BACKSLASH DESIGN

アーキテクチャ中心開発の勘所

2018年7月6日

山田 大介
ビースラッシュ株式会社

3つの勘所

1．静動分離

- 末広がり から 構造化 へ
  - 末広がりは静動混在
  - 静解析ファースト

2．アーキテクチャテンプレート

- コンポーネントの全体レイアウトを決める
- 骨格＋肉付けプログラミング

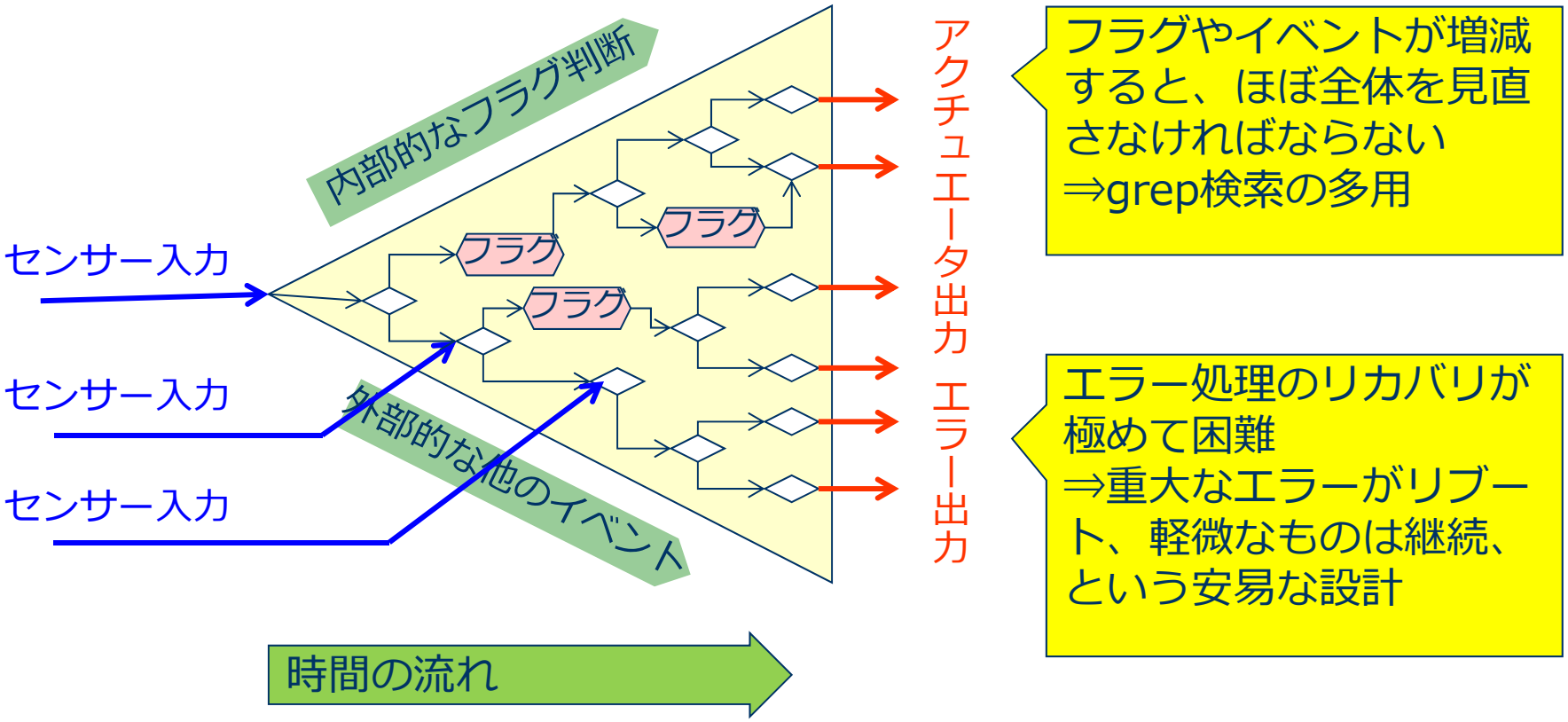
3．WhatとHowの補助線

- 複数ビューを統合し経営とテクノロジを繋ぐ
  - 多重下請け構造からはアーキテクトは生まれにくいなぁ

1. 静動分離

静動混在：末広がり形状

- 制御仕様をそのままプログラミングすると「末広がり」になる
  - 内部的なフラグ判断の積み重ね
  - 外部的な他のイベントの積み重ね

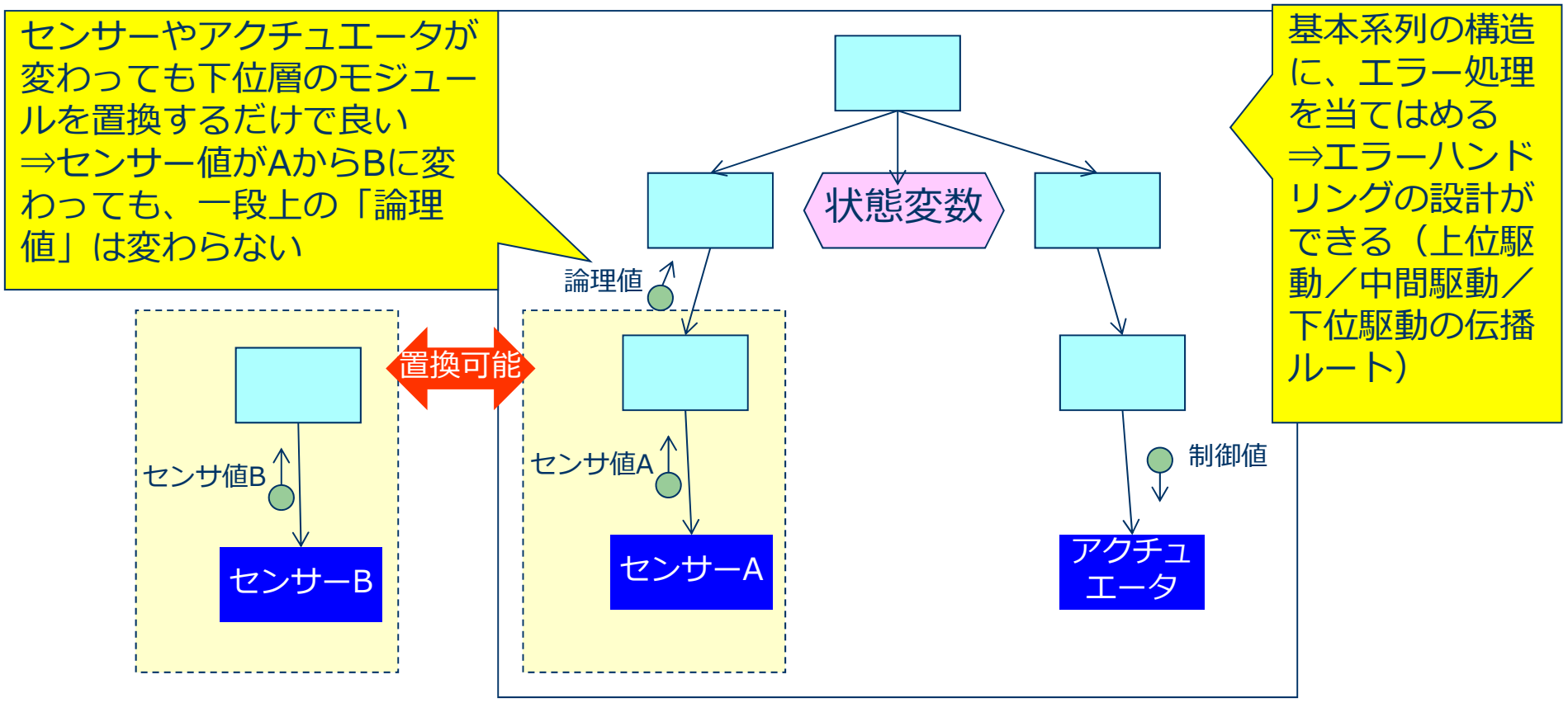


フラグやイベントが増減すると、ほぼ全体を見直さなければならない
⇒grep検索の多用

エラー処理のリカバリが極めて困難
⇒重大なエラーがリブート、軽微なものは継続、という安易な設計

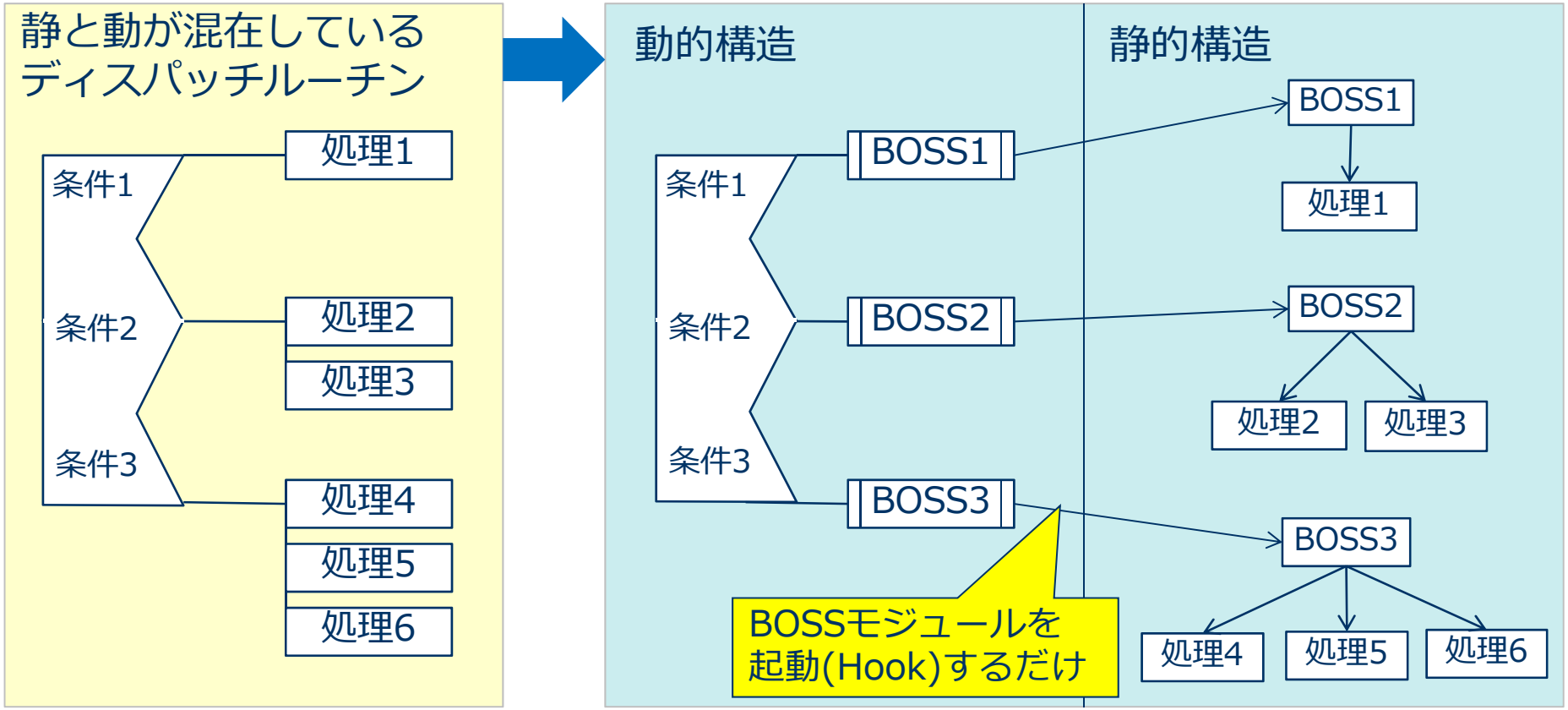
静的構造を設計すると：構造化

- 論理的な階層化設計
  - 上位層で、状態変数を定義
  - 下位層で、センサーやアクチュエータのプリミティブな制御のみ



静動分離

- 静的構造 と 動的構造を分離する
  - 静的構造はWhatの視点で図面化　⇒ 問題ドメインの用語
  - 動的構造は、mainループなどを図面化　⇒ ソフトウェアの用語



BACKSLASH DESIGN

2．アーキテクチャテンプレート

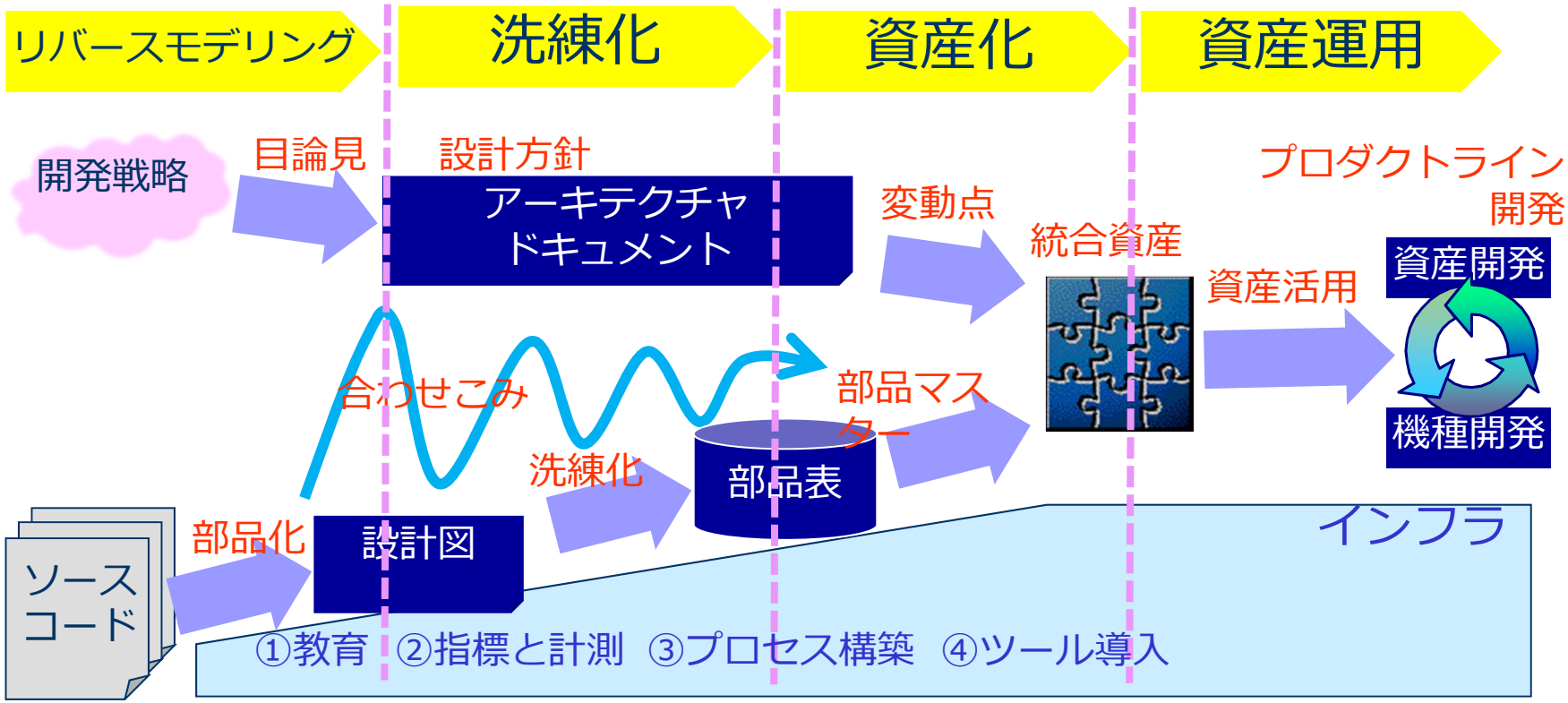
ソースコードと設計のシンクロ開発

アーキテクチャを遵守したコード

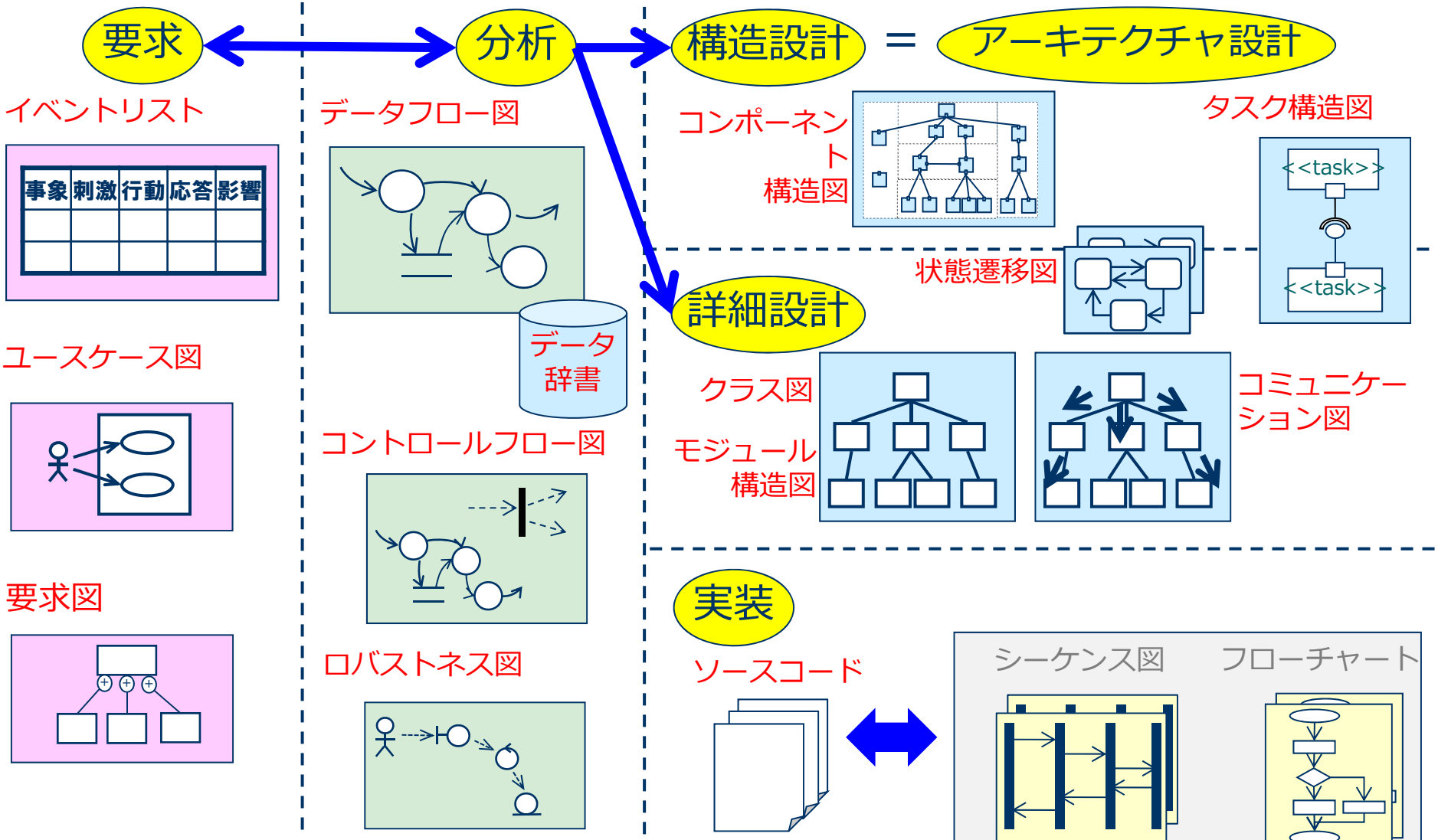
- トップダウン：アーキテクトが、設計意図を明示し、コードと合わせ込む

ソースコードと設計の同期

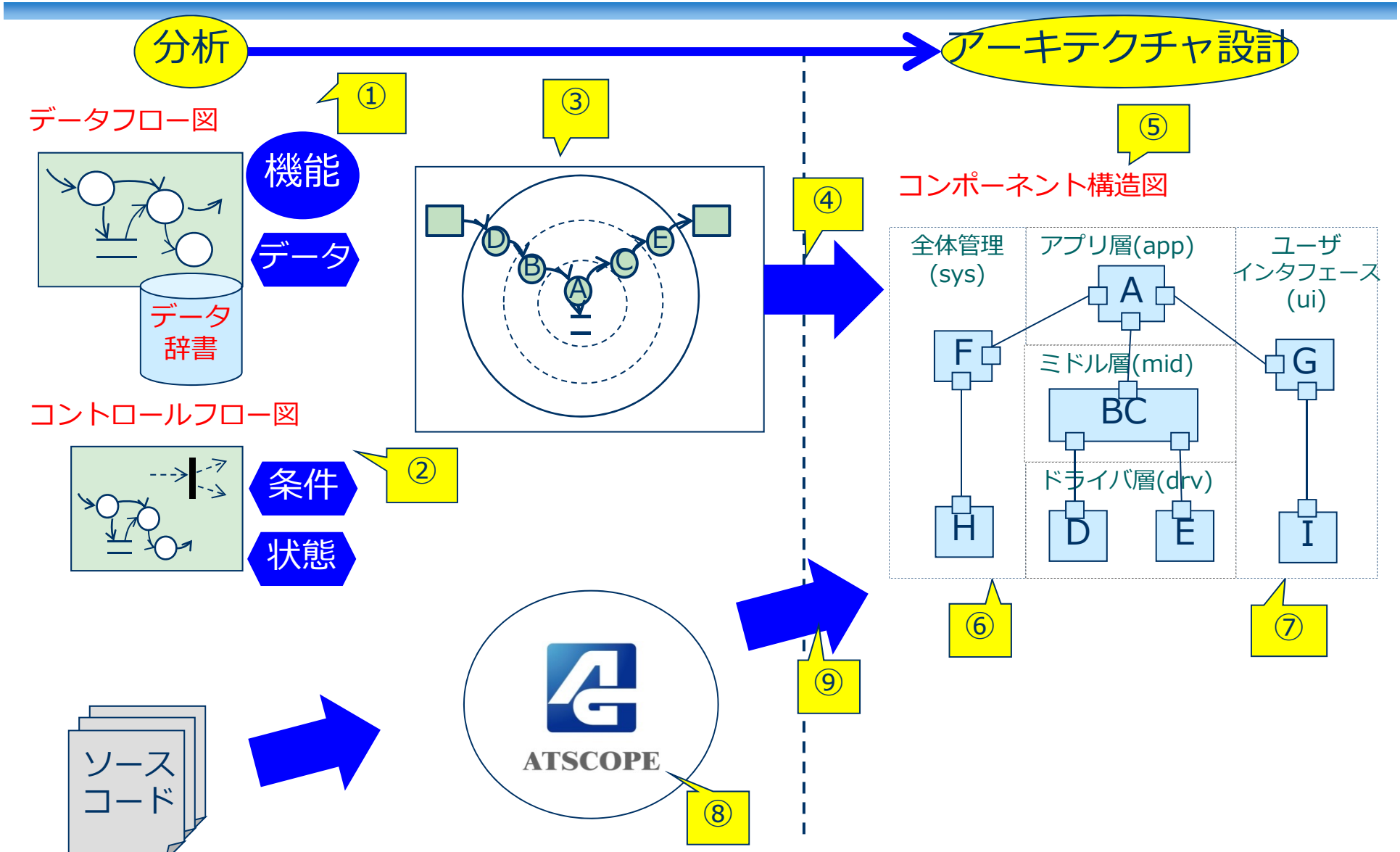
- ボトムアップ：各エンジニアが、コードを洗練化し、部品化していく



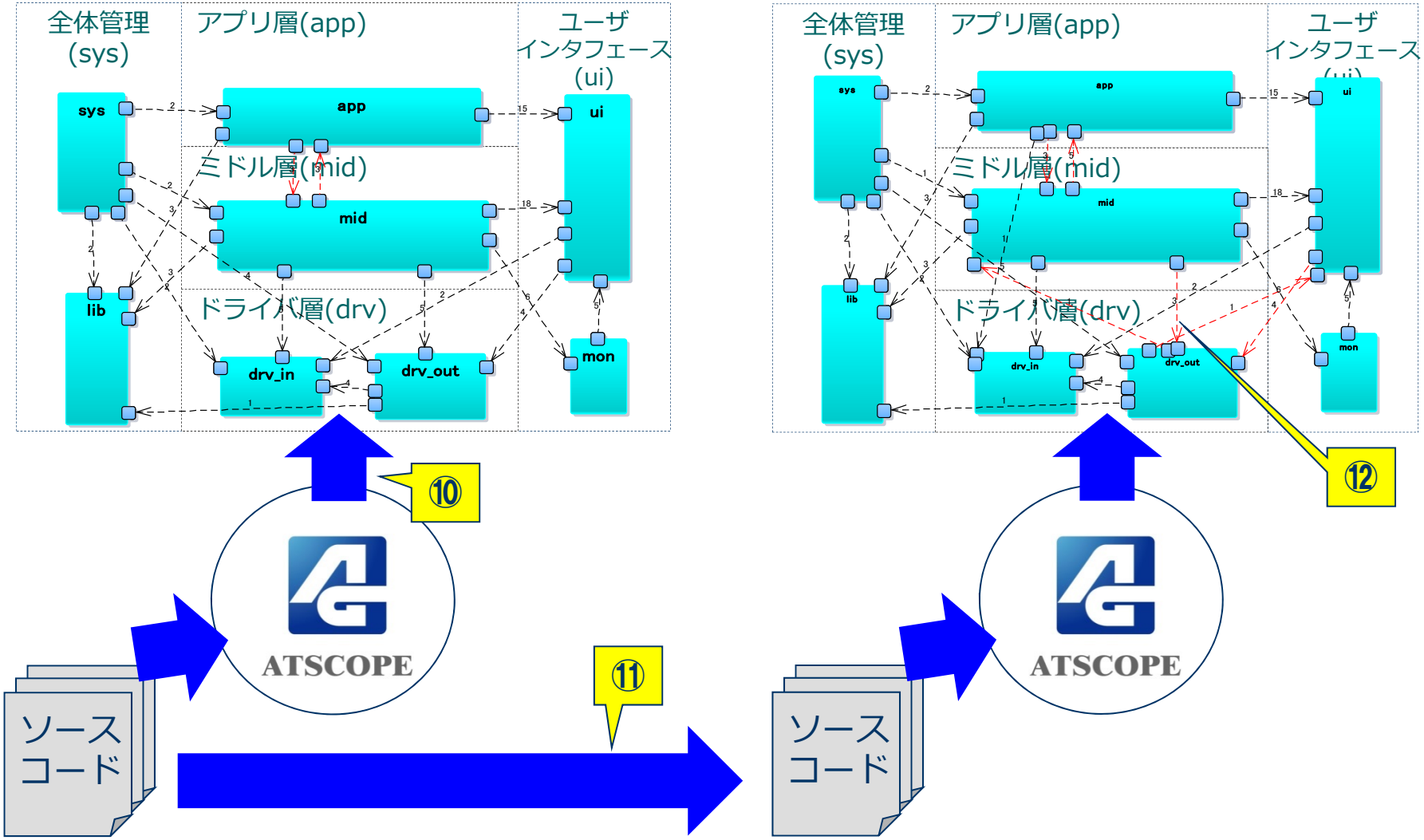
アーキテクチャ構築の主な成果物



設計意図発掘ツールの活用（左ページ）



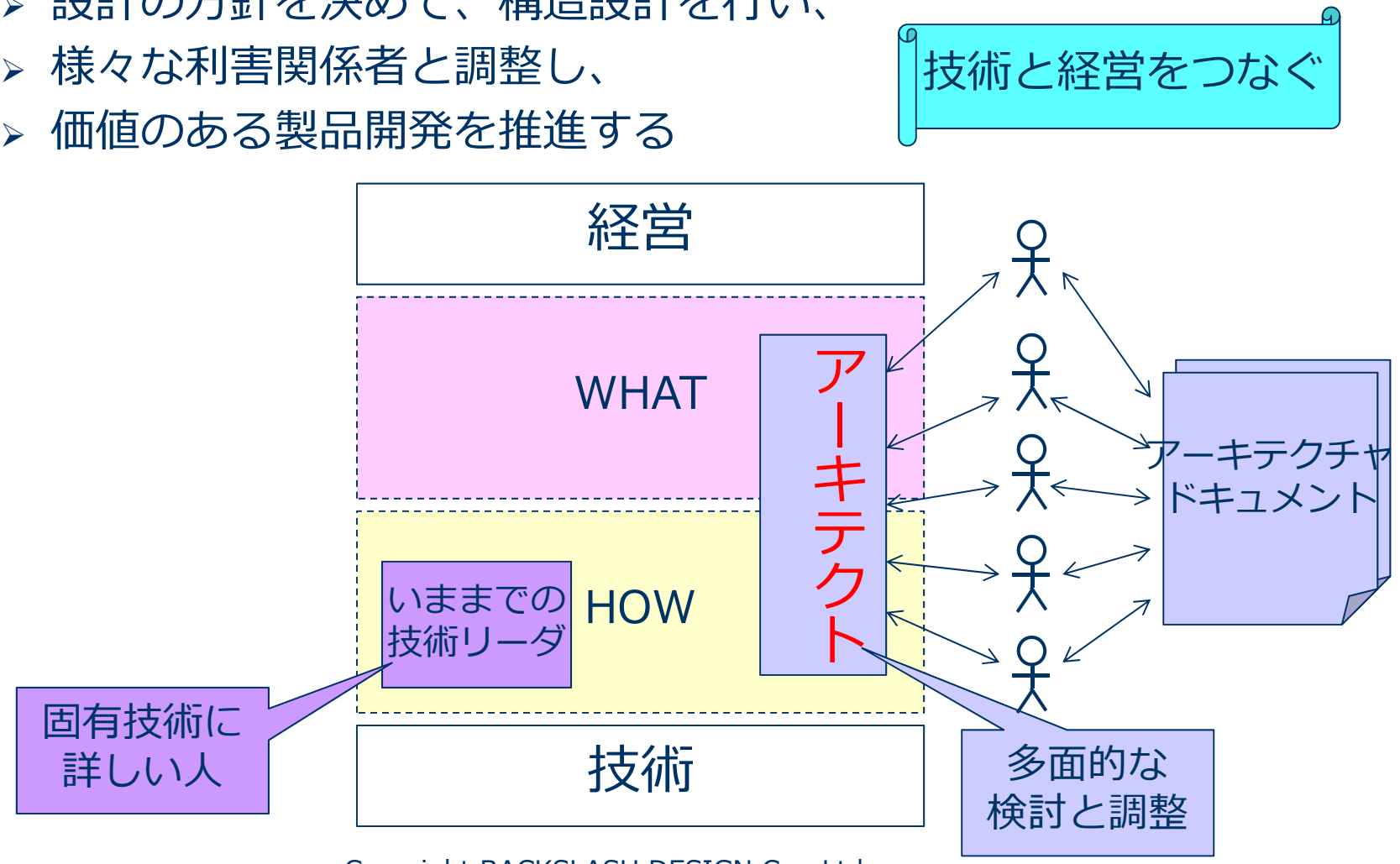
設計意図発掘ツールの活用（右ページ）



3．WhatとHowの補助線

アーキテクトとは

- 『何を、どのように、作るのか』のリーダーシップを発揮する人
  - 設計の方針を決めて、構造設計を行い、
  - 様々な利害関係者と調整し、
  - 価値のある製品開発を推進する

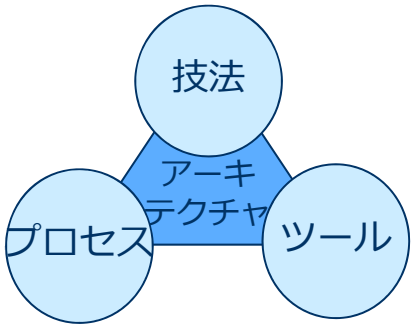


アーキテクチャ中心開発

既存資産を起点として戦略的な開発の実現へ

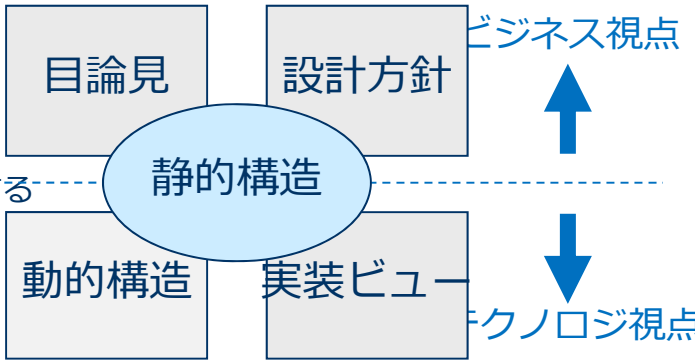
アーキテクチャを中心に、技法／プロセス／ツールを連動させます

- ボトムアップ：既存コードを部品化して、洗練化・資産化していく
- トップダウン：設計意図を明確にして、ソースコードへ反映していく



アーキテクチャ設計とは

- 複数ビューで設計意図を明確にします
- 静的構造で、全体アーキテクチャを俯瞰する
- 目論見と設計方針で、商品の特徴や差別化を定義する
- 動的構造で、パフォーマンスのボトルネックなど設計の勘所を明確にする
- 実装構造で、ソースコードの構成管理方法や書き方を統一する

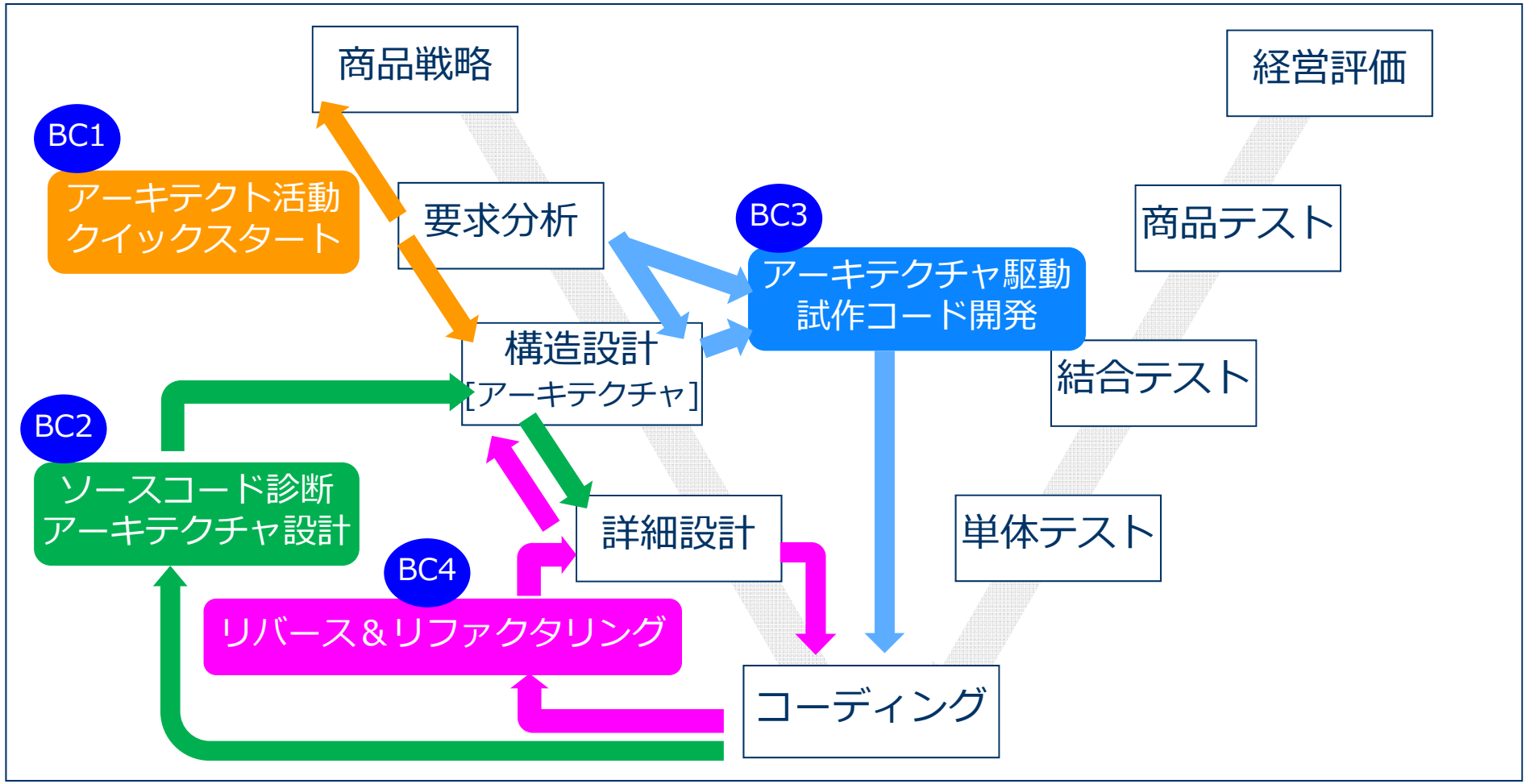


アーキテクチャ中心の試作駆動開発

- 常に動く状態を保ちながら、徐々に内部を作成していく方法です
  - コンポーネント間のインタフェースを定義する
    - 責務分割し、重要な引き渡すデータをデータ辞書で定義する
    - プログラミングしながら、インタフェースの妥当性を確認する
  - コンポーネント内はまずはスケルトンコードで繋ぐ
    - インタフェース部分を作り、いつでも動く状態を保つ（骨格）
    - 徐々に内部処理を書いていく（肉付け）

アーキテクト活動支援＆アーキテクチャ構築サービス

ビースラッシュではソースコード起点と要求分析起点でアーキテクト活動を支援します



主なセミナー