EdgeTech+ 2022

A3-05

11/18(金)14:30-15:15

派生開発におけるモデルの活用

株式会社エクスモーション　斎藤 賢一

派生開発推進協議会　星野 充史

Adenda

- 世の中の変化に対する開発の現状 斎藤
- モデリングによる素早い設計判断
  - コードを見える化し、開発効率アップ
  - 設計を劣化させない、モデル変更のポイント
  - モデルを活用した価値変化への対応
- モデル活用による素早い設計判断のための派生開発ナレッジ適用事例
  - 開発の状況
  - モデルの活用

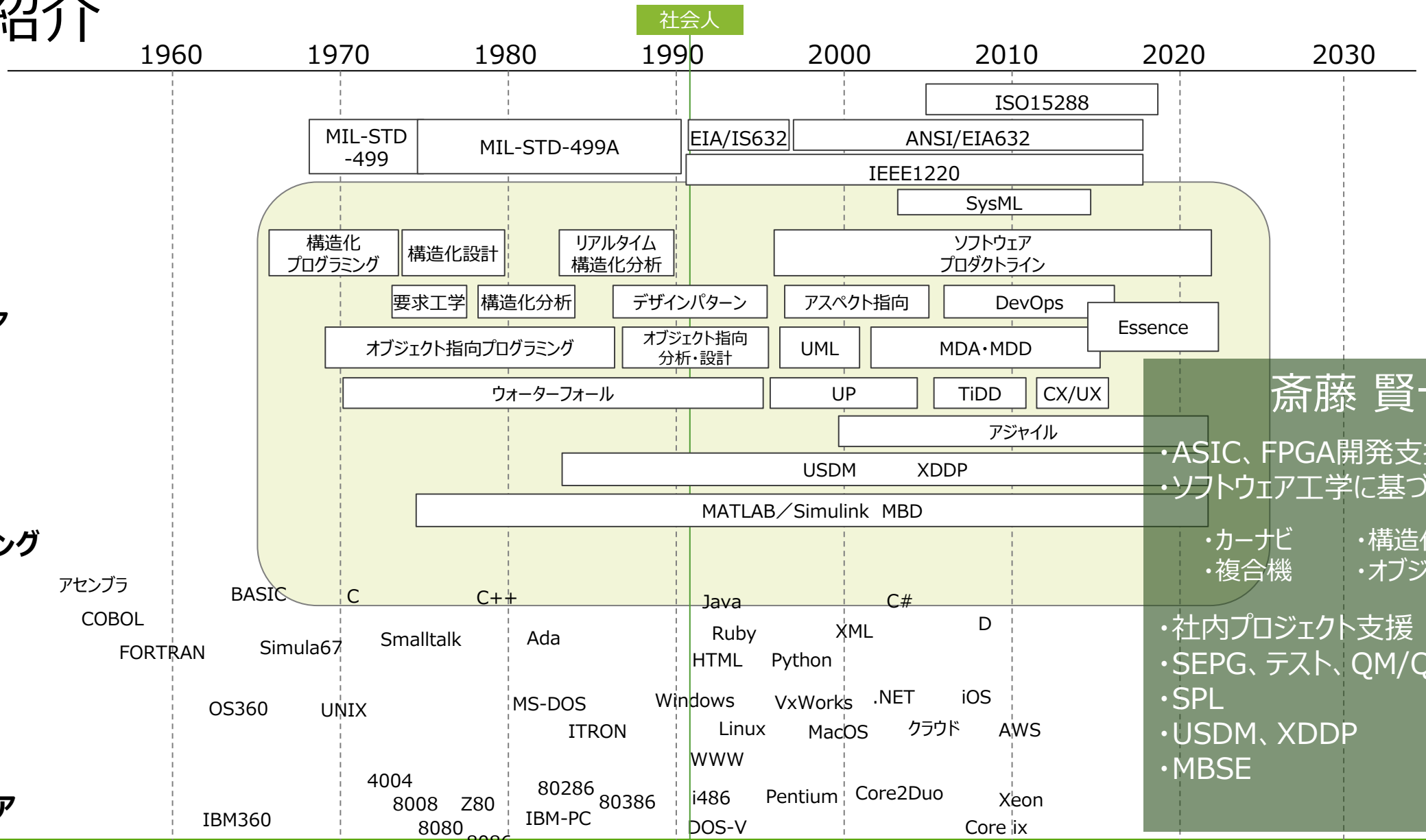
星野

Adenda

- 世の中の変化に対する開発の現状 斎藤
- モデリングによる素早い設計判断
  - コードを見える化し、開発効率アップ
  - 設計を劣化させない、モデル変更のポイント
  - モデルを活用した価値変化への対応
- モデル活用による素早い設計判断のための派生開発ナレッジ適用事例
  - 開発の状況
  - モデルの活用

星野

自己紹介



斎藤 賢一

- ASIC、FPGA開発支援
- ソフトウェア工学に基づく製品開発
  - カーナビ
  - 複合機
  - 構造化手法
  - オブジェクト指向
- 社内プロジェクト支援
- SEPG、テスト、QM/QA
- SPL
- USDM、XDDP
- MBSE

世の中の変化に対する開発の現状

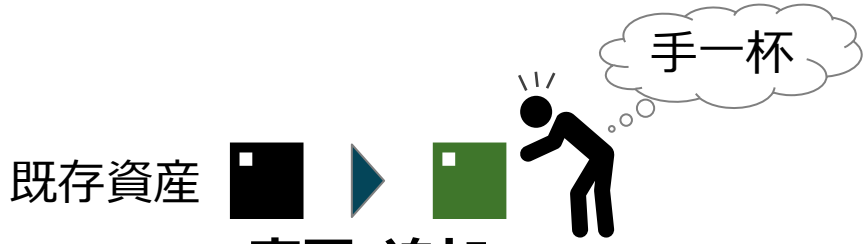
デジタル変革・データ活用により、
顧客価値の創出**スピードは速まる**ばかり

製品**そのもの**が価値 ▶ 製品による**体験**が価値

プロダクトライフサイクル　成熟期 5年から10年 ▶ 成熟期 **1年から2年**

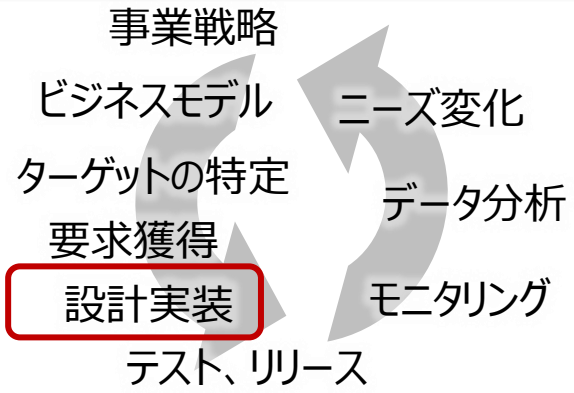
しかし

システムは**大規模**かつ**複雑化**し、既存資産の**メンテナンスで手一杯**な開発も多い



現状の開発を**効率的**にこなし、**顧客価値**の創出スピードを**上げていく**には

モデリングによる素早い設計判断



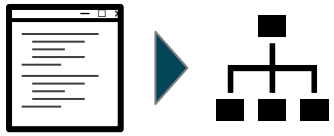
現状の開発を効率的にこなし、顧客の価値創出スピードを上げていくには

解決策の1つ　**モデリング**により**素早く設計判断**を行う

3つの開発状況に応じたモデルの活用

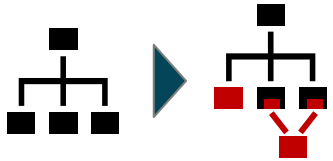
これからモデルを活用する

コードを見える化し、
開発効率化アップ



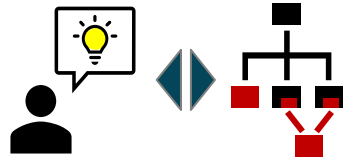
すでにモデルを活用している

設計を劣化させない、
モデル変更のポイント



価値変化にすばやく対応

モデルを活用した
価値変化への対応



ちなみに、開発工程とモデル・モデリング言語

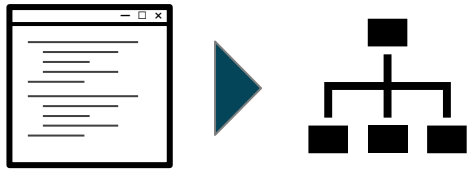
工程	モデル	モデリング言語	
事業戦略	ビジネスモデル	BPML	
市場・商品・技術戦略	サービス・製品 コンセプトモデル	ConML	
システム開発	システム要求モデル	SysML	
	システム設計モデル		MATLAB / Simulink
ソフトウェア開発	ソフトウェア要求モデル	UML	
	ソフトウェア設計モデル		MATLAB / Simulink
	詳細設計モデル／コード		

これからモデルを活用する

コードを見える化し、開発効率アップ

コードの見える化　モデルで表現

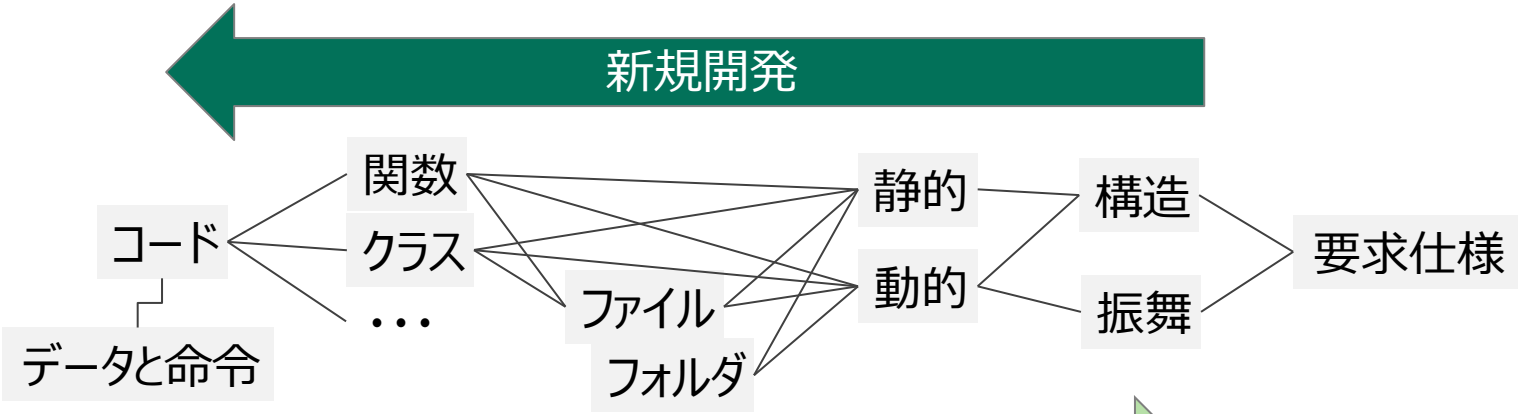
コードをモデルで表現

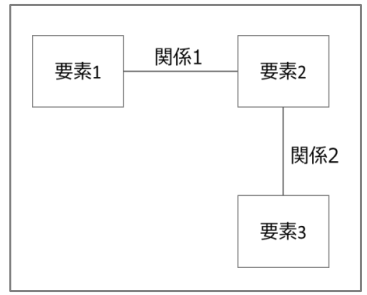


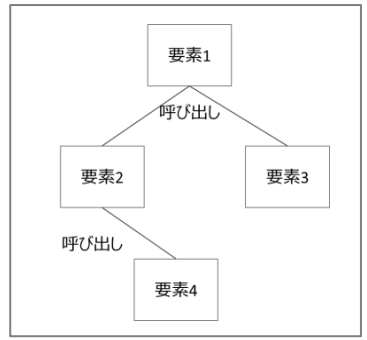
- 重要な部分を浮かび上がらせ、理解を促進する
- 記憶に残りやすい
- 変更の影響に気づきやすくなる

四角 —— 意味のある括り、まとまり（要素）

線 —— 要素間の関係







（資産がコードのみの）派生開発 → **変更部分のみ**を明らか ← スペックアウト

リバース → 全部もしくはある範囲のみ

四角と線で図にしてみる

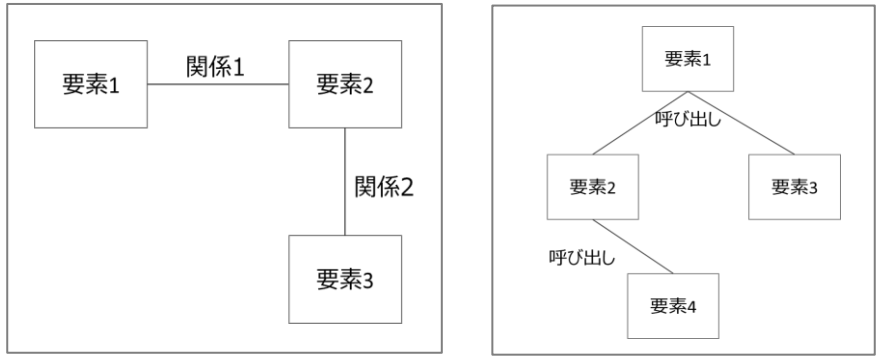
四角と線で図にしてみる

最初は表記法が間違っていても構わない
以下の2つの観点でモデルを書いて、記憶に残し、レビューする

▶ 変更の正しさを早期に確認

構造

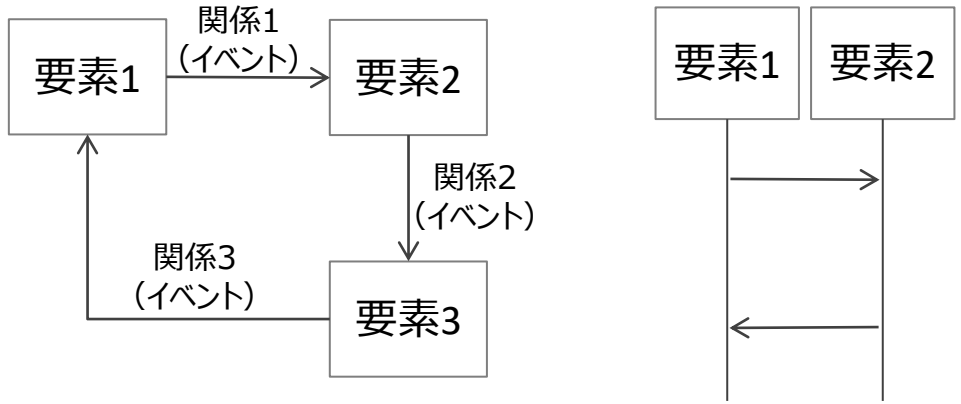
意味的な要素と関係性



シナリオ非依存

振舞

時間的な要素と関係性

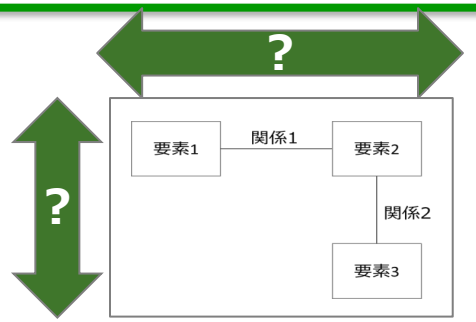


シナリオ依存

コードを調べた後はモデルを描くモチベーションが沸かなくなるため、
コードを**調べながらモデリング**する

軸を意識すると理解度アップ

縦軸・横軸の**意味**を意識してモデリングすることで、
メンバー内で共有すると理解性が向上し、レビュー効率が上がる



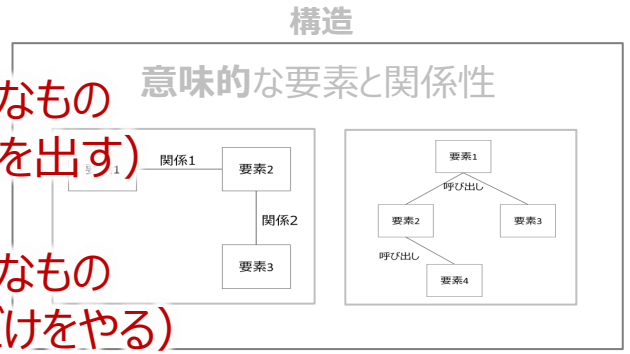
軸の例

構造

横軸
左：入力　　右：出力

縦軸
上：意味的に重要なもの
（判断して指示を出す）

下：意味的に簡素なもの
（言われたことだけをやる）

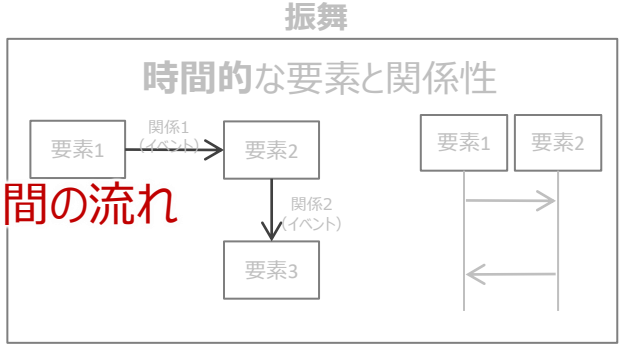


振舞

横軸
左⇔右：時間の流れ

縦軸
上
⇕
下

時間の流れ



すでにモデルを活用している

設計を劣化させない、モデル変更のポイント

設計劣化を未然に防ぐ

変更前

四角 —— 意味のある括り、まとまり（要素）

線 —— 要素間の関係

変更時に崩さない

変更後

意味のある括り、まとまり（要素）

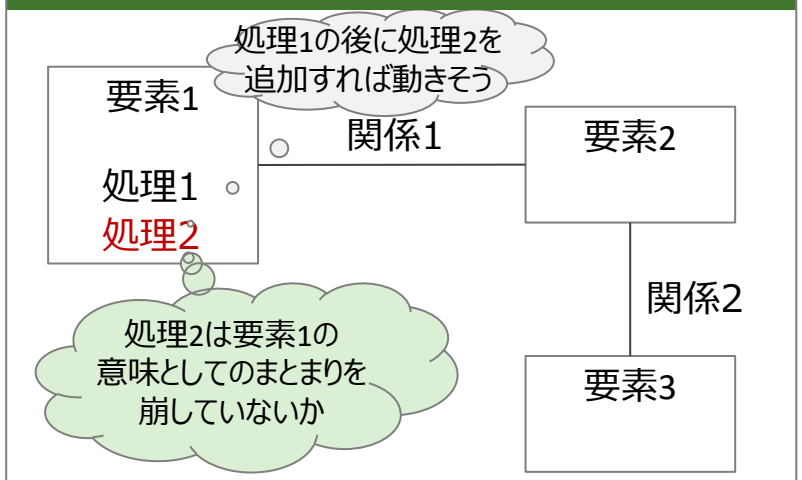
要素間の関係

変更要求仕様に対して

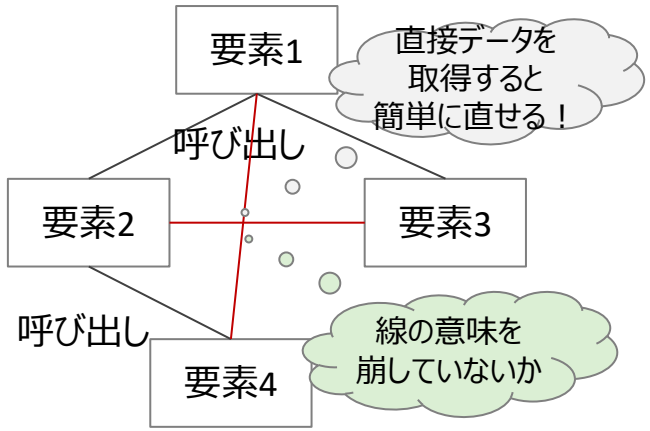
- 変更後も**四角の意味**が設計観点上で**単一**
- 変更後も**線の意味**が「**四角間の関係**」を表している

よくある「設計が**崩れる**傾向」

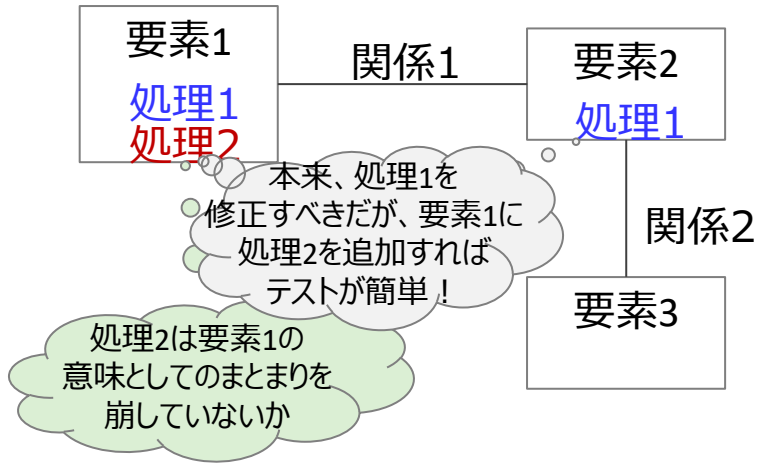
ここ新しい処理を追加すれば**動きそう**



データを直接取得、修正が**少なく済む**



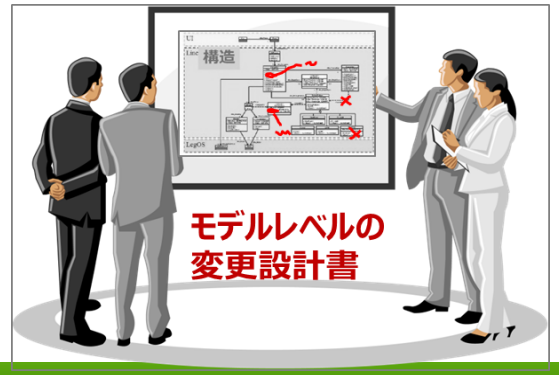
テストが大変 変更箇所は**最小限に**



モデルをいきなりいじらない

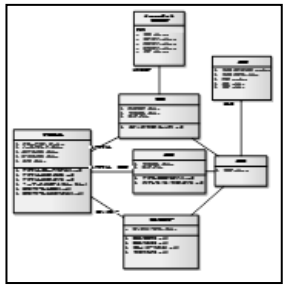
設計は要求に対する**最適解**
変更に対する個別対応では最適解は得られない

どのように変更しようとしているかを**メモ**で残し、全体最適解を検討する



新規開発の場合

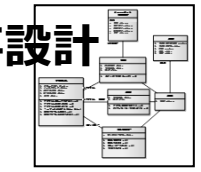
要求仕様1
要求仕様2
要求仕様3
・・・

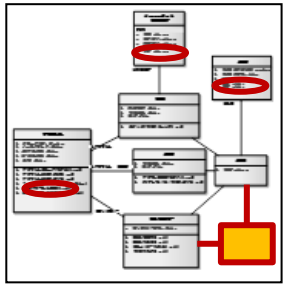


要求仕様全体に対する**最適解**を検討

変更時は？

変更要求仕様1
変更要求仕様2
変更要求仕様3
・・・
＋
既存設計





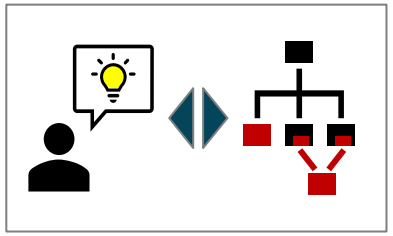
変更設計

変更要求仕様全体と**既存設計**に対する**最適解**を検討

価値変化にすばやく対応

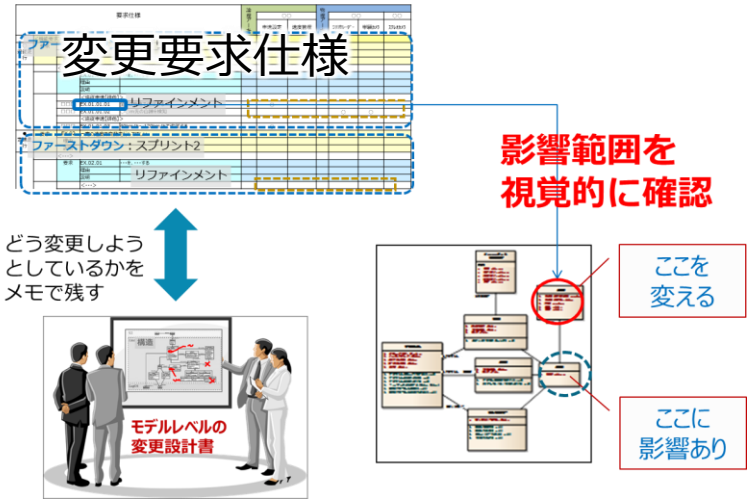
モデルを活用した価値変化への対応

モデルを活用した価値変化への対応

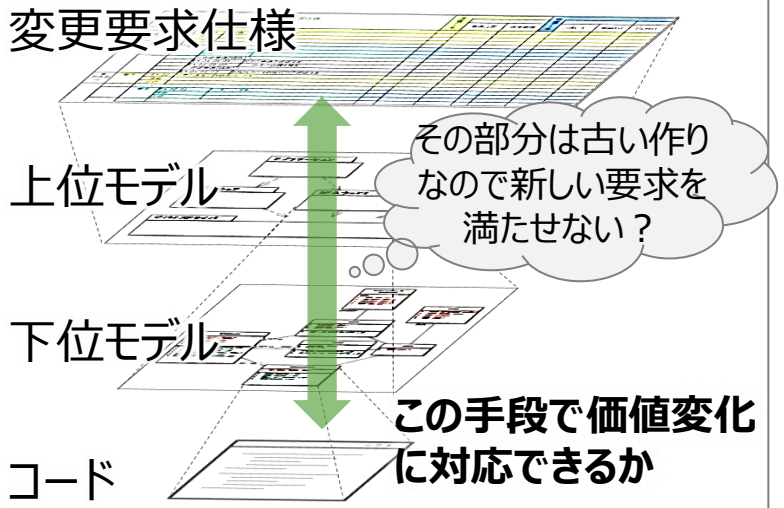


ニーズの変化を捉え、素早く解決策に落とし込むには、
モデルの活用により設計を判断する

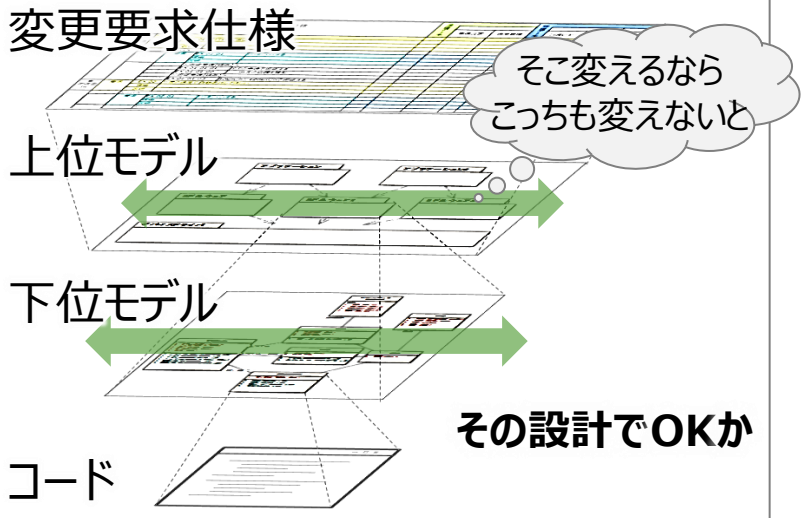
価値→要求と設計のトレース



実現手段への落とし込み検討



変化対応への設計判断



Adenda

- 世の中の変化に対する開発の現状 斎藤
- モデリングによる素早い設計判断
  - コードを見える化し、開発効率アップ
  - 設計を劣化させない、モデル変更のポイント
  - モデルを活用した価値変化への対応

- モデル活用による素早い設計判断のための派生開発ナレッジ適用事例
  - 開発の状況
  - モデルの活用

星野

背景

モデルを活用した背景

企業にて開発力向上を目的とした有志のワーキンググループ(WG)が発足
テーマ「**モデルを活用した製品開発の改善**」
活動期間は1年間
製品開発チームをWGメンバーが支援しながら実践

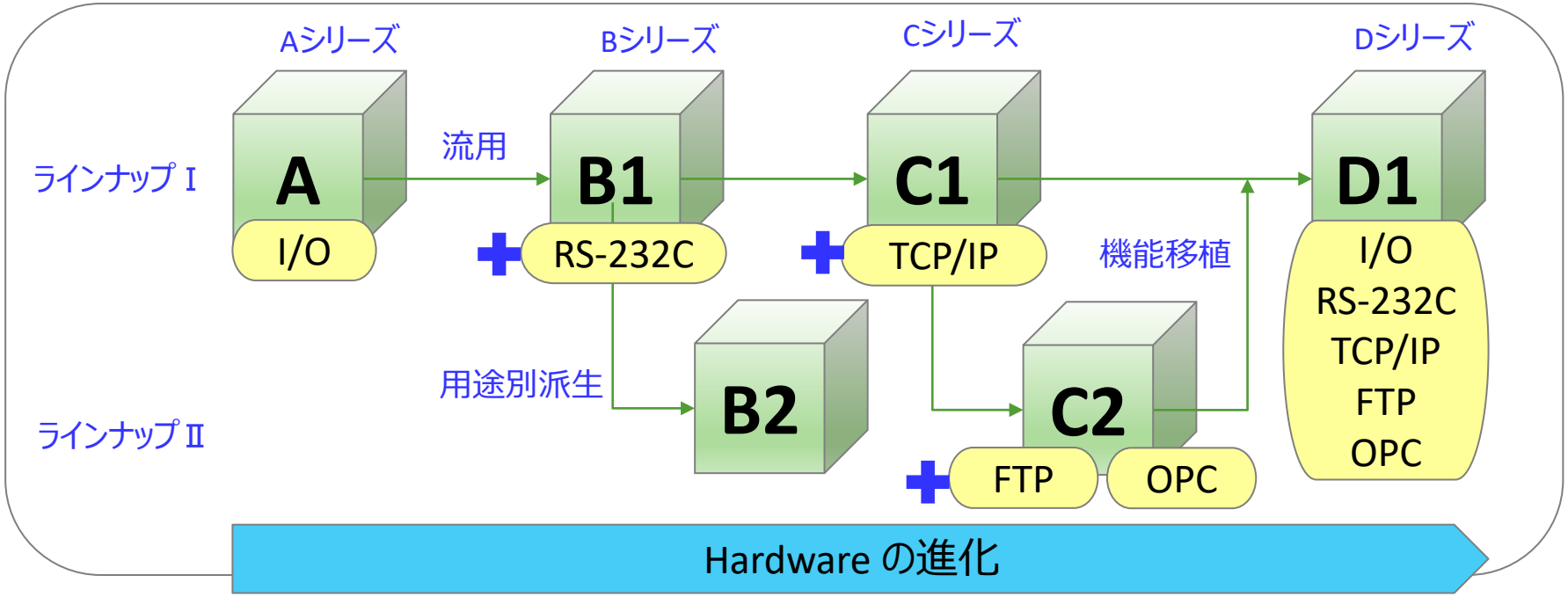
1年を掛けてモデル駆動型の変更設計に取り組み、
早期に設計判断できる仕組みを構築した事例

開発の概要

製品：User Interfaceを有する検査機器
　　25年以上開発が続くシリーズ製品
　　多彩な製品ラインナップ・オプション機器と連動
開発：C言語を主とした組み込み型の開発
　　プログラム流用、移植を繰り返して開発
　　開発期間は1カ月程度が主流
開発チーム：3名
　　2017年から派生開発プロセス(XDDP)に取り組む

製品の特長

製品の多様化と機能移植の開発例



長年の派生開発で**プログラム構造の秩序を保つことは容易ではない**
　モジュールの依存関係の複雑化、複数責務による意図しない巨大化
　グローバル変数の多用化と排他処理の増殖、クローンコードの増殖
　状態変数やフラグ条件分岐の増殖、デッドコードの存在

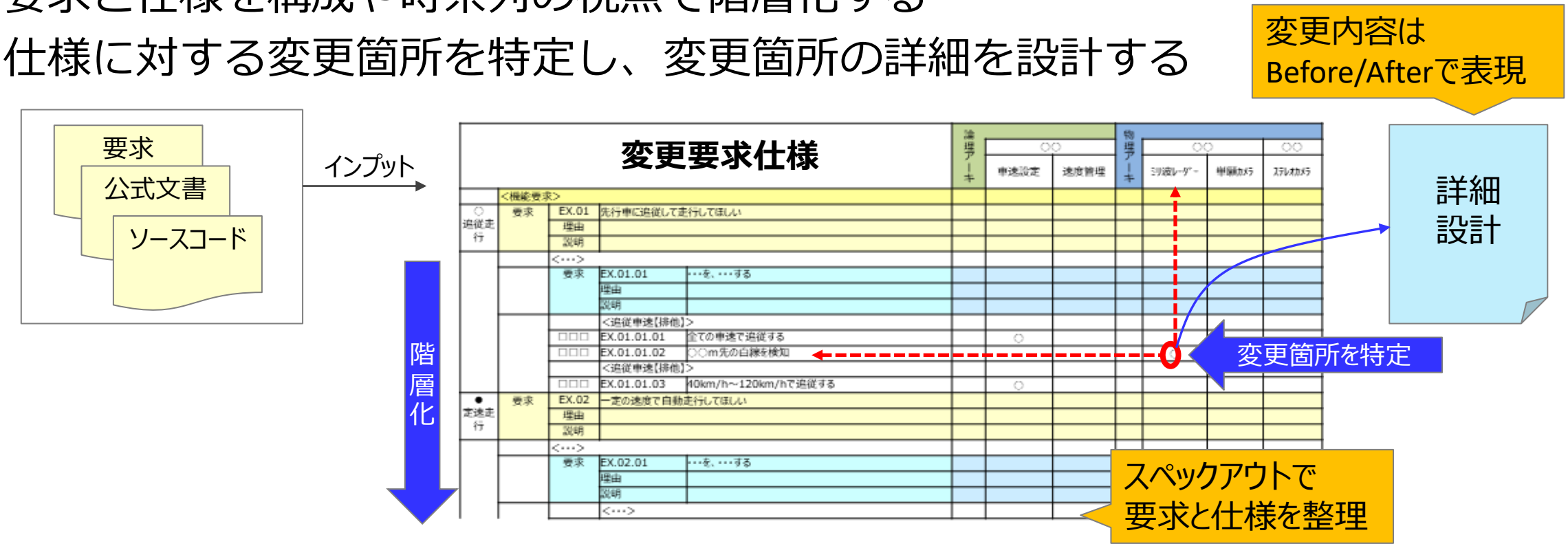
モデルを活用する以前の開発チーム

開発の状況

XDDPによる製品の派生開発

要求から仕様を導出し、変更箇所を特定する

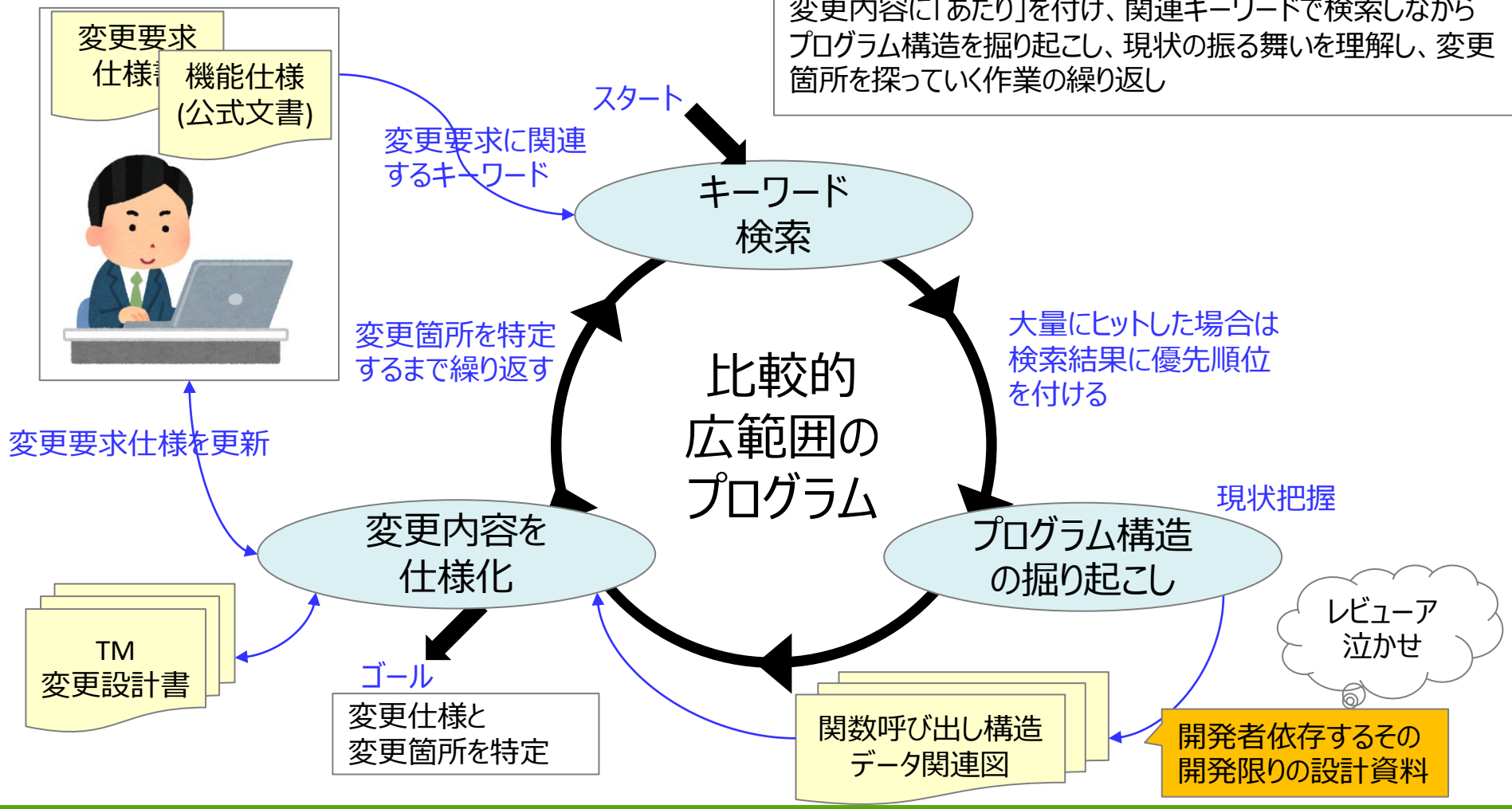
- 要求を目的語と動詞で記述し、要求の背景や理由を捉える
- 要求と仕様を構成や時系列の視点で階層化する
- 仕様に対する変更箇所を特定し、変更箇所の詳細を設計する



変更設計の状況

キーワード検索よるスペックアウト

変更内容に「あたり」を付け、関連キーワードで検索しながらプログラム構造を掘り起こし、現状の振る舞いを理解し、変更箇所を探っていく作業の繰り返し



問題点

長年の派生開発でプログラム構造は劣化し
プログラム構造を理解するための情報も不足していた

変更設計の問題点

- キーワード検索に頼った非効率なスペックアウト
  ⇒**変更設計に時間が掛る**
- 現構造における適切な変更箇所の判断が困難
  ⇒**プログラム構造の劣化が進行する**

限られた開発期間の中で、効率よく変更設計したい
⇒しかし、劣化した構造はそう簡単には改善困難

プログラム構造を徐々に理解しながらモデルを活用した変更設計へ変えていく

モデルの活用

モデルの活用

第一歩

**プログラム構造を可視化し、理解することで
これ以上劣化させない仕組みを構築する**

モデルに期待すること

- 適切な範囲でスペックアウトするための “道しるべ” である
- 設計レビューにおける変更箇所の妥当性を確認できる

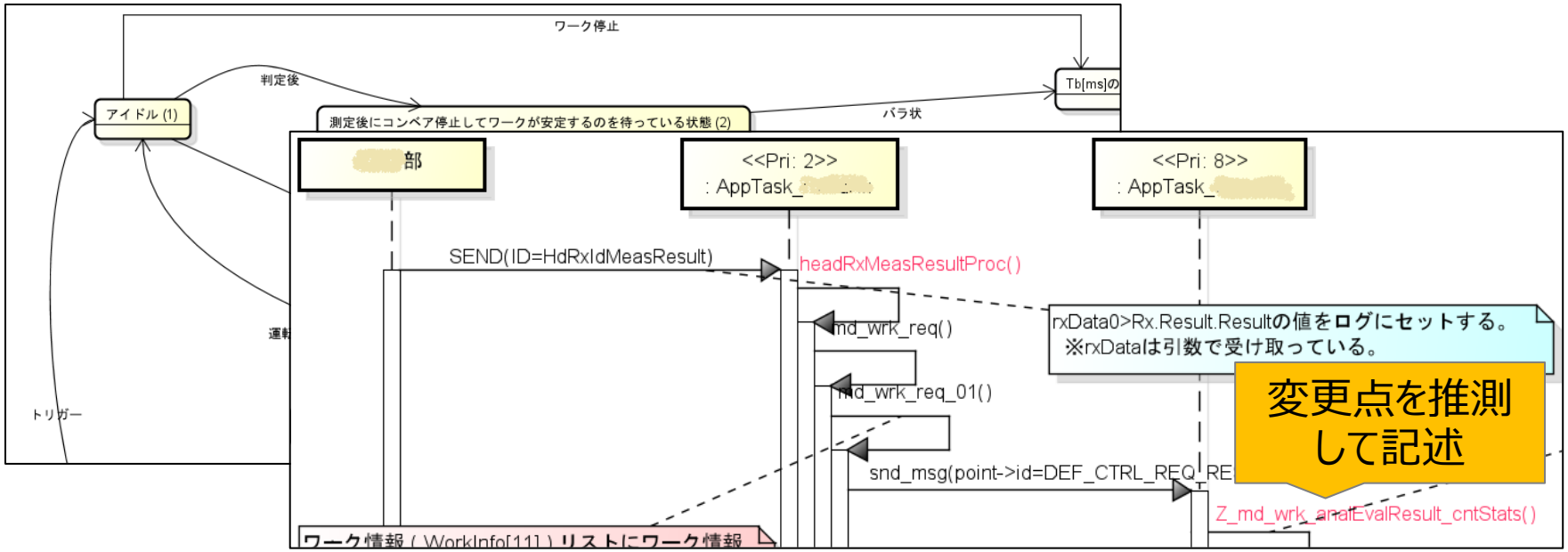
取り組み方針

- UMLにより表記法を統一
- 無理なく継続可能な取り組み

モデルの活用

プログラム構造を可視化

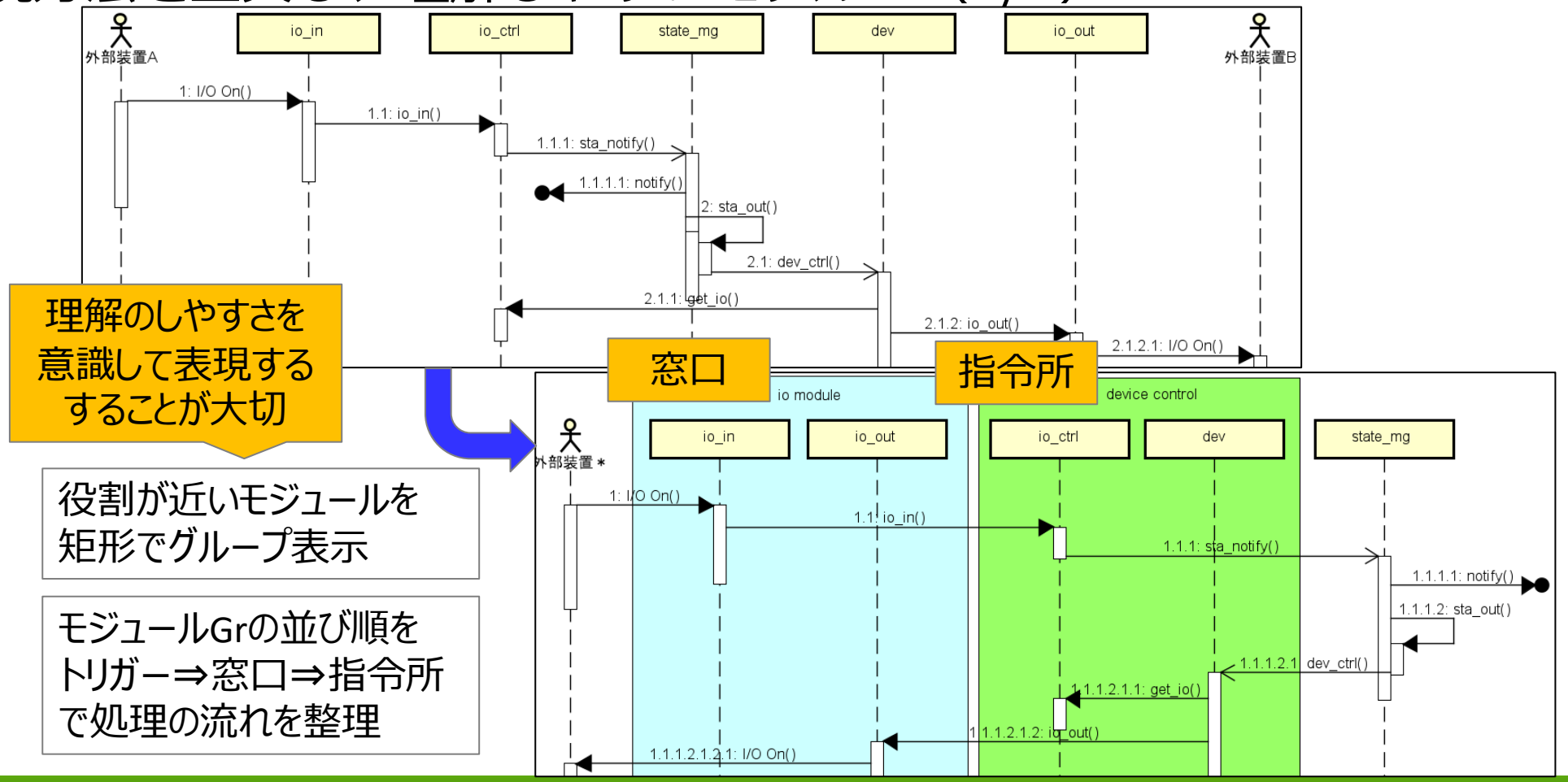
- 機能の振る舞いごとに現構造のモデルを作成
- 機能仕様書で可視化した振る舞いを確認
- 変更要求に該当する箇所に変更点を記述
- 開発チームの中でモデルを共感することを重視



モデルの活用

プログラム構造を徐々に理解

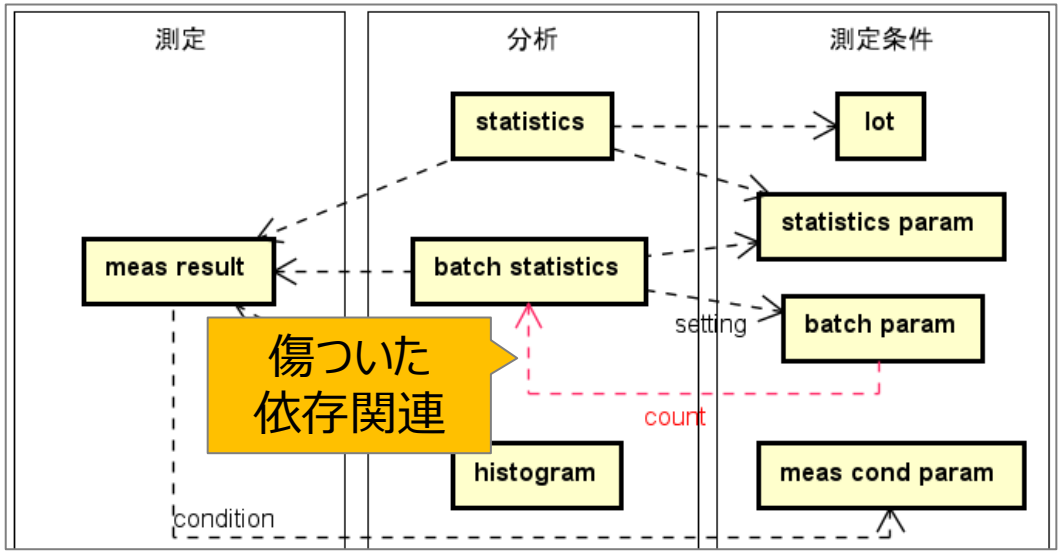
表現方法を工夫し、理解しやすいモデルへ（1/2）

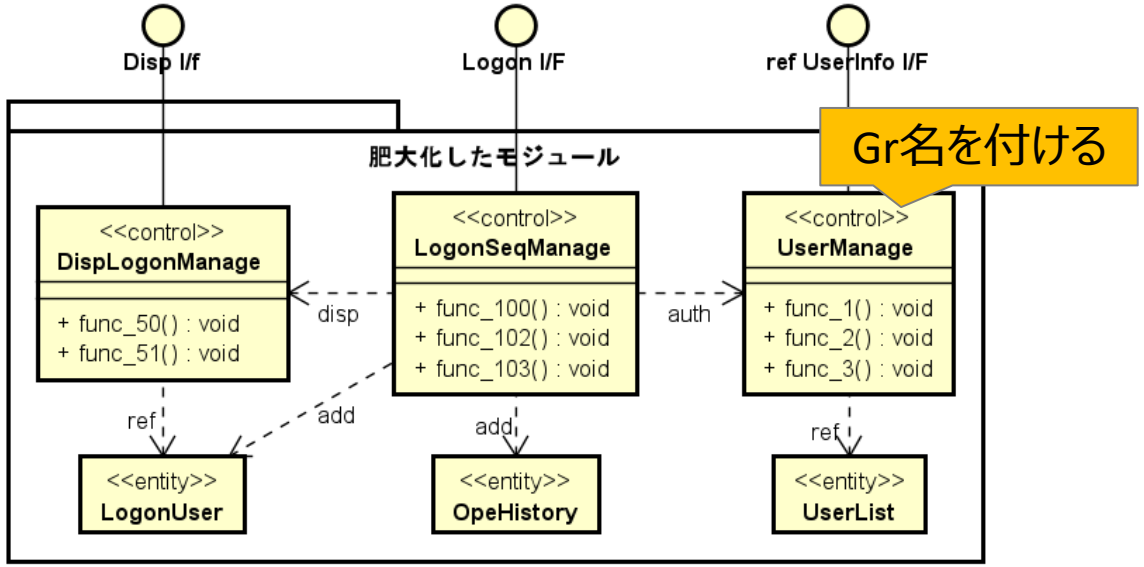


モデルの活用

プログラム構造を徐々に理解

表現方法を工夫し、理解しやすいモデルへ（2/2）





データ依存関係の着目点

- データを機能の境界で表現
- 暗黙知として埋もれやすい情報も表現
- クローンデータの存在を表現

肥大化モジュールを細分化する着目点

- 制御処理とデータ処理の関数Grに分けて表現

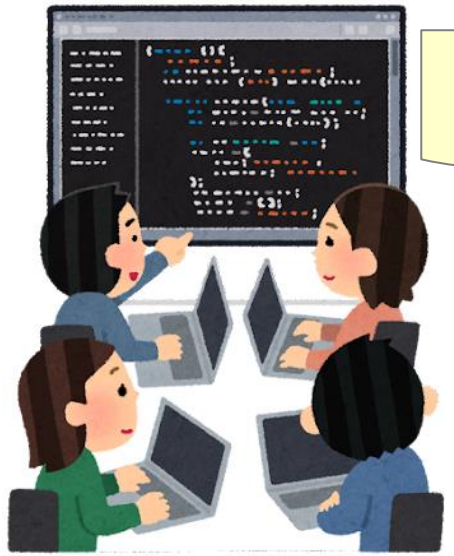
モデルの活用

書き溜めたモデルを活用して変更方法を検討

開発チーム内で変更内容に“あたり”を付ける

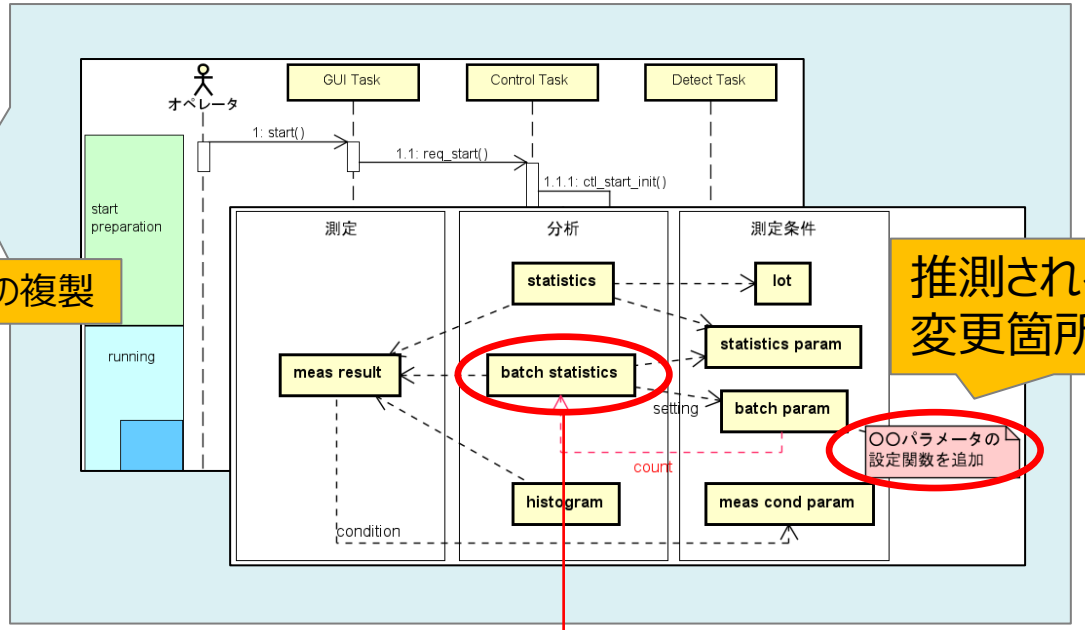
要求

機能仕様
(公式文書)

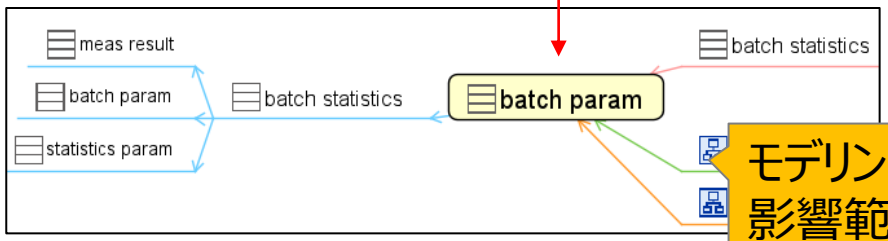


モデル
(開発用)

公式文書の複製



推測される
変更箇所にマーク



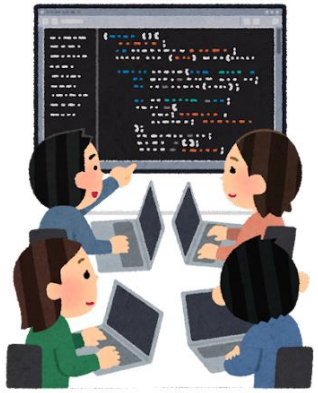
モデリングツールの機能を使い
影響範囲を確認

要求に対する変更内容を共感
変更要求仕様書へ要求と理由を記載

モデルの活用

モデルを活用した変更設計

設計資産のモデルから変更範囲を絞り込み
仕様をスペックアウトする。同時にモデル(開発用)を更新し、
開発後に公式文書へ反映する。



機能仕様
(公式文書)

変更要求
仕様書

モデル
(開発用)

スタート

モデルから変更
範囲を絞り込み

絞り込んだ範囲を
順に解析

必要な情報を
スペックアウト

設計メモ

ゴール
変更仕様と
変更箇所を特定

変更内容を
仕様化

変更要求仕様を更新
モデルに変更箇所を記載

変更箇所を特定
するまで繰り返す

変更範囲を
絞り込んだ
プログラム

変更箇所の
現状を理解

TM
変更設計書

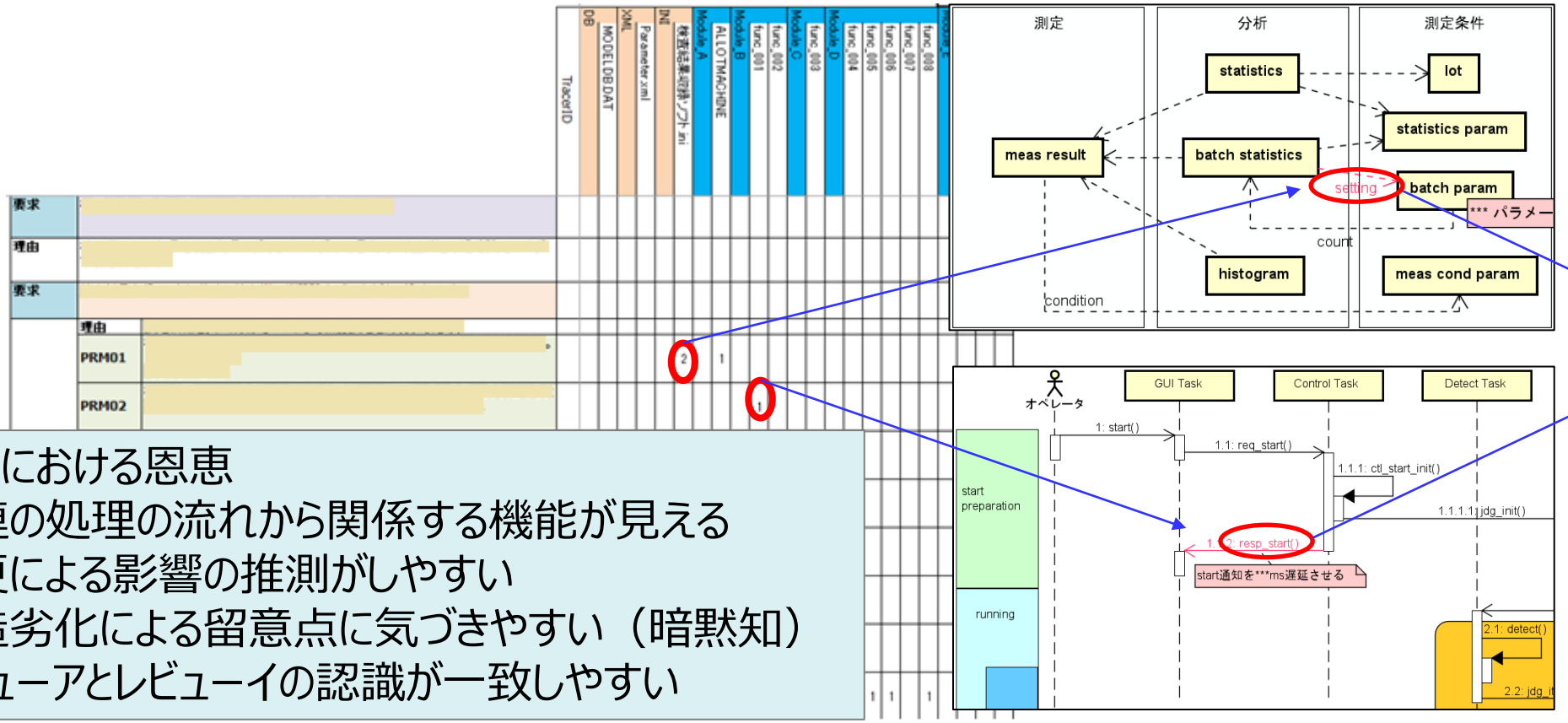
開発後
公式文書へ反映

モデル
(公式文書)

モデルの活用

モデルを活用した設計レビュー

要求から導出した仕様に対し、変更箇所とモデルを付け合わせて
プログラム構造における適切な箇所かどうかを確認する



レビューにおける恩恵

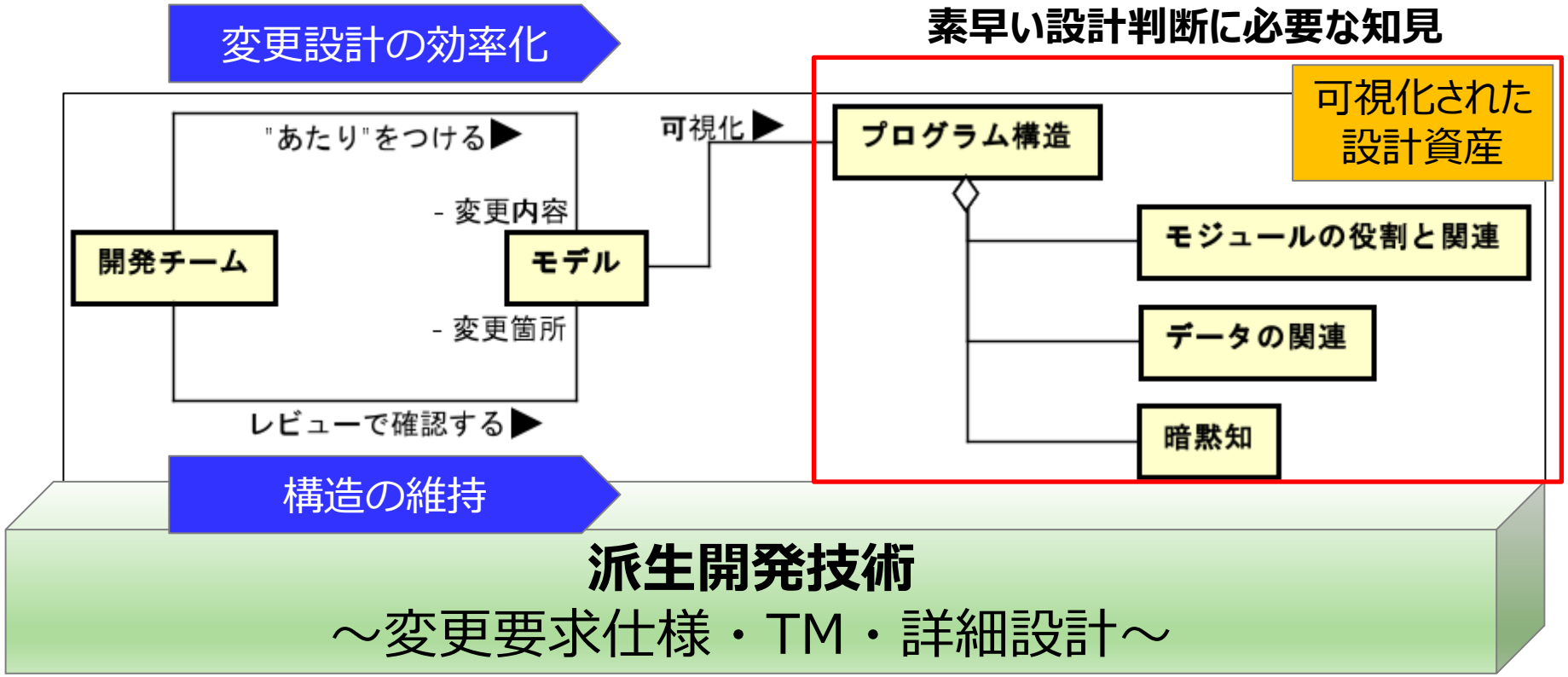
- 一連の処理の流れから関係する機能が見える
- 変更による影響の推測がしやすい
- 構造劣化による留意点に気づきやすい（暗黙知）
- レビューアとレビューイの認識が一致しやすい

まとめ

モデルを活用した効果

効果１

- 変更設計に掛かる時間を短縮できた
- プログラム構造を維持しやすい変更設計に変化してきた



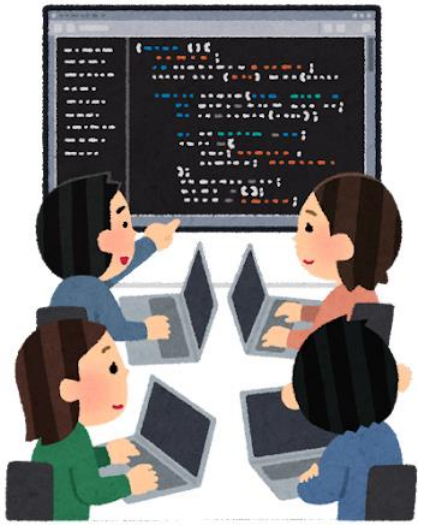
モデルを活用した効果

効果２

- 開発チームのコミュニケーションが活性化してきた
  ⇒設計技法やモデリングツールの使い方の共有
  　モデリング技術習得に対するモチベーション向上

開発チームにとってモデルは

共通理解しやすい設計支援ツールであり
プログラム構造の可視化により設計を議論できる場である



今後の課題

プログラム構造を維持しやすくなったとは言え、、、

- 傷ついた構造は徐々に劣化していく
- ひどく傷ついた構造はモデルも複雑なため効果も薄い
- 機能の凝集度が低いモジュールはテスト量も減らない

さらに、
製品シリーズはプログラム流用・機能移植を繰り返えす
今できることを少しずつ取り組むことが大切

課題

設計資産のモデルから得た知見を活用し
より変更設計しやすいプログラム構造へ改善していく
～劣化したプログラム構造のリファクタリング～

ご清聴
ありがとうございました