分析・設計と実装を繋ぐもの(J2EE)

(株)シーエーシー　技術研究部
山田 喜彦

Contents

- J2EE コンポーネント
- 共有化に潜む問題
- 設計の原則
- どう実装？
- アーキテクチャ設計

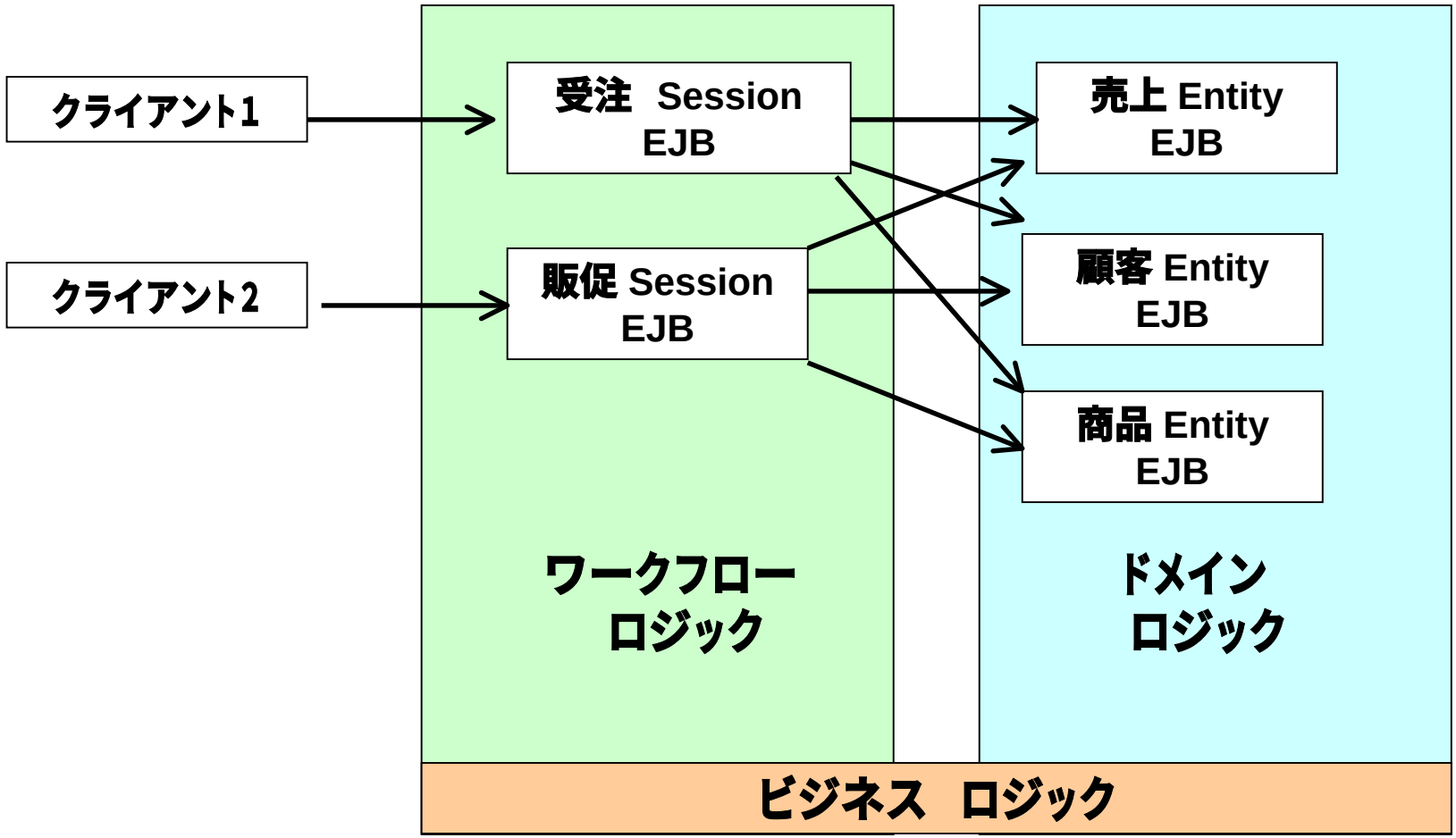
自己紹介

以下を専門に研究しています

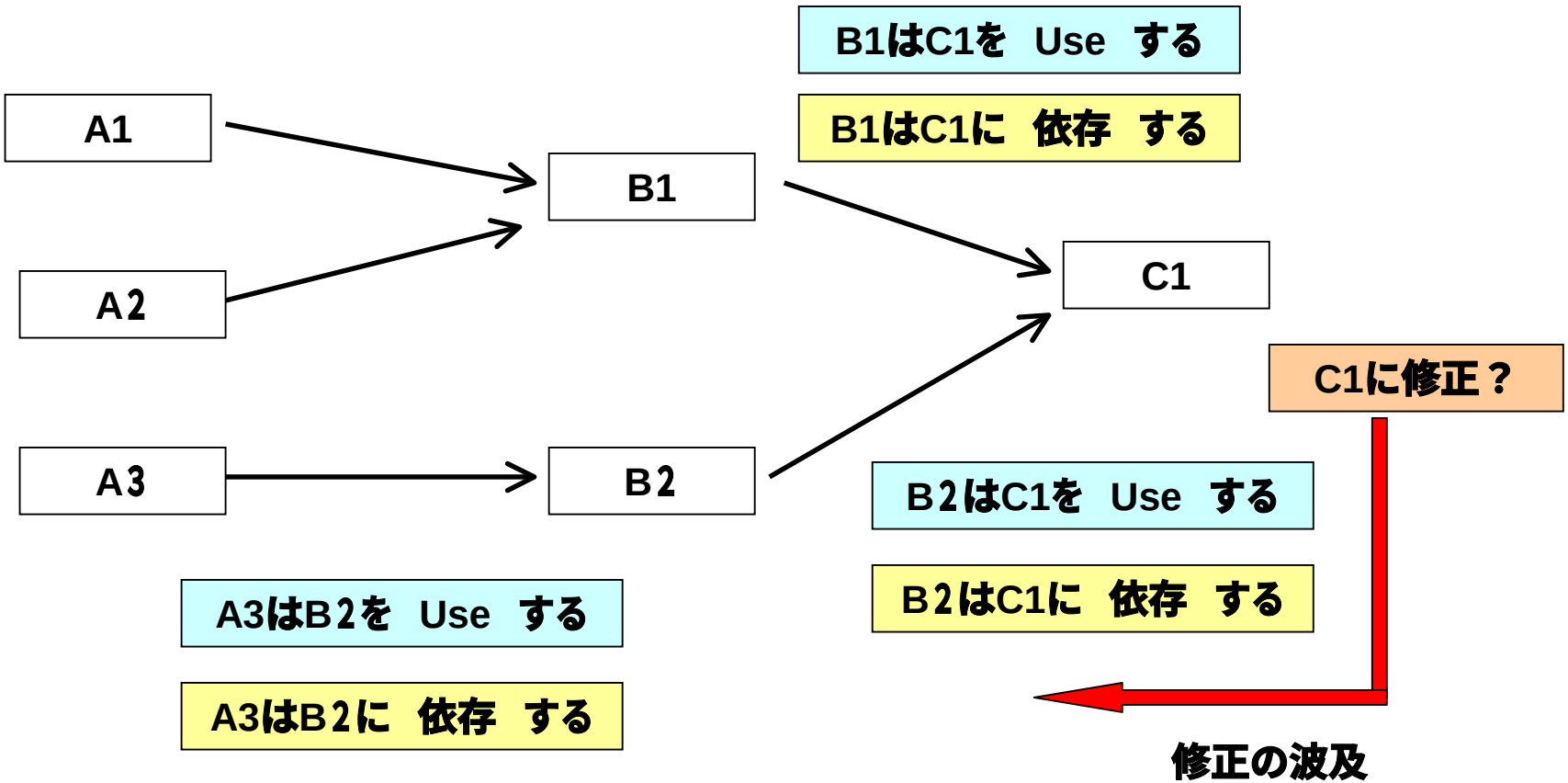
- オブジェクト指向
- J2EE
- Web Service

理想的なモデリングと実装のギャップ。その解決に必要な知恵を考えます

J2EEコンポーネント



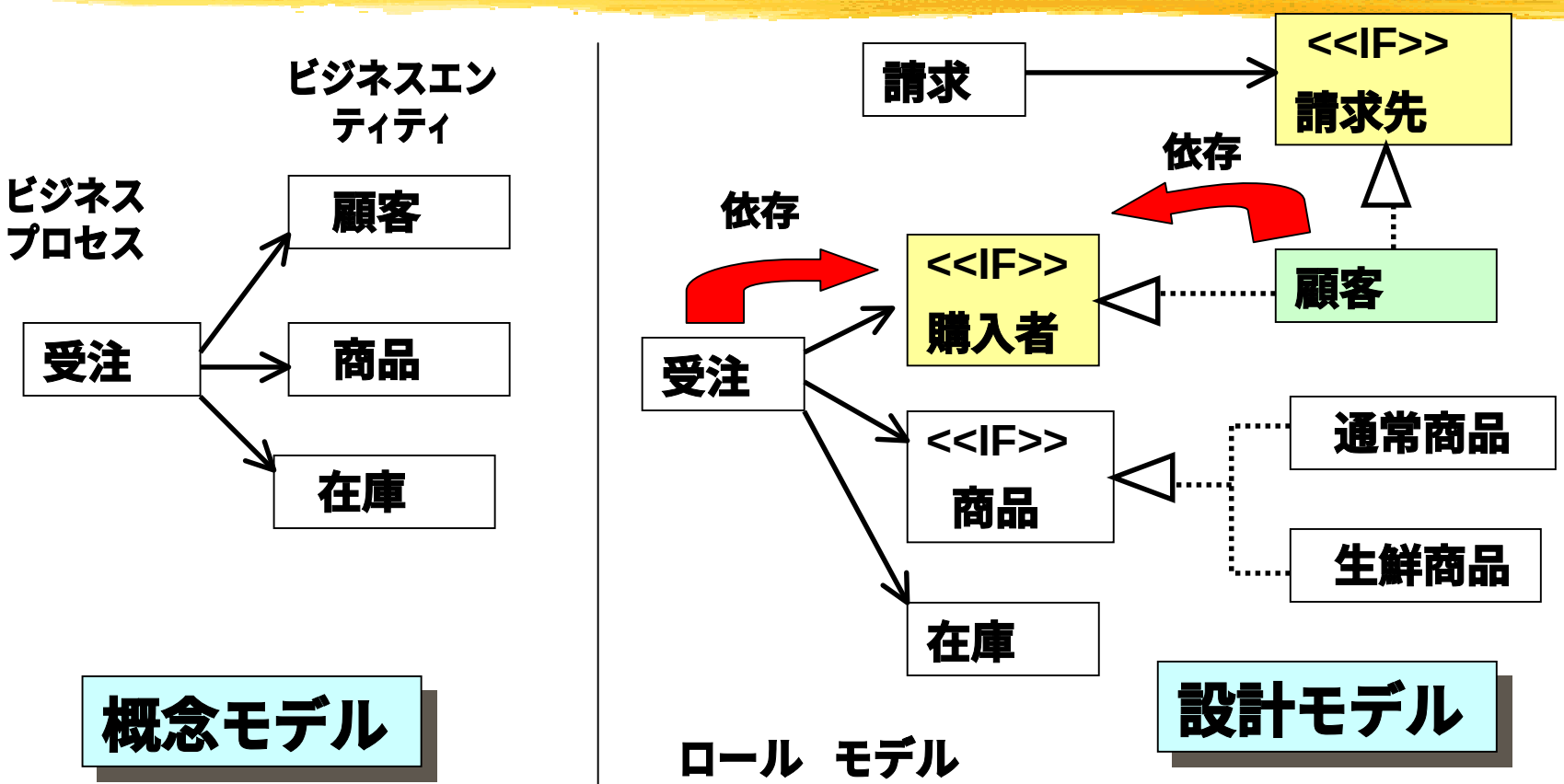
共有化に潜む問題


B1はC1を Use する
B1はC1に 依存 する
A1
B1
A2
C1
C1に修正？
A3
B2
B2はC1を Use する
B2はC1に 依存 する
A3はB2を Use する
A3はB2に 依存 する
修正の波及

設計の原則(1)

- インターフェース分離の原則
  (Robert C. Martin「Agile Software Development」)
- インターフェースの原則
  (Bertrand Meyer「オブジェクト指向入門」)
  - 明示的なインターフェース
  - 少ないインターフェース
  - 小さいインターフェース (弱い結びつき)

設計の原則(2)



依存反転の原則：抽象に依存させる。具象には依存させない

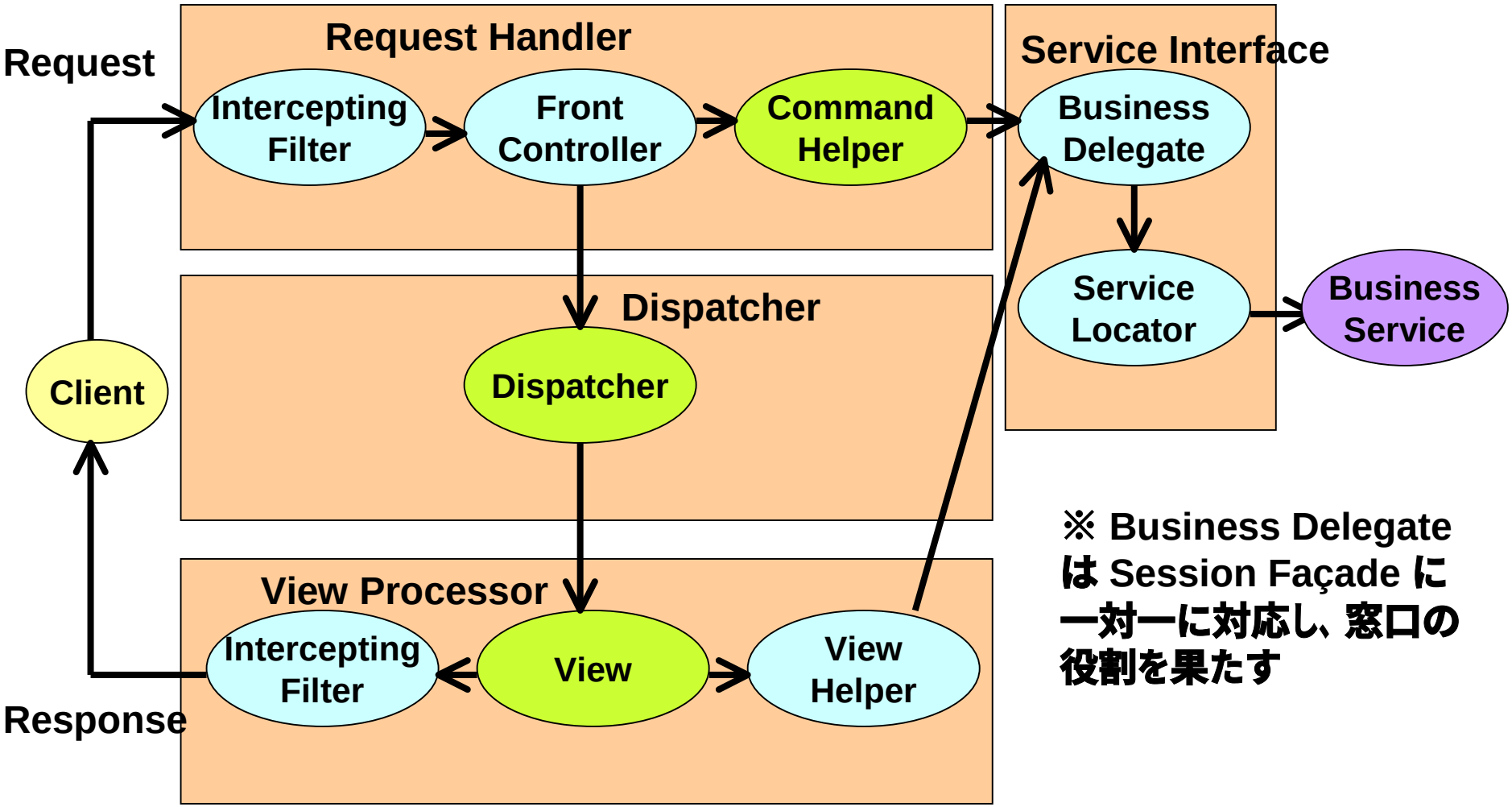
設計の原則(3)

- 時の試練に耐えた原則
  - Encapsulation
  - Interface
  - Loose Coupling
  - Appropriate Granularity
  - High Cohesion
  - Parameterization
  - Deferral
- 「Beyond Software Architecture」(Luke Hohmann)

どう実装？：アーキテクチャ選択

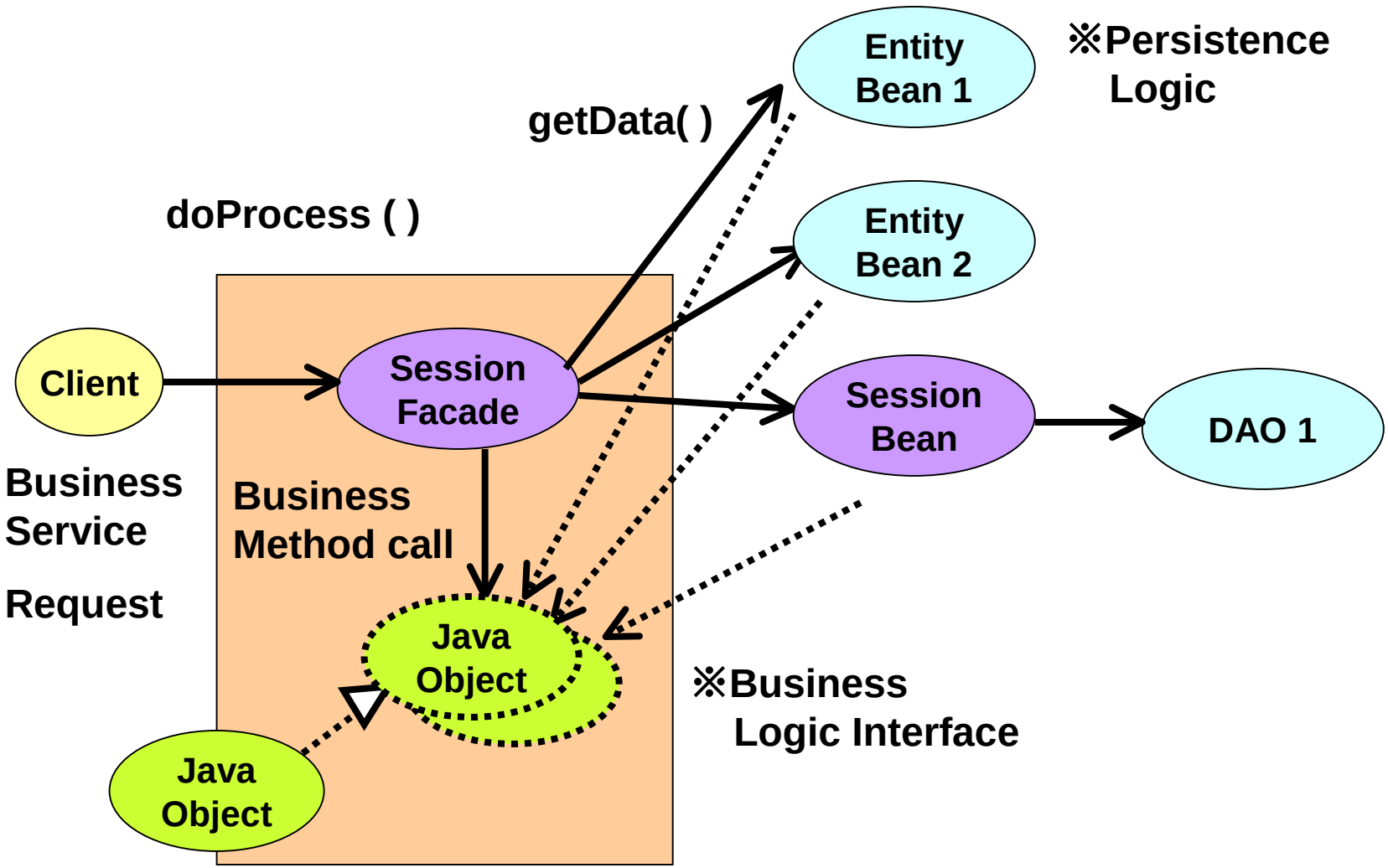
- ビジネス・ロジック
  - EJB　vs.　Stateless Session Bean
- Session State
  - Web Tier　vs.　EJB Tier
- Persistent
  - Entity Bean　vs.　JDBC+DAO　vs.　JDO
- Transaction
  - CMT　vs.　BMT

どう実装？：Web層のパターン

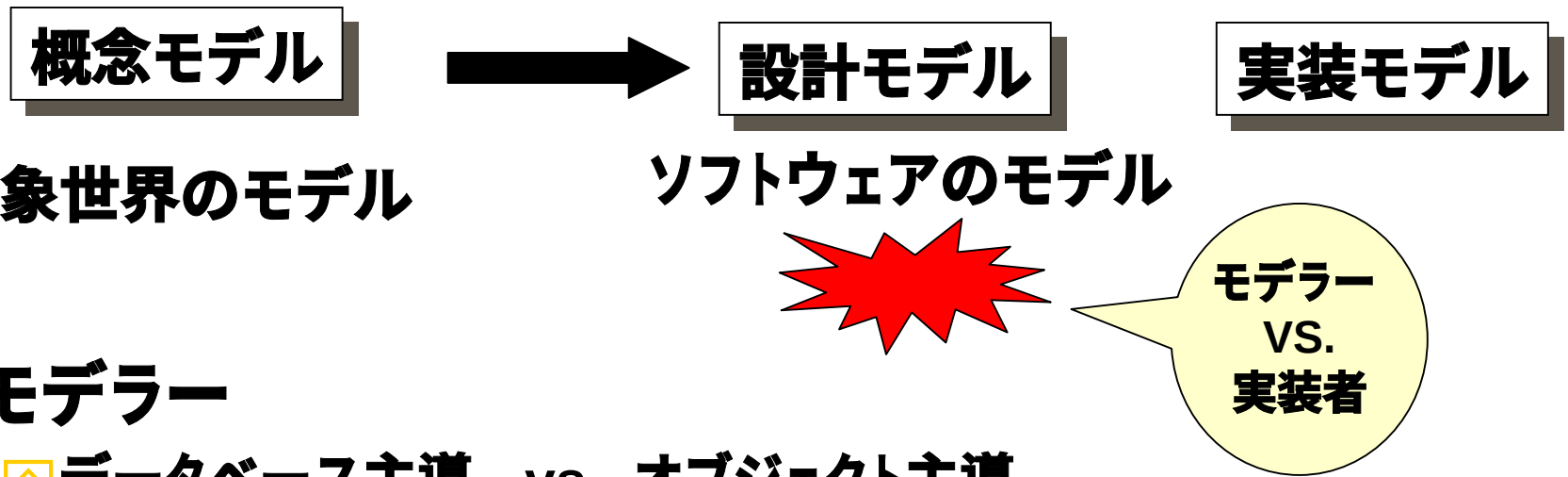


※ Business Delegate は Session Façade に一対一に対応し、窓口の役割を果たす

どう実装？：EJB層のパターン

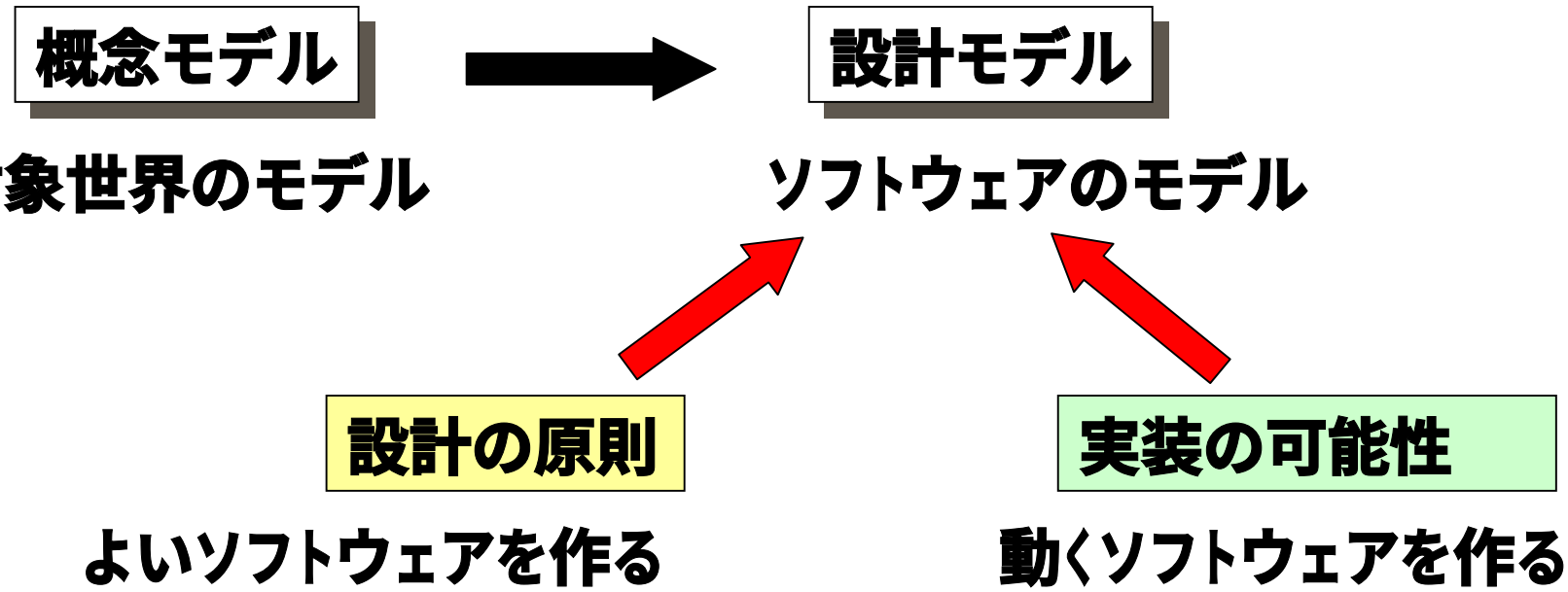


アーキテクチャ設計(1)



- モデラー
  - データベース主導　vs. オブジェクト主導
  - 概念クラスをそのまま設計クラス図 (論外)
  - 実装を無視した理想的なクラス図 (設計の原則)
- 実装者
  - パフォーマンス主体(J2EE Pattern)

アーキテクチャ設計(2)



- パターン？

パターンWG

- 実践タスク
  - J2EEでの分析・設計と実装を継ぐパターン適用の模索　サブタスク
    - 分析・設計の方法論＆実装の方法論　(vs. ではなく)
    - 最適なJ2EE設計を探る
    - 新しいJ2EEパターンをめざして