Základní koncepty DDD

Workshop

PeckaDesign

Vojtěch Buba

Obsah workshopu

- Co je a co není Domain Driven Design
- Strategický návrh - co jsou to ohraničené kontexty
- Taktický návrh - agregáty, entity, value objekty, eventy …

Co je to doména?

Co je to DDD?

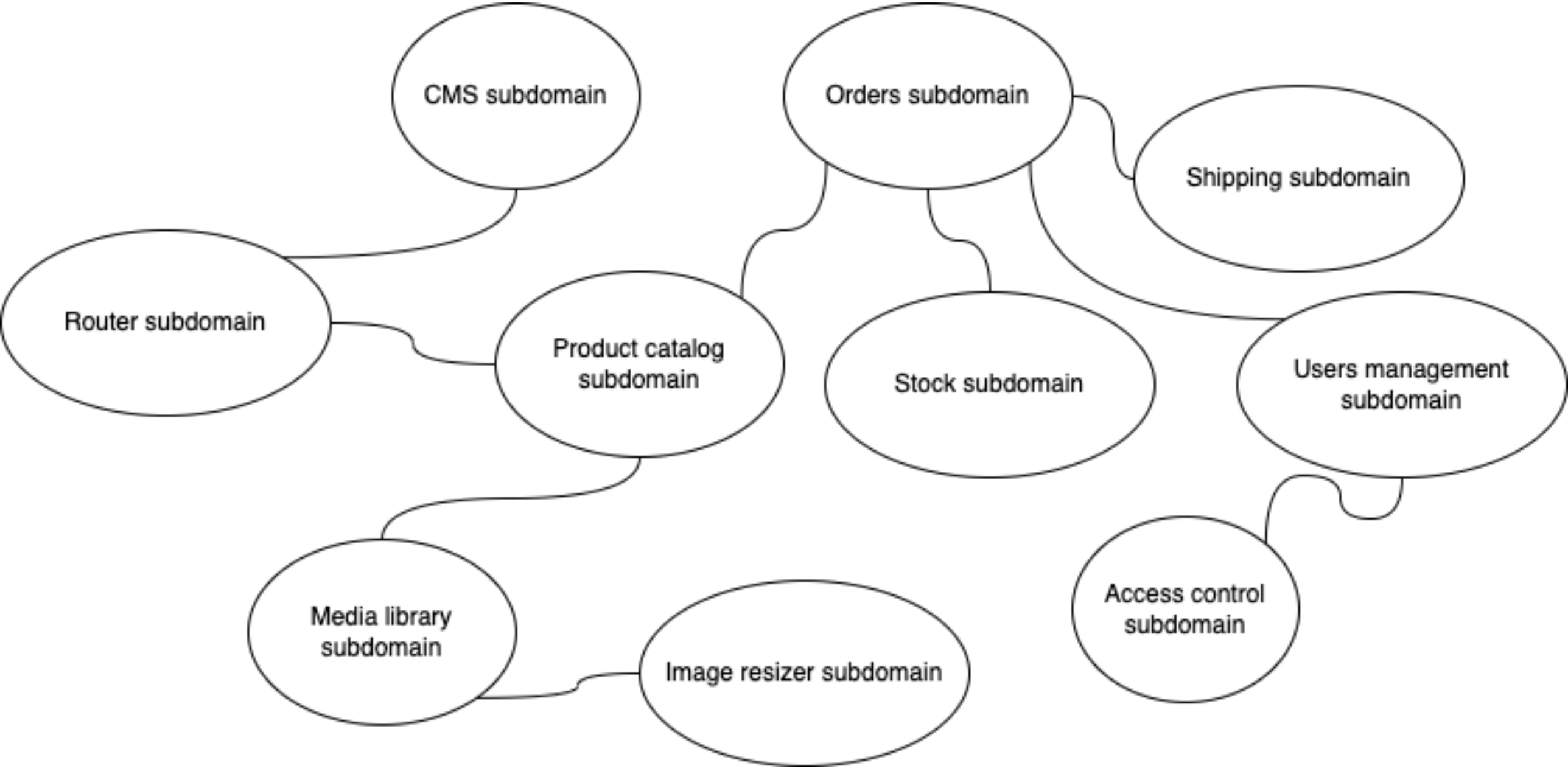
Domain Driven Design

- Metodika
- Sada teoretických nástrojů
- Důraz na modelováni
- Komunikace - doménový expert, všudypřítomný jazyk

Strategický návrh

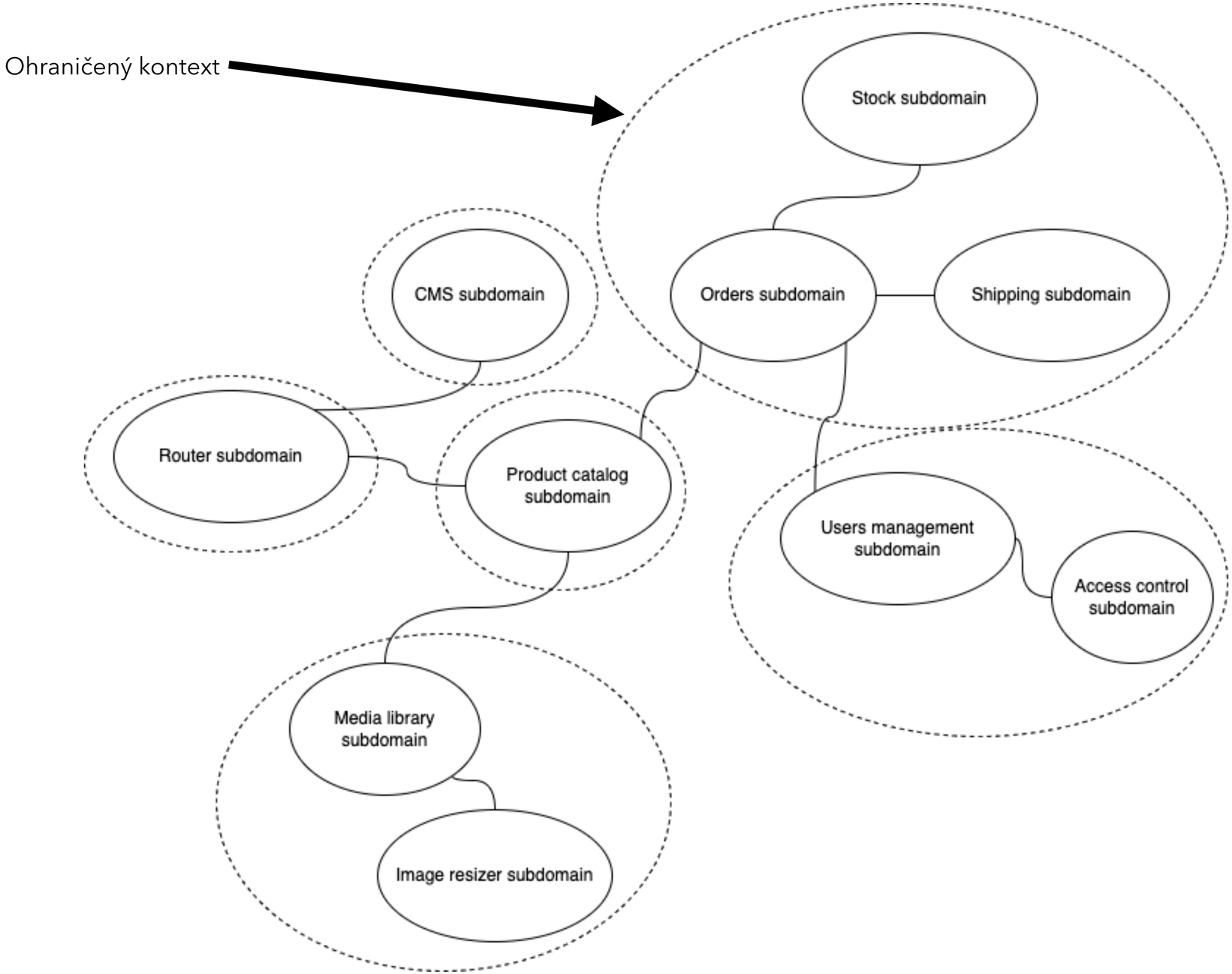
Pojmy

- Doména a poddoména
- Ohraničený kontext
- Všudypřítomný jazyk


CMS subdomain
Orders subdomain
Shipping subdomain
Router subdomain
Product catalog subdomain
Stock subdomain
Users management subdomain
Media library subdomain
Image resizer subdomain
Access control subdomain

Proč je to důležité

- Odstranění nejednoznačnosti
- Kontext mění význam
- Soustředíme se na to co je důležité


Ohraničený kontext
Stock subdomain
CMS subdomain
Orders subdomain
Shipping subdomain
Router subdomain
Product catalog subdomain
Users management subdomain
Access control subdomain
Media library subdomain
Image resizer subdomain

User

Author, Contributor, Owner, Moderator…

Anémický model

Jak jej poznat

- Entity mají pouze set / get metody
- V kódu vidíme několik voláni get / set metod za sebou

Proč je špatně

- Získáváme velmi málo za - object-relational impedance
- Ztráta vyjádření úmyslu
- Objekt je pouze hloupá obálka na data

```php
<?php

interface User {

    public function getFirstName(): string;

    public function setFirstName(string $firstName): void;

    public function getLastName(): string;

    public function setLastName(string $lastName): void;

    public function getStreet(): string;

    public function setStreet(string $street): void;

    public function getCity(): string;

    public function setCity(string $city): void;

    public function getZip(): string;

    public function setZip(string $zip): void;

}
```

```php
<?php

interface User {

    public function changePersonalName(string $firstName, string $lastName): string;

    public function getFirstName(): string;

    public function getLastName(): string;

    public function postalAddress(Address $address): void;

    public function relocateTo(Address $newAddress): void;

    public function getAddress(): Address;

```

Taktický návrh

Identita

Identita

- Uživatelsky definovaná
- Přirozená
- Umělá (pro účely persistence)

Entita

Entita

- Unikátnost definovaná identitou
- Musí být unikátní
- Změna vlastností nemění identitu

Value objekt

Value objekt

- Identita definovaná vlastnostmi
- Dva objekty se stejnými vlastnostmi jsou stejné
- Měří, kvantifikují, popisují
- Neměnnost

Agregát

Agregát

- Soubor entit a value objektů
- Reprezentovaný entitou jako kořenem agregátu
- Hranice konzistentnosti

Doménový event

Doménový event

- Reprezentuje událost která se stala
- Je neměnný
- Je součásti všudypřítomného jazyka
- Komunikace mimo ohraničený kontext

```php
<?php declare(strict_types = 1);

namespace Pd\EventModule;

interface EventInterface
{
    public function occurredOn(): \DateTimeImmutable;

    public function getWeb(): int;

    /**
     * @return array<string, mixed>
     */
    public function getContent(): array;

    public function getType(): string;
}
```

```php
<?php declare(strict_types = 1);

namespace Pd\ProductModule\Event;

class WarehouseInStockEvent implements \Pd\EventModule\EventInterface, \App\ProductModule\Event\HasProductAndWeb
{
    public const TYPE = 'WarehouseInStock';

    private int $itemId;
    private int $productId;
    private int $warehouseId;
    private \DateTimeImmutable $occurredOn;
    private int $webId;


    public function __construct(int $itemId, int $productId, int $warehouseId, int $webId)
    {
        $this->itemId = $itemId;
        $this->productId = $productId;
        $this->warehouseId = $warehouseId;
        $this->occurredOn = new \DateTimeImmutable();
        $this->webId = $webId;
    }


    public function occurredOn(): \DateTimeImmutable
    {
        return $this->occurredOn;
    }


    public function getWeb(): int
```

Ukázka

Proč usilovat o DDD

- Vytvoření modelu který odráží realitu fungováni firmy
- Zlepšení pochopení jak firma funguje
- Modelováni na základě zpětné vazby doménových expertů
- Vydefinováni jasných hranic co je a co není důležité
- Společný jazyk napříč firmou

Doporučené zdroje

- Domain-Driven Design in PHP
- Implementing Domain-Driven Design
- Matthias Noback - Hexagonal Architecture - Message-Oriented Software Design
- Domain-Driven Design: Tackling Complexity in the Heart of Software

Otázky