ソフトウェアプロダクトライン エンジニアリング (SPLE) の紹介

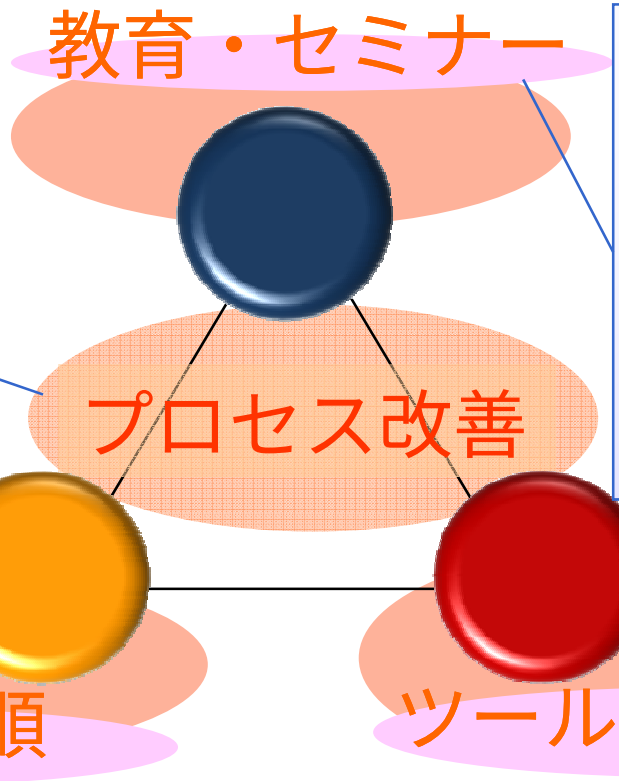
2009/10/23

㈱ SRA　産業開発統括本部
エンベデッド・プロダクト開発部
林　好一
yosikazu@sra.co.jp

First, a Message from My Employers…

会社紹介：(株) SRA

ソフトウェア
エンジニアリング
サービス



- Rational University
  - オブジェクト指向分析
  - RUP入門
- プロセス改善
  - SEI認定[CMMI®入門]
  - SPICE入門
- ソフトウエアテスト
- ソフトウェアプロダクトライン
  - 管理者/技術者向けコース
- Qt トレーニングコース
- その他、各種技術者教育

- CMMI/SPICE適用支援
- 課題ベースのプロセス改善
- 簡易診断・ギャップ分析
- レトロスペクティブ支援

- アジャイル開発手法導入
- ソフトウェアプロダクトライン導入
- UML/オブジェクト指向導入

- ツール導入コンサルティング
  - Qt,, Rational関連, etc
- ツール運用支援サービス
- アプリケーション脆弱性・品質診断サービス（Fortify)

And Now, Today's Feature Presentation…

アジェンダ

- 初めに
  - Engineering の目的
  - システムは変種を生む
- SPLE
  - SPLE とは？
  - SPLE 適用事例
  - これまでとの違い
- 似ているが異なるものを扱う方法
  - 共通性と可変性
    - フィーチャ
- SPLE のプロセス
- SPLE の戦略的重要性
- まとめ

初めに

SPL Engineering

初めに＞Engineeringの目的

“Engineering”
は何のため？

⇒ 良いものを作るため

どうやって？

- **良いものを作るには、良い設計のやり方、そして良くできた設計**
- **良い設計にはそもそも的確な仕様が必要**

的確な要件とは？

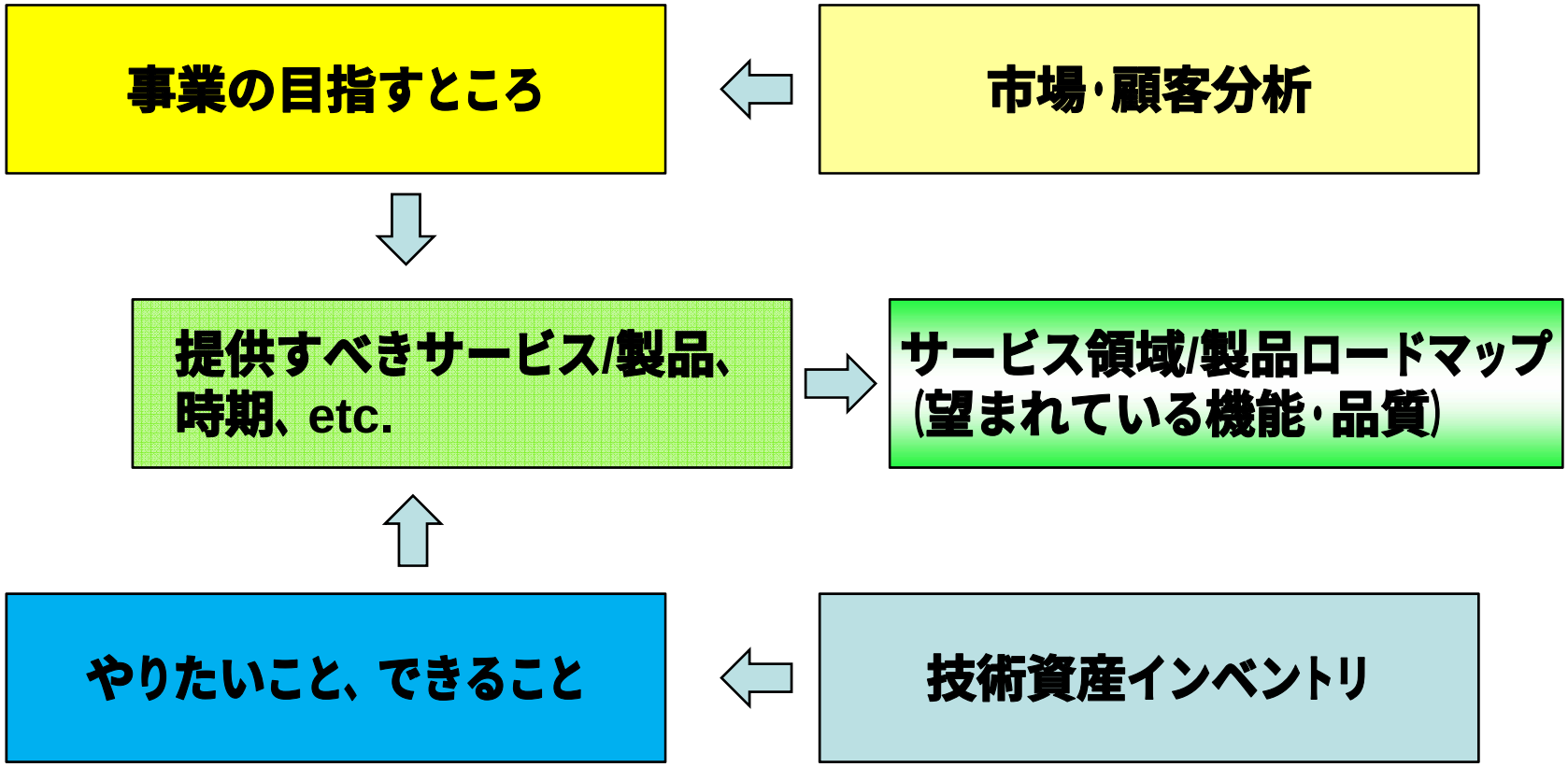
初めに＞Engineeringの目的

- 顧客が欲しいと「言って」いるものを書き留めたもの？
- 顧客が言わなくても欲しがっているもの？
  - 聞いているだけでは不充分。掘り起こす
- 顧客がお金を出しても前より喜ぶもの？
  - 見えているところからさらに深いところを読む
- 顧客じゃなかった人が顧客になるもの？
  - 見えているところからさらに広い範囲を読む

- 超上流活動からつなげる必要がある
  - 要求情報源の分析（顧客分析、市場分析）
  - 何を、誰のために、何のために

事業目標が「良いもの」を基礎付ける

初めに＞Engineeringの目的



名前は一つ、種類は複数

初めに＞システムは変種を生む

- 例えば…
  - 給与計算システム
    - 複数の企業で異なるが似ているシステムを使っている
  - 携帯音楽プレーヤー
    - 一つの企業が似ているが異なる製品を売っている

システムを全く新しくゼロから作ることは少ない

⇒似たようなシステムを作るための
Engineering が求められる

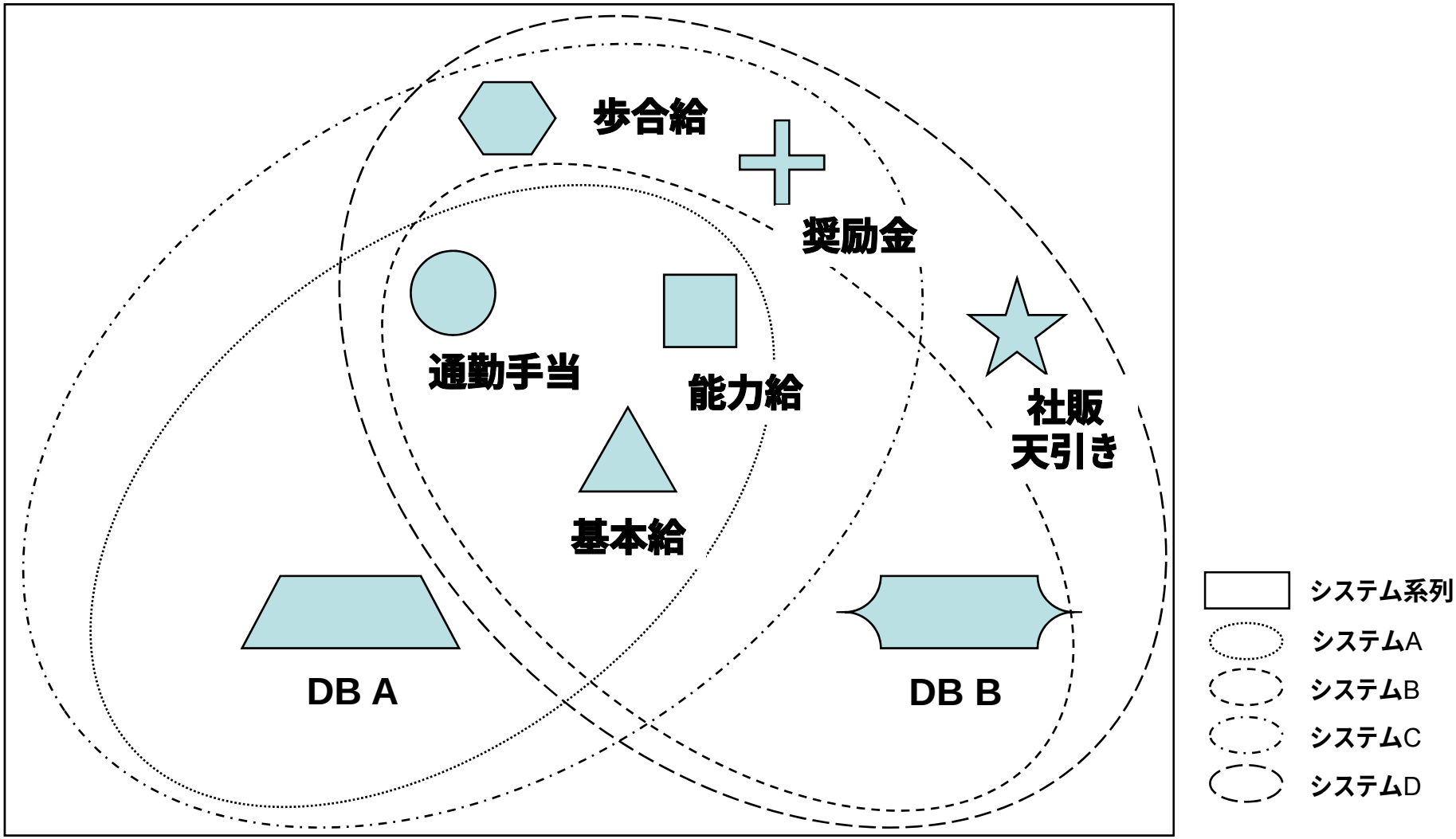
SPLE

SPL Engineering とは？

SPLE＞説明

- Software Product Line Engineering (SPLE)
  ＝ ソフトウェアの「製品（システム）系列」
  の作り方
- ソフトウェア再利用を体系的・計画的に行ない、
- 類似するソフトウェア集約型システム群の開発において低コスト、高品質、短納期を実現するためのパラダイム

システムとシステム系列
製品と製品系列



SPLEに似ているもの

SPLE＞説明

少ない労力で多様性に対応しようとする方策

- モデル駆動開発 (MDA)
  - PIM は固定で PSM がバリエーション
- サービス指向アーキテクチャ (SOA)
  - ごく汎用的な「アーキテクチャ」の上で部品を足し引き
  - アーキテクチャの見直しは…？
- コンポーネントベース開発／部品化
  - 再利用による節約を図る
  - 部品が使われるようにする仕組みは…？
- 派生開発 (以前の開発成果をベースに行なう開発)
  - 体系だっているとは限らない (普通は違う)
  - アーキテクチャが当初の意図どおりであるとは限らない
  - 組込みシステム開発の「常道」

SPLE適用事例 1/3

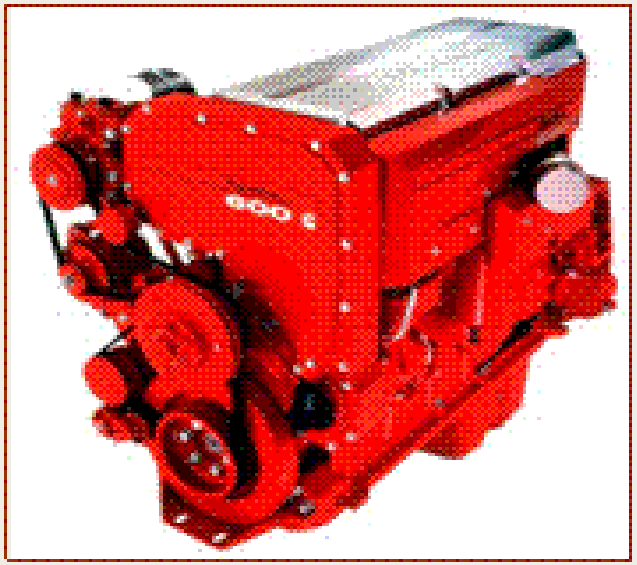
SPLE＞事例

Carnegie Mellon
Software Engineering Institute

カミンズ社：ディーゼルエンジン制御システム

1,000以上の個別のエンジン制御アプリケーションからなる20以上の製品群において、

- 製品サイクルタイムが250人月から数人月に短縮
- ビルドおよび統合にかかる時間が1年から1週間に減少
- 品質目標以上の品質を達成顧客満足度が高い
- 製品スケジュールが守られている



◆ http://www.sei.cmu.edu/productlines/essentials/essentials_11.htm

SPLE適用事例 2/3

SPLE＞事例

Carnegie Mellon
Software Engineering Institute

ノキア・モバイル・フォン

毎年25～30の新製品が出る製品系列

製品間には…

- キーの数の相違
- 画面の大きさの相違
- 基本機能の相違
- 58の異なる言語
- 130ヶ国で利用
- 複数のプロトコル
- 旧製品との互換性
- コンフィギュア可能な機能
- リリース後に機能変更可能にしておくニーズ

◆ http://www.sei.cmu.edu/productlines/essentials/essentials_14.htm

SPLE適用事例 3/3

SPLE＞事例

野村総合研究所
流通業向けエンタープライズシステムの受託開発

- 属人的「ノウハウ」の外化
- 開発時および保守時の読解工数の削減
- 可変性管理の容易化
- 設計再利用による生産性向上
- 類似性がやや低い他業種にも応用

[Ishida07]参照

参考：SPLE適用事例サマリ 1/2

SPLE＞事例

導入企業	ビジネス	開発者数	導入時の開発状態	導入による主な効果
AKVAスマート(*1)	魚の養殖産業向け給餌制御・養殖管理ソフトウェア	6～10人	レガシーシステムのリプレース	・70%以上のコード削減 ・使用感の統一 ・技術基盤およびコードスタイルの共通化 ・再利用、保守、統合の容易化
Boschガソリンシステム(*1)	エンジン制御システム	～1,000人	既存資産をベースにした戦略重視型	・較正工数（20%）および保守工数の削減 ・リソース消費を20～30%削減 ・市場の多様性に応じた製品系列の定義
Cummins(*2)	エンジン制御システム	不明	多数の類似のソフトウェアを個別に開発	・製品サイクルタイムが250人月から数人月に短縮 ・ビルドおよび統合にかかる時間が1年から1週間に減少 ・品質目標以上の品質を達成、高い顧客満足度
DNVソフトウェア(*1)	海運、石油・ガス、鉄道等産業向け情報処理システム	100人	三つの清算センタを共通支援	・ソフトウェア基盤の共通化 ・早期市場投入、ライフサイクルコスト削減と高品質 ・マーケティング、販売、開発の連携の強化
日立製作所(*3)	エンジン制御システム	不明	レガシーシステムからの再利用可能資産の切り出し	・既存資産の可視性の向上（予定） ・品質を維持した再利用性の向上（予定）
九州日立マクセル(*4)	マッサージチェア	不明	変更可能性の高い部分をなるべく集約したアーキテクチャ	・基本アーキテクチャを温存したままシリーズ展開 ・顧客満足度指標を容易に既存および新規の資産に対応付け
マーケットメーカー・ソフト(*1)	株式市況および金融市況の管理・表示ソフトウェア製品	25人	スクラッチからの戦略的多品種開発	・市場投入期間が1/2～1/4（推定） ・製品5本出荷後に従来開発方法より安価に ・保守コスト～60%削減
ノキア・モバイルフォーン(*1)(*2)	モバイル電話機製造	＞1,000人	既存資産をベースにした戦略重視型	・ソフトウェア進化の理解向上 ・共通性に対する洞察の深化
ノキア・ネットワークス(*1)	ネットワークインフラストラクチャ、通信・ネットワークサービス基盤、およびオペレータおよびサービスプロバイダ向け	＞1,000人	既存資産をベースにした戦略重視型	・高複雑度のシステムの管理 ・利用可能資産の再利用と可視性向上

*1: [vdLinden07]に基づく。一部[JASPIC08]を参考にした。
*2: [Clements01]に基づく。
*3: [Yoshimura07]に基づく。
*4: [Abeta08]に基づく。

参考：SPLE適用事例サマリ 2/2

SPLE＞事例

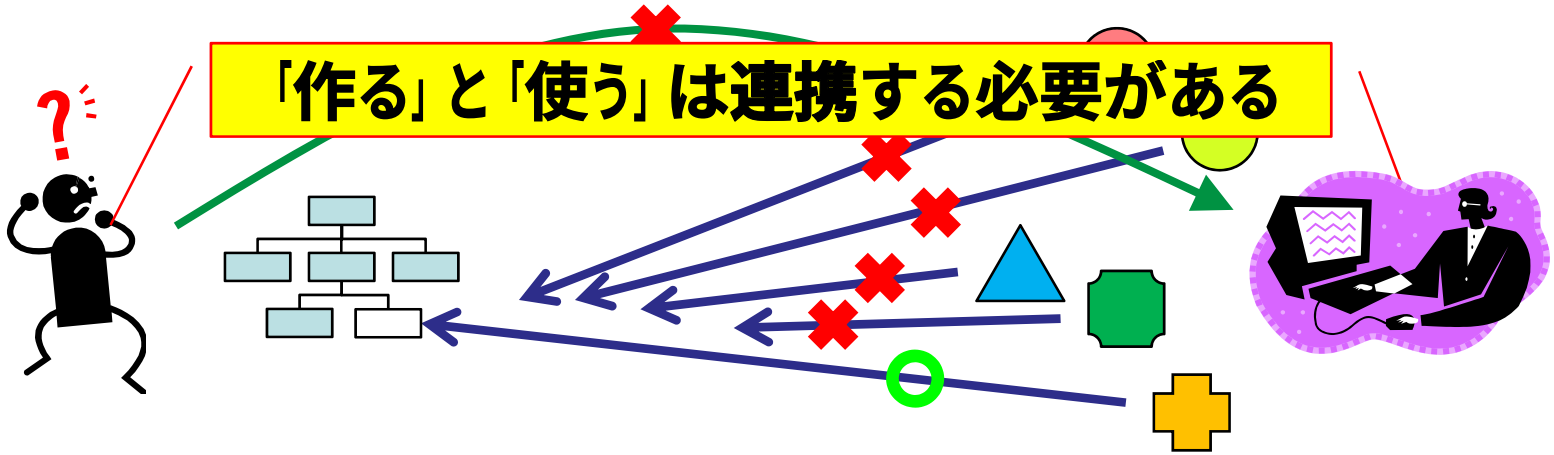
導入企業	ビジネス	開発者数	導入時の開発状態	導入による主な効果
野村総合研究所(*5)	流通業向けエンタープライズシステム受託開発	不明	過去の経験の蓄積でシステム構築	・開発および保守時の読解工数の削減 ・可変性管理の容易化 ・設計再利用による生産性向上 ・類似性がやや低い他業種にも応用
フィリップス・コンシューマー・エレクトロニクステレビ向けソフトウェア(*1)	末端消費者向け電子製品	250人	新アーキテクチャ、リバース・エンジニアリング・コード	・全ての中～高価格帯テレビ向けソフトウェアを一つのソフトウェアプロダクトラインで対応 ・マーケティングによる望ましい可変性を生み出す能力 ・製品開発におけるソフトウェア開発のクリティカルパスからの脱出 ・6年後にも新規アーキテクチャは不要（以前は5年以内に再開発）
フィリップス医療システム(*1)	X線、MRI、CT、超音波等の医療向け画像機器	＞1,000人	既存資産をベースにした戦略重視型	・工数が1/2～1/4に低減 ・再利用機能に関して市場投入期間を50%以下に短縮 ・再利用機能に関して製品欠陥密度を50%に低減 ・使用感の共通化 ・製品計画およびロードマップ使用の向上 ・一つの製品から他の製品へフィーチャを容易に
シーメンス医療ソリューション(*1)	X線管、MRI・CTスキャナからインフラストラクチャサポートまでを含む病院および医療従事者向けハードウェアおよびソフト	100人	ボトムアップ	・再利用レベル：～50% ・開発サイクル期間短縮：～25% ・品質コスト削減：～57%
Telvent(*1)	エネルギー、交通管制、運輸、環境の四つのセクター向けのソリューション	＞1,000人	多くの顧客要件変更に設定で対応	・サーバコア資産を他市場へも振り向け ・実行時可変性の導入 ・参照プロセスフレームワークの改善 ・コア資産のロードマップの集中管理

*1: [vdLinden07]に基づく。一部[JASPIC08]を参考にした。
*5: [Ishida07]に基づく。

従前の「再利用」例 1

SPLE＞これまでとの違い

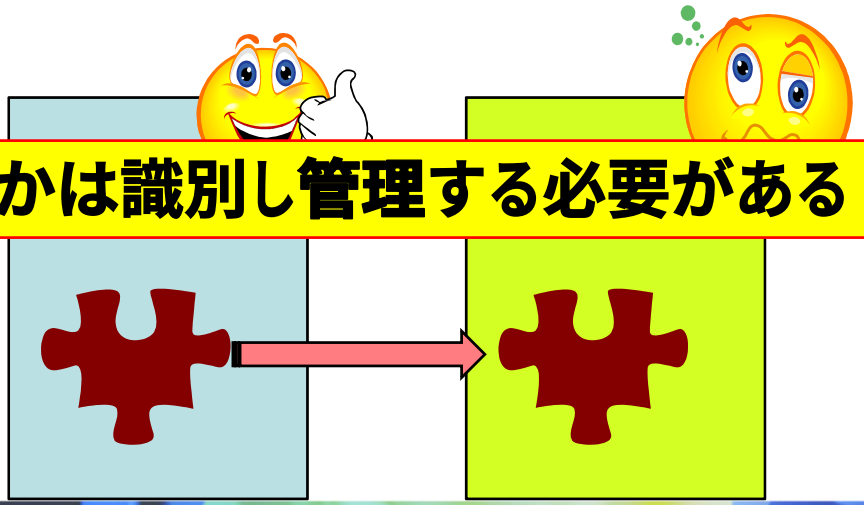
- 流用開発
  - 流用元を探す手間、どこをどう変えるかを考える手間
  - 流用元へのフィードバックが不十分/無し
  - ＝「あるものを使う」という程度
- 部品開発
  - 使われる仕組み・努力が不足/不適切なまま作る
    →知らない・使いにくいので使われない
  - ＝「使うべきものを作る」という程度



従前の「再利用」例 2

SPLE＞これまでとの違い

- 無制約の改変
  - どこをどう変えるかは一技術者の裁量で決まる
  - 多種多様の改変が発生して、何が根幹で何が枝葉かわからなくなり、保守・拡張・さらなる再利用を困難にする
- 無計画な再利用
  - 仕様が似ているから（より典型的にはコードが似ているから）流用する
  - 想定外の事故が起こる（例：Ariane 5, Therac 25）



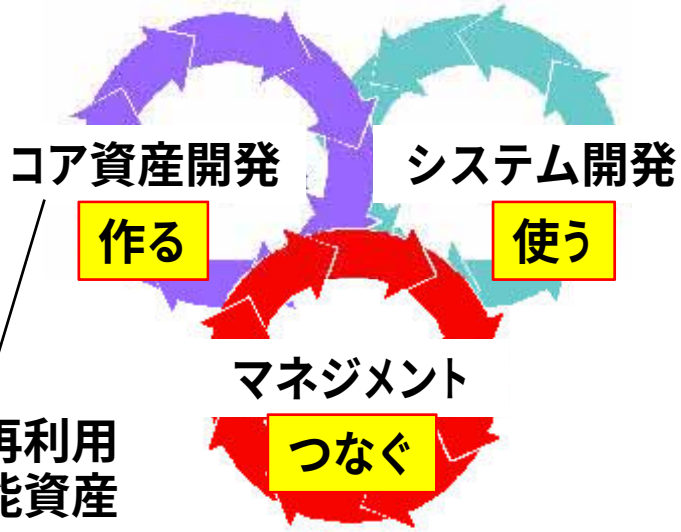
どこをどう変えるか・変えないかは識別し管理する必要がある

SPLE の特徴

SPLE＞これまでとの違い

連携

- 再利用可能資産の構築とその活用が連携している
  - 当たり前ながら、容易でない
  - このプロセスを実施できれば、再利用のうま味が味わえる

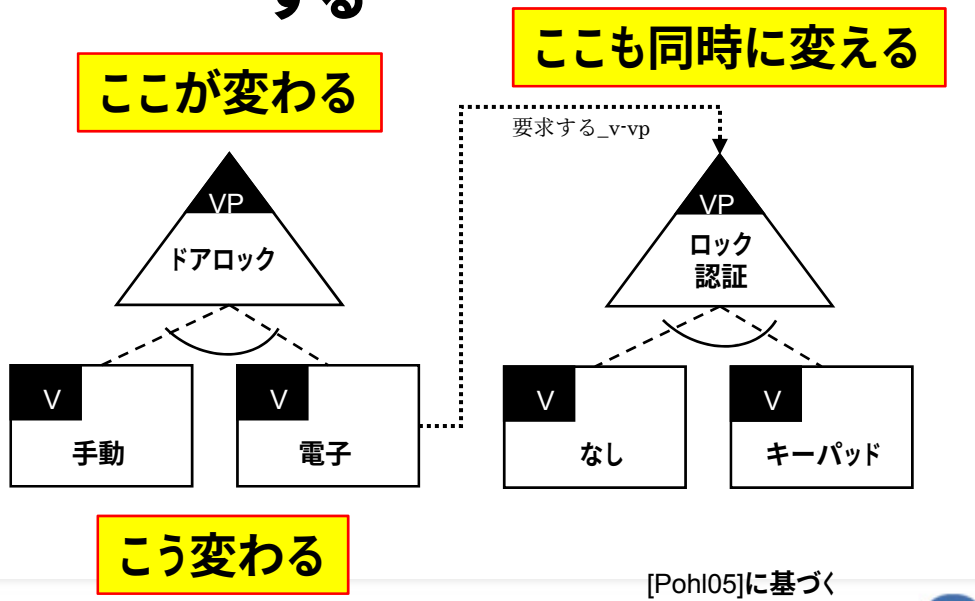


＝再利用可能資産

米 Software Engineering Institute による

可変点管理

- 可変点を基に変更可能点を識別・共有・管理する
  - どこをどう改変可能かを押さえているから計画・実施が容易になる
  - そして、計画に沿って実施する

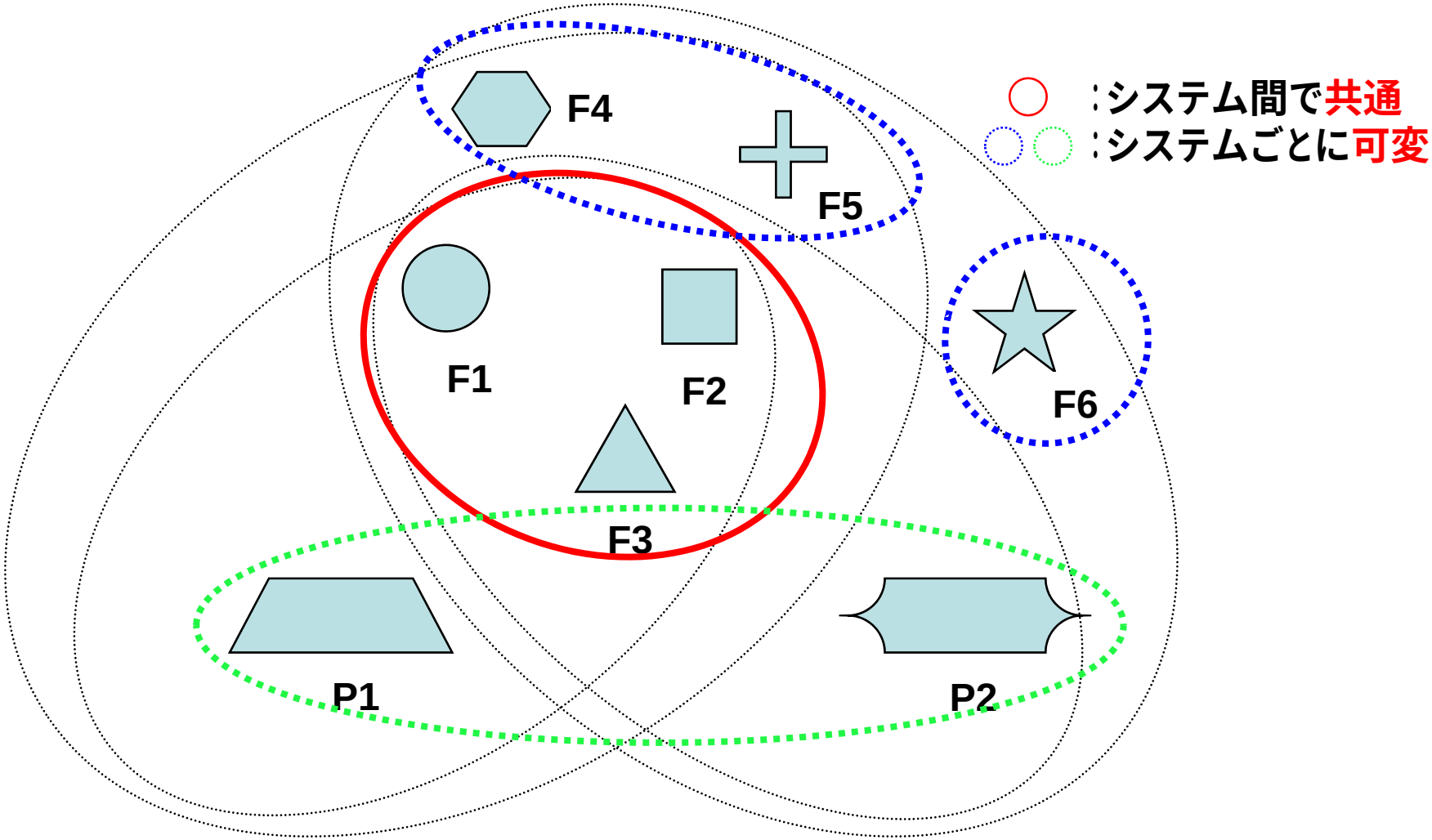


[Pohl05]に基づく

似ているが異なるものを扱う方法

共通性と可変性

似ていて異なる＞共通性と可変性



フィーチャ

似ていて異なる＞共通性と可変性

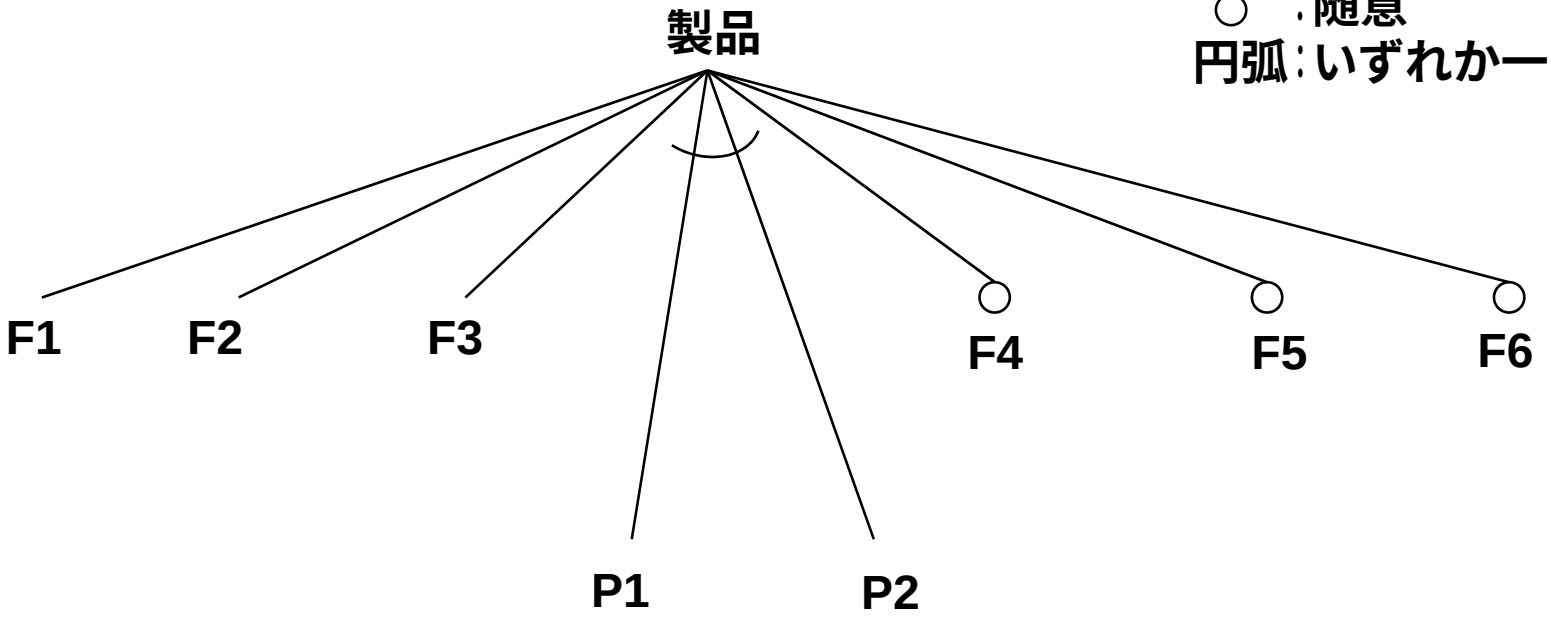
- SPL で何が可変かを表現するものとして、「フィーチャ」がよく使われる
  - 対象ドメインにおける専門家や末端利用者の「語彙」に相当
  - または、「マニュアルに出てくる語彙」
    - ユーザ向け/運用管理者向け/保守担当者向け/…
- システム（製品、サービス）の利害関係者にとって意味のある機能または品質を一言で表わしたもの
  - 仕様として展開される
  - SPL 全体に共通のフィーチャ
  - システムごとに選択/非選択されるフィーチャ
  - システムによってパラメータを調整するフィーチャ

  などがある

フィーチャモデル

似ていて異なる＞フィーチャ

無印：システム間で共通
○　：随意
円弧：いずれか一つ

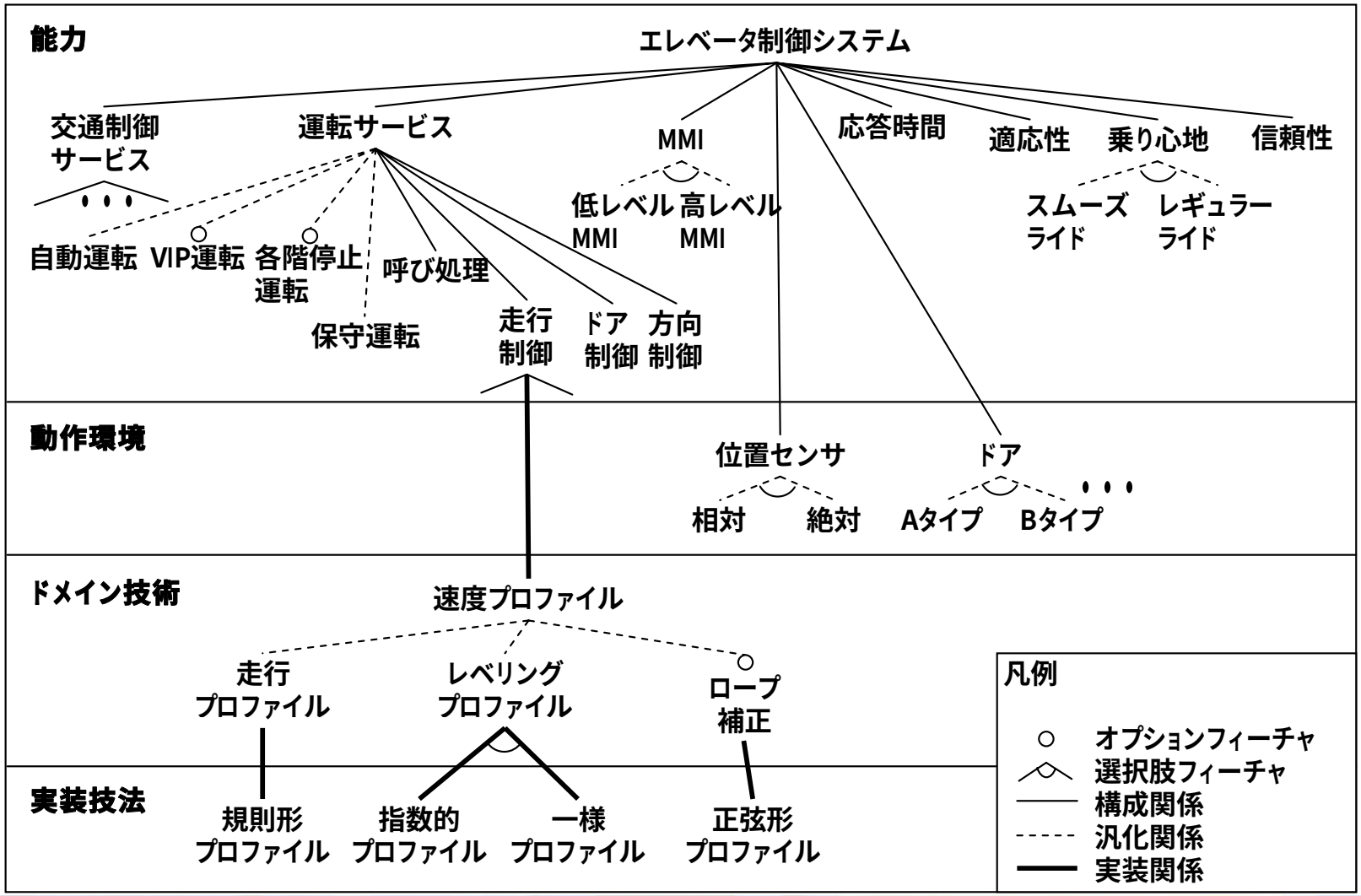


構成規則

No.1: F6 は P1 を要求する

フィーチャモデル例

似ていて異なる＞フィーチャ



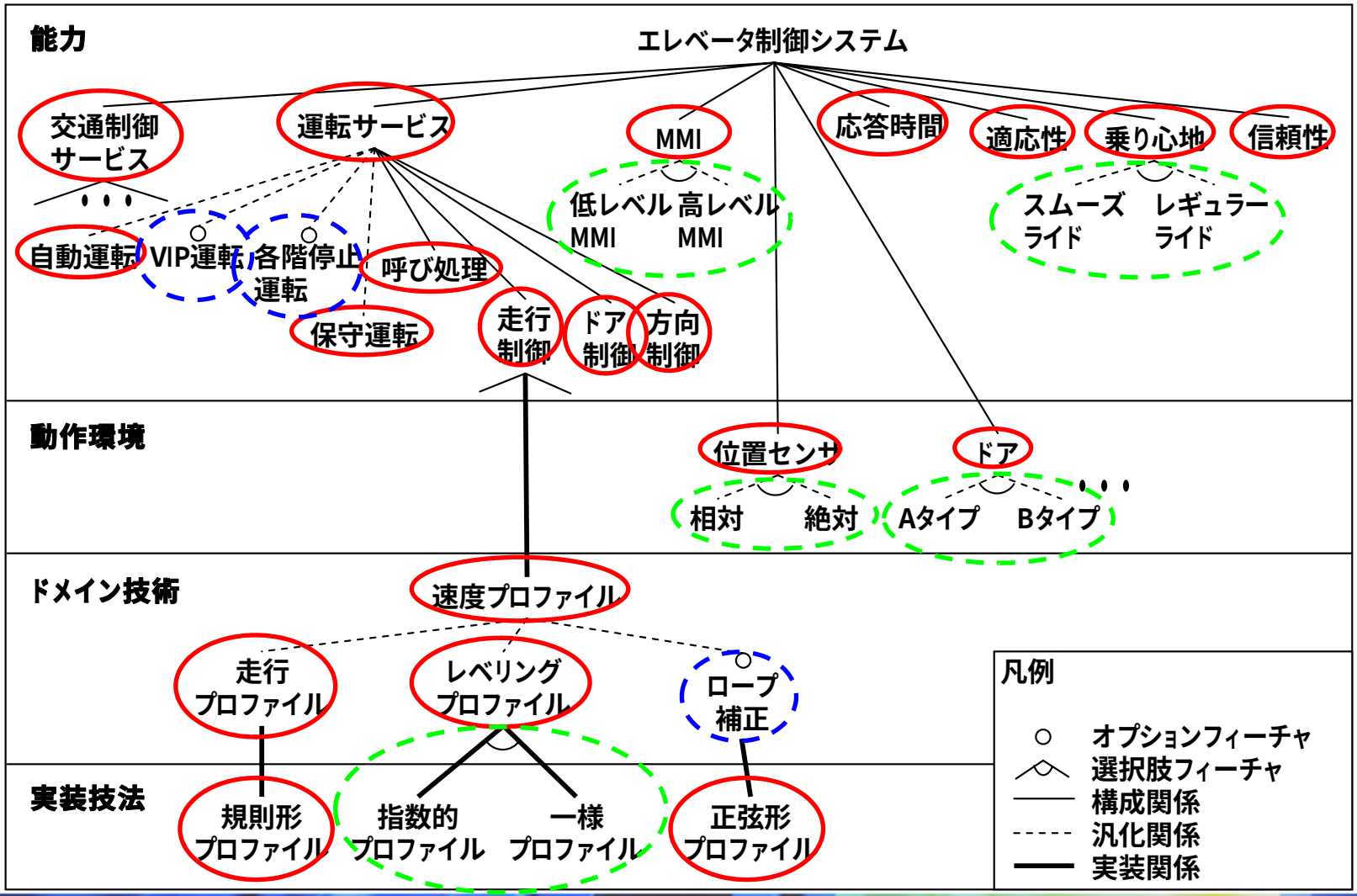
*: [Kang02a]に基づく。

共通性と可変性

似ていて異なる＞フィーチャ

○ ：システム間で共通

○○ ：システムごとに可変



*: [Kang02a]に基づく。

フィーチャモデル：コミュニケーション

似ていて異なる＞フィーチャ

- 非技術者にも理解可能
  - 末端利用者、経営者、企画・マーケティング担当者、運用担当者、インストール担当者、保守要員、アーキテクト、設計者、プログラマ、試験担当者、…
- 各利害担当者の関心事と、他の利害関係者の関心事の間の依存関係がわかる
  - 特定のフィーチャの：
    - 実現に必要な技術/既存サービス、稼働環境における制約/利用者に課せられる制限、…

フィーチャモデル：利用価値

似ていて異なる＞フィーチャ

- システム（群）の特徴（機能、品質）の把握
  - 各利害関係者の視点において、
  - 何が共通でどのような変種がありうるか
- 最終システムの価値の積算 ⇔ 見積り
- システム構築に必要な部品の識別
- アーキテクチャ設計への入力
  - 品質フィーチャ　→　アーキテクチャスタイル
  - 機能フィーチャ　→　コンポーネントへの割り当て

Apple iPod 製品系列のフィーチャ

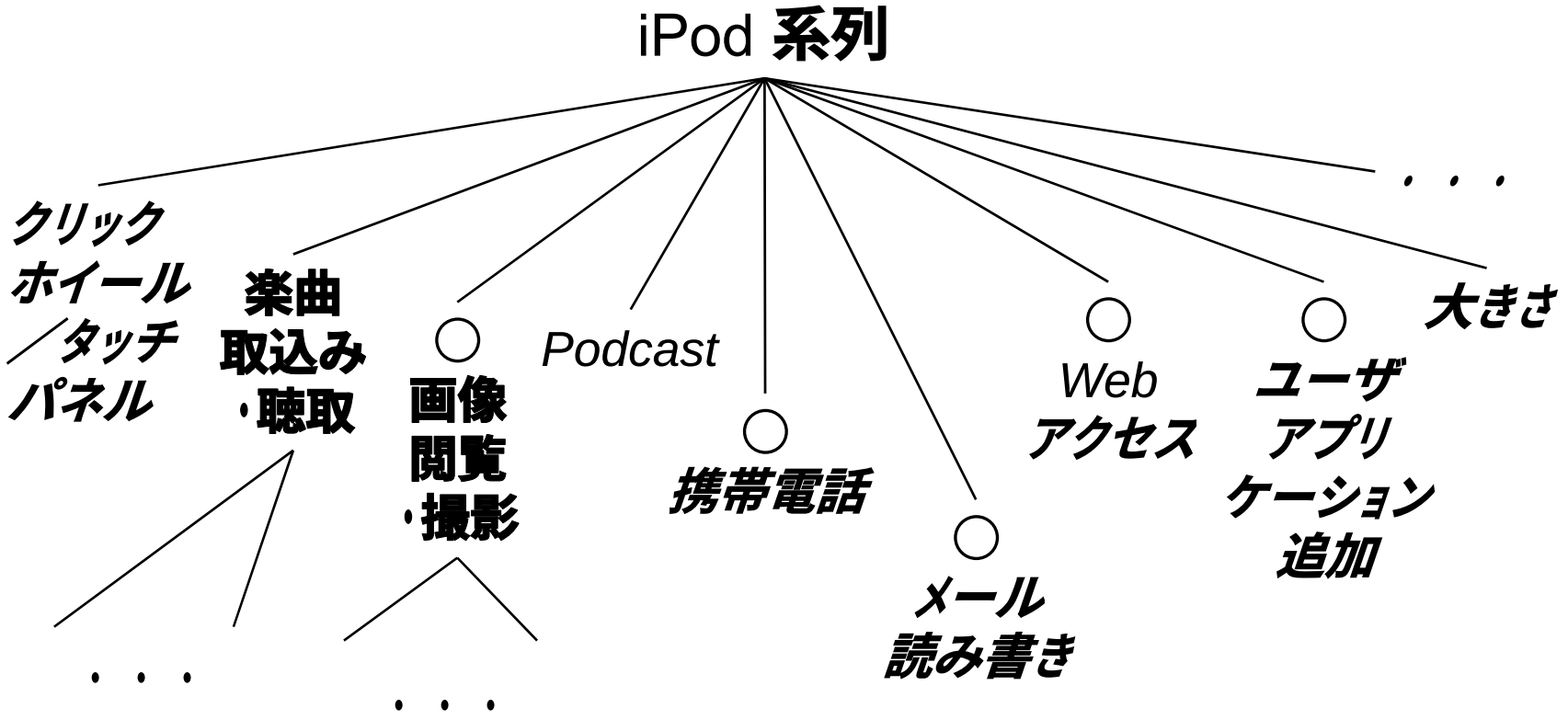
似ていて異なる＞フィーチャ

iPod 系列 (iPhone を含む) のフィーチャ

- 楽曲取り込み・聴取
- 画像取り込み・閲覧・撮影
- タッチパネル/クリックホイール/音声案内
- ユーザアプリケーション追加
- メール読み書き
- Web アクセス
- 小型、軽量
- 、、、

iPod 製品系列のフィーチャモデル

似ていて異なる＞フィーチャ



共通性を把握する意味

戦略的重要性

- 共通性は効率のみを意味するのではない
- 共通性が多いということは、そこに何かがあるということ
  - 市場が求めてきているもの
  - 自組織の強み
  - 行なっている/行なおうとしている事業の中核
- その共通性（コア）が持つゴールと個々のシステムのゴールは異なる
  - 事業展開を支える資産の形成　と
  - 事業展開の具体的な手段であるシステムの構築
- そのため開発は二段階に分かれる
  - ただし人/部門を分けるとは限らない

広い、長い

ピンポイント

SPLE のプロセス

SPL を作るプロセス

プロセス

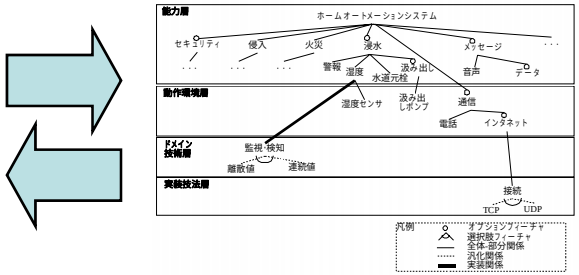
- 以上のような考え方で効率良く高品質のシステムを作ろうとすると、例えば次のページのようなプロセスになる

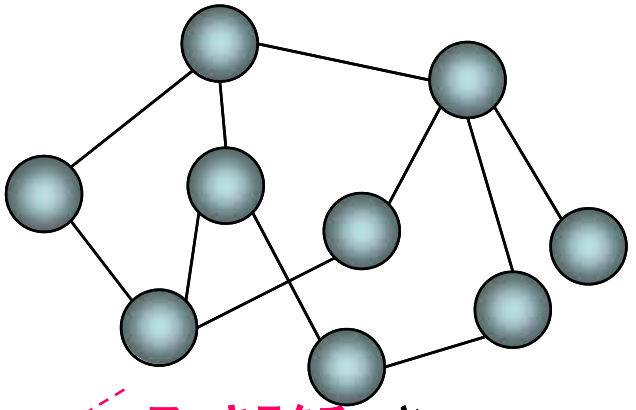
※SPLE 実施の方法は一つではない

コア資産の開発

プロセス

事業目標

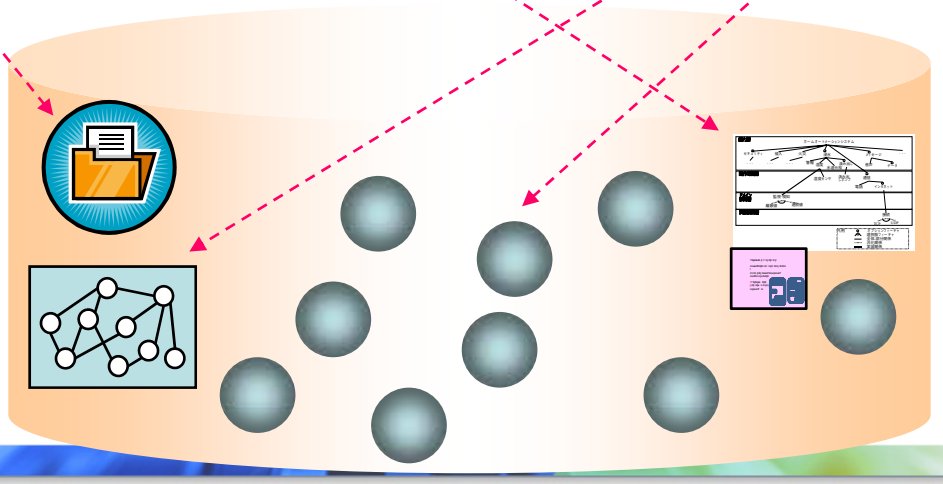




市場/顧客分析
サービス領域/製品計画

フィーチャモデルと SPL要求
(**共通性**と**可変性**)

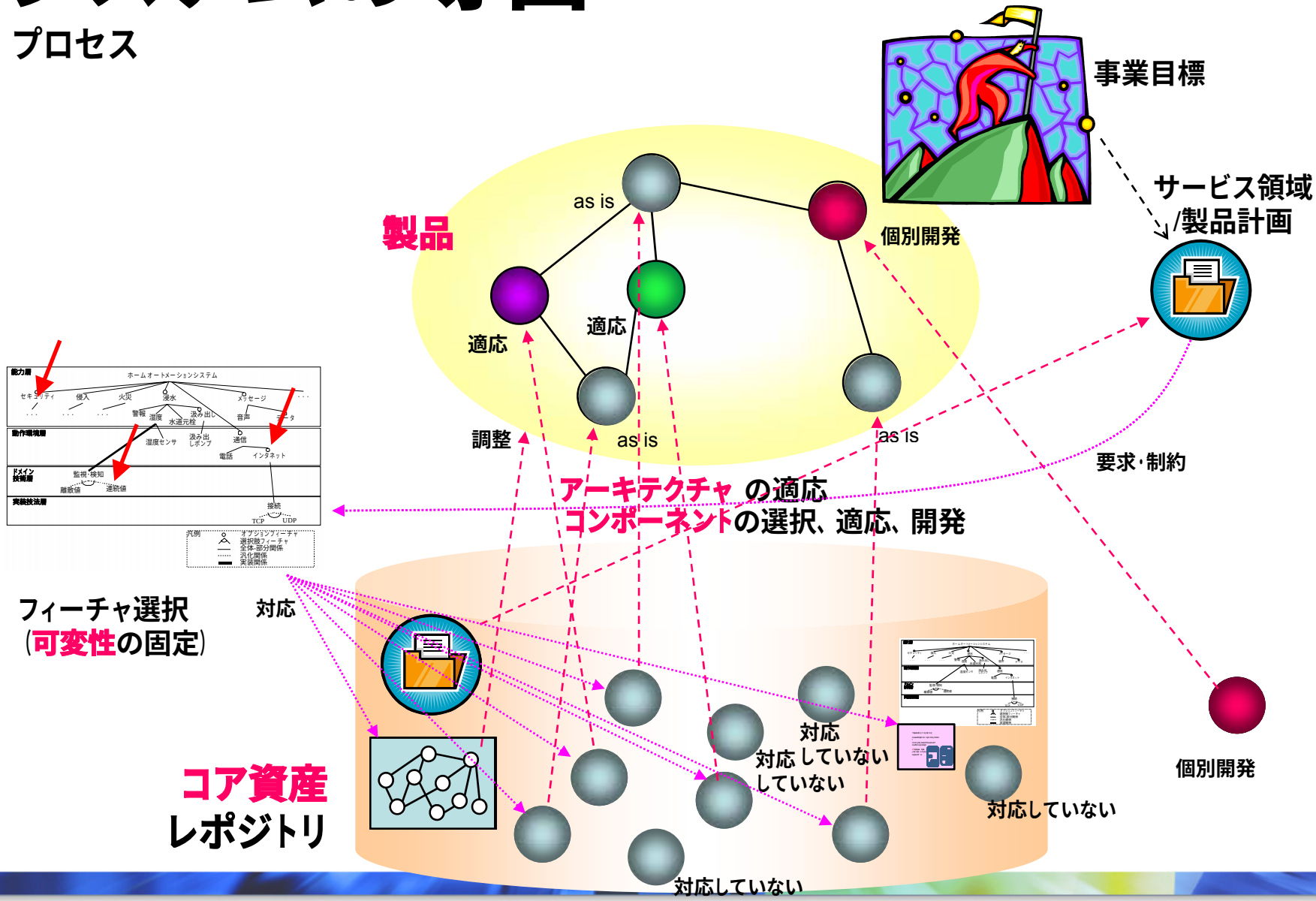
アーキテクチャ と
コンポーネント



コア資産
レポジトリ

システムの導出

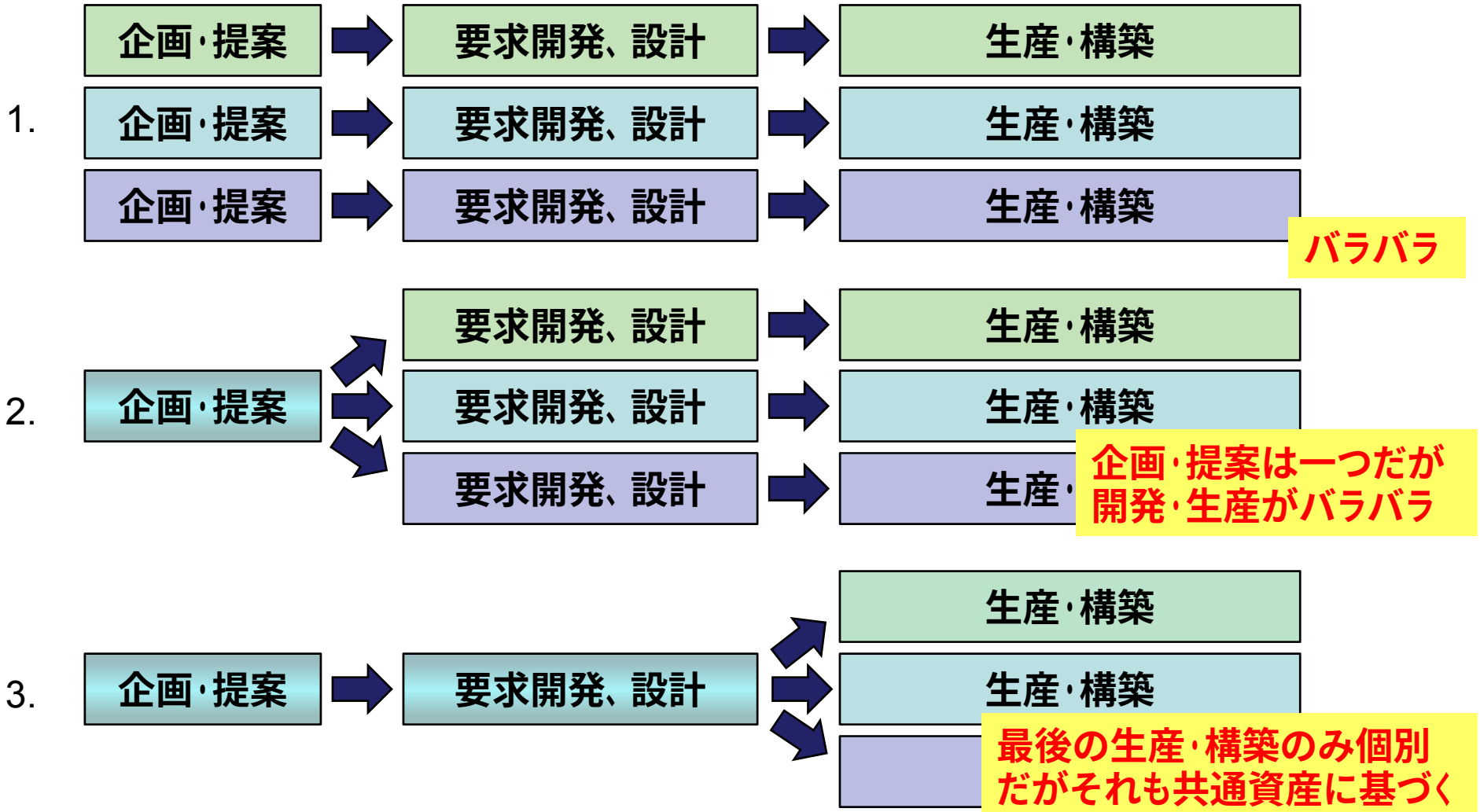
プロセス



SPLE の考え方

プロセス

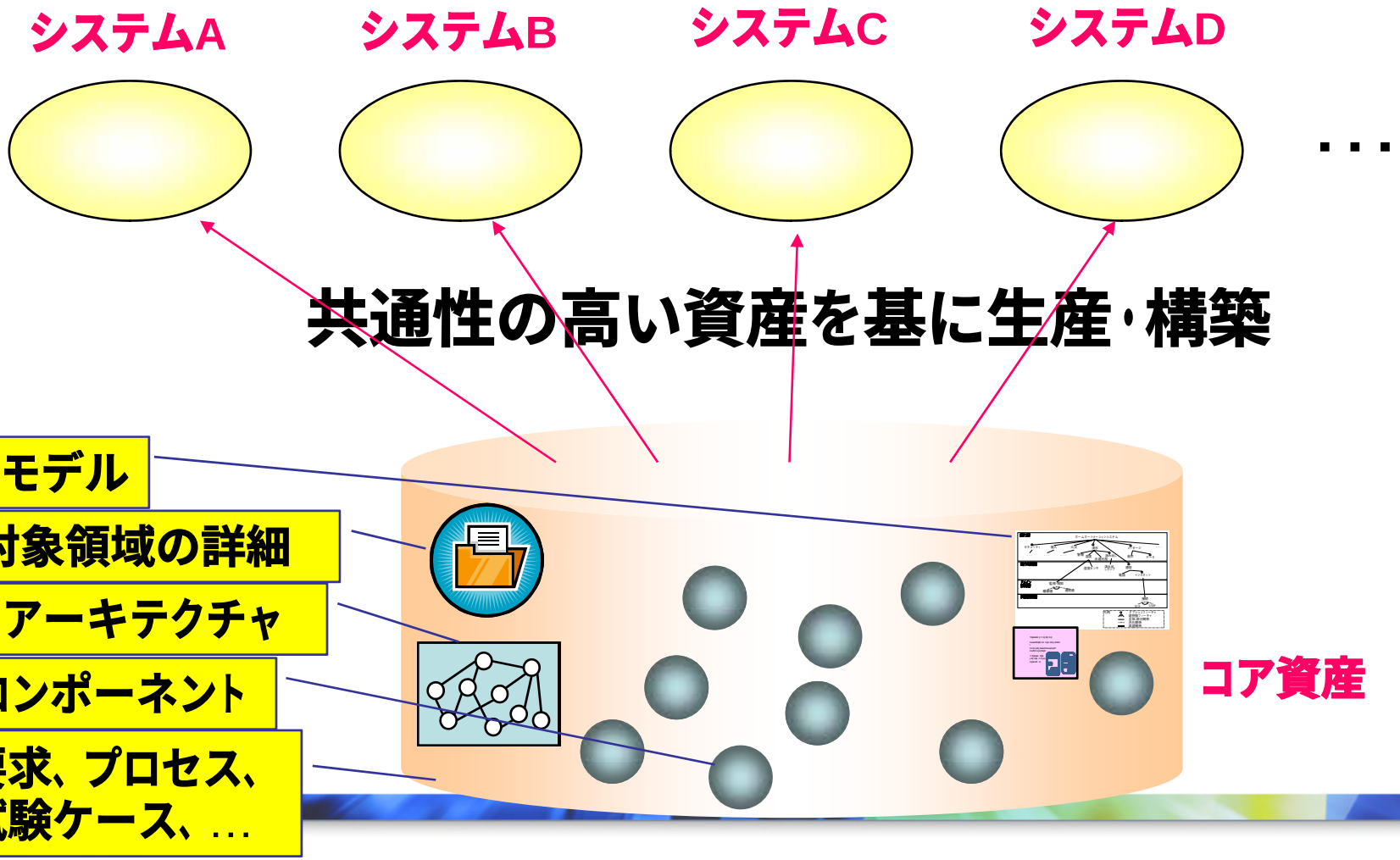
(同じことは二度しない)



SPLE の仕組み

プロセス

(一度作って何度も使う)



終わりに

SPL の戦略的重要性

戦略的重要性

- 事業戦略・目標からの連続性を確保することになる
  - 超上流活動の認識
  - 無駄なく網羅的に目標をカバーするプロセス
- 事業のコアを識別することになる
- 品質を確保した上で生産性を向上させることになる
  - 桁を上げる向上は再利用以外ではありえない

まとめ — SPLを裏側から見て

- 良いシステムとは、顧客満足をもたらすシステム
  - その開発のためには、顧客/市場 - 企画/提案 - 設計 - 生産・構築を、縦も横も切れ目なくつなげなくてはならない
- 事業にはコアとなるものがある
  - システムを開発する上では、そのコアを中心に据え、さらに周辺部を押さえなくてはならない
- 多くのシステムは変種を生む
  - そこでは再利用を体系的に進めることが鍵

参考情報

SPLE のプロセスと技術 インタネット リソース リスト

- Software Product Line Conferences
  - http://www.splc.net/
- Feature-Oriented Reuse Method
  - http://selab.postech.ac.kr/classes/eece700A/materials/papers/1_Feature-Oriented%20Product%20Engineering.pdf (IEEE Software 特集記事)
  - Feature-Oriented Domain Analysis
    - http://www.sei.cmu.edu/str/descriptions/foda.html
- SEI Software Product Line Home Page
  - http://www.sei.cmu.edu/productlines/index.html
  - A Framework for Software Product Line Practice, Version 5.0 (SEI)
    - http://www.sei.cmu.edu/productlines/framework.html
- BigLever社による SPLE 情報ページ
  - http://www.softwareproductlines.com/

SPLE のプロセスと技術 文献リスト

- ライフサイクルプロセス、組織、移行戦略
  - [Pohl05] (日本語)
- 方法論
  - [Kang02] (概要)
  - [Gomaa04] (組込みに特化)
- 事例研究、SPLE対応度評価フレームワーク (FEF)
  - [vdLinden07]
- SPLE を取り巻く課題
  - [Clements01]
- ジャーナル
  - [IPSJ09] (日本語)
  - [Yoshimura07] (日本語)

参考文献

[Abeta08]
安部田 章、組込みシステム産学官連携技術交流会in熊本－プロダクトライン開発ワークショップ講演資料、2008

[Clements01]
Paul Clements, "Linda Northrop; Software Product Lines; Practices and Patterns"; Addison-Wesley, 2001 ［邦訳：前田 卓雄 訳『ソフトウェアプロダクトライン――ユビキタスネットワーク時代のソフトウェアビジネス戦略と実践』日刊工業新聞社］

[Gomaa04]
Hassan Gomaa, “Designing Software Product Lines with UML – From Use Cases to Pattern-based Software Architectures,” Addison-Wesley, 2004

[Ishida07]
Yuzo Ishida; "Software Product Lines Approach in Enterprise System Development", In: Proceedings of the 11th Software Product Line Conference, Kyoto, Japan, September 10-14, IEEE Computer Societyt, 2007, pp. 44-53.

[IPSJ09]
情報処理 50巻 4号　特集「ソフトウェア再利用の新しい波 –広がりを見せるプロダクトライン型ソフトウェア開発–」

[JASPIC08]
日本SPIコンソーシアム (JASPIC) プロダクトライン分科会2008年度成果物（JASPIC会員にのみ公開）

[Kang02]
KyoChul Kang, Jaejoon Lee, and Patrick Donohoe, “Feature-Oriented Product Line Engineering,” IEEE Software, Vol. 9, No. 4, July/August 2002, pp. 58-65.

[Kang02a]
KyoChul Kang, Patrick Donohoe, Eunman Koh, Jaejoon Lee, and Kwanwoo Lee, “Using a Marketing and Product Plan as a Key Design Driver for Product Line Asset Development.” G. Chastek, editor, Software Product Lines: Proceedings of the Second Software Product Line Conference (SPLC2), San Diego, U.S.A., Aug. 19-22, 2002, Heidelberg, Germany: Springer Lecture Notes in Computer Science Vol. 2379, 2002, pp. 366-382.

[Pohl05]
Klaus Pohl, Günter Böckle, Frank van der Linden, "Software Product Line Engineering - Foundations, Principles, and Techniques," Springer Verlag, Heidelberg, Germany, 2005. ［邦訳：林 好一、吉村 健太郎、今関 剛 訳『ソフトウェアプロダクトラインエンジニアリング――ソフトウェア製品系列開発の基礎と概念から技法まで』エスアイビー・アクセス］

[vdLinden07]
Frank van der Linden, Klaus Schmid, Eelco Rommes (eds.), "Software Product Lines in Action – The Best Industrial Practice in Product Line Engineering", Springer Verlag, Berlin, 2007

[Yoshimura07]
吉村 健太郎、「製品間を横断したソフトウェア共通化技術」、情報処理 48巻 2号 2007/02 pp. 171-176