Orientação a Objetos com Java

Domain Driven Design (DDD)

Gustavo de Miranda Gonçalves
gustavo.miranda@prof.infnet.edu.br

Domain-Driven Design (DDD)

Definição de Domínio

"A sphere of knowledge or activity."

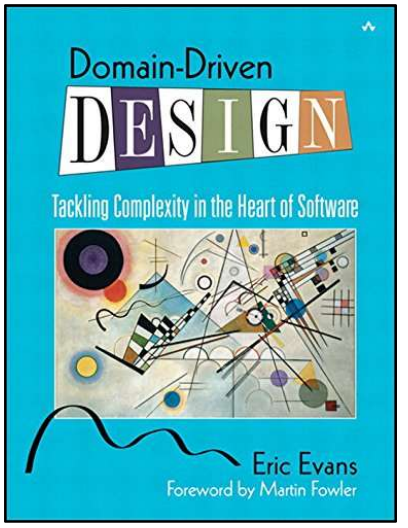
"Sphere of knowledge and activity around which the application logic revolves."

Origem do DDD

Domain Driven Design

- Projeto Orientado a Domínio.
- Proposto por Eric Evans, autor do livro e proprietário da DomainLanguage, empresa de treinamento e consultoria focada em DDD.
- O livro (Domain-Driven Design) é um catálogo de padrões.
- Padrão = Contexto + Problema + Solução

Eric Evans



https://domainlanguage.com/

Orientação a Objetos & Domain Driven Design

- **Código alinhado ao negócio**
  Contato dos desenvolvedores com os especialistas do domínio.
- **Reutilização**
- **Decoplamento**
- **Independência da Tecnologia**

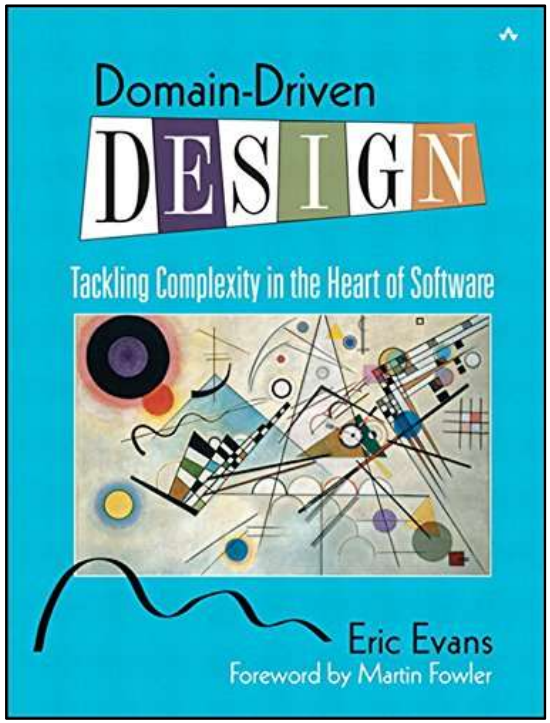
Domain-Driven Design

Part I - **Putting the Domain Model to Work**

Part II - **The Building Blocks of a Model-Driven Design**

Part III - **Refactoring Toward Deeper Insight**

Part IV - **Strategic Design**



Domain-Driven Design (DDD)

Part I - Putting the Domain Model to Work

(Colocando o Modelo de Domínio para Funcionar)

Ubiquitous Language
(Linguagem Ubíqua)

“Nossas notícias são compostas por um título, um conteúdo e às vezes possui uma foto.”

Ubiquitous Language
(Linguagem Ubíqua)

"Nossas **notícias** são compostas por um **título**, um **conteúdo** e às vezes possui uma **foto**."

Ubiquitous Language
(Linguagem Ubíqua)

"Nossas **notícias** são compostas por um **título**, um **conteúdo** e às vezes possui uma **foto**."

"As notícias são publicadas diariamente e precisamos nos lembrar da data de expiração."

Ubiquitous Language
(Linguagem Ubíqua)

"Nossas **notícias** são compostas por um **título**, um **conteúdo** e às vezes possui uma **foto**."

"As **notícias** são **publicadas** diariamente e precisamos nos lembrar da **data de expiração**."

Ubiquitous Language
(Linguagem Ubíqua)

"Nossas **notícias** são compostas por um **título**, um **conteúdo** e às vezes possui uma **foto**."

"As **notícias** são **publicadas** diariamente e precisamos nos lembrar da **data de expiração**."

"Geralmente, a data de expiração é o dia seguinte mas algumas notícias podem ser fixadas por mais de uma semana."

Ubiquitous Language
(Linguagem Ubíqua)

"Nossas **notícias** são compostas por um **título**, um **conteúdo** e às vezes possui uma **foto**."

"As **notícias** são **publicadas** diariamente e precisamos nos lembrar da **data de expiração**."

"Geralmente, a **data de expiração** é o dia seguinte mas algumas **notícias** podem ser fixadas por mais de uma semana."

Linguagem Ubíqua

Entidades

- **Notícia**
- **Foto**

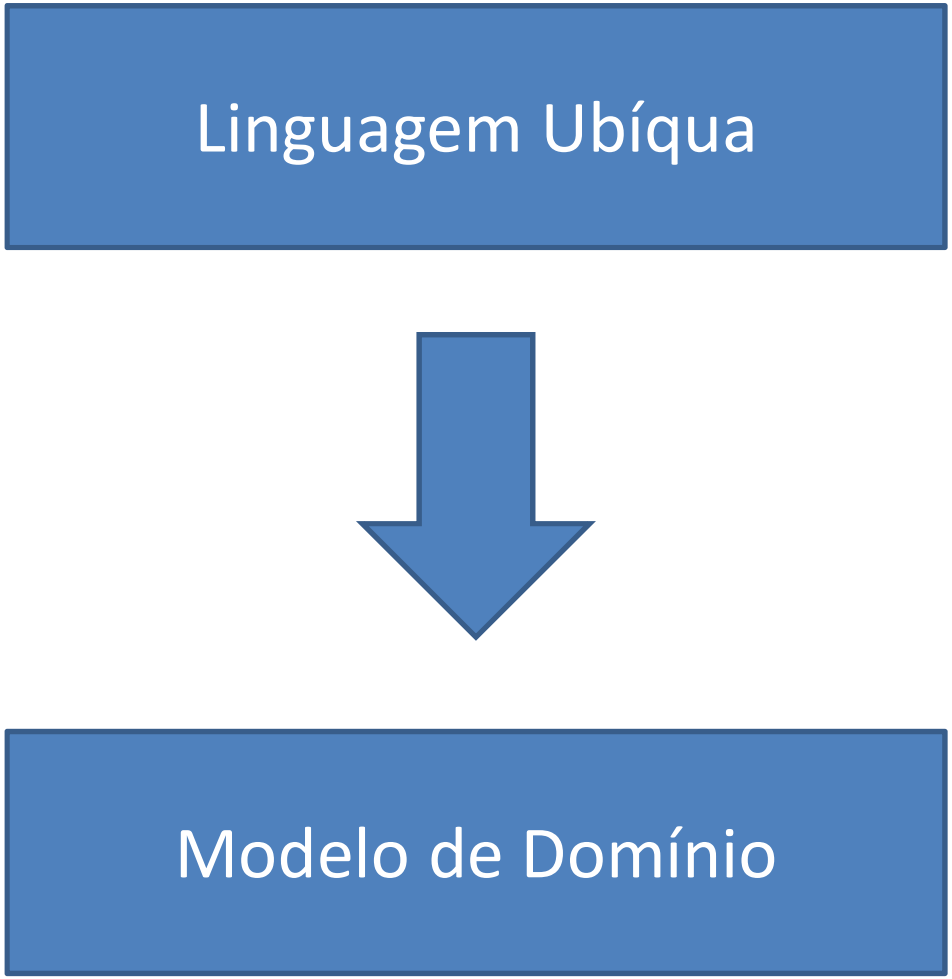
Atributos

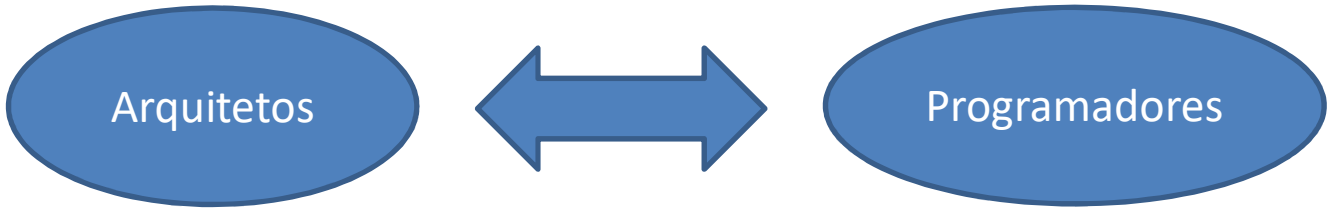
- **Título**
- **Conteúdo**
- **Data Expiração**
- Fonte/Autor da Foto

Serviços

- **Publicação**

Ambiguidades **não** são permitidas.


Linguagem Ubíqua
Modelo de Domínio



Processo de maturação de um sistema desenvolvido usando Modelo de Domínio deve ser **contínuo**.

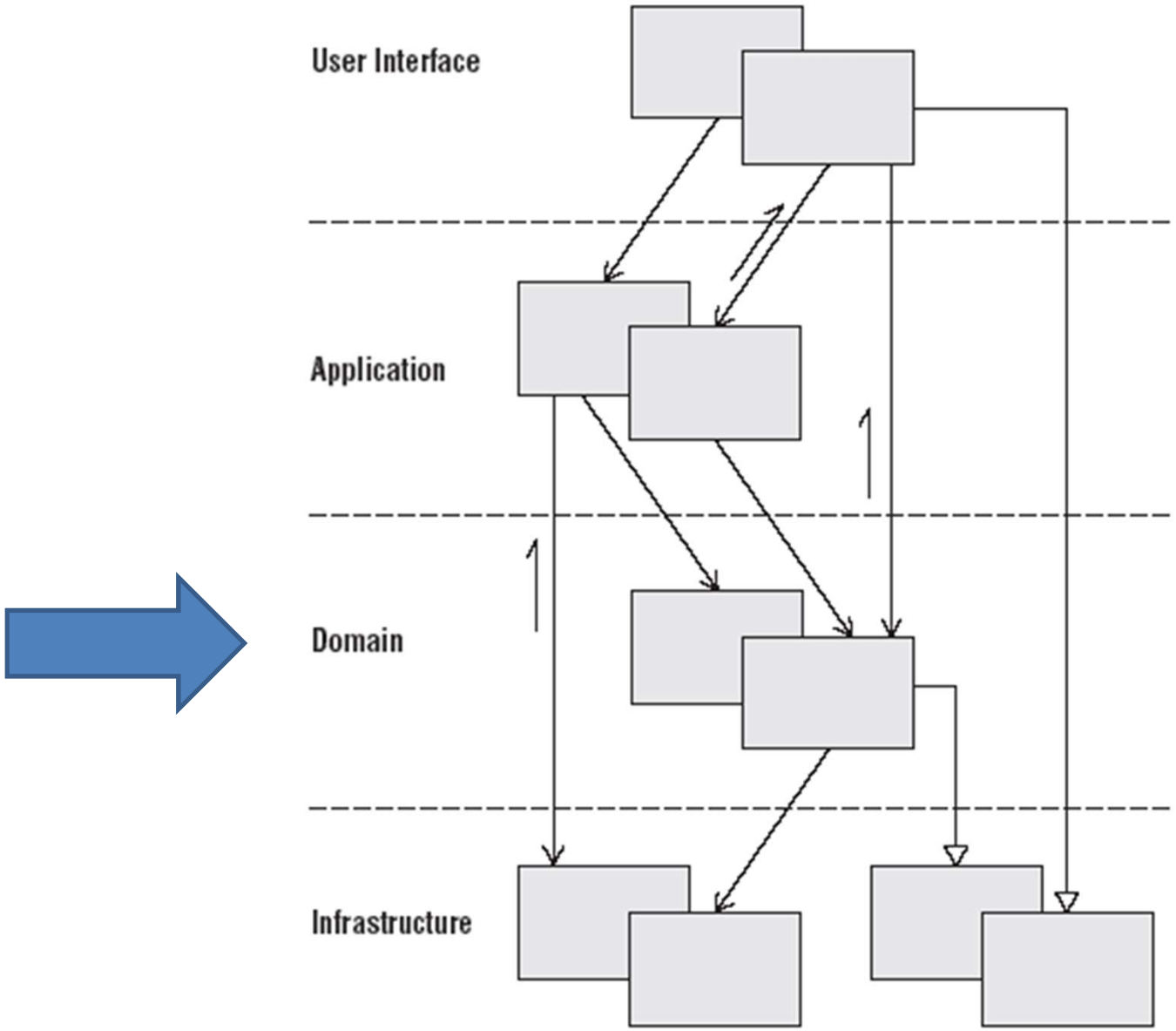
Refatoração deve ser feita **no código e no modelo**.

Domain-Driven Design (DDD)

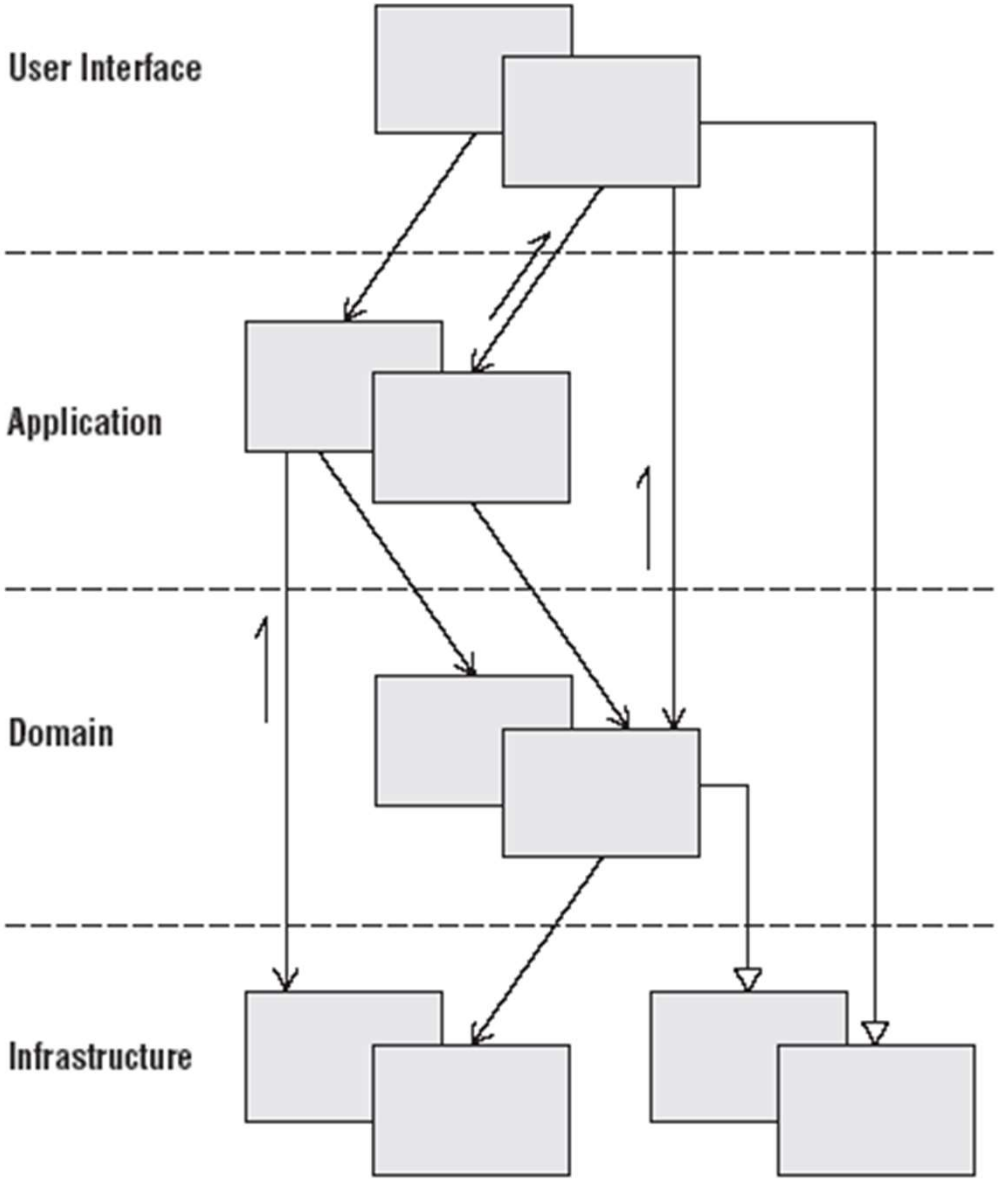
Part II *- The Building Blocks of a Model-Driven Design*
(Blocos de Construção do Model-Driven Design)

Part II - **The Building Blocks of a Model-Driven Design**
(Blocos de Construção do Model-Driven Design (MDD)

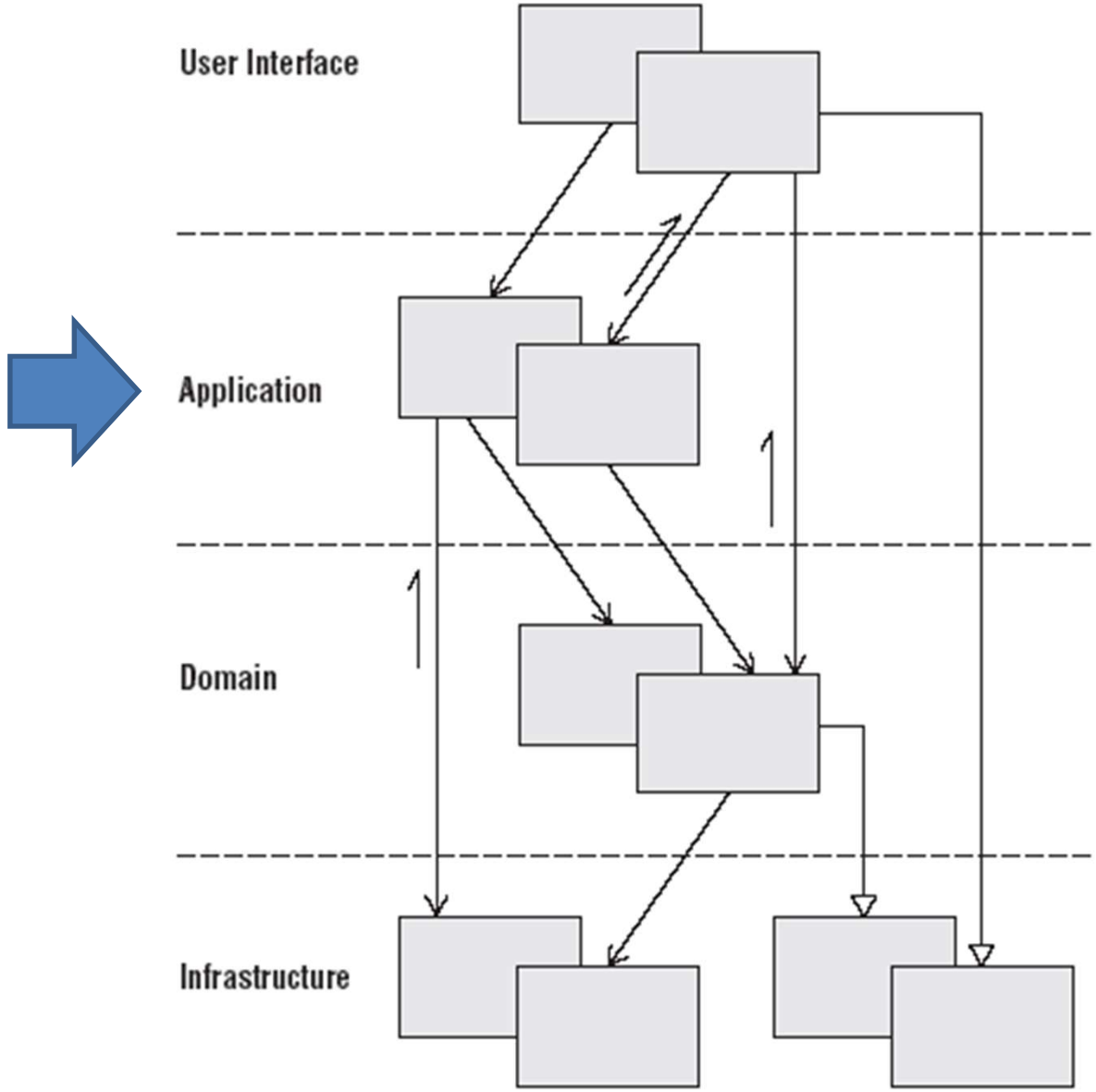
Separar o Modelo de Domínio


User Interface
Application
Domain
Infrastructure

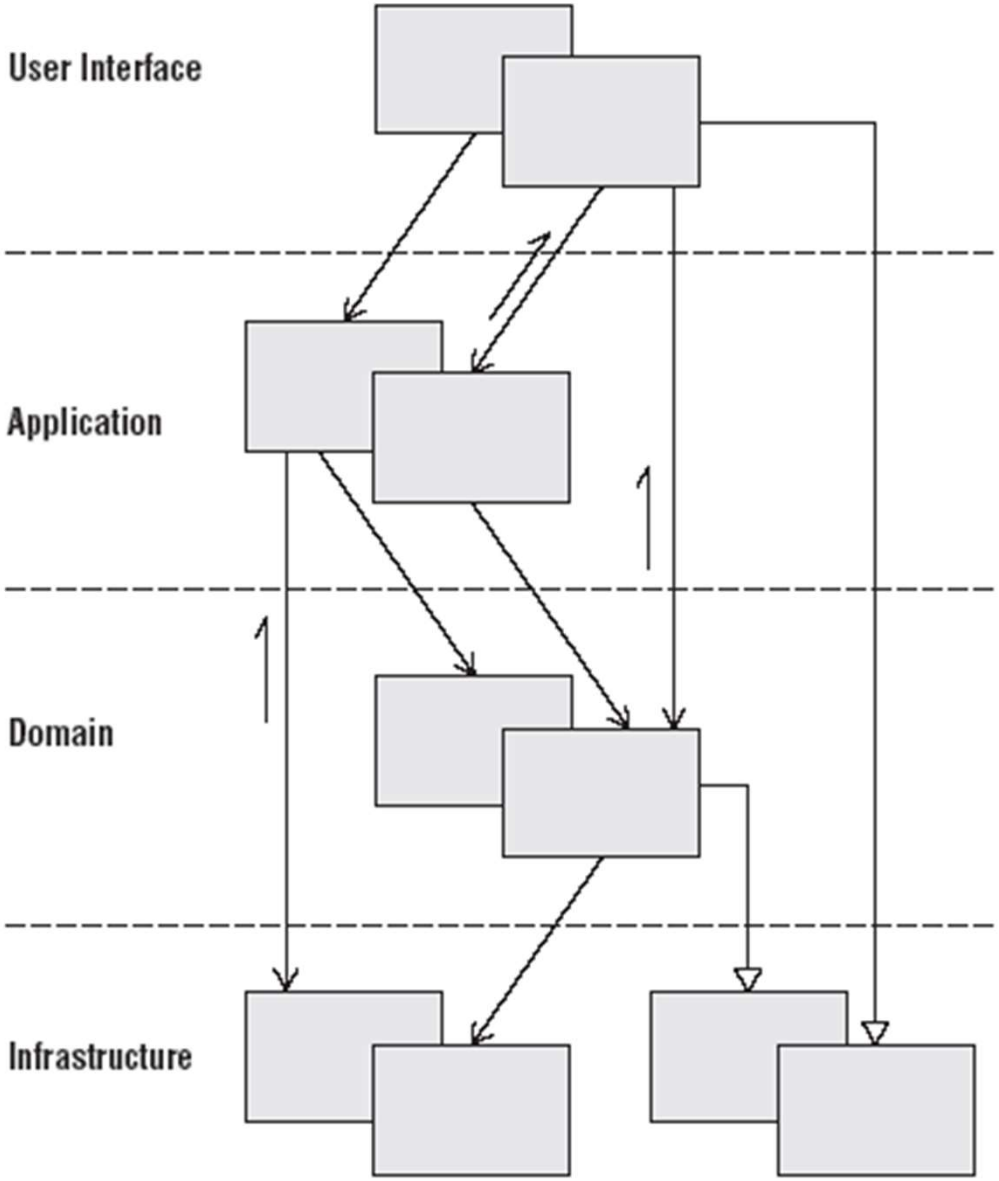
Presentation Layer
(Camada de Apresentação)
Exibir informações ao usuário
e interpretar comandos.

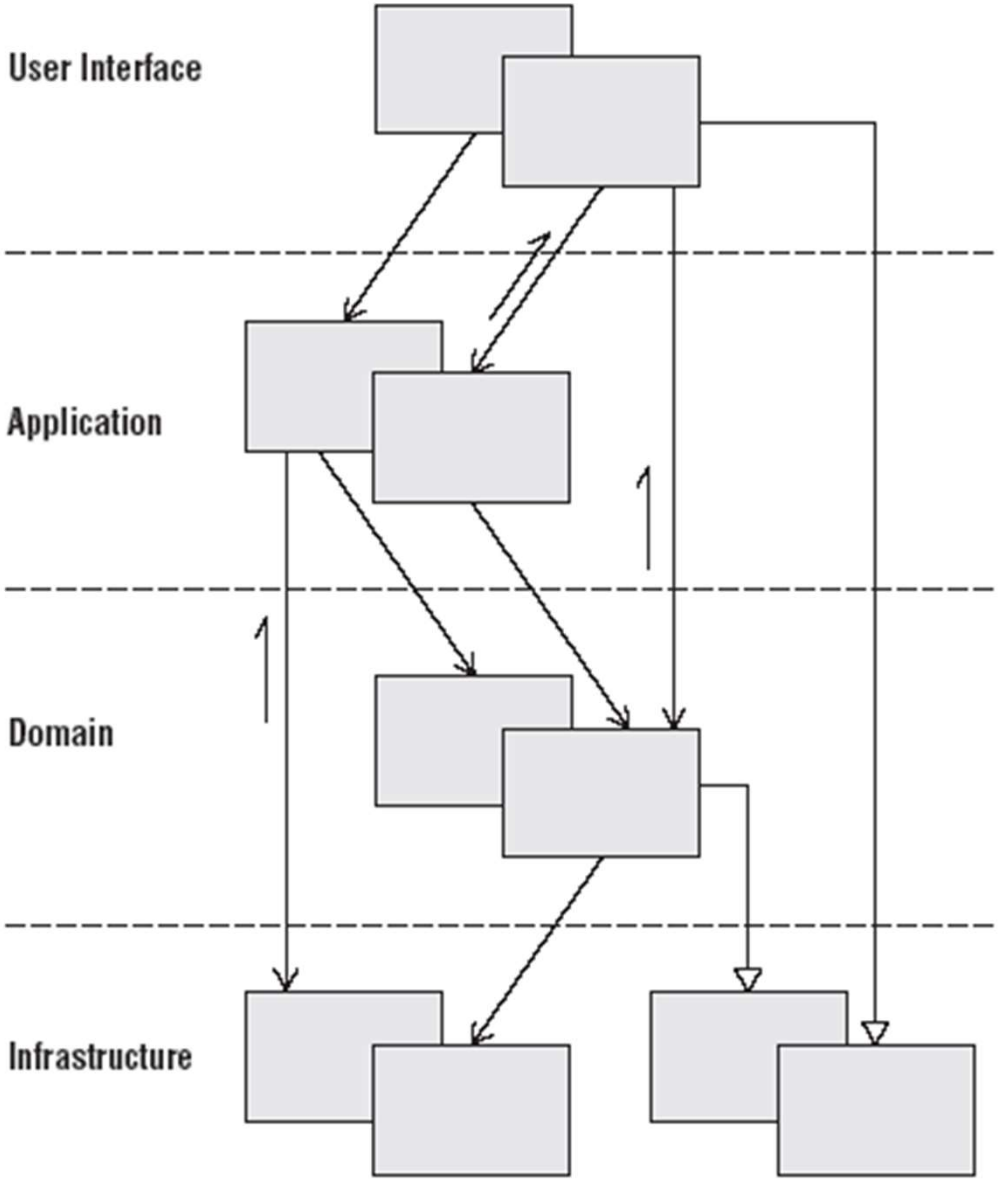


Application Layer
(Camada de Aplicação)
Não possui lógica do negócio.

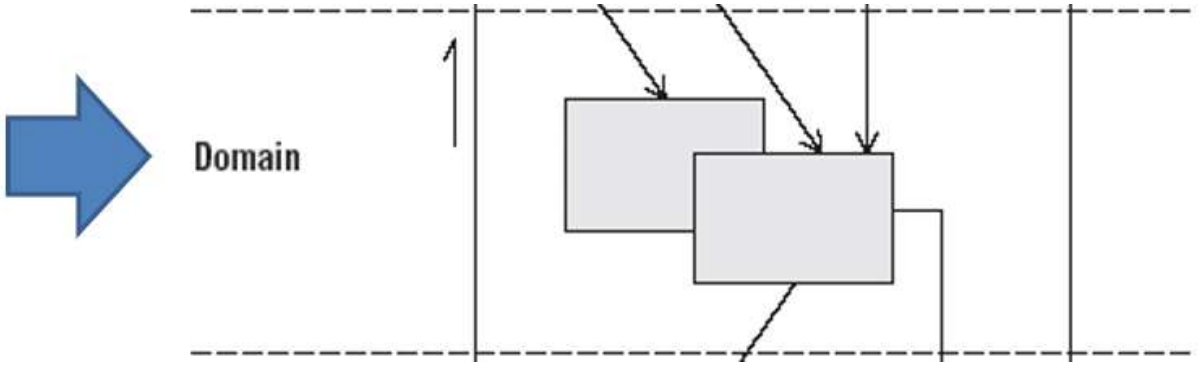


Domain Layer
(Camada de Domínio)
Conceitos, regras e lógicas de negócio. Foco do DDD.





Infrastructure Layer
(Camada de Infra-estrutura)
Recursos técnicos que dão suporte as outras camadas.



O domínio será composto por **blocos de construção** (padrões propostos pelo DDD):

- Entidades
- Objetos de Valor
- Agregados
- Fábricas
- Serviços
- Repositórios

Entidades

Classes de objetos que necessitam de uma identidade.

Objetos de Valor

- São compostos de valores.
- Não possuem distinção de identidade.
- Imutáveis

Exemplos: strings; números; cores; DateTime; Guid.

Agregados

Compostos de Entidades ou Objetos de Valores que são encapsulados numa única classe.

Fábricas

Classes responsáveis pelo processo de criação dos Agregados ou dos Objetos de Valores.

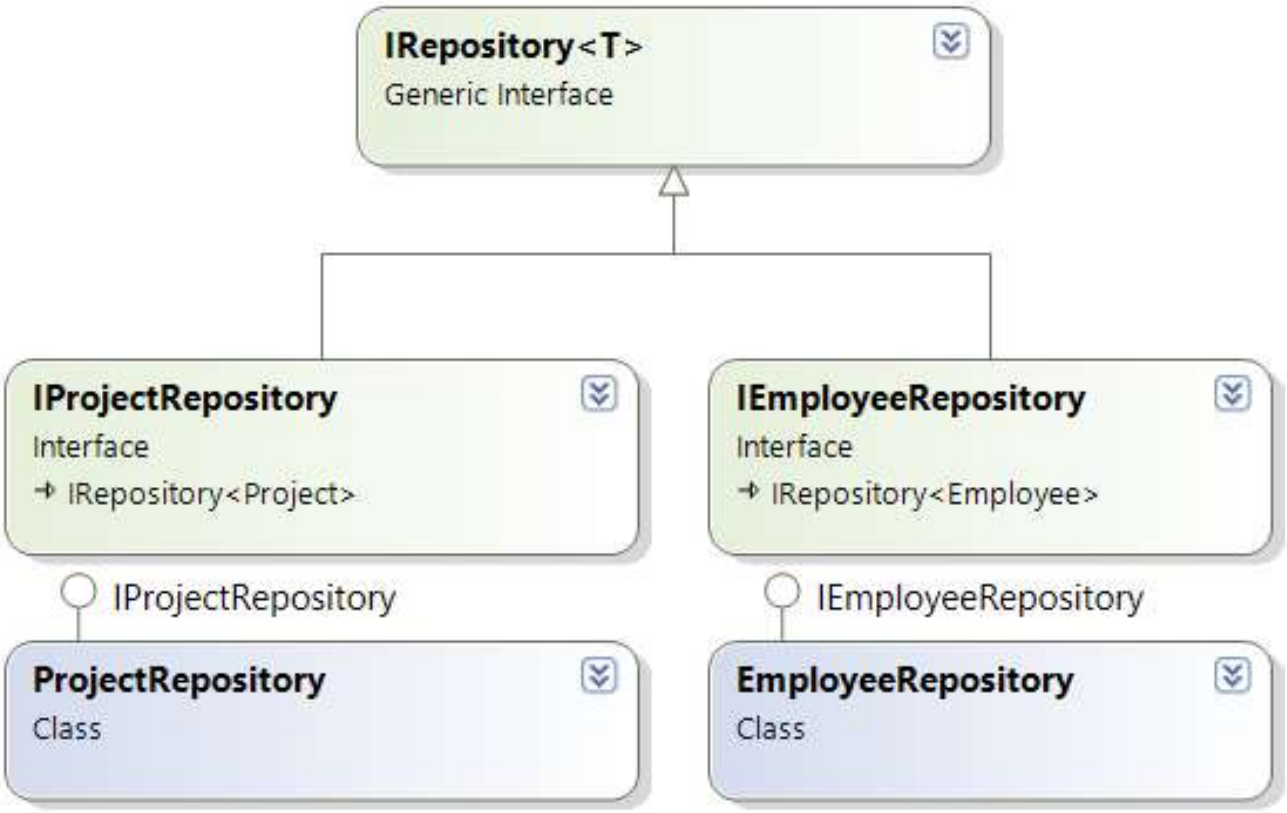
Serviços

- Contém Lógica do Negócio
- Não guardam estado
- Não pertence a nenhuma entidade ou objeto de valor.

Repositórios

- Gerencia o ciclo de vida de Entidades, Objetos de Valor e Agregados.
- Possuem operações de CRUD (**C**reate, **R**ead, **U**pdate e **D**elete).

Repositórios



Domain-Driven Design (DDD)

Part I - Putting the Domain Model to Work

(Colocando o Modelo de Domínio para Funcionar)

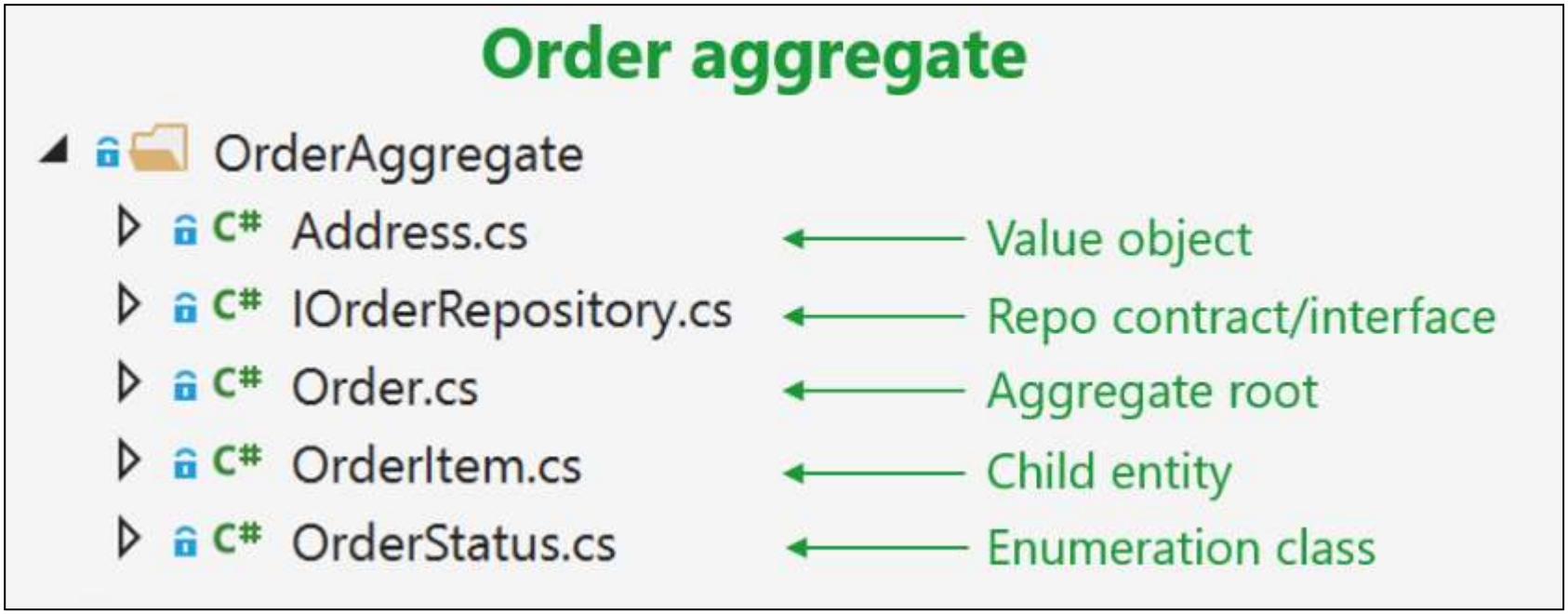
Prática – Exemplo I

Domain-Driven Design (DDD)

Part I - Putting the Domain Model to Work

(Colocando o Modelo de Domínio para Funcionar)

Prática – Exemplo II



https://docs.microsoft.com/en-us/dotnet/standard/microservices-architecture/microservice-ddd-cqrs-patterns/net-core-microservice-domain-model

Domain-Driven Design (DDD)

Part III - Refactoring Toward Deeper Insight

(Refatorando para compreender profundamente o modelo)

Intention-Revealing Interfaces

(Interface de Intenção Revelada)

Nomes de métodos ou classes devem deixar claro o que fazem, mas não detalhes de implementação (respeitando o encapsulamento).

Side-Effect-Free Functions

(Funções sem Efeitos Colaterais)

Métodos podem ser divididos em duas categorias:

- Commands (Comandos)
- Queries (Consultas)

Separar operações de consulta (Queries) de operações de comandos (Comandos).

Usar objetos de valor quando possível para evitar modificar o domínio de objetos (operações com objetos de valor costumam criar novos objetos de valor).

Assertions

(Asserções)

Para ajudar os desenvolvedores a entender os efeitos dos comandos:

- Intention-Revealing Interfaces
- Asserções

Criar testes unitários para os comandos que alterem estados e/ou validar as chamadas dos comandos.

Designed by kjpargeter / Freepik