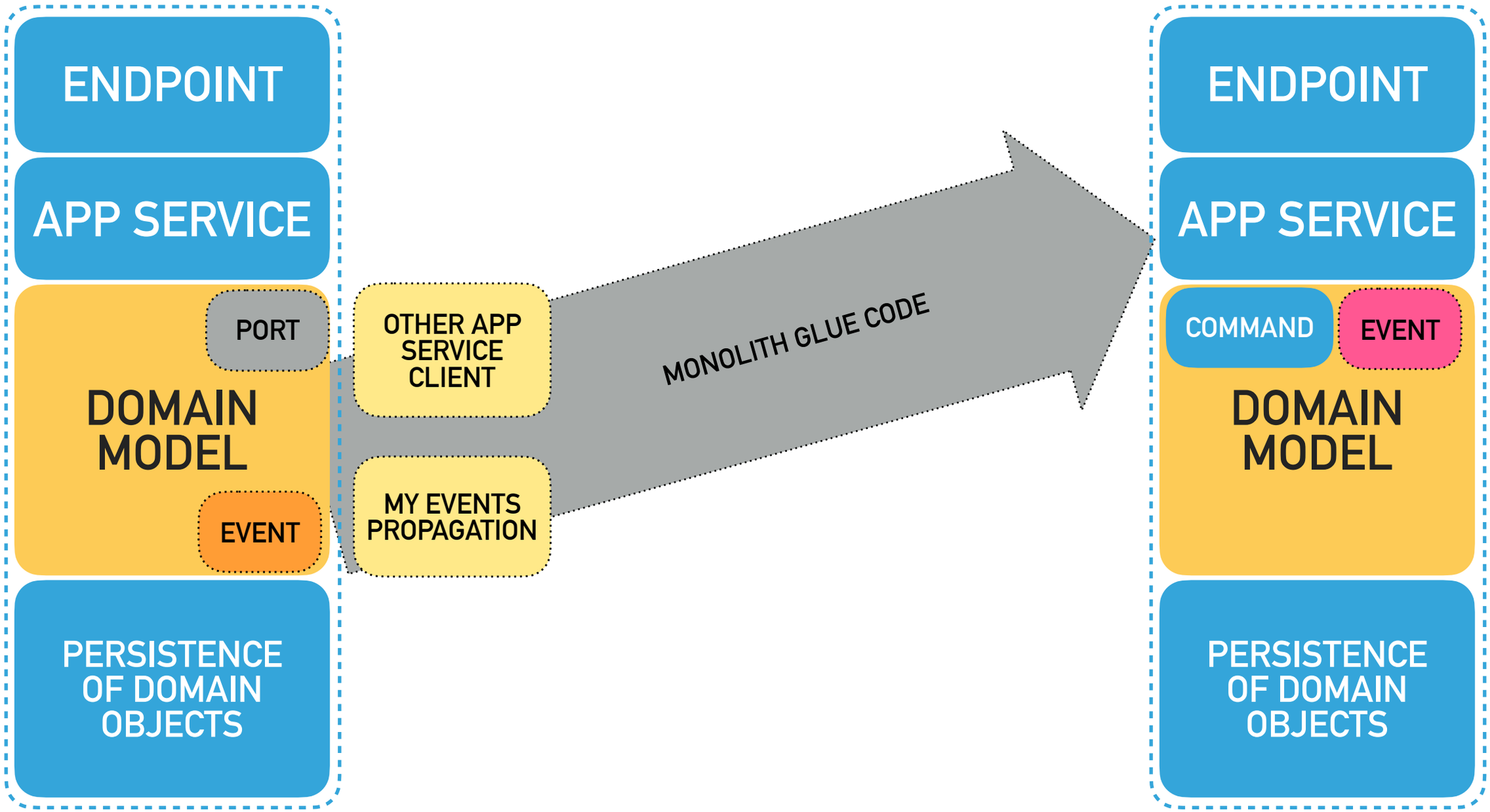
System księgowy

KSIĘGOWOŚĆ

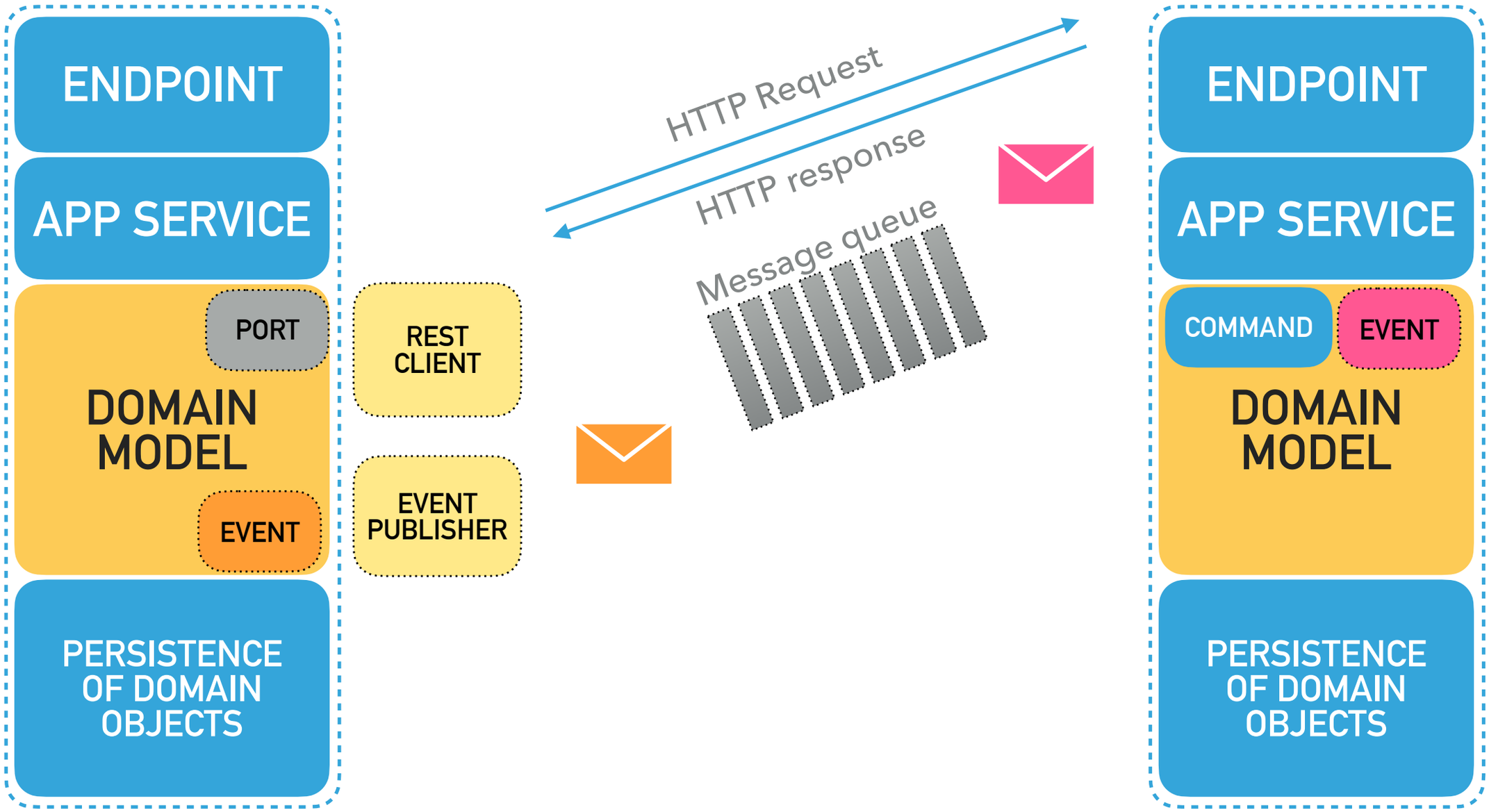
DEFENSYWNA MODULARYZACJA

- Asekuracyjne podejście do dzielenia na mikroservisy
  by minimalizować koszt pomyłki / przeprojektowania
- Jeśli projektujesz microservisy w ramach zespołu
  - Implementację warto zacząć od modularnego monolitu
- Modularny monolit:
  - Osobne **niezależne** moduły gradle / maven
  - Zawierają zarówno logikę, persystencję i endpointy dla frontendu
  - Nie zawierają klientów restowych czy listenerów eventów
    integrujących z innymi microservice-ami
  - Dodatkowy moduł spinający porty wyjściowe z wejściowymi między modułami
  - Deployment w jednym kontenerze

MODUŁY DEPLOYOWANE JAKO MONOLIT



MODUŁY JAKO MICORSERVICE-Y



DEFENSYWNA MODULARYZACJA

- Pułapką może być tranzakcyjność ACID przy przetwarzaniu event-ów zewnętrznych
- To zachowanie zmieni się przy przejściu na microservice-y

STEREOTYPY MIKROUSŁUG

- **Bounded Context** as a Microservice
- **Backend for Frontend** as a Microservice
- **Invariant** as a Microservice
- **Long running process** as a Microservice
- **Read Model** as a Microservice
- **Algorithm** as a Microservice
- **CRUD** as a Microservice

VM PL
SOFTWARE HOUSE

PYTANIA

VM PL
SOFTWARE HOUSE

DZIĘKUJĘ