Camada de Domínio/ Negócio e Dados

Leandro Garcia

Como desenhar a parte de Domínio/ Negócio e Dados

O que é?

- Muitos softwares tem necessidade de armazenamento de dados
- Separar armazenamento de dados das regras de negócio é importante
- Encapsula complexidades para as outras camadas
- Baixo acoplamento

Onde estamos?

- Armazenamento de dados
- Regra de negócio
- Barramento de Serviços
- Interfaces com o usuário

Como se inicia o desenho de um Software?

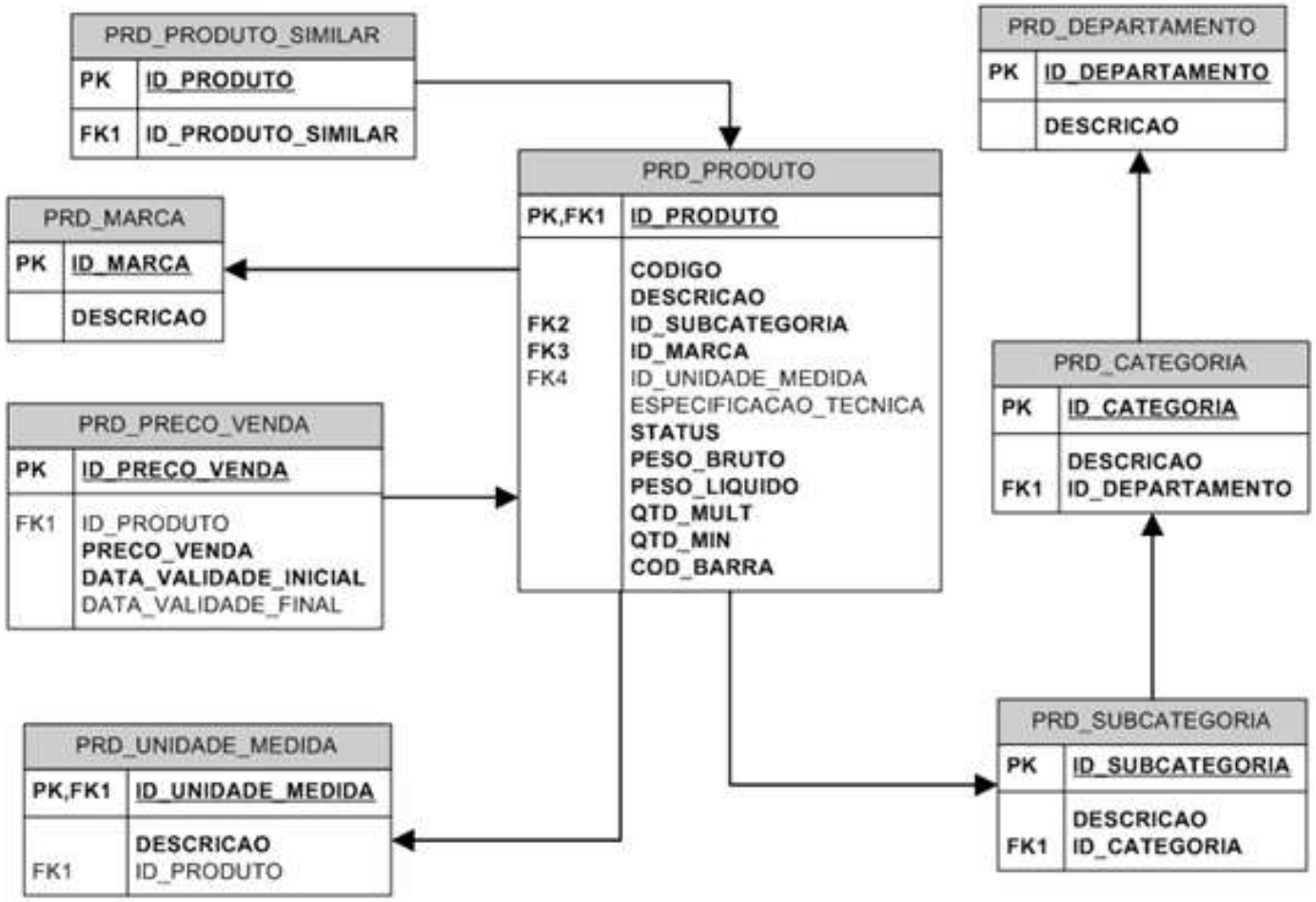
- Após entendimento com cliente
- Após construção de artefatos
- Após prototipagem

Dois tipos de modelo...

- Guiado por Dados
- Guiado por Domínio/Negócio

Programação Orientada a Objetos

Diagrama de Entidade e Relacionamento



Padrões para dados e domínio

Padrão	Tipo	Descrição
Transaction Script	Dados	Foco na ação do usuário
Table Module	Dados	Objetos com representação tabular – DataSet
Active Record	Meio termo	Lógica de domínio simples
Domain Model	Domínio	Domínio independente de dados

Desenvolvimento Guiado a Dados

Desenvolvimento Guiado a Dados

Inicia o desenho do sistema pelo DER (diagrama de entidades e relacionamentos) , pelo banco de dados

Os próximos passos acabam seguindo o modelo de banco: domínio, serviços, etc

Vantagens

- Velocidade de produção
- Facilidade de persistência dos dados
- Recuperação de dados simplificada (linq)

Desvantagens

- Ancorado nos recursos do banco
- Fugindo de recursos valiosos de Orientação a Objetos
- Tendência de aumento de Acoplamento (dependência do banco)
- Controle especial das versões do banco e da aplicação

Quando optar por Dados?

- Foco em registros
- Regras de negócios mais simples

Transaction Scripts

Transaction Scripts - Comandos no banco

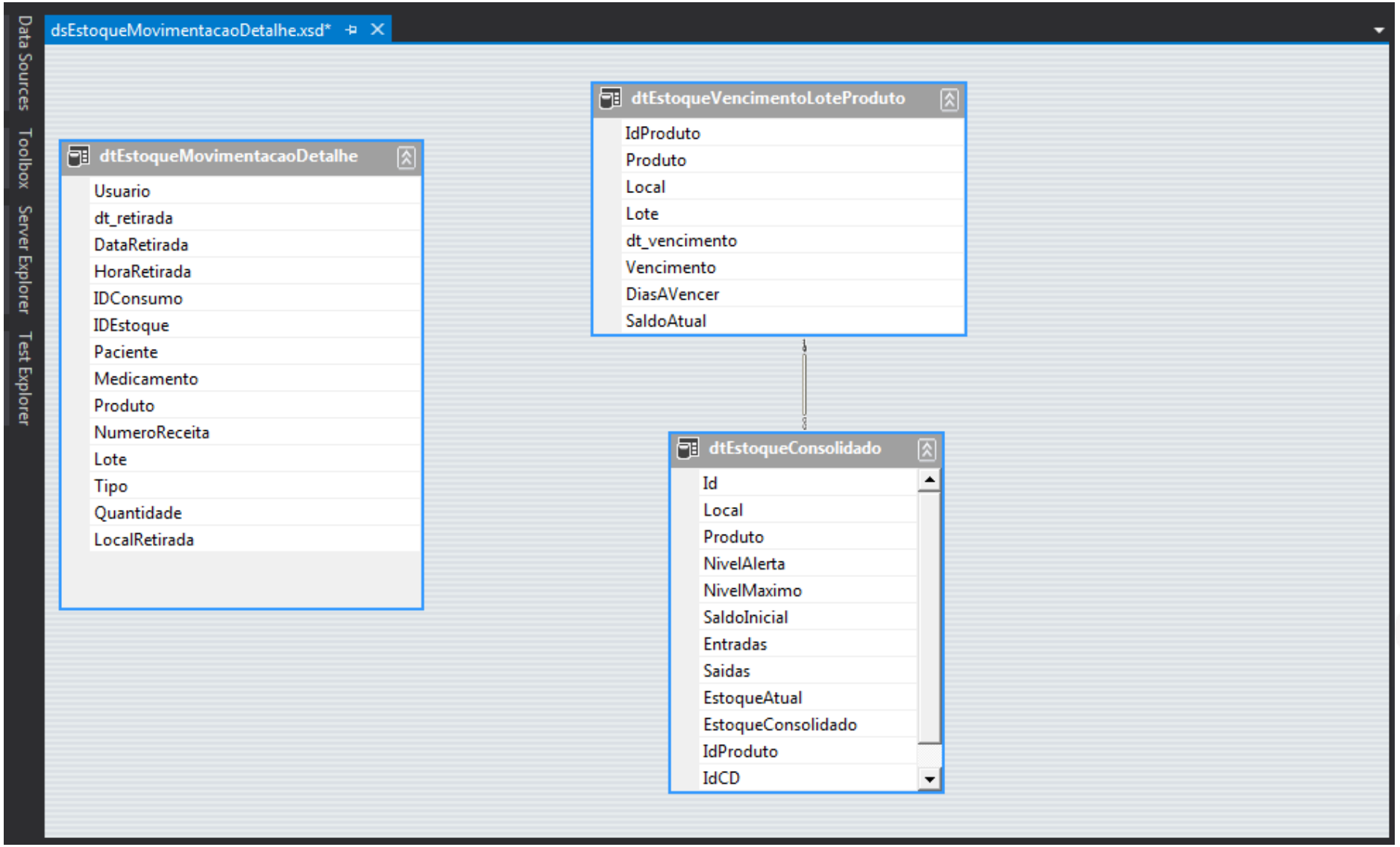
- DbCommand – SqlCommand, OracleCommand, etc
- Uso de **queries** para construção do *DataSet*
- Tipos de dados definidos no momento da consulta
- Não possui validações em tempo de compilação

Transaction Scripts – Vantagens e Desvantagens

- Vantagens
  - Fácil entendimento para quem conhece a linguagem SQL
  - Execução rápida, sem custo de mapeamento entre objeto e banco
  - Independência de bancos de dados: SQL, Oracle, MySql, etc
- Desvantagens
  - Dificuldade de manutenção
  - Fuga de padrões de código Orientado a Objetos

Table module

DataSet


Data Sources
Toolbox
Server Explorer
Test Explorer
dsEstoqueMovimentacaoDetalhe.xsd*
dtEstoqueMovimentacaoDetalhe
Usuario
dt_retirada
DataRetirada
HoraRetirada
IDConsumo
IDEstoque
Paciente
Medicamento
Produto
NumeroReceita
Lote
Tipo
Quantidade
LocalRetirada
dtEstoqueVencimentoLoteProduto
IdProduto
Produto
Local
Lote
dt_vencimento
Vencimento
DiasAVencer
SaldoAtual
dtEstoqueConsolidado
Id
Local
Produto
NivelAlerta
NivelMaximo
SaldoInicial
Entradas
Saidas
EstoqueAtual
EstoqueConsolidado
IdProduto
IdCD

DataSet – Não tipado

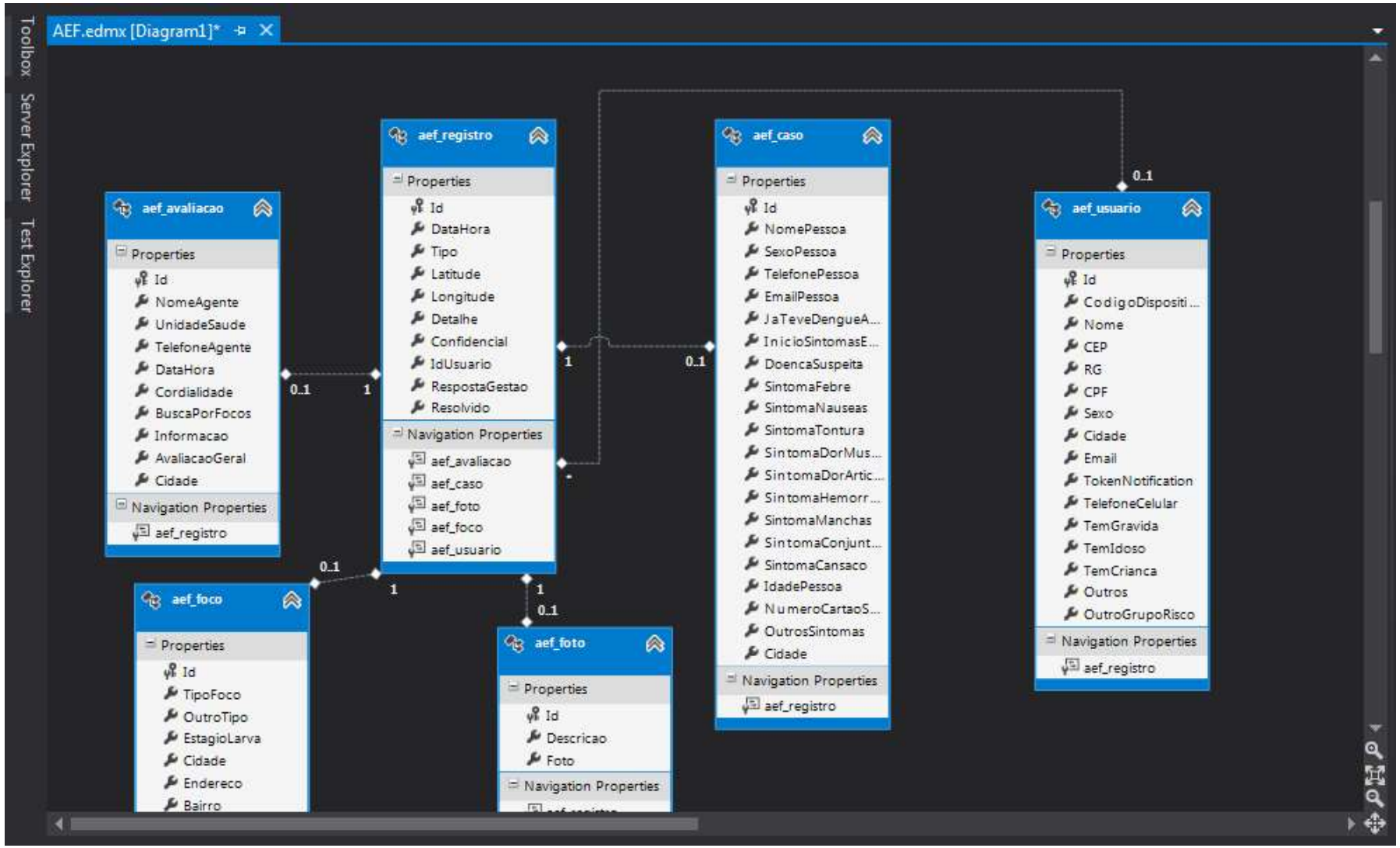
- Uso de ***queries*** para construção do *DataSet*
- Tipos de dados definidos no momento da consulta
- Não possui validações em tempo de compilação

DataSet – Vantagens e Desvantagens

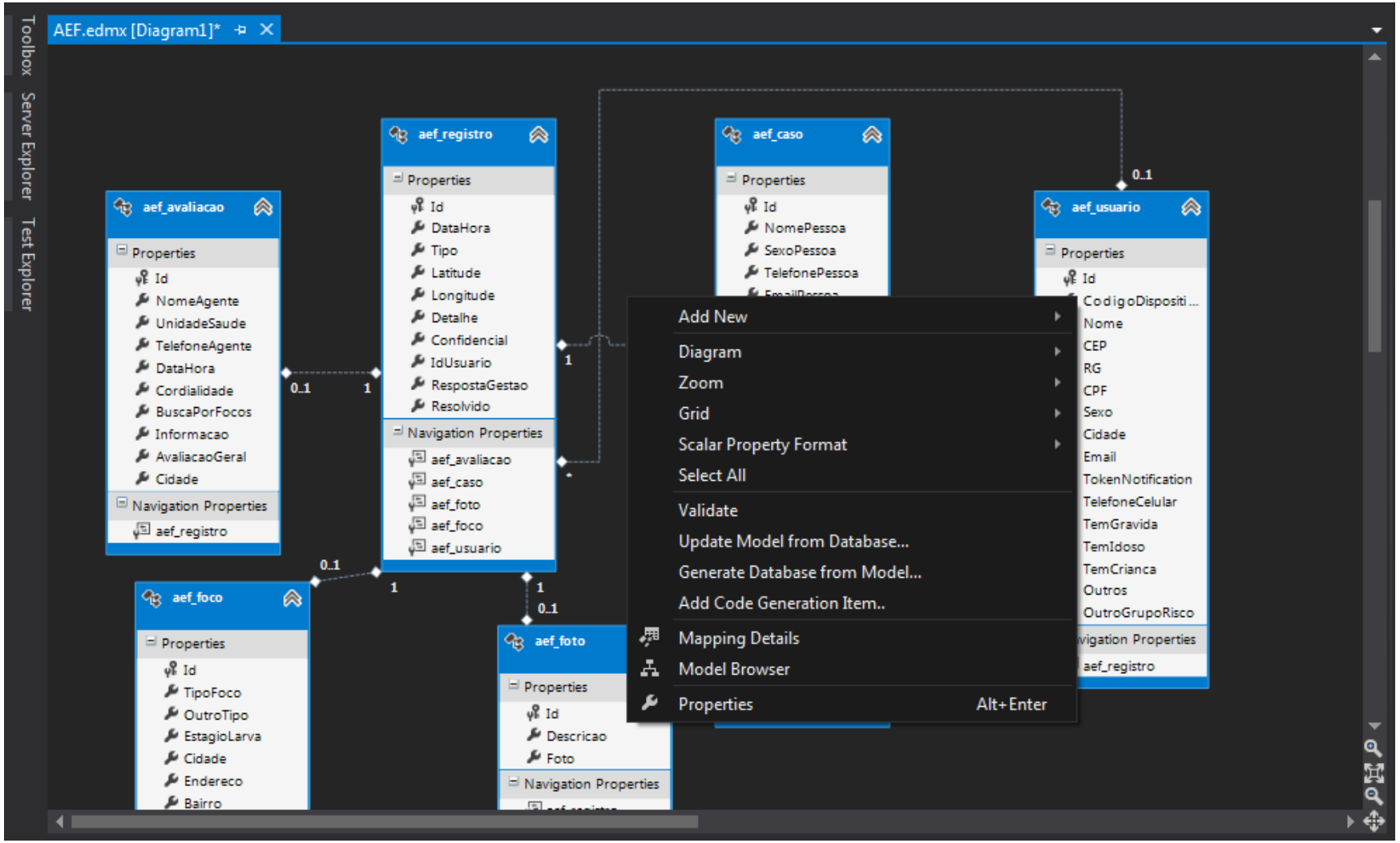
- Vantagens
  - Fácil mapeamento inicial do banco
  - Fácil manutenção
  - Grande velocidade de produção
- Desvantagens
  - Baixa performance (alto custo de transformação do DataSet e do Banco)
  - Fuga de padrões de código Orientado a Objetos

Active Record

Entity Framework



Entity Framework



Entity Framework – Vantagens e Desvantagens

- Vantagens
  - Fácil mapeamento inicial do banco
  - Fácil manutenção
  - Grande velocidade de produção
  - Possibilidade de desenho Orientado a Domínio e utilização de padrões OO
- Desvantagens
  - Baixa performance (alto custo de transformação)
  - Alto custo de manutenção de banco quando o modelo *Entity* não o espelha, gera algum retrabalho

Desenvolvimento Guiado a Domínio

Desenvolvimento Guiado a Domínio

Inicia-se o desenho do sistema construindo o diagrama de classes, usualmente um projeto do tipo *Class Library,* com foco no negócio e comportamento; não nos dados

O restante do projeto acaba se adaptando a este diagrama

Vantagens

- Flexibilidade de funcionamento
- Melhor representação do negócio
- Camada de negócios mais independente (menor acoplamento)
- Menos regras de negócio fora do código-fonte (no banco de dados)
- Melhor utilização de princípios de Orientação a Objetos

Desvantagens

- Maior complexidade na persistência de dados
- Maior complexidade de recuperação de dados
- Pode haver maior custo de transformação
- Menor capacidade de leitura direto do banco

Quando optar por Domínio?

- Foco no negócio
- Negócio e regras mais complexas

Formas de armazenar os dados

Modelos facilitados – DataSet, Command ou Entity Framework

- Pode-se usar facilitadores e transformar manualmente, com bancos relacionais
- Modelo relacional diferente do modelo de classes
- Transformação manual entre os modelos
  - Modelo de domínio diferente do modelo de dados
  - Uma tabela pode virar um campo serializado
  - Uma classe pode ser armazenada em mais de uma tabela
  - Um registro no banco pode gerar mais de um objeto

Nhibernate – Mapeador Objeto-Relacional

- Mapeador flexível
- Possibilita diferença do modelo de banco para o modelo de classes
- Cabe ao programador definir as transformações

NoSql

- Não relacionais
- Dados menos estruturados ou esquemas flexíveis
- Melhor desempenho em alguns cenários

- Colunas, Chave-valor, Grafos ou Documentos

Outras formas de armazenamento

- Arquivos XML
- Arquivos de Texto
- Caches em memória

Exemplos e decisão de formato

Dados ou Domínio/Negócio?

Qual a melhor escolha?

Sistema de cadastro escolar, onde o aluno ou pai entra em uma página da internet, encontra a escola, turma e turno.

O gestor da rede escolar acompanha e aprova o processo da parte de administração do sistema.

Qual a melhor escolha?

Sistema de análise de variação de ativos financeiros e decisão de entrada e saída de investimento, utilizando modelagem matemática complexa e IA.

Baseado em dados buscados em API de plataforma de Trading.

Qual a melhor escolha?

Grande rede social focada em relacionamento entre pessoas, como Facebook, por exemplo. Com imenso conjunto de dados e demanda por velocidade de recuperação de dados.

Qual a melhor escolha?

Sistema de controle financeiro pessoal desenvolvido por uma *StartUp.*

MVP precisa de ser lançado rapidamente para testar o mercado e decidir se o negócio é viável ou não.

Qual a melhor escolha?

A que se adequa mais à realidade do cliente, do negócio e do time de produção.

Baixo acoplamento das outras camadas é importante (para futura evolução desta camada).

Desenho da camada de domínio

Desenho de um cenário para cada padrão, contextualizando e explicando a escolha pelo padrão.

Exemplos reais ou fictícios

Desenho da camada de domínio do seu projeto

- ✓ Escolha da abordagem: Dados ou Domínio
- ✓ Como fica a separação entre Dados e Negócio?
- ✓ Desenho do negócio superficialmente
- ✓ Escolha do modelo de persistência
- ✓ Escolha das tecnologias

- ✓ Razões pelas quais se deu as escolhas

Camada de Domínio/Negócio e camada de Dados

Leandro Garcia


PUC Minas
Virtual