Uma introdução a

Domain Driven Design

Daniel Cukier
danicuki@ime.usp.br

IME-USP

Padrões

Um padrão é uma regra de três partes que expressa a relação entre um certo contexto[1], um problema[2] e uma solução[3]

O que é DDD

- Livro de Eric Evans (2004)
- Padrões
- Boas práticas
- Experiência

Foco: Domínio

- Alinhamento com negócio
- Isolamento entre domínios
- Reutilização
- Mínimo acoplamento
- Independente de tecnologia

DDD

A volta da Orientação a Objetos?

Padrões no DDD

[Contexto.]

[Discussão do problema.]

Resumo do problema.

Discussão da solução.

Por esta razão:
Resumo da solução.

Conseqüências, implementação, exemplos.

[Contexto resultante.]

DDD

Colocando o modelo de domínio para funcionar

Blocos de construção do MDD

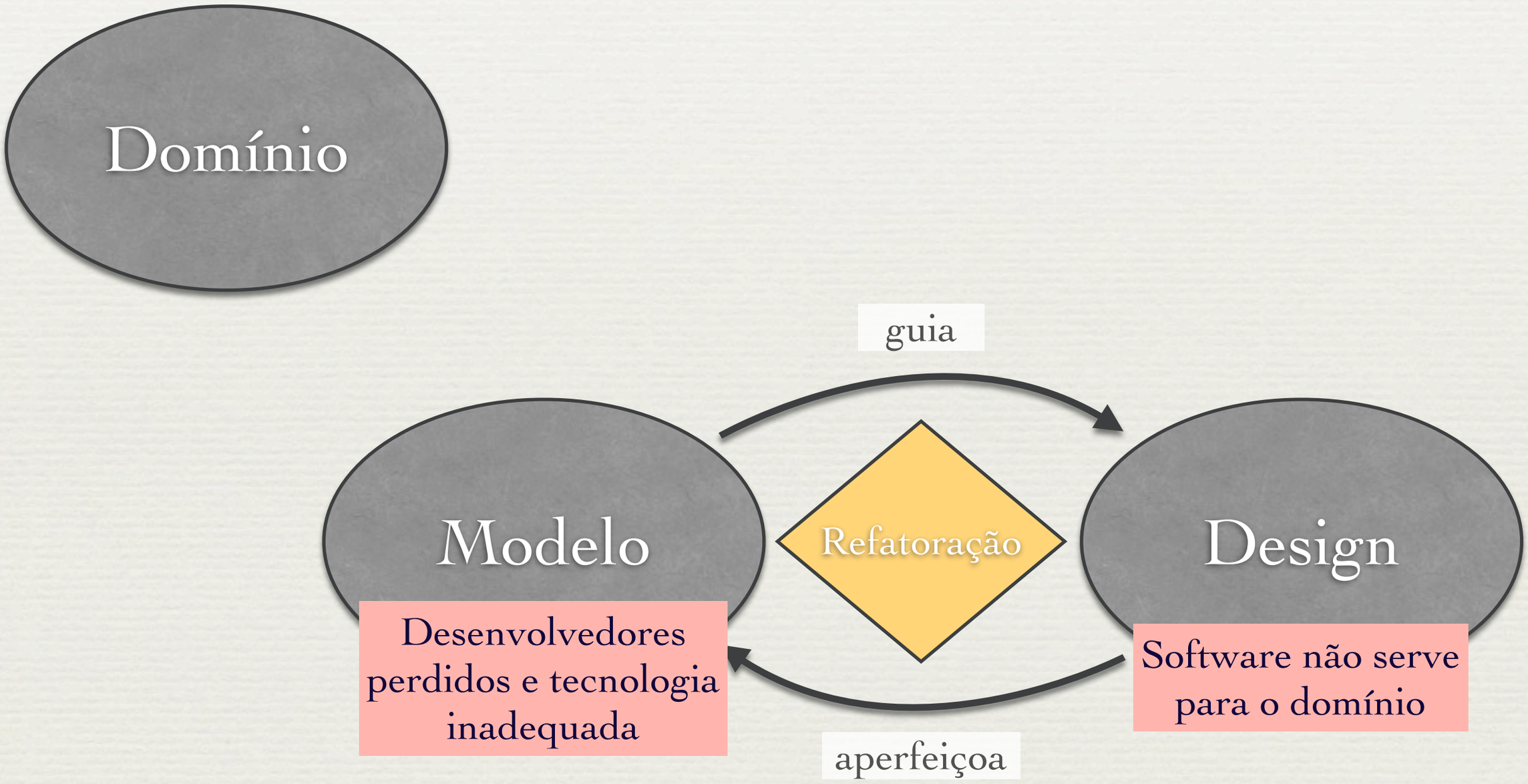
Refatorando para compreender profundamente o modelo

Projeto estratégico

Colocando o modelo de domínio para funcionar

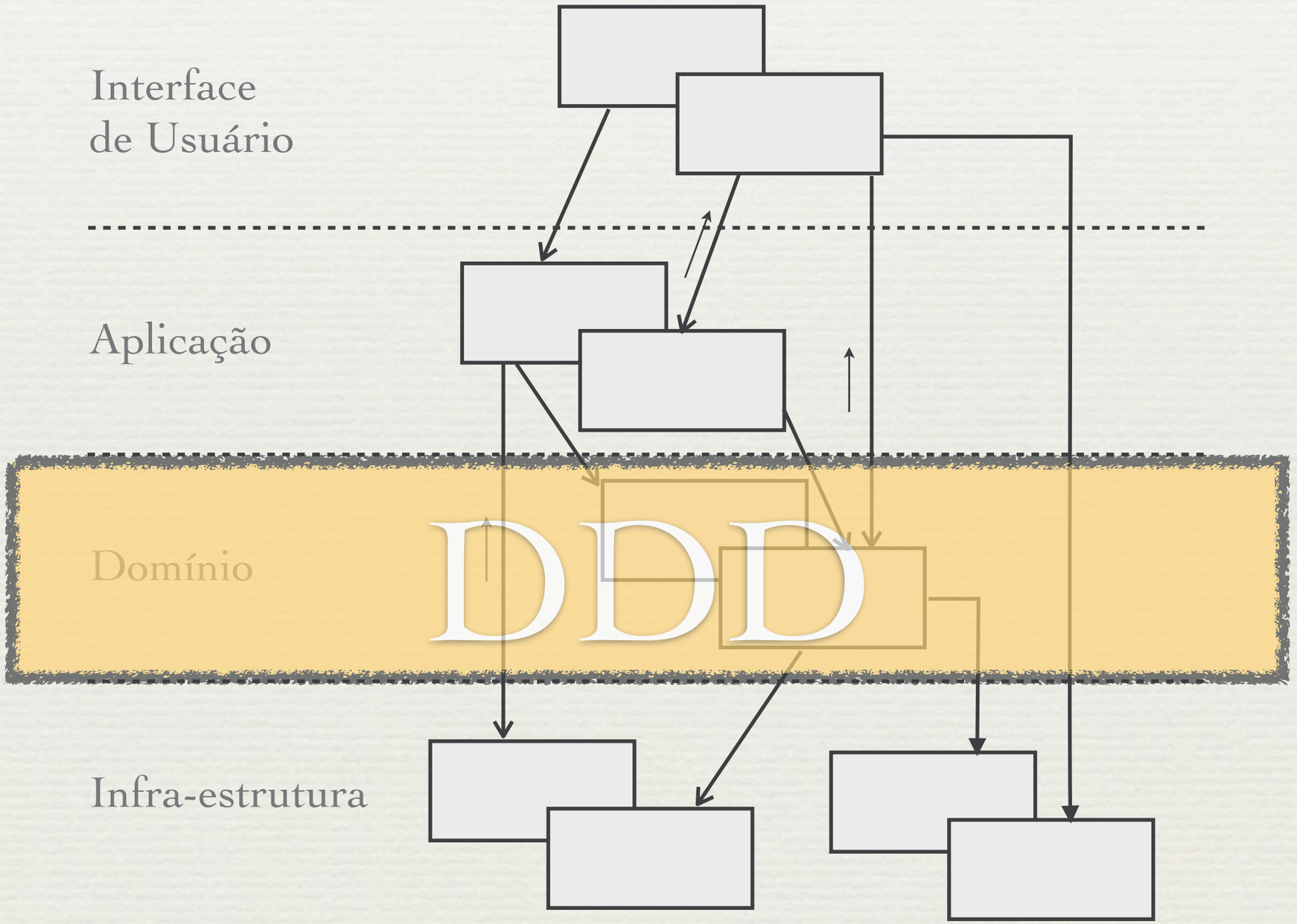
Linguagem Ubíqua

Model Driven Design



Blocos de Construção do Model Driven Design

Arquitetura em Camadas

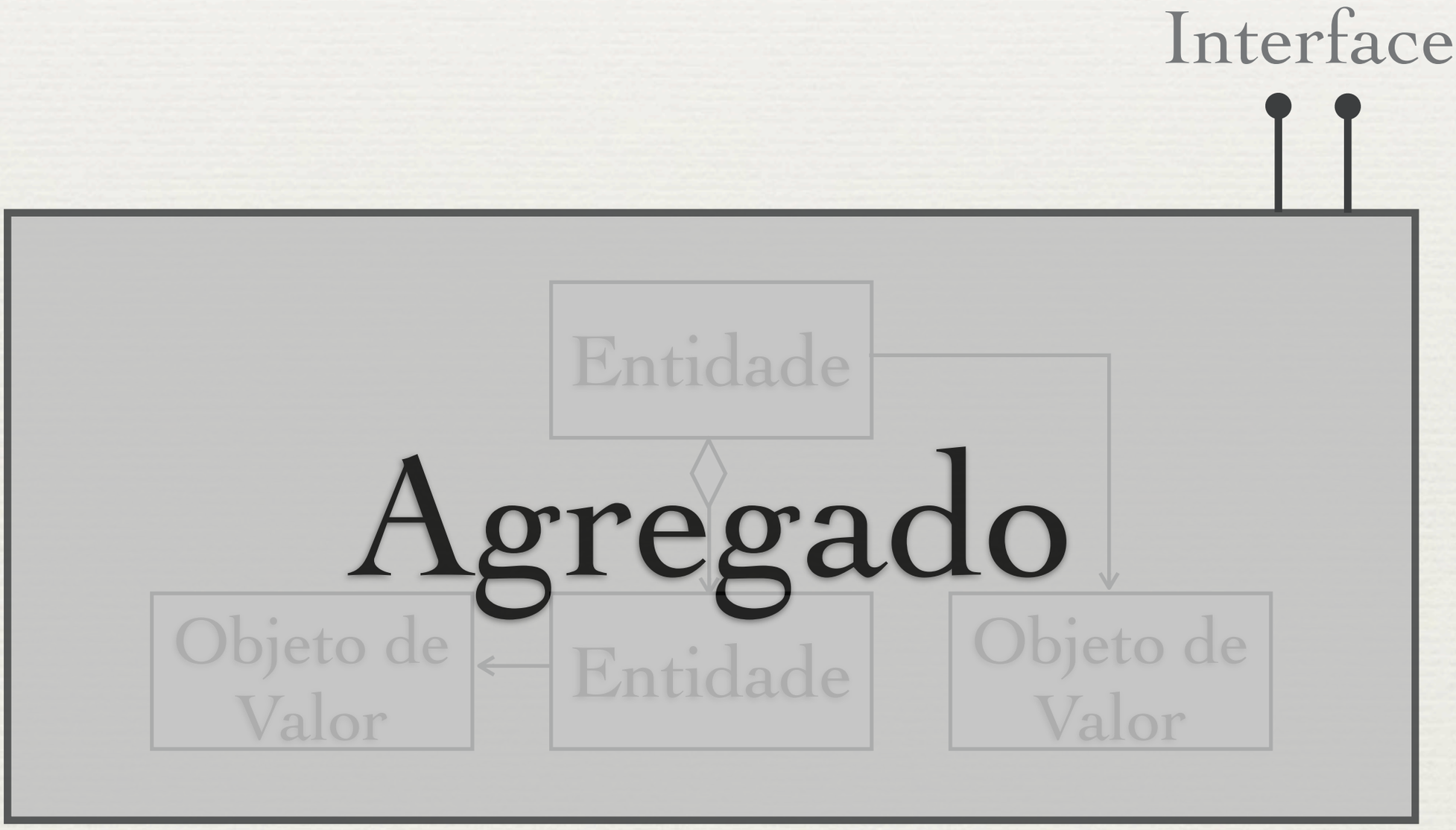


Entidades

Objetos de Valores

- Pra quem não precisa de identidade
- Imutáveis
- Descrevem coisas
- Ciclo de vida rápido
- Exemplos: Strings, números, cores

Agregados


Interface
Entidade
Agregado
Objeto de Valor
Entidade
Objeto de Valor

Fábricas

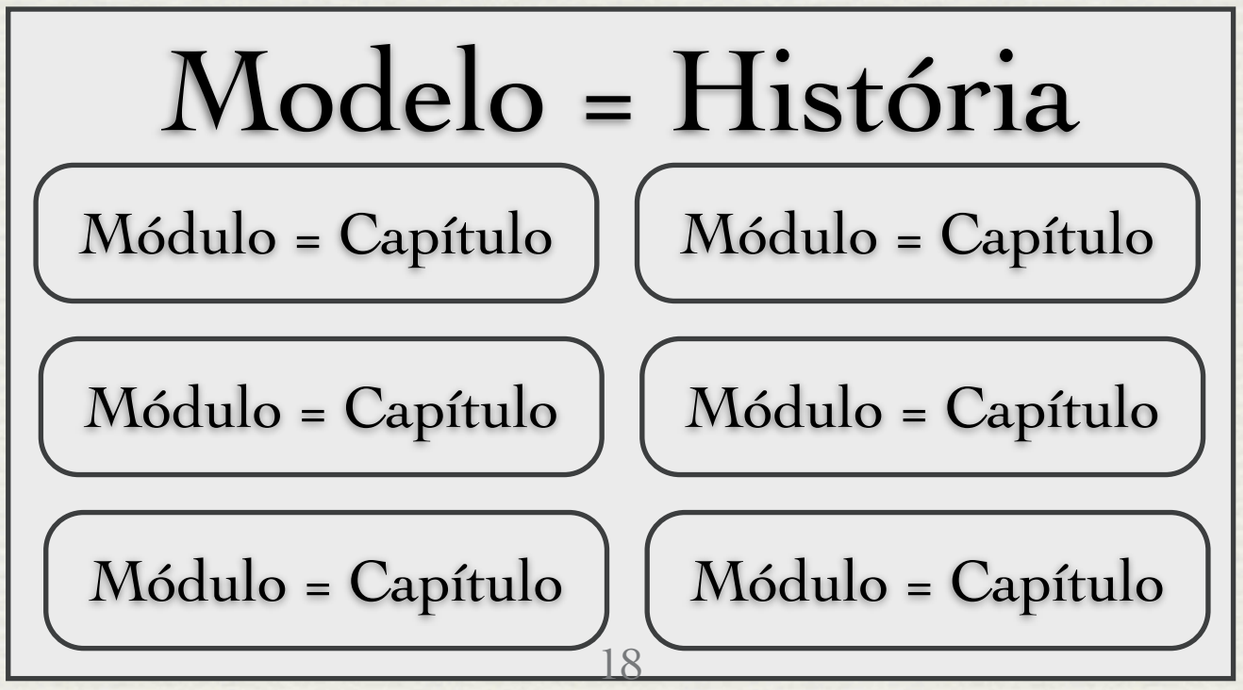
- Quando construir (Agregados) for complexo
- Devem ser sempre abstratas
- Não fazem parte do modelo, mas do domínio

Serviços

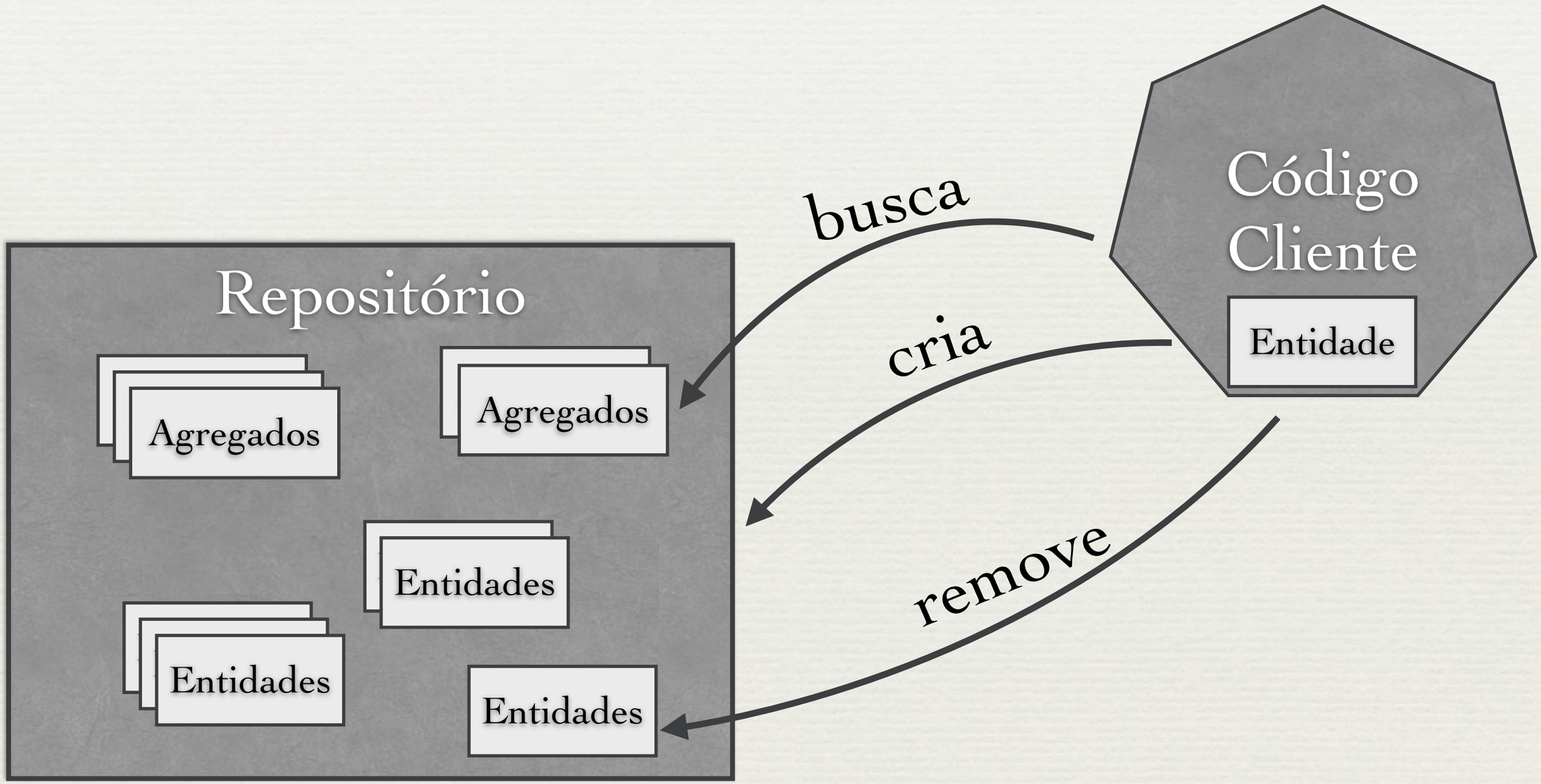
1. A operação não faz parte de nenhuma Entidade ou Objeto de Valor
2. Interface fala a língua do Domínio
3. Sem estado

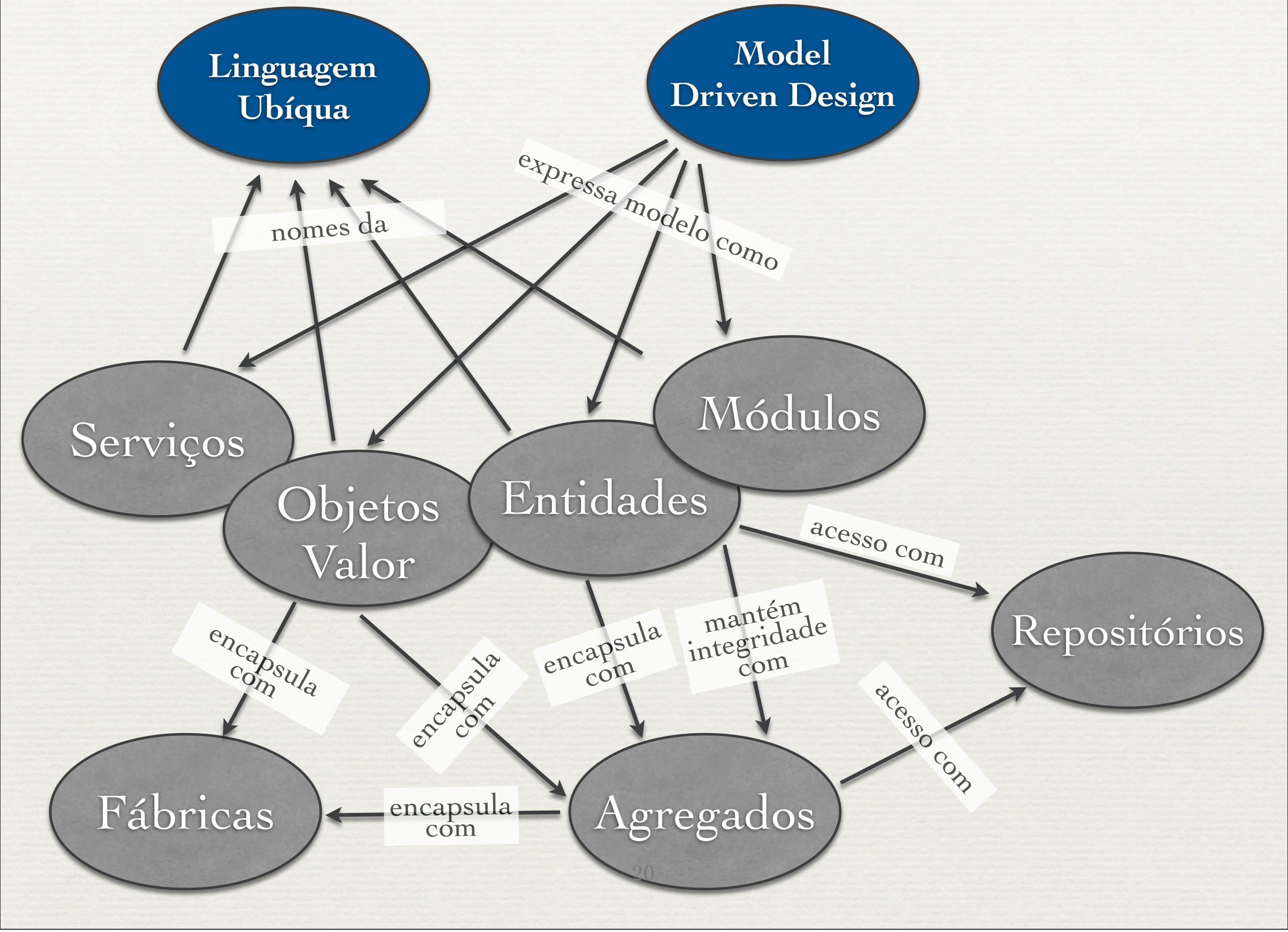
Módulos

- Agrupe conceitos do modelo, não código
- Baixo acoplamento / alta coesão
- Nomes da Linguagem Ubíqua



Repositórios

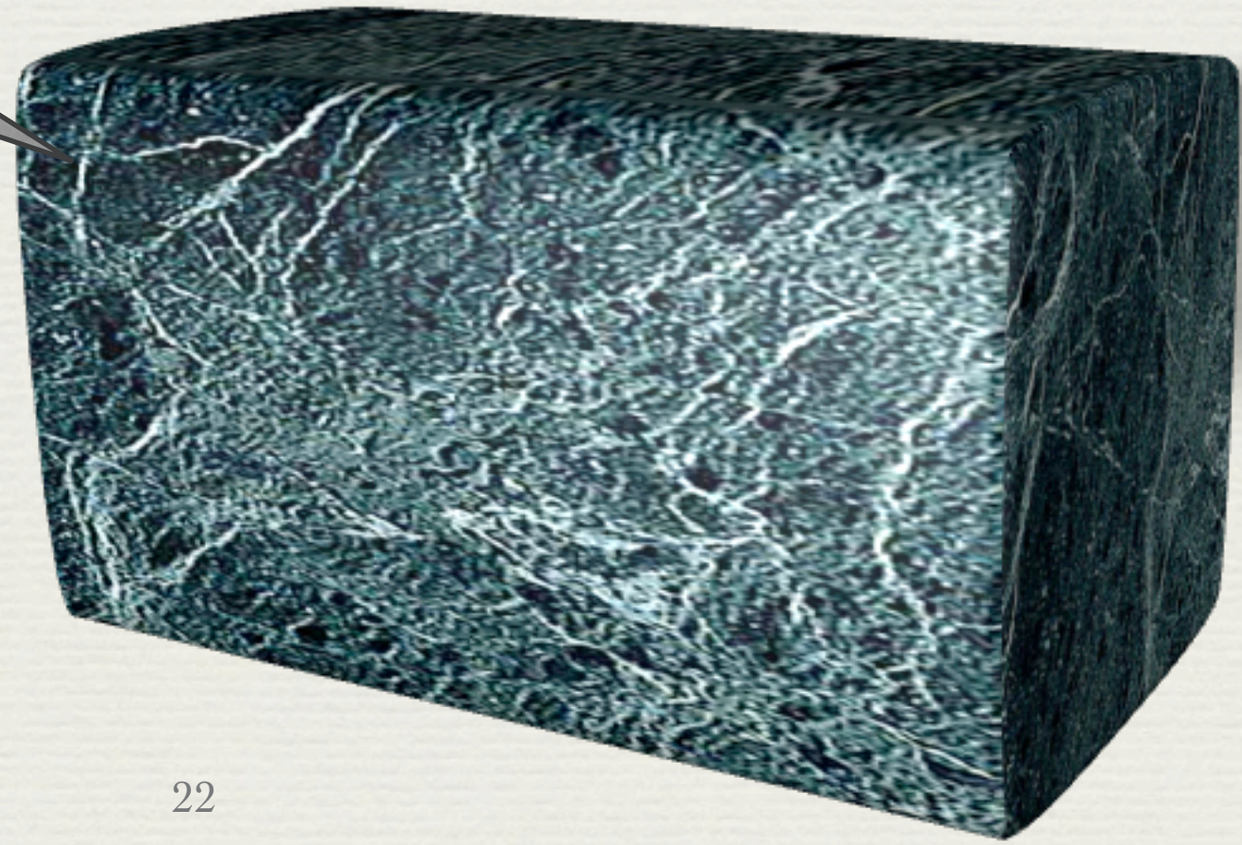



Linguagem Ubíqua
Model Driven Design
nomes da
expressa modelo como
Serviços
Objetos Valor
Entidades
Módulos
acesso com
Repositórios
encapsula com
encapsula com
encapsula com
mantém integridade com
acesso com
Fábricas
encapsula com
Agregados

Refatorando para compreender profundamente o modelo

Interfaces de Intenção Revelada

Eu faço exatamente isso. Não precisa se preocupar como

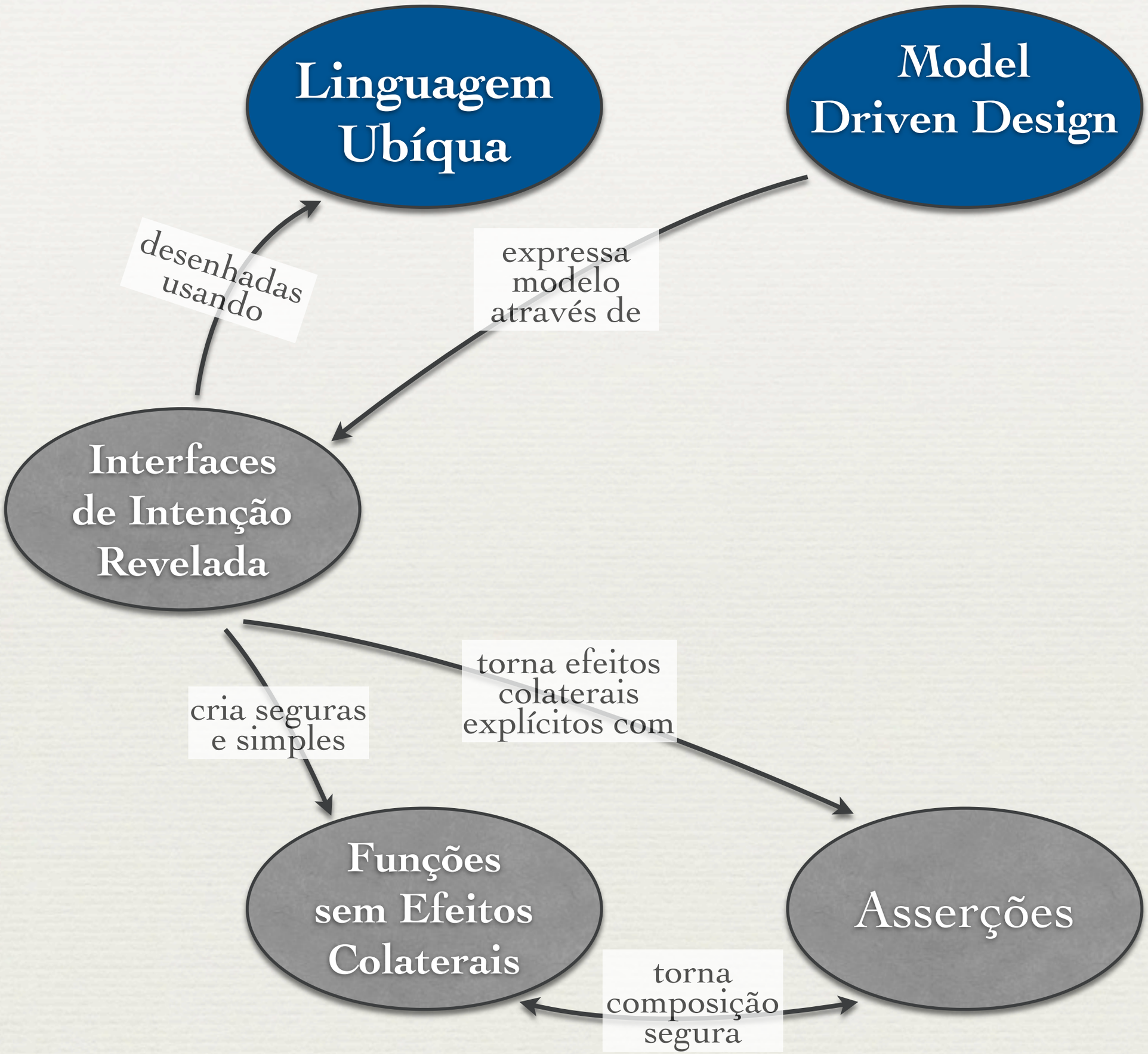


Funções sem Efeitos Colaterais

- Colocar tudo que for possível em funções, principalmente em cálculos complexos
- Onde não der, usar **Comandos** que fazem poucas operações e não retornam objetos do domínio

Asserções

- Testes de unidade
- Usar facilidades da linguagem
- Testam o comportamento dos **Comandos**


Linguagem Ubíqua
Model Driven Design
desenhadas usando
expressa modelo através de
Interfaces de Intenção Revelada
cria seguras e simples
torna efeitos colaterais explícitos com
Funções sem Efeitos Colaterais
Asserções
torna composição segura

Projeto Estratégico

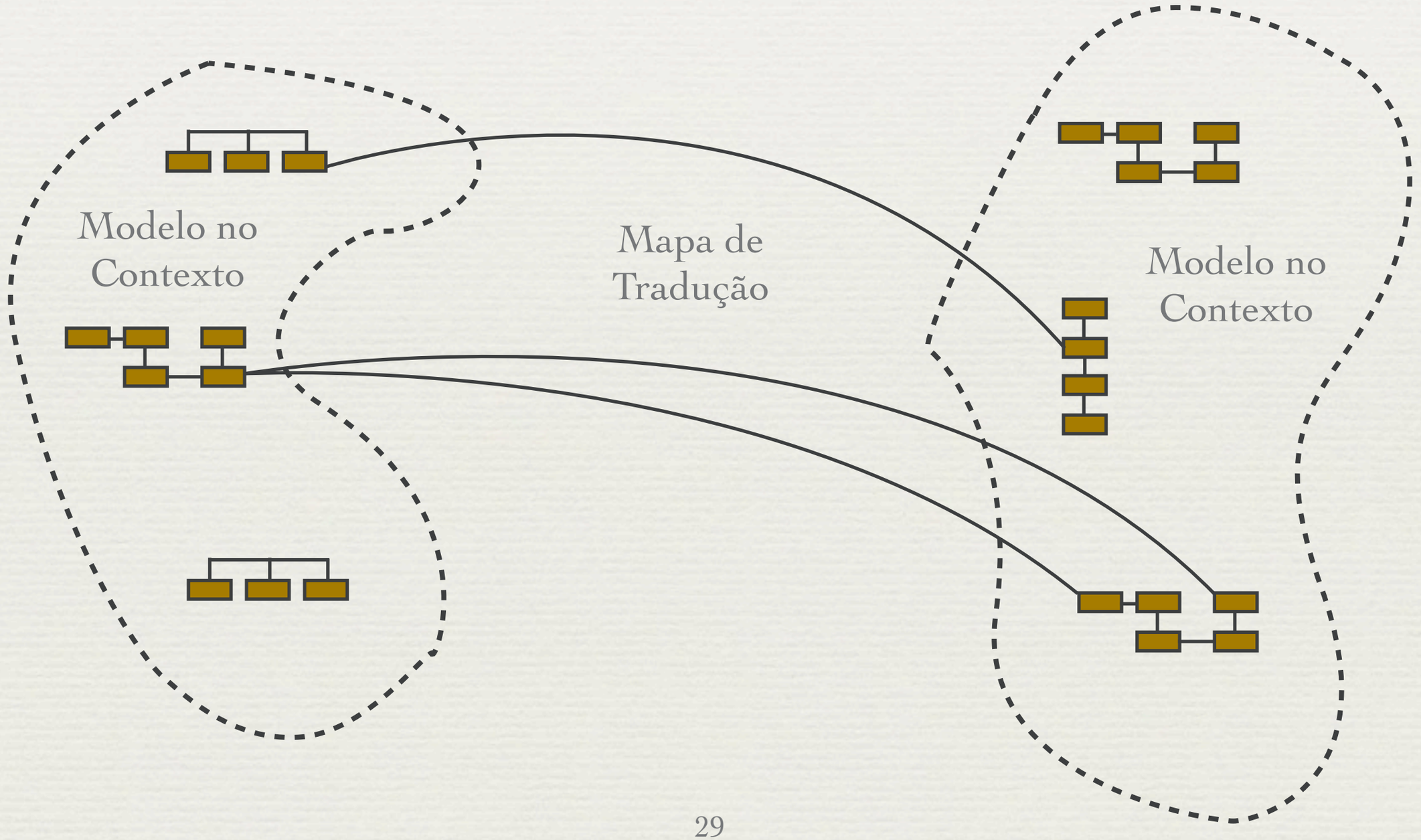
Contexto Delimitado

As células existem porque sua membrana define o que está dentro ou fora e determina o que pode entrar

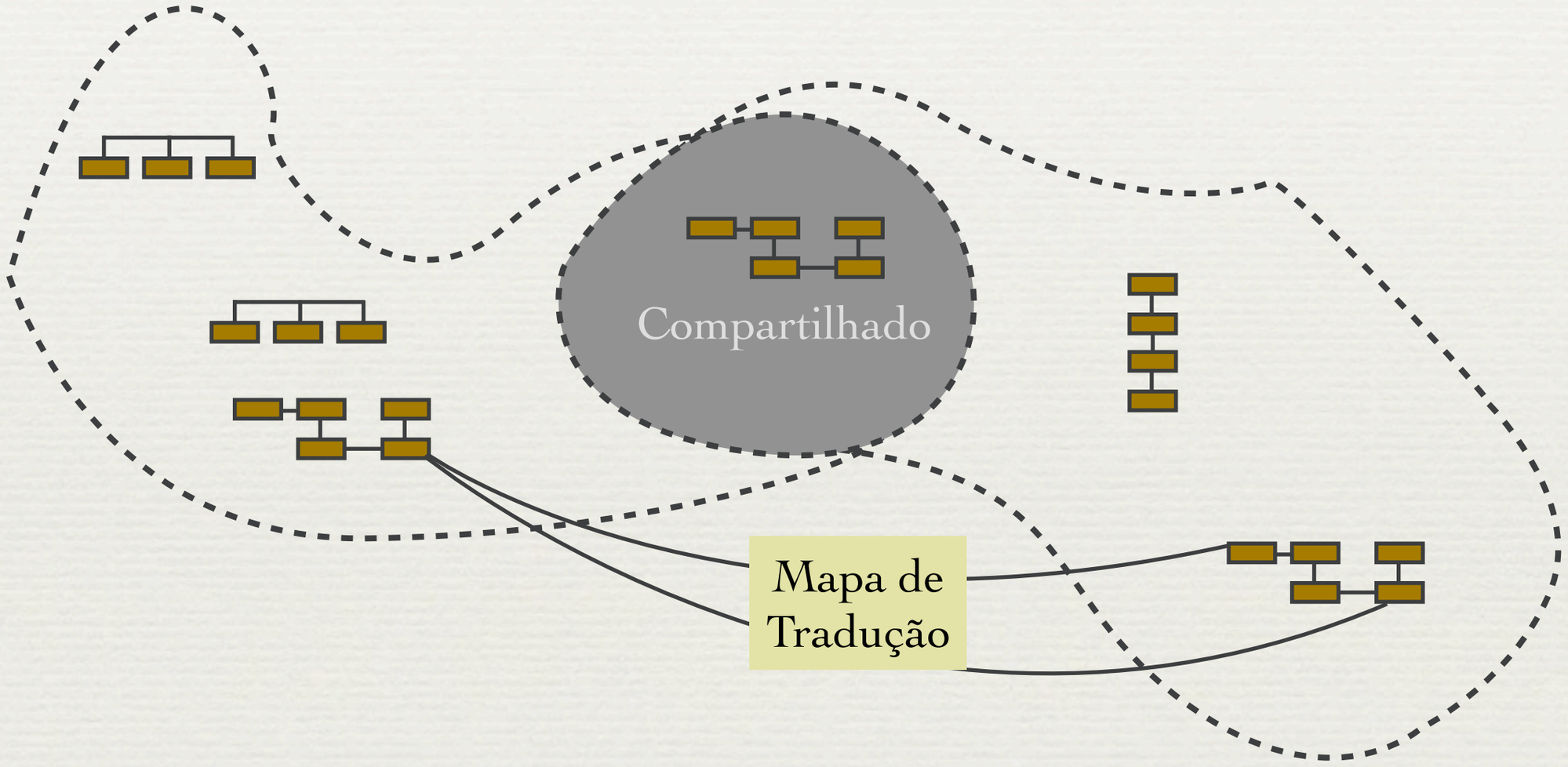
Integração Contínua

- Testes automatizados
- Processo de build automático
- Sincronização diária
- Relatórios

Mapa do Contexto



Núcleo Compartilhado



- É mais difícil fazer mudanças
- Evita (mas não elimina) duplicações

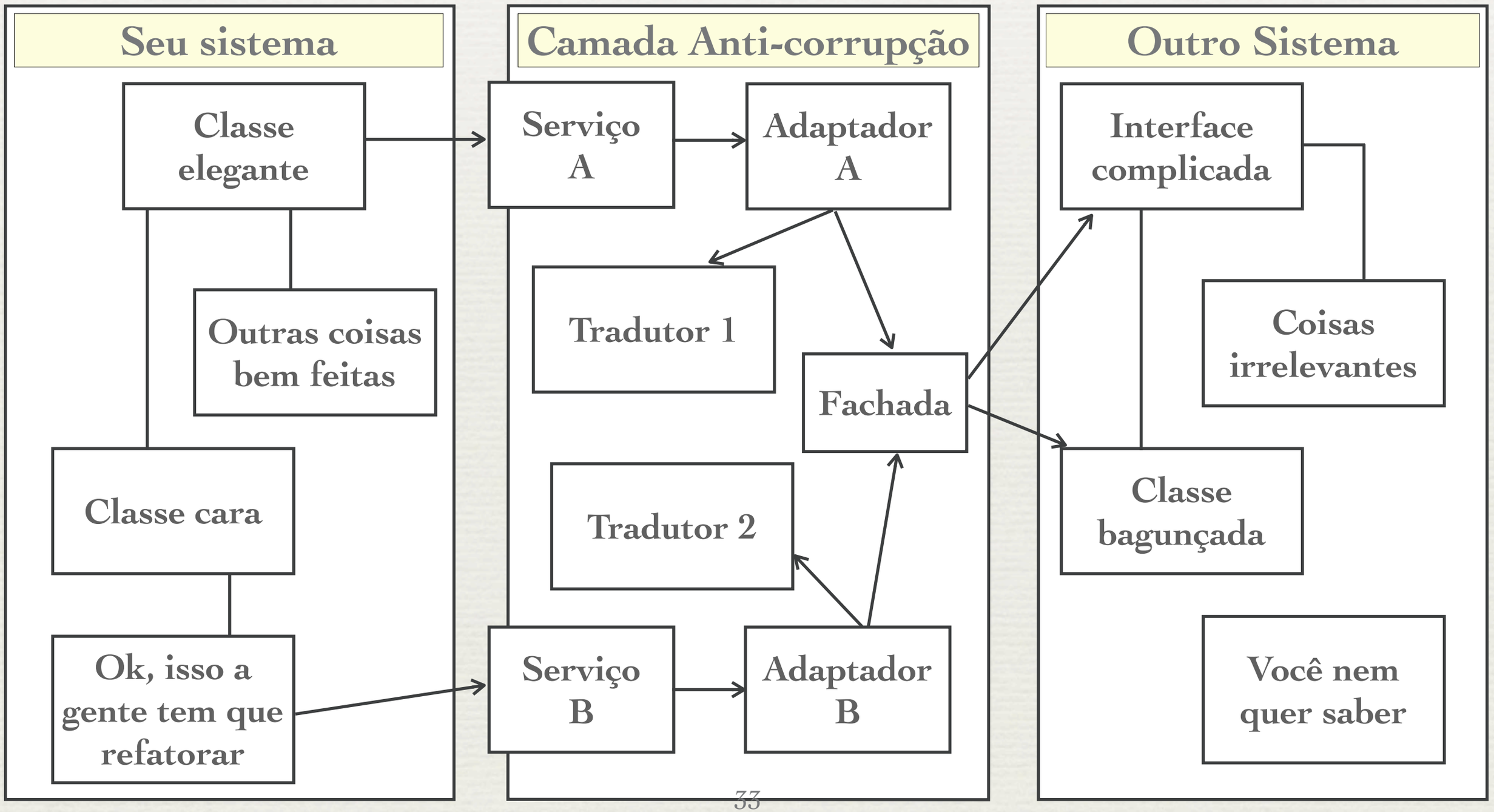
Produtor-Consumidor

- Testes automatizados
- Cliente-Fornecedor

Conformista

- Time fornecedor não tem interesse em ajudar
- Tira complexidade de tradução entre contextos
- Mesma linguagem ubíqua
- Parecido com **Núcleo Compartilhado**, mas cliente não tem poder de modificação

Camada Anti-corrupção

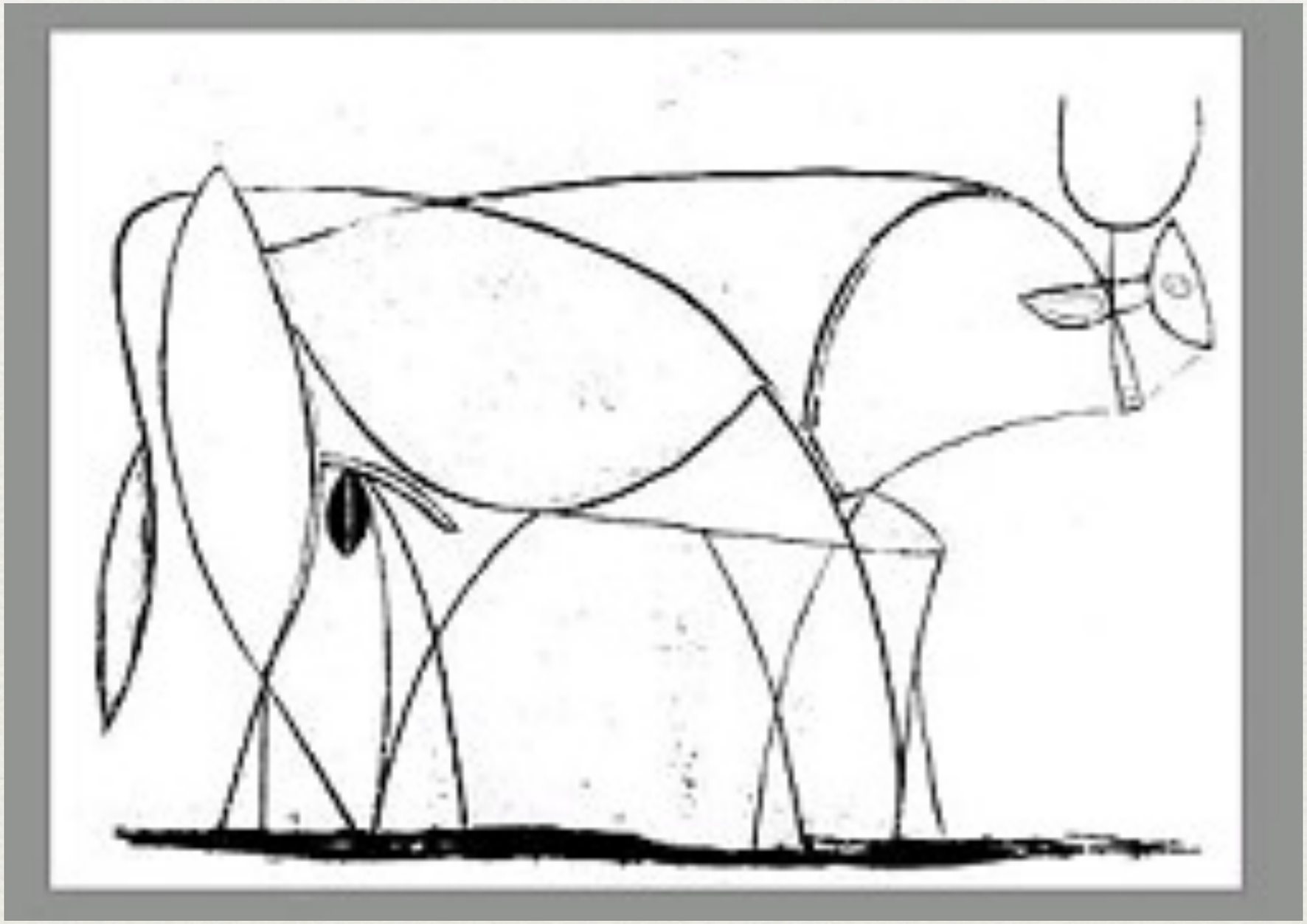


Caminhos Separados

- Quando integrar custa caro e o benefício é pequeno
- Contexto delimitado que não tem nenhuma conexão com os outros

Resumindo

1. Trabalhando com um modelo
2. Blocos de construção
3. Refatorando e evoluindo
4. Refinando, destilando




Conformista
Núcleo Compar-tilhado
Serviço de Hospedagem Aberto
Linguagem Publicada
Contexto Delimitado
Integração Contínua
Camada Anti-corrupção
Contornos Conceituais
Interfaces de Intenção Revelada
Linguagem Ubíqua
Model Driven Design
Arquitetura em Camadas
Mapa do Contexto
Caminhos Separados
Fechamento de Operações
Funções sem Efeitos Colaterais
Núcleo Abstrato
Núcleo Segregado
Entidades
Objetos de Valores
Classes Isoladas
Asserções
Subdomínios Genéricos
Núcleo do Domínio
Mecanismos Coesivos
Serviços
Módulos
Nível de Conheci-mento
Ordem que Evolui
Sentença da Visão do Domínio
Núcleo Destacado
Agregados
Fábricas
Arcabouço Componente Plugável
Camadas de Responsa-bilidade
Metáfora de Sistema
Repositórios

Referências

- Evans, Eric - *Domain Driven Design*, Addison-Wesley - 2004
- http://www.infoq.com/resource/minibooks/domain-driven-design-quickly/en/pdf/DomainDrivenDesignQuicklyOnline.pdf