課題解決型アーキテクチャ事例とアーキテクト育成の取り組み

１．課題解決型アーキテクチャ

２．アーキテクチャ事例紹介

３．アーキテクト育成の取り組み

４．まとめ

三菱電機メカトロニクスソフトウエア(株）
和歌山支所　　岩橋正実
Iwahashi.Masami@wak.msw.co.jp

１．課題解決型アーキテクチャ

モデル・アーキテクチャ・アーキテクト

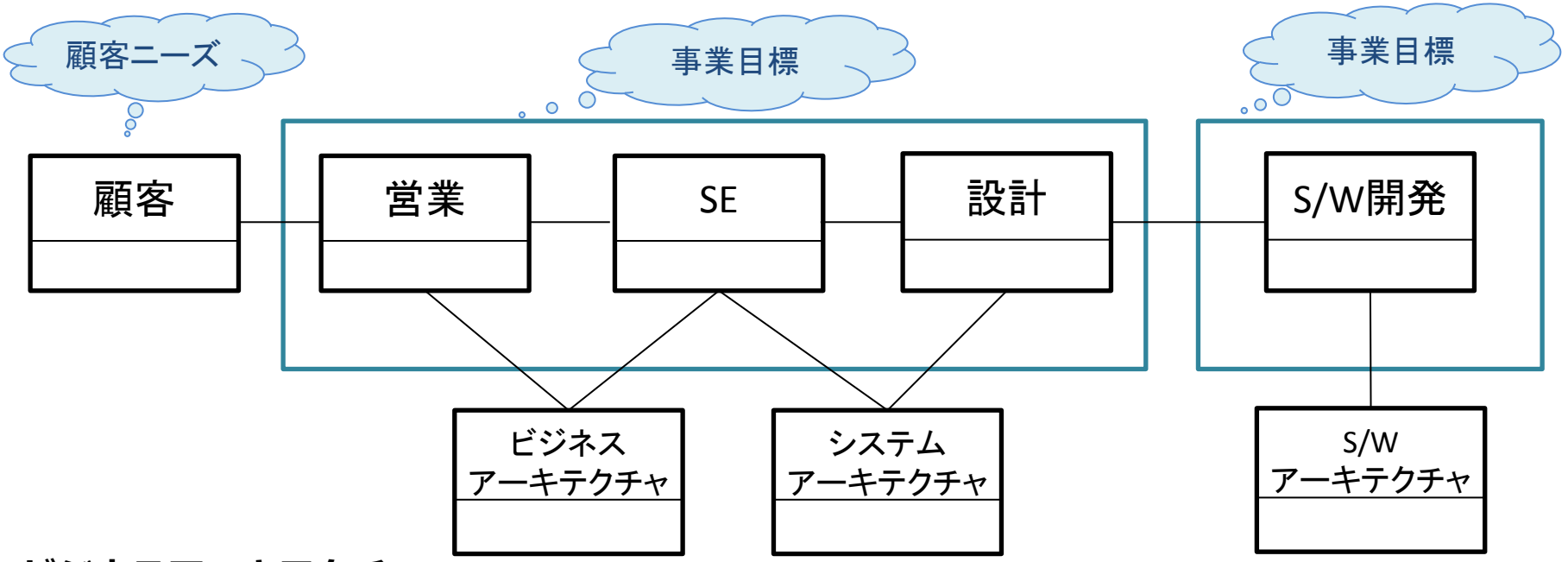
モデル
ソフトウェアで実現したい機能を定義して機能を実現する
ソフトウェアの構造と振る舞いの定義。

アーキテクチャ
特定の目的を達成する指針に基づく要素の構造と振る舞い。

アーキテクト
組織ゴールを達成する為のアーキテクチャを開発できる人。

アーキテクチャの種類

アーキテクチャは、特定の要求を達成する為の要素の構造と振る舞い。
要素の粒度でビジネス、システム、ソフトウェアのアーキテクチャがある。



ビジネスアーキテクチャ
顧客ニーズに基づき事業目標を達成する為にキャッシュフローを含むアーキテクチャを構築。

システムアーキテクチャ
顧客ニーズに基づき事業目標を達成する為に社会インフラ含め複数のサブシステム間のアーキテクチャを構築。

ソフトウェアアーキテクチャ
顧客ニーズに基づき事業目標を達成する為のアーキテクチャの構築。

顧客ニーズ及び事業目標をビジネス/システム/ソフトウェアのアーキテクチャで実現。
ビジネスを成功させる為の要求を高信頼性と高生産性で実現するアーキテクチャが大切。

課題と課題解決方法

課題を抑制するアーキテクチャを用いることでソフトウェアの高品質/高生産性を実現する

課題		課題解決方法
開発のバラツキ	要求定義のバラツキ	要求分析定義手法
	工程間の変換バラツキ	シーケンスパターン
重複記述		クラス分析設計手法
要求・S/Wの発散		SPL（ソフトウェアプロダクトライン）
文書精度（曖昧表現）		日本語による形式記法の推進DSL
要求精度		目的指向開発
競合及び制約	機能競合	競合分析設計
	出力競合	
	例外競合	
	時間制約	絶対時間分析設計
安定しない要求	段階的仕様詳細化	仕様安定度に基づくプロセス最適化（ｲﾃﾚｰﾃｨﾌﾞ開発）
	段階的仕様追加	変更粒度によるプロセス最適化（インクリメンタル開発）
見積り精度		見積り手法

アーキテクチャとプロセス

アーキテクチャ

特定の課題を解決するための構造と振る舞い

アーキテクチャは、要素と要素間の関連で表現される。
課題解決の為のアーキテクチャを構築することが大切。

※分析・設計・試験のアーキテクチャは**様式（文書フレーム）**で確立する。
※実装のアーキテクチャは、**フレームワーク**で確立する。

開発プロセス

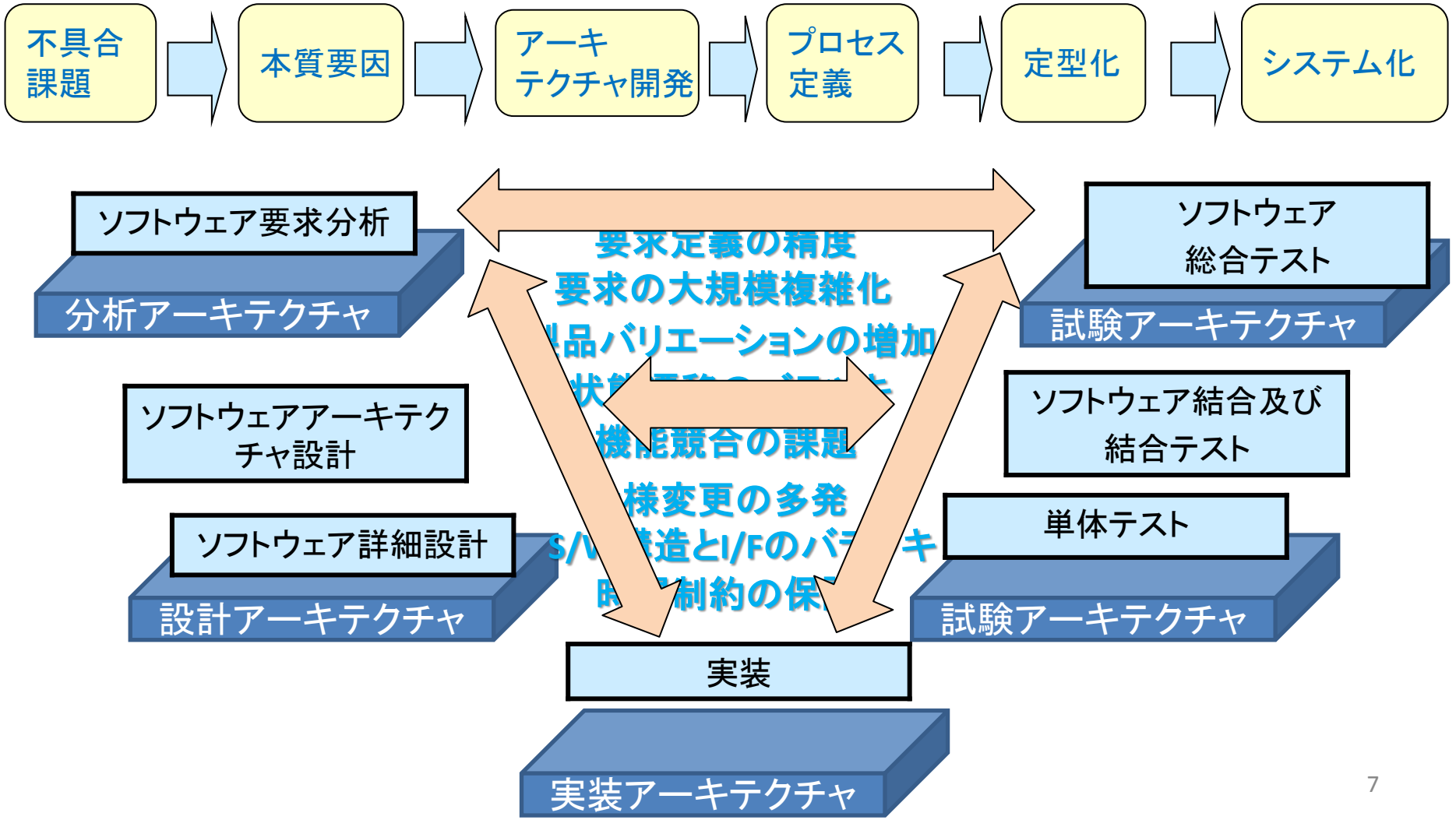
アーキテクチャを指針に基づき正しく使用する作業手順

分析：要求を分析アーキテクチャへ落し込む手順
設計：分析を設計アーキテクチャへ落し込む手順
実装：設計を実装アーキテクチャへ落し込む手順
試験：分析・設計を試験アーキテクチャへ落し込む手順

課題解決のアーキテクチャの使い方を開発プロセスで定義する事で組織的な適用効果が得られる。更にアーキテクチャの発散の抑制にも繋がる。

課題解決型アーキテクチャ開発

不具合の発生源と作り込み要因を分析して、再発させない
アーキテクチャ定義と開発プロセスの定義が重要。



アーキテクチャ開発プロセス

アーキテクチャで解決する目的/課題を定義。目的/課題解決を達成する指針を定義した上でアーキテクチャ定義とタスクの定義を行う事により組織全体の品質を改善する。

アーキテクチャ定義：目的/課題＋指針＋モデル

1. 目的指向アーキテクチャ開発プロセス

2. 課題解決アーキテクチャ開発プロセス

目的指向アーキテクチャ開発プロセス

顧客要求及び組織目標を定義。現状とのギャップ分析を実施して目的(実行目標)を定義する。
目的達成の指針とアーキテクチャ定義と目的達成のタスク定義により組織全体の品質を改善する。

アーキテクチャ定義:目的＋指針＋モデル

①顧客ニーズを定義

②対象組織の目標を定義

③顧客ニーズ及び組織目標と現状とのギャップを分析して目的を定義

④目的を達成する為の指針とアーキテクチャを定義

⑤アーキテクチャの使用プロセスを定義

課題解決アーキテクチャ開発プロセス

課題の発生源を特定して作り込み要因を分析して課題を定義。
課題を作り込まない技術開発を行い、課題解決の指針とアーキテクチャ定義と課題解決のタスクの形式化により組織全体の品質を改善する。

アーキテクチャ定義：課題＋指針＋モデル

①課題の発生源の特定
②作り込み本質要因を特定
③課題を定義
④課題抑制の指針とアーキテクチャを定義
⑤アーキテクチャの使用プロセスを定義
⑥プロセスの形式化による安定化
⑦形式化したプロセスのシステム化

２．アーキテクチャ事例紹介

アーキテクチャ事例：*A*開発手法紹介

自律オブジェクト指向（AOO：Autonomic architecture base Object-Oriented development technique）が1998年に組込みソフトウェア開発向けオブジェクト指向の開発手法として発表。その後、AOOは、プロセス(AOO_PRS)、プロダクトライン(AOO_SPL)、見積り(AOO_EST)、形式手法(AOO_DSL)を拡張。組込みソフトウェアの最強のオープンな開発手法として総称を*A*「エース」(Autonomic architecture base embedded software development technique)と呼んでいる。

Autonomicは、「自律」という意味と「自律神経」という意味も含まれる。自律は、他からの支配・制約などを受けずに、自分自身で立てた規範に従って行動するアーキテクチャを基準に開発手法を定義。

要求を1機能1目的で階層的にカテゴリで分類整理して、機能間に依存関係を持たせず自分自身の機能目的達成のための要求を定義する。

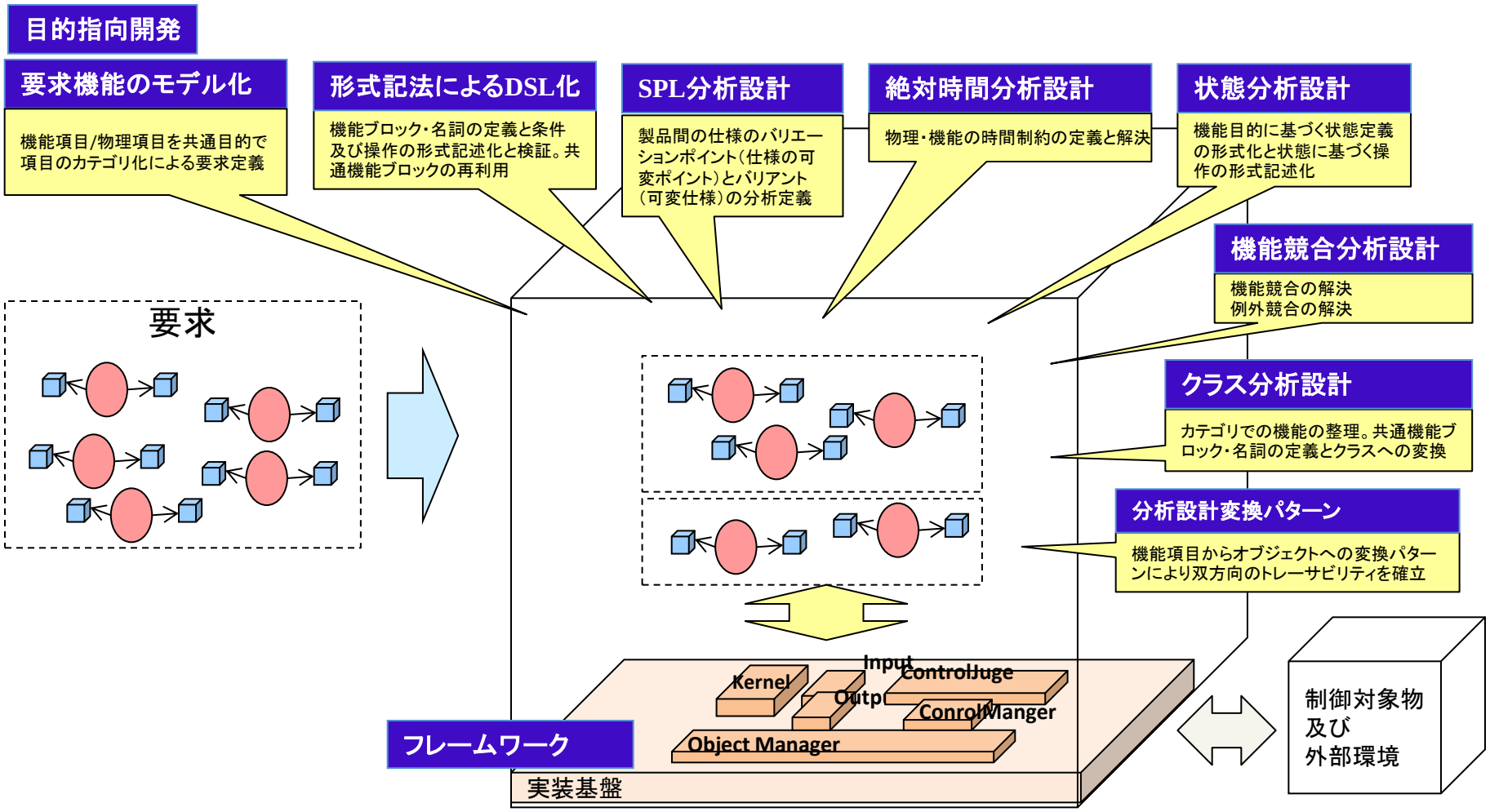
更に、共通機能、機能ブロック（操作）、名詞（属性）、バリエーション（ポイント＋バリアント）を定義して要求モデルの洗練化して日本語によるDSL化を進める。

要求から変換されるオブジェクトも目的単位で自律して振る舞い、オブジェクト間の関連は、静的結合で自分自身の1つの目的達成の為に振る舞う。

要求を人の認知方式に基づきモデル化され自律神経モデルに基づきフレームワークに変換することで要求・設計・実装・試験への双方向の紐付けを可能にして、高品質を確保した上で派生機種開発、SPL開発を可能にする。

アーキテクチャ事例：A開発手法の概要

要求を1機能1目的として階層的に機能ブロック（操作）、属性、時間制約、変換パターン、etcのプロパティを定義して要求定義での洗練化を進めて想定可能な競合・例外事象を定義、製品内・製品間でのバリエーションを定義することで高品質・高生産性を確保する



課題に基づくアーキテクチャ開発

事業戦略、事業特性、開発上の課題に基づくアーキテクチャ

①事業特性に基づくアーキテクチャ開発
②開発上の課題に基づくアーキテクチャ開発
③事業戦略に基づくアーキテクチャ

課題	課題解決方法
①仕様変更の多発	仕様変更影響範囲の局所化のためのアーキテクチャ
②S/W構造とI/Fのバラツキ	分析設計変換バラツキ抑制のためのアーキテクチャ
②機能競合の課題	機能競合解決のアーキテクチャ
②状態遷移定義のバラツキ	状態定義のアーキテクチャと状態抽出方法の定義
②時間制約の保証	時間制約解決のアーキテクチャ
③要求の大規模複雑化	要求の分解整理のアーキテクチャと形式記述化
③製品バリエーションの増加	バリエーションポイント及びバリアント定義の枠組み
③リードタイムの短縮	目的指向開発の推進

ソフトウェアの構造・振る舞いの決定の明確な理由がある。事業戦略・特性・課題の組織的な解決を推進するのがアーキテクトである。

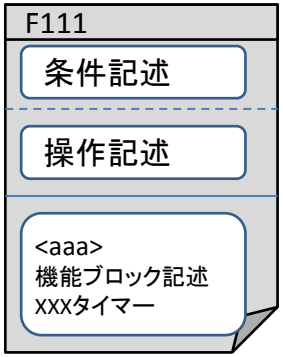
AOO_PRS：①～③プロセスによるアーキテクチャの安定化

ソフトウェア要求分析

③【要求の大規模複雑化】の抑制

機能項目			目的	属性	操作	状態	優先度	変換パターン	時間	例外
F1	F11	F111		xxx	aaa					E01
		F112								
	F12	F121								
F2	F21	F211								

③【リードタイムの短縮】



②【機能競合の課題】の解決

機能マトリクス			F1			F2
			F11		F12	F21
			F111	F112	F121	F211
F11	F11	F111		○	×	○
		F112	○		○	○
	F12	F121	○	×		○
F21	F21	F211	×	×	×	

例外マトリクス			E01	E02	E03
F11	F11	F111	—	—	—
		F112	—	—	—
	F12	F121	—	—	○
F21	F21	F211	—	○	○

②時間制約の保証

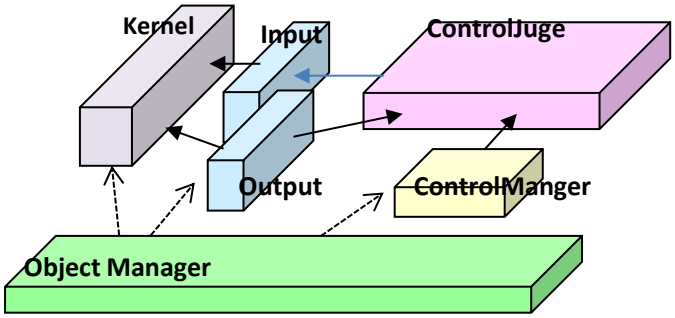
物理項目			目的	属性	操作	状態	優先度	変換パターン	時間制約	例外
P1	P11	P111								E02
		P112								
P2	P21	P211								E032

②【状態遷移定義のバラツキ】の抑制

①【仕様変更の多発】の抑制

②【S/W構造とI/Fのバラツキ】の抑制

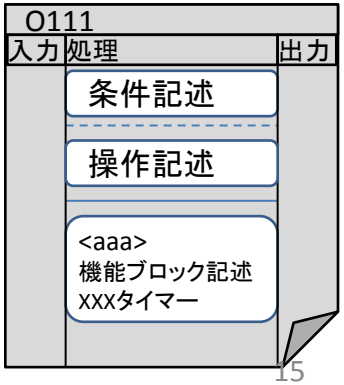
ソフトウェアアーキテクチャ設計



②【S/W構造とI/Fのバラツキ】の抑制

ソフトウェア詳細設計

オブジェクト			属性	操作	状態	優先度	時間
F1	F11	F111	xxx	aaa			
		F112					
	F12	F121					
F2	F21	F211					
P1	P11	P111					
		P112					
P2	P21	P211					
ControlManager							
ObjectManager							
Kernel							



AOO_DSL：③要求の大規模複雑化の解決
表形式と日本語による形式記法による複雑化の抑制

1）要求のカテゴリ化と機能ブロック/属性の形式化

2）形式記法を用いた機能仕様の日本語による形式化

3）機能ブロック/属性の再利用によるDSL化の推進

要求

要求

要求

機能項目リスト

機能項目			目的	操作	目的	属性
1.xxx	1.1.xxx					
	1.2.xxx					
2.xxx	2.1.xxx					
	2.2.xxx	2.2.1.xxx				
		2.2.2.xxx				
制御マネージャ						

物理項目リスト

物理項目			目的	操作	目的	属性
1.xxx	1.1.xxx					
	1.2.xxx					
	1.3.xxx					
2.xxx	2.1.xxx	2.1.1.xxx				
		2.1.2.xxx				
デバイス						

1機能項目1目的として
階層的にカテゴリ化を進める

機能ブロック

機能ブロック

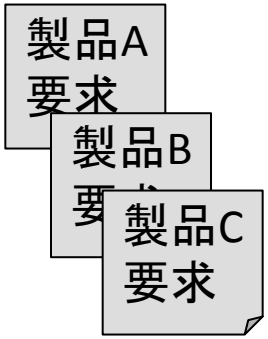
①and②
①外気温度＞10℃
②運転モード＝冷房
・XXX制御状態←制御中
【XXX制御状態＝制御中】

SelectSetDataTB:

JD		SD	
JV		SV	
JV		SV	
JV		SV	

機能仕様で扱う名詞の定義して形式記法で仕様定義
更に1機能項目内を目的で分解して機能ブロックを定義
機能ブロックの再利用による要求分析定義の生産性向上
※文書コード生成検証の実現

AOO_SPL：③製品バリエーションの増加の解決



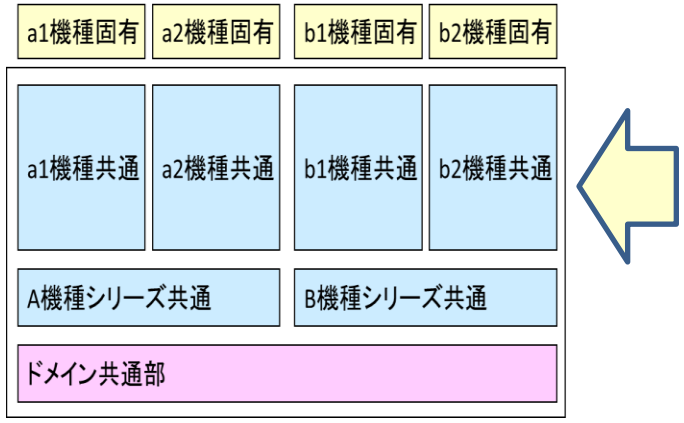
1機能項目１目的で

カテゴリで分類整理

機能項目		機能仕様 目的	操作	目的	属性	バリエーションポイント	バリアント	バリアントID	機種A	機種B	機種C	機種D
1.xxx	1.1.xxx					開始条件	標準	0	0	0	1	2
							北米	1				
							欧州	2				

物理項目			機能仕様 目的	操作	目的	属性	バリエーションポイント	バリアント	バリアントID	機種A	機種B	機種C	機種D
2.													
1.xxx	1.1.xxx												
	1.2.xxx												
2.xxx	2.1.xxx	2.1.1.xxx											

オブジェクトに変換（可変性の継承）



オブジェクト			機能仕様 目的	操作	目的	属性	バリエーションポイント	バリアント	バリアントID	機種A	機種B	機種C	機種D
1.xxx	1.1.xxx						開始条件	標準	0	0	0	1	2
	1.2.xxx							北米	1				
2.xxx	2.1.xxx	2.1.1.xxx						欧州	2				
1.xxx	1.1.xxx												
	1.2.xxx												
2.xxx	2.1.xxx	2.1.1.xxx											

機能項目名称	１．１．xxxx		
機能目的	XXXXXXXXXXXXXXXXXX		
バリエーションポイント	バリアント		
	機種A，B	機種C	機種D
開始条件	標準	北米	欧州
条件記述 ＜＜開始条件＞＞ XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX			
操作記述 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX			
＜標準＞ XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX			
＜北米＞ XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX			
＜欧州＞ XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX			

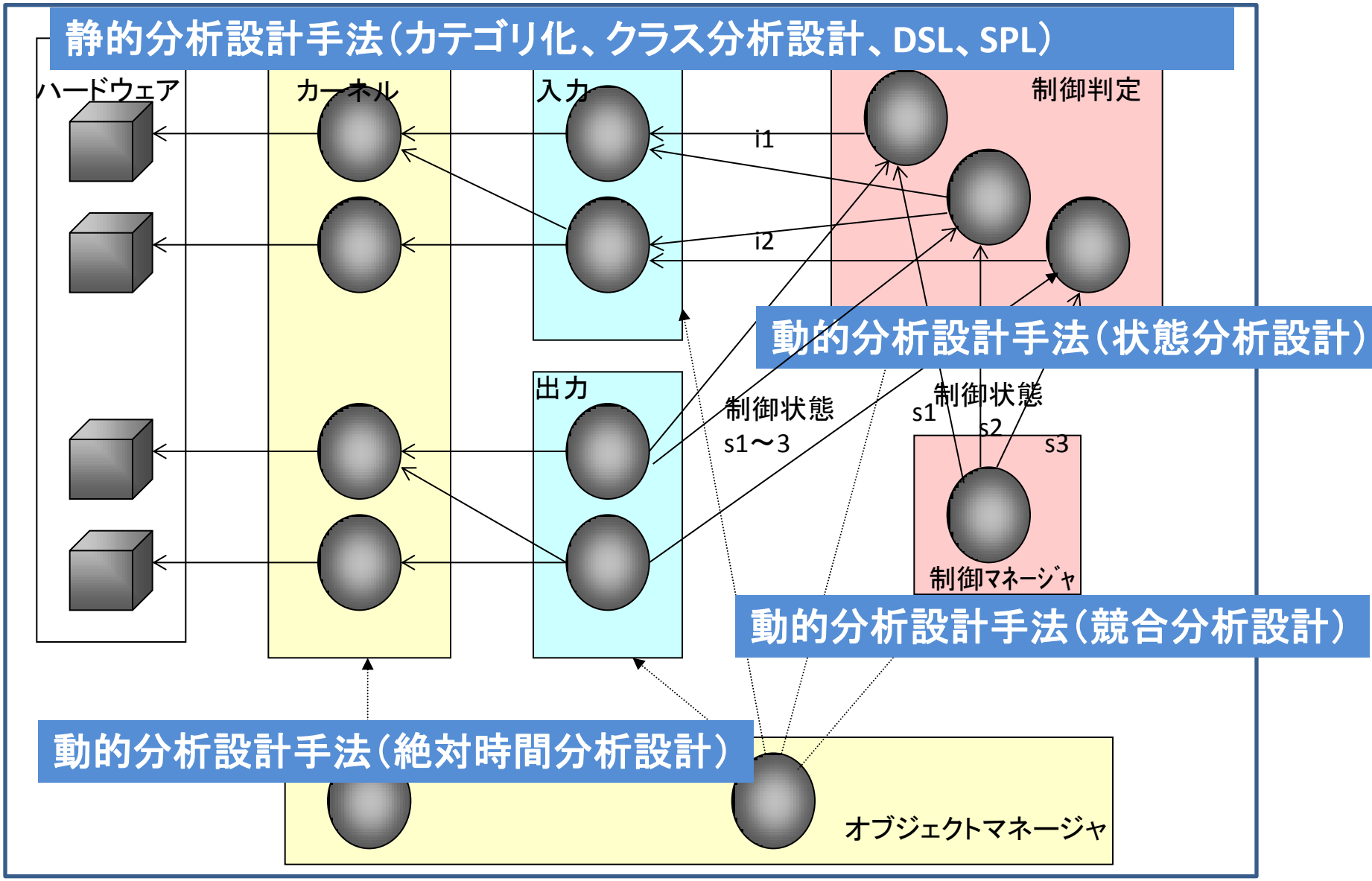
製品間の可変性を分析して可変ポイント（バリエーションポイント）と可変部分（バリアント）を定義する

①要求の物理と目的（機能）で分解整理してソフトウェアと双方向に紐付することでSPL開発を可能にする。

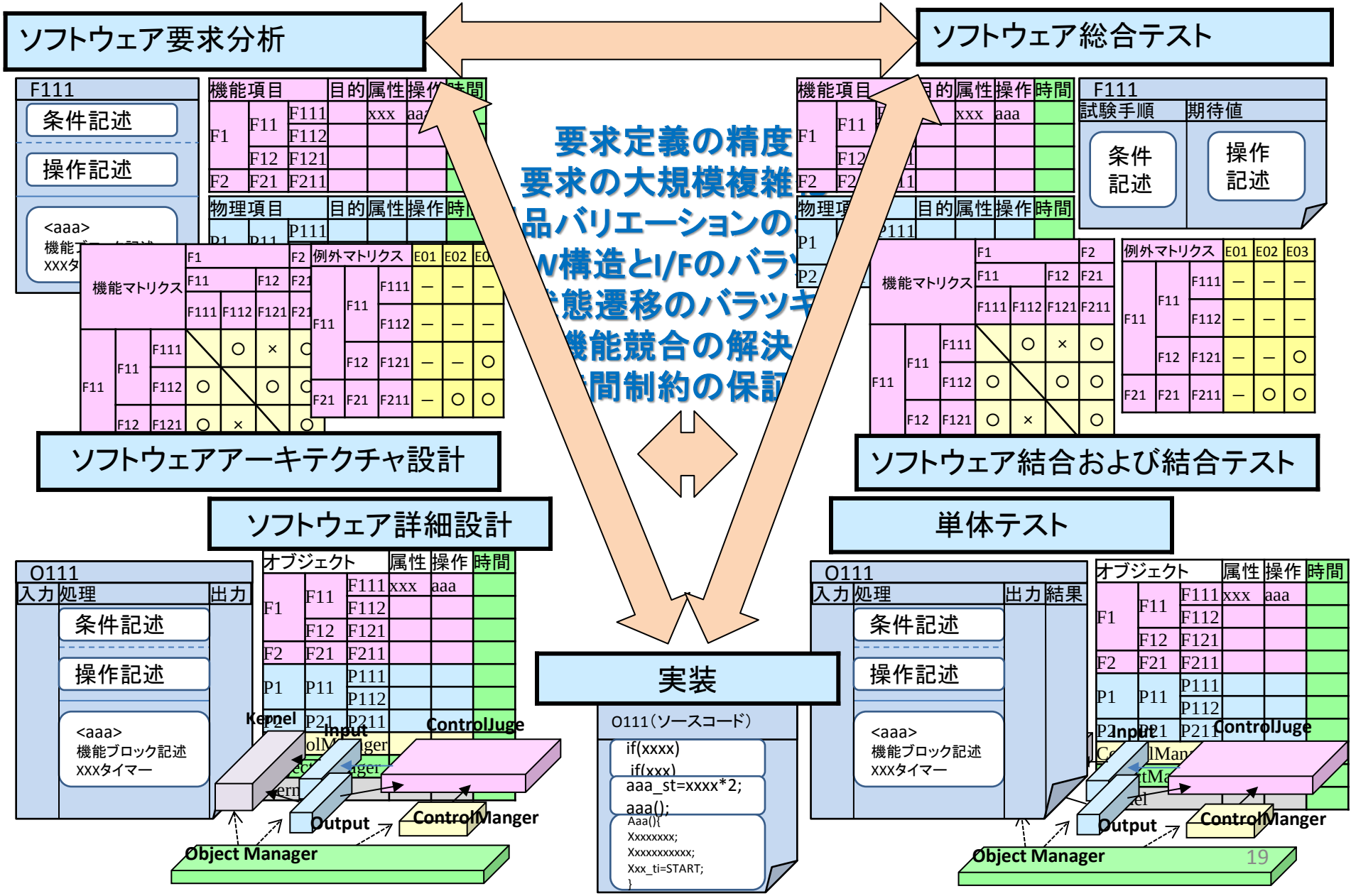
②フレームワークをドメイン依存として構築すると中長期開発で　破綻するリスクがある。

状態遷移、状態に基づく操作、競合解決、バリエーション解決は、　多くの組込み制御システムに共通であり、共通フレームワーク上　で実現することでSPL開発を成功させる。

AOO：①～③フレームワークによるアーキテクチャの安定化



AOO_PRS：①～③プロセスによるアーキテクチャの安定化



３．アーキテクト育成の取り組み

アーキテクトに求められるスキル

アーキテクトには以下のスキルが必要である。

①アーキテクチャ使用スキル（モデル化とプロセスがポイント）

②アーキテクチャ構築スキル（本質の定義がポイント）

- 抽象化スキル
  対象の本質を定義するスキル
- 汎用化スキル
  複数の対象から共通的な本質を定義するスキル
- 構築スキル
  アーキテクチャを構築できるスキル

アーキテクト育成の取り組み（使用スキル）

①フレームワーク定義

分析・設計・試験及び実装の汎用アーキテクチャ定義
分析・設計・試験の様式フレーム定義と実装のフレームワーク定義
目的：優れたアーキテクチャの形式知化と共有
　　アーキテクチャ発散の抑制と改善スキルの向上
教育：課題解決型フレームワーク教育

②プロセス定義

分析/設計/試験/実装のアーキテクチャの使用方法の定義
目的：アーキテクチャの正しい使用法の徹底と発散の抑制
育成：課題解決型プロセス教育

③システム化

アーキテクチャとプロセスの安定化の支援
目的：アーキテクチャの正しい使用の徹底と発散の抑制の効率化
　　ガイダンスによるOJT負荷の低減

アーキテクト育成の取り組み（構築スキル）

①抽象化スキル：対象の本質を定義するスキル

既存コードのアーキテクチャのモデル化による抽象化スキルを育成

目的：アーキテクチャ発散防止

抽象化能力の向上とコード解析精度スピード向上

②汎用化スキル：複数の対象から共通的な本質を定義するスキル

既存アーキテクチャ間の共通性を分析して汎用化スキルを育成

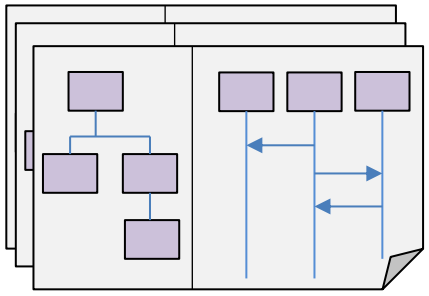
目的：アーキテクチャの再利用と発散防止

汎用化能力向上による組織全体の設計品質向上

③構築スキル：小さな課題解決のアーキテクチャとプロセス定義

目的：開発上の小さい課題を解決する為のアーキテクチャとプロセス定義

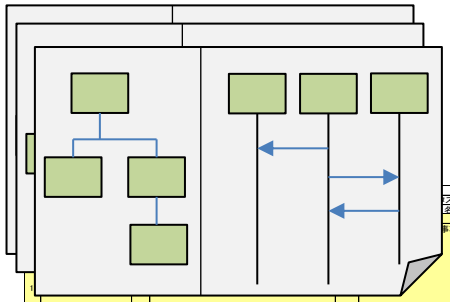
により組織全体の開発力向上とアーキテクトを育成

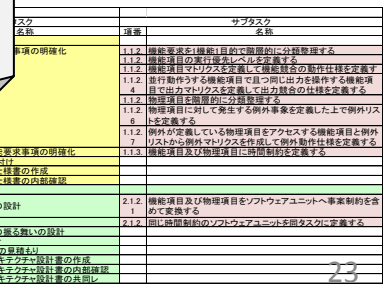


抽象化

既存ソースコード

一般化と抽象化





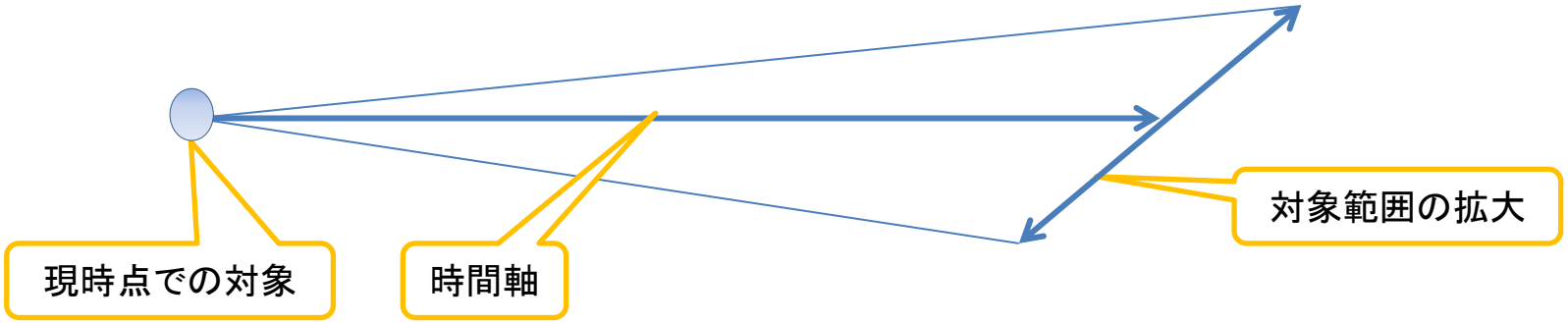
4. まとめ

アーキテクチャ開発ポイント　まとめ

1）解決する目標/課題を定義する
　顧客ニーズ/事業目標と現状のギャップを分析して定義
　バリエーション増加，リードタイム短縮、低コスト開発、品質特性・・・
　要求や課題を分類整理する事でアーキテクチャを発掘

2）時間軸を考慮する
　製品の長いライフサイクルの中での変動に対応できる
　アーキテクチャを設計する。

3）対象範囲の拡大
　対象ドメインの範囲を可能な限り広げる。
　制御機器の組込みシステム及びIT系システムまで対象領域
　を拡大する事で安定したアーキテクチャを設計する。



アーキテクト育成の取り組み　まとめ

1. 優秀なアーキテクトが開発したアーキテクチャの形式知化

優秀なアーキテクトは、事業戦略・製品特性・開発上の課題解決を継続的に対応可能なアーキテクチャを開発している。優秀なアーキテクトのスキルを形式知化して育成。
①アーキテクチャの定義とフレームワーク化（課題/目的定義＋方針定義＋モデル定義）
②アーキテクチャを使用する為のプロセス定義

2. 実務中心アーキテクト育成の取り組み

継続的なアーキテクチャとプロセスの改善と教育
①定期教育(QP-Meeting)：アーキテクチャとプロセスを開発関連要員全員に継続実施
②プロジェクト開始時の導入教育
③プロジェクトレビューでの個人に合わせた育成

3. ボトムアップ的アーキテクト育成の取り組み

本質を見極める抽象化能力を育成する事によるアーキテクトの育成
①既存資産のモデル化（アーキテクチャ定義：課題＋方針＋モデル）
②複数モデルからの共通モデル化（アーキテクチャ定義：課題/目的＋方針＋モデル）
③発生した課題に対するアーキテクチャ開発

ご清聴ありがとうございました。

詳細お問い合わせ先
三菱電機メカトロニクスソフトウエア株式会社
岩橋正実
073－436－0776
iwahashi@est.hi-ho.ne.jp
Iwahashi.Masami@wak.msw.co.jp