

PREVIOUSLY ON READING POFEAA

2005-05-22

kdmsnr - <http://capsctrl.que.jp/kdmsnr/>

前回までの読書会

- » (1) PofEAAを読み、に談話室滝沢に行こうオフ。
 - 日時:2004.02.28(土)17:00~19:00
 - 場所:談話室滝沢 お茶の水駅前店
 - 参加者:13名

- » (2) 2回目でもやろうか。ファウラーも来日するしオフ。
 - 日時:2004.03.28(日)17:00~19:00
 - 場所:渋谷ルノアール
 - 参加者:12名

前回までの主な内容

エンタープライズ アプリケーションとは

- » **ビジネス アプリケーション**
 - 給与支払システム、医療管理システム、請求システム、信用度採点システム、物流追跡システム
- » **データがたくさん**
 - 数百テーブルあるギガバイト データベース
- » **複数のユーザーに対応**
- » **ビジネス ロジックは複雑で不合理**
- » **その他の多くのシステムと統合**

レイヤリング

レイヤリングの特徴

- ✓ OSIの階層みたいなイメージ
- ✓ 各レイヤは独立している
- ✓ 下位レイヤの上に複数の上位レイヤ

- ✓ Tier: 物理的な層
- ✓ Layer: 論理的な層

- ✗ カプセル化されないこともあったり(影響の波及)
- ✗ レイヤリングによりパフォーマンスが落ちることも

基本的な3つのレイヤ

プレゼンテーション

ドメイン

データソース

基本的な3つのレイヤ

ユーザーとやり取り

プレゼンテーション

ビジネスロジック

ドメイン

データの場所を示す

データソース

基本的な3つのレイヤ

ユーザーとやり取り

プレゼンテーション

外部スキーマ

ビジネスロジック

ドメイン

概念スキーマ

データの場所を示す

データソース

内部スキーマ

基本的な3-1つのレイヤ

ユーザーとやり取り

プレゼンテーション

外部スキーマ

データの場所を示す

データソース

内部スキーマ

基本的な3つのレイヤ

ユーザーとやり取り

プレゼンテーション

外部スキーマ

意味モデル、データベース論理モデル、オブジェクトモデル

データの場所を示す

データソース

内部スキーマ

基本的な3つのレイヤ

ユーザーとやり取り

プレゼンテーション

外部スキーマ

問題領域

データの場所を示す

データソース

内部スキーマ

基本的な3つのレイヤ

ユーザーとやり取り

プレゼンテーション

外部スキーマ

解決領域

問題領域

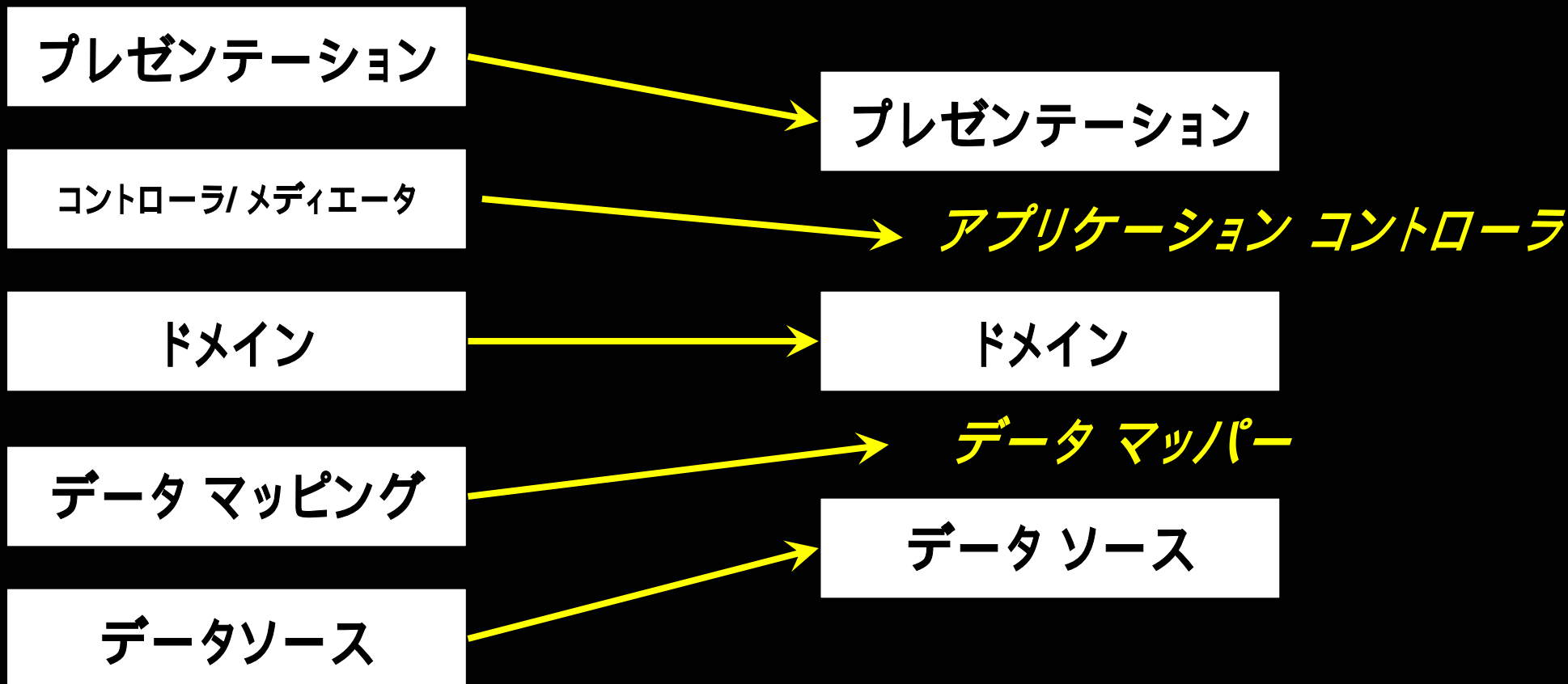
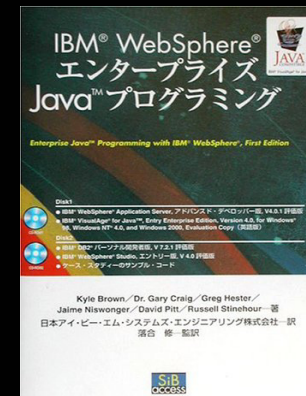
データの場所を示す

データソース

内部スキーマ

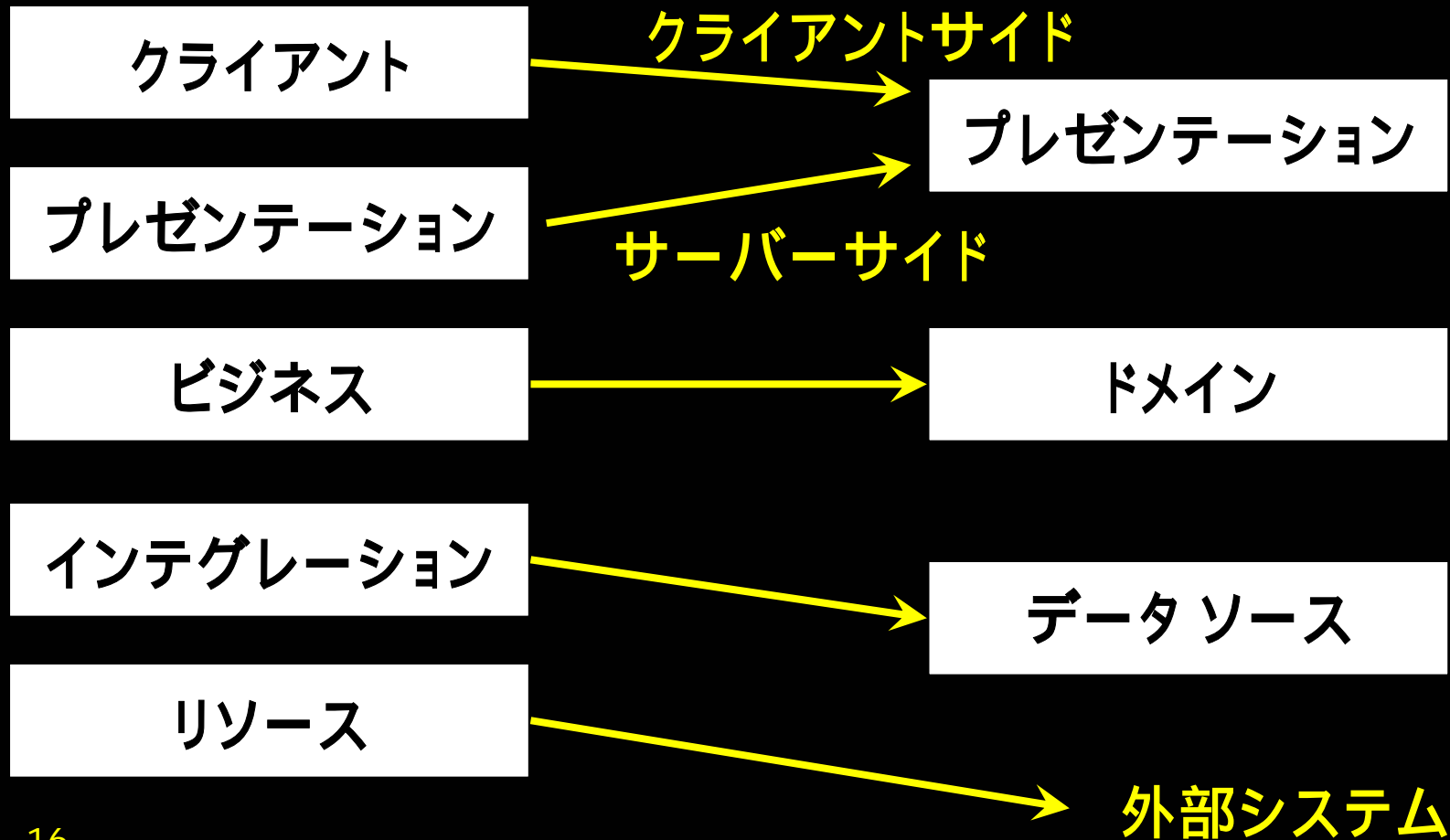
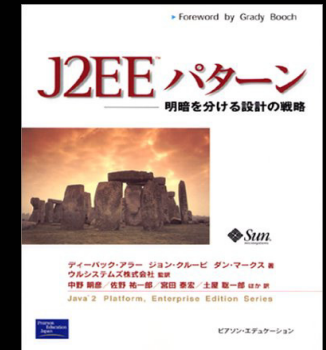
その他のレイヤリング

IBM Websphere エンタープライズJava プログラミング

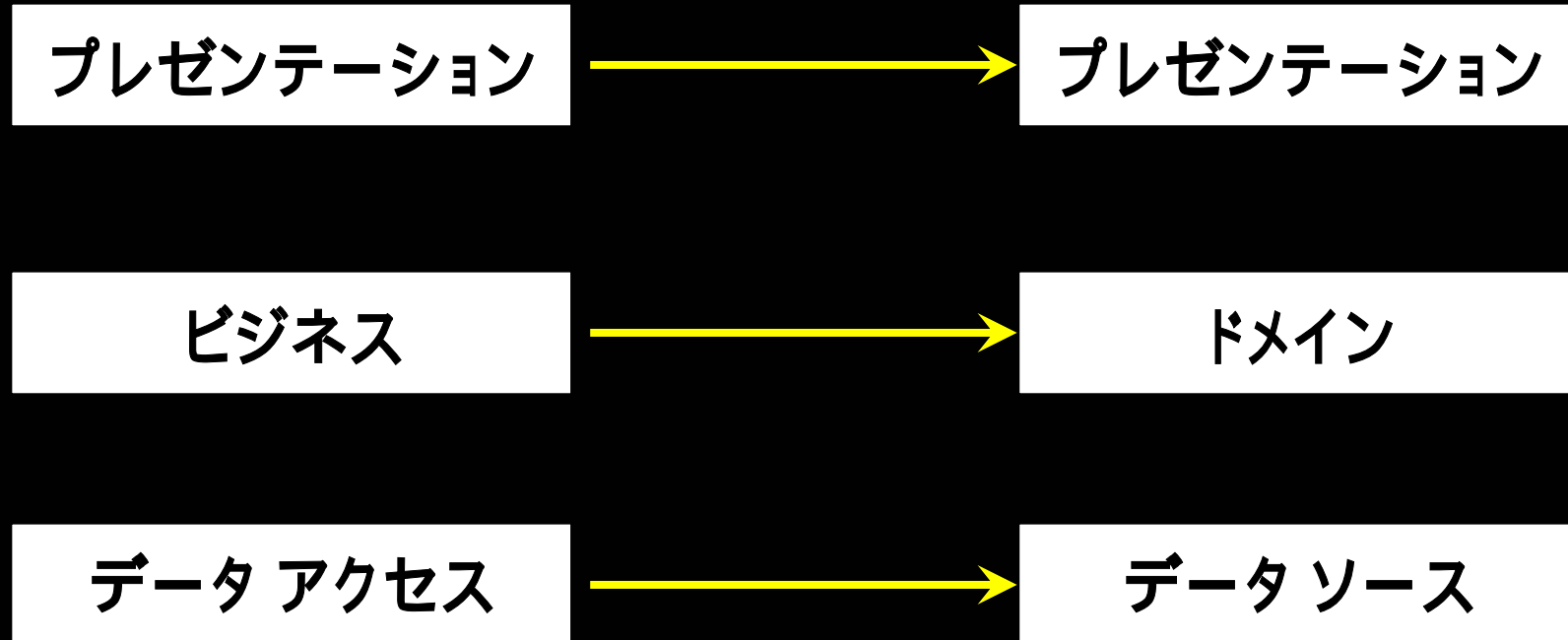


J2EEパターン

明暗を分ける設計の戦略

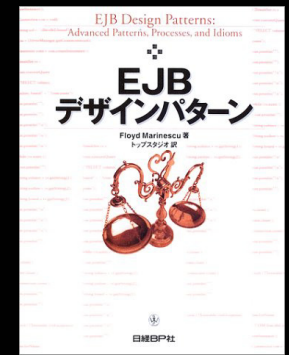


Microsoft DNA Layers



Kirtland, *Designing Component Based Architectures*,
Microsoft Press

EJBデザインパターン



プレゼンテーション



プレゼンテーション

アプリケーション



アプリケーションコントローラ

サービス



サービスレイヤ

ドメイン



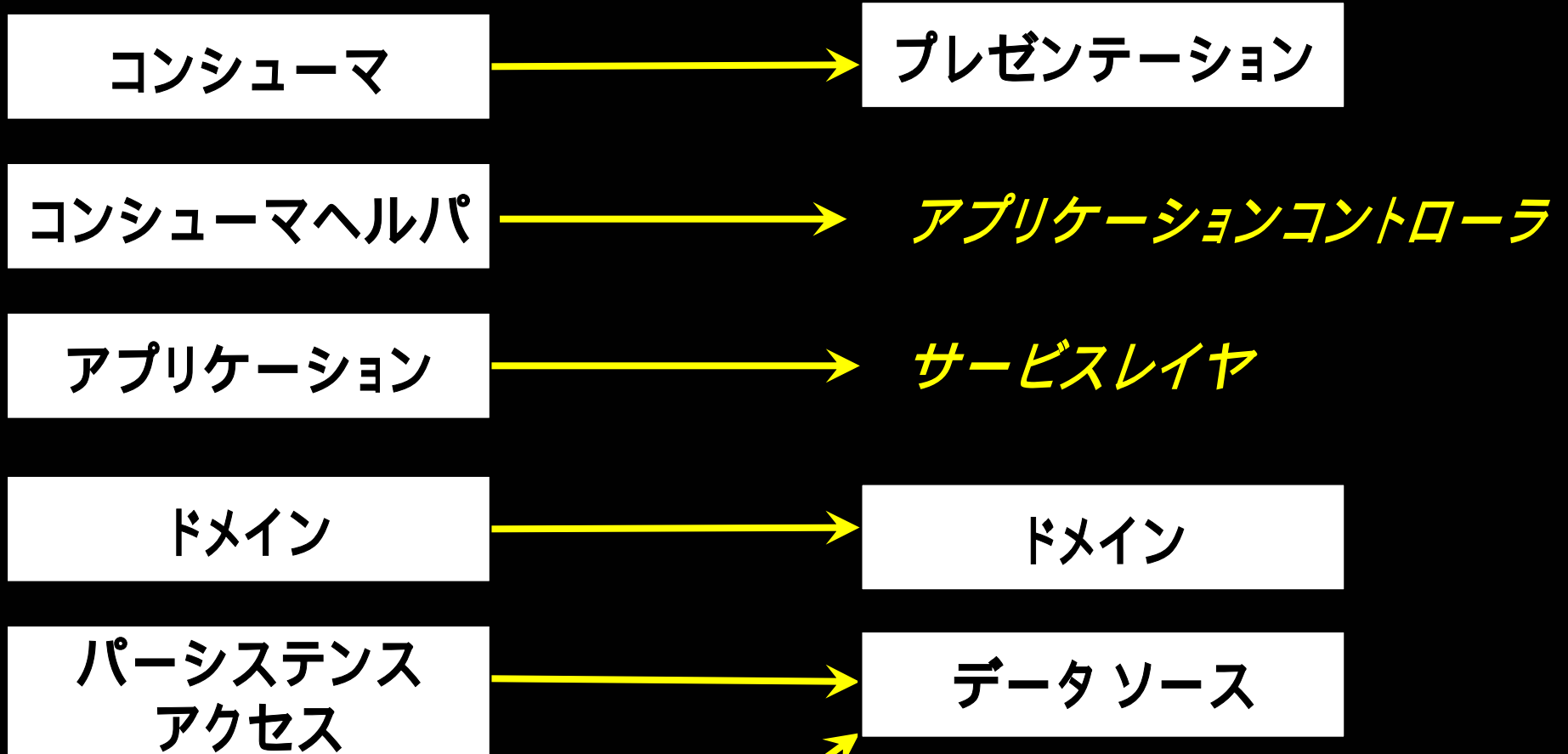
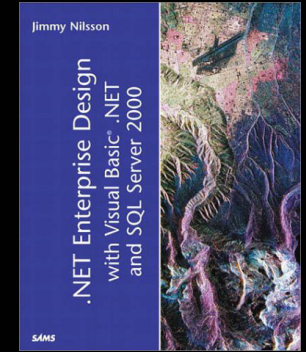
ドメイン

パーシステンス



データソース

. Net Enterprise Design With Visual Basic .Net and SQL Server 2000



ドメイン ロジック

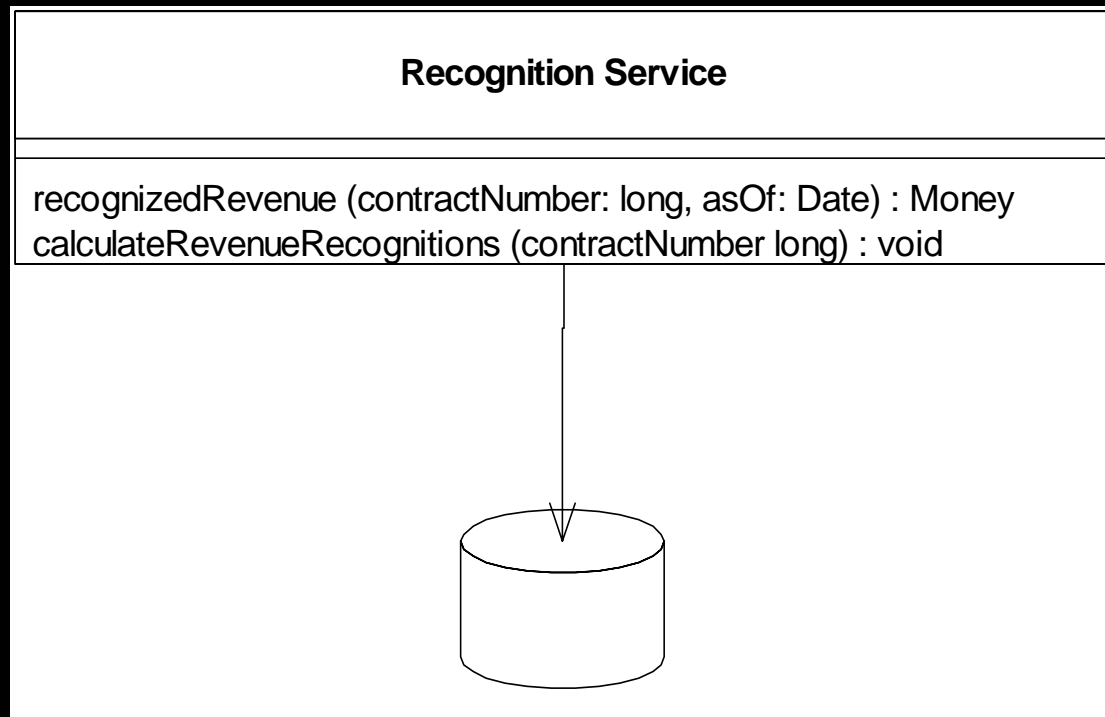
3つのアプローチ

- » *Transaction Script*
 - シンプル
- » *Domain Model*
 - オブジェクト指向
- » *Table Module*
 - 上記の2つの中間

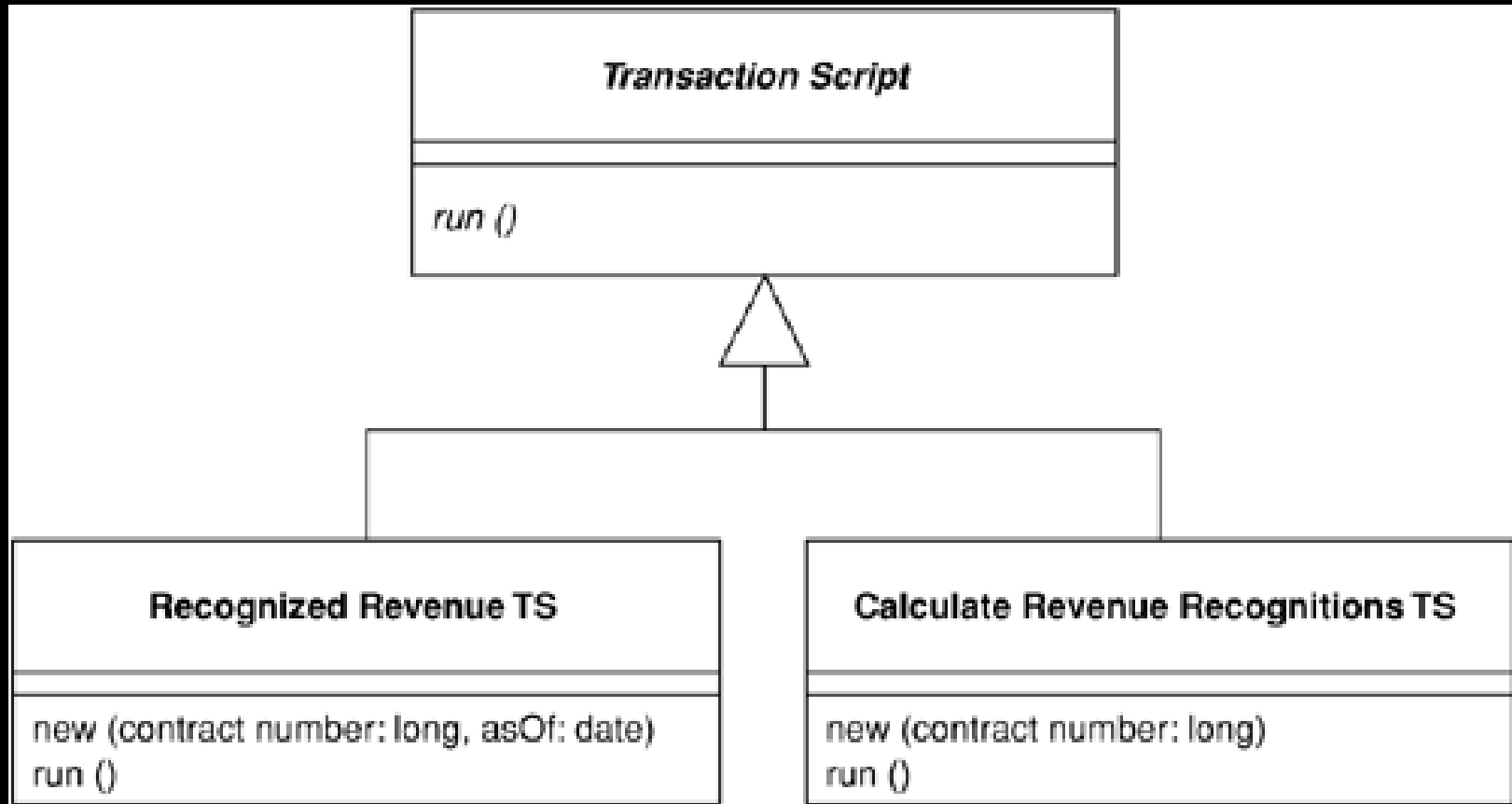
Transaction Script

- ✓ トランザクションごとに処理を分ける
- ✓ シンプルなプログラミング モデル
 - ✓ CGIスクリプトなどで利用される
- ✓ データベースとのマッピングもシンプル
 - ✓ 行データゲートウェイ、テーブルゲートウェイ
- × ドメイン モデルが複雑になるにつれ難しくなる
- × スクリプト間でのコードの重複

ひとつのクラスに複数の処理



ひとつのクラスにひとつの処理



Domain Model

- ✓ (純粹な)オブジェクト指向的
- ✓ 複雑なドメイン ロジックにも対応できる
- × 構築が難しい
- × RDBMSとのマッピングが複雑になる
データマッパーの使用

Domain Model

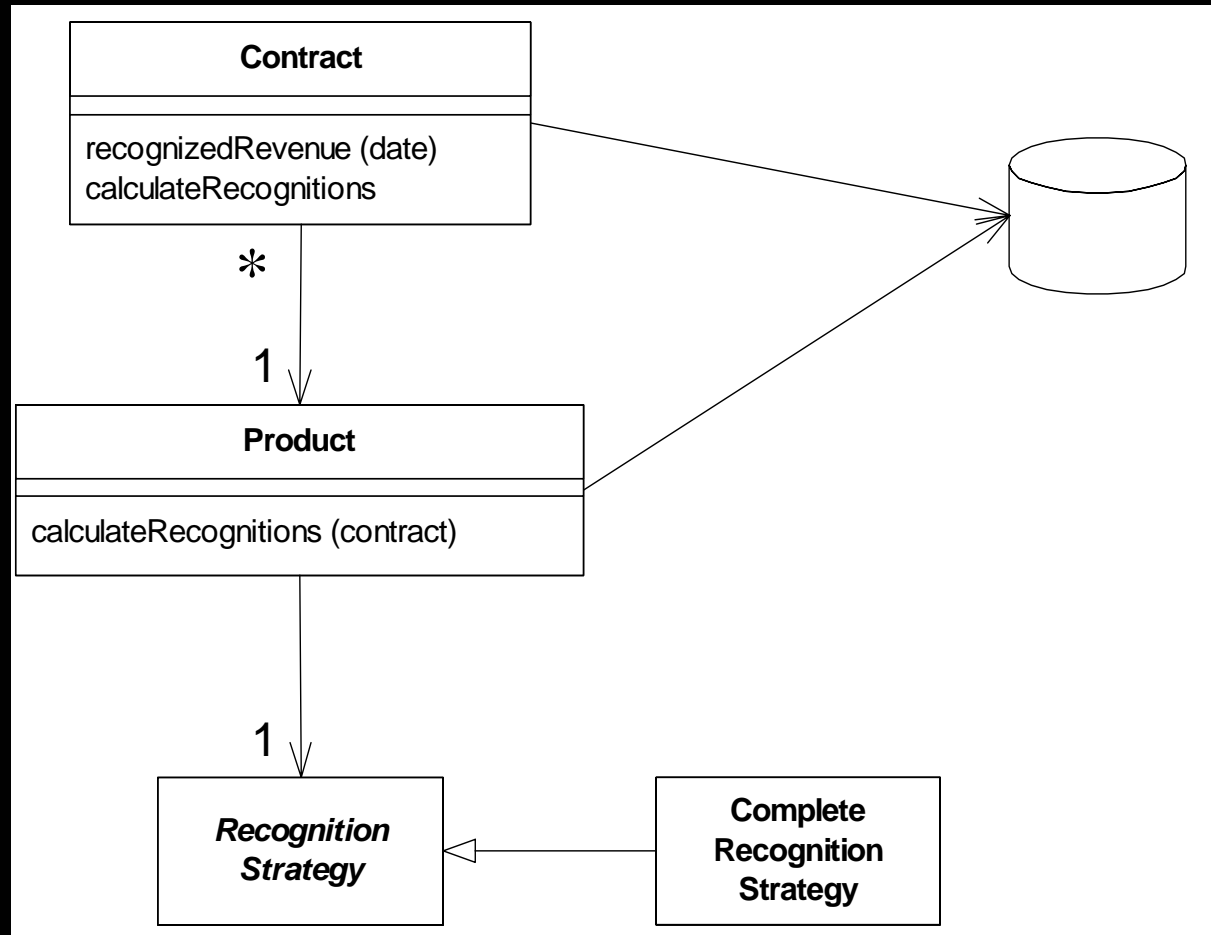
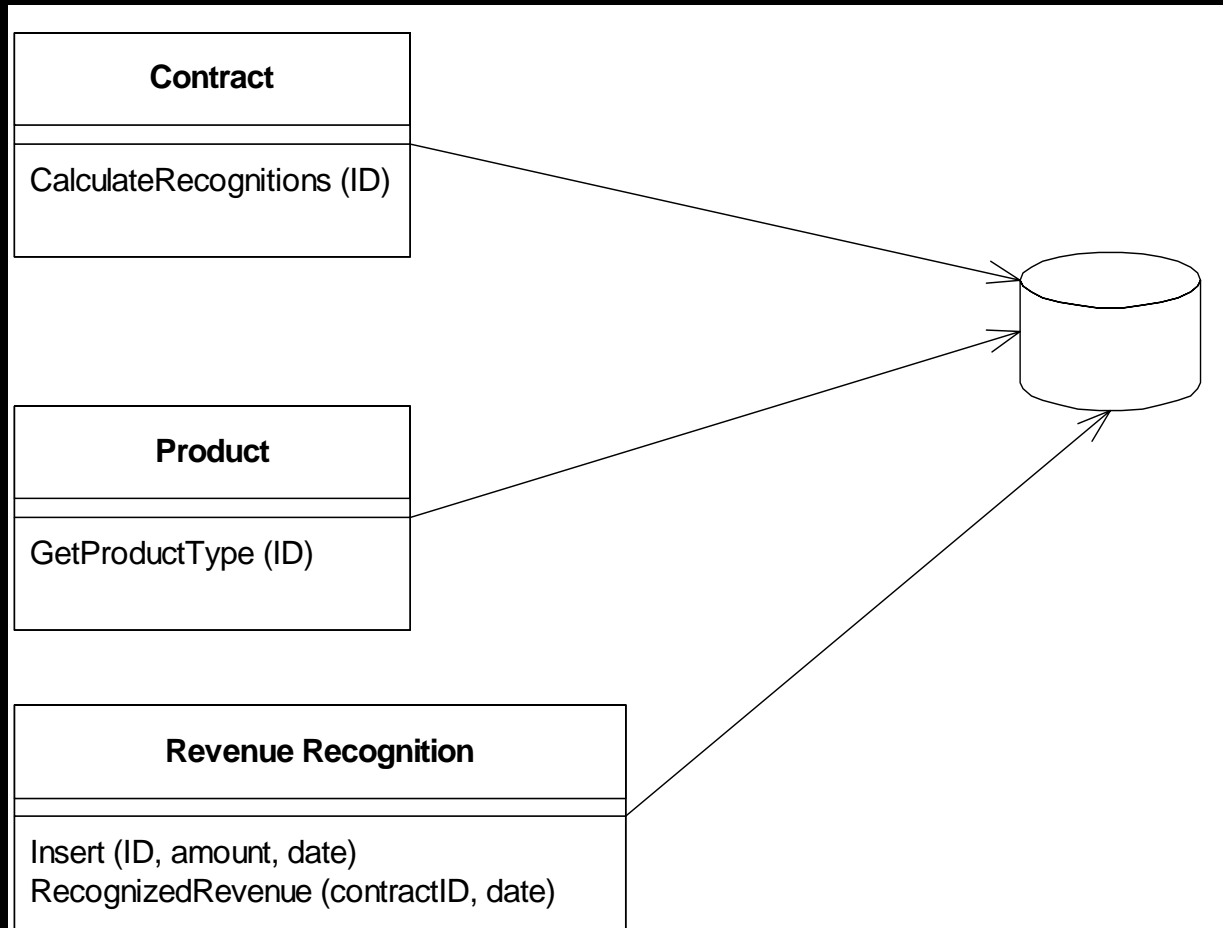


Table Module

- ✓ ツールのサポート(Visual Studio.NET)
- ✓ ひとつのデータ構造をすべてのレイヤで使用
- × 複雑なロジックに対応できない
- × オブジェクト指向のパターンが適用できない

Table Module



+ *ServiceLayer*

» ビジネスロジック

- アプリケーションロジック(ワークフローロジック)
- ドメインロジック

