事例に見る
反復開発からアジャイルへ
プロジェクト管理の勘所

＜　非ウォーターフォール事件簿　そこから学ぶこと　＞

豆蔵
mamezou

2011年 7月 8日

(株)豆蔵　堀江 弘志

非ウォーターフォールの基本的なこと

非ウォーターフォール事件簿

非ウォーターフォール成功の秘訣

自己紹介

- 略歴
  - 専門分野　プロジェクト管理、プロセス改善
  - ユーザ系SI企業でシステム開発やプロジェクト管理に携わった後、2005年より株式会社豆蔵に在籍。プロジェクト管理に関するコンサルティング、PM育成、CMMIによるプロセス改善などを手がける
  - PMP、認定スクラムマスター
- 主な執筆・記事など
  - 「プロジェクトマネジャーに贈るプロセス改善事例集」（TechTarget）
  - 「BABOK2.0を読んでみよう」（@IT情報マネジメント）
  - 「仕組みがわかるBABOK（仮称）」（翔泳社より8月頃出版予定）

非ウォーターフォールの基本的なこと

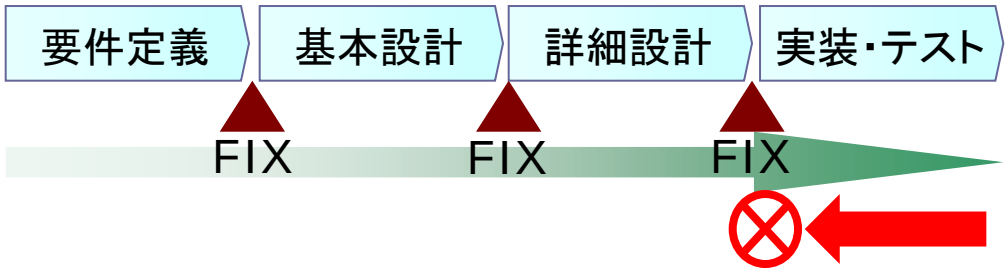
- 開発現場の実情
- 非ウォーターフォールとは
- 乱立する開発プロセス
- UP（反復型プロセス）
- Scrum（アジャイルプロセス）
- 認定スクラムマスター研修の様子
- それでもウォーターフォール

開発の実情

- 業務が複雑かつ曖昧
- 開発期間中に要求仕様が変化しがち
- 開発初期段階ですべての要求を見通せない
- 異種混合アーキテクチャ
- 技術刷新が激しい
- 短期開発
- 開発経験者が少ない

非ウォーターフォールとは

- ウォーターフォールとは



プロジェクトの終盤に
重大な技術問題発覚！

プロジェクトの終盤に
仕様漏れが発覚！

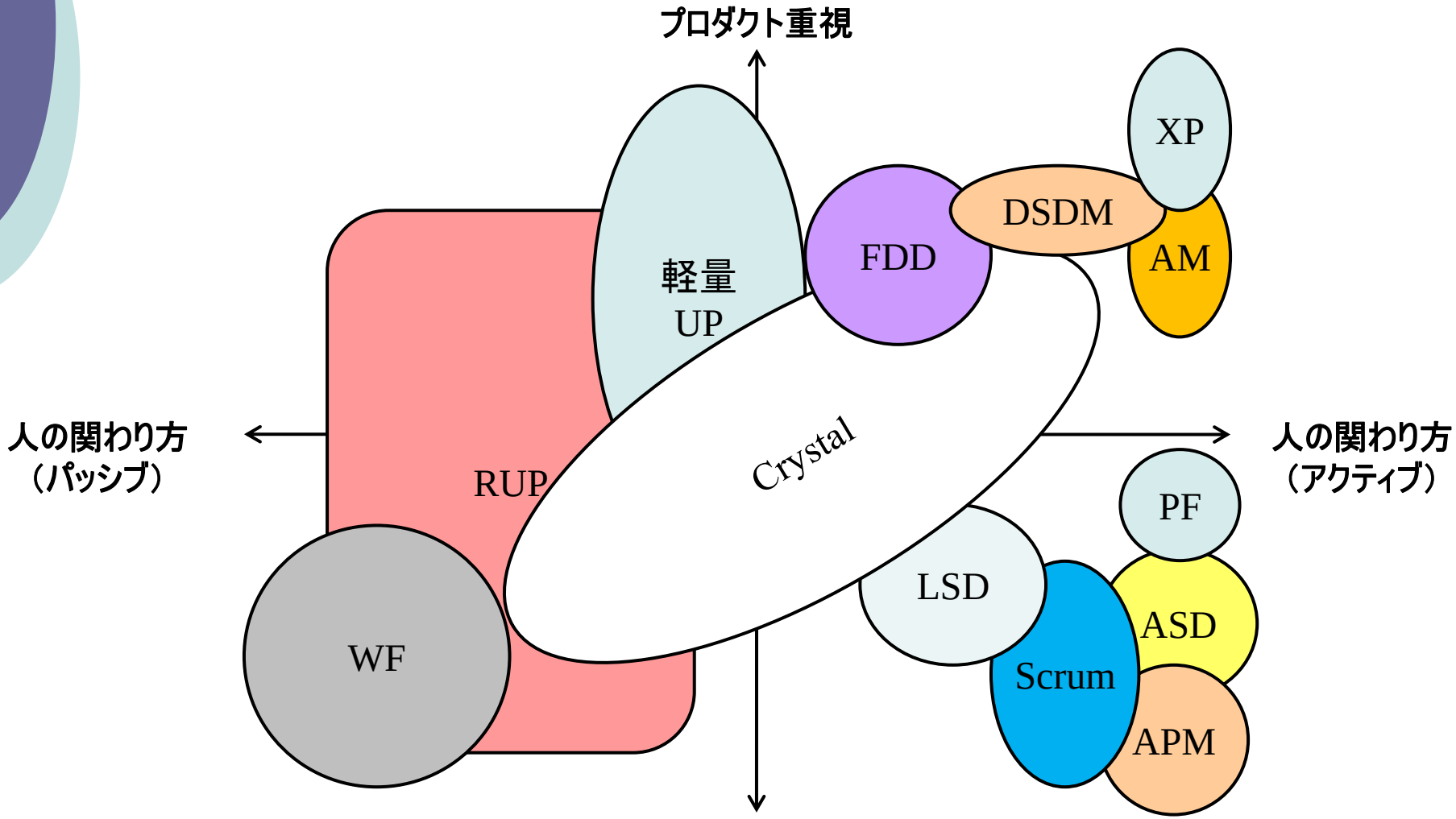
顧客の期待とは
異なるシステムが！

- 非ウォーターフォールとは
  - 前工程へ遡れる=変更の柔軟な受け入れ

    ビジネス価値から正しい要求とは何かを追求できるプロセス
  - 上・下流が近づく=リスクの早期顕在化

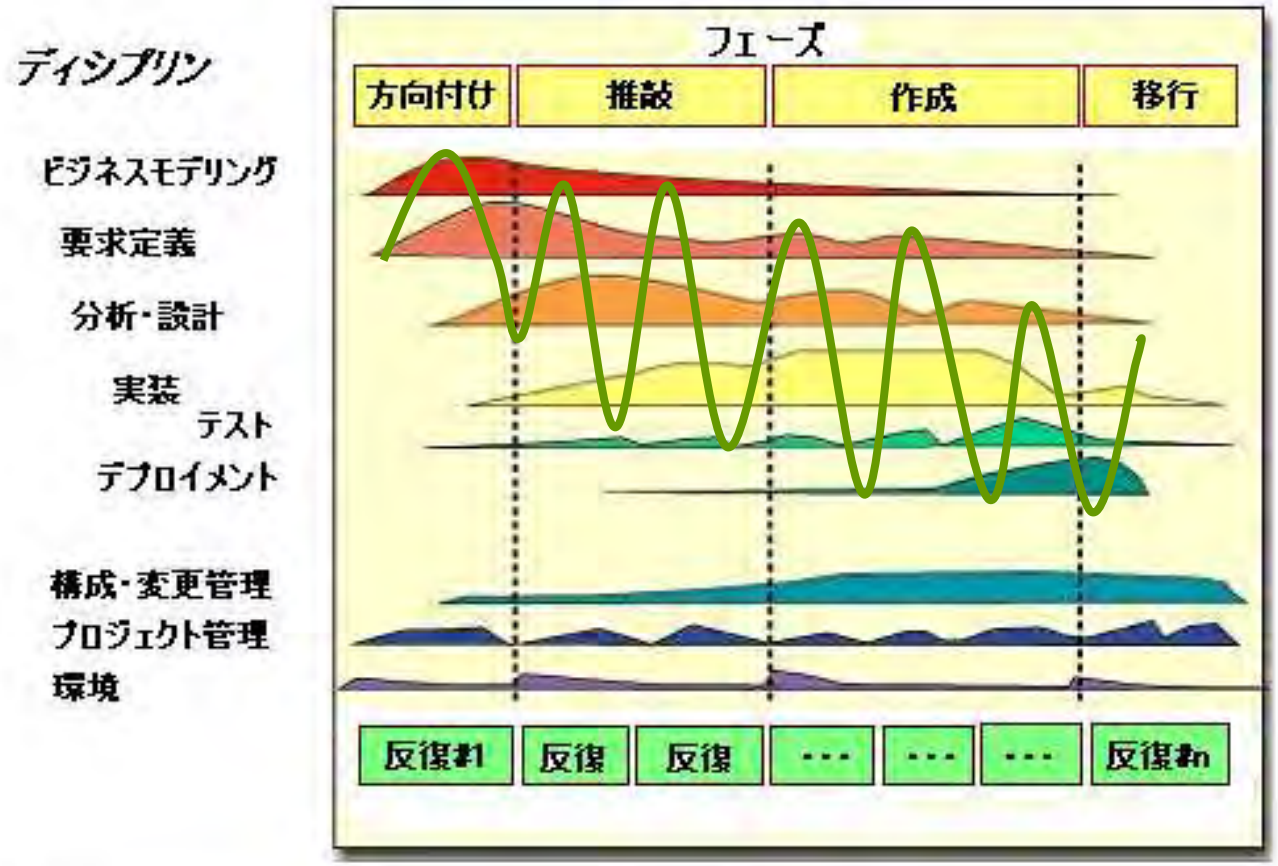
    さまざまな技術問題を早期に対処できるプロセス

乱立する開発プロセス



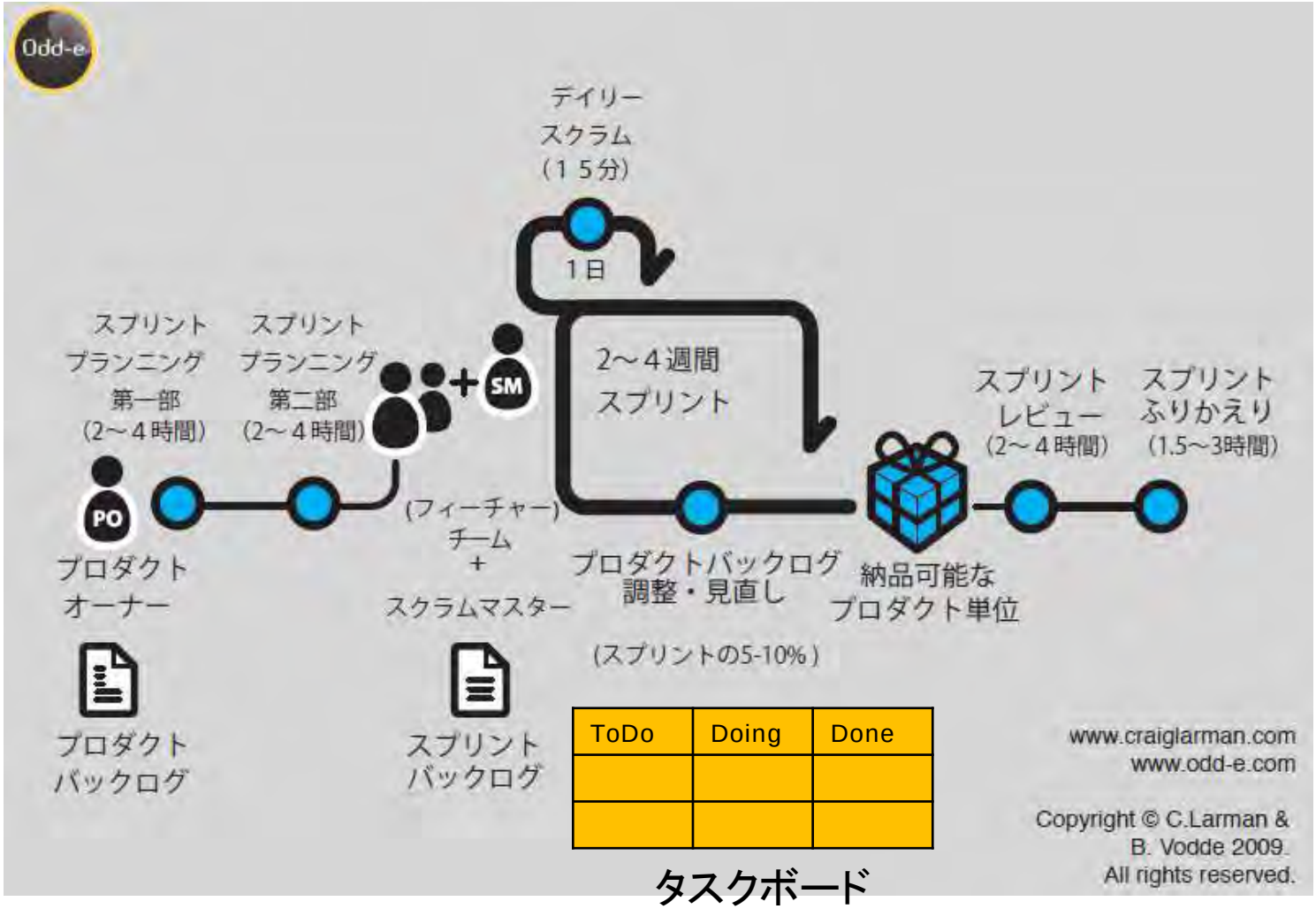
資料引用：アジャイルプロセス協議会

UP（反復型プロセス）



出典:「The Rational Unified Process: by Philippe Kruchten, 2000, Addison Wesley」

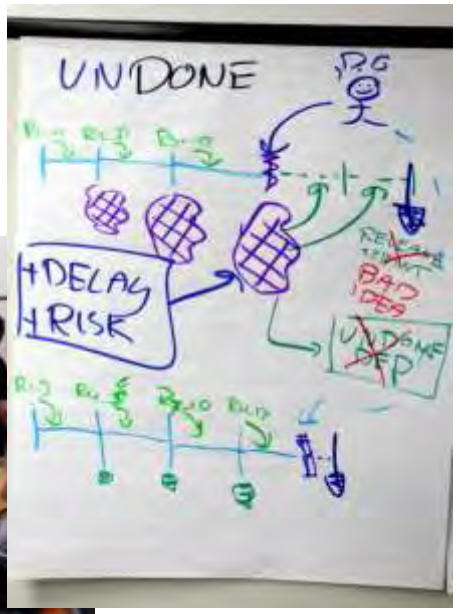
Scrum（アジャイルプロセス）



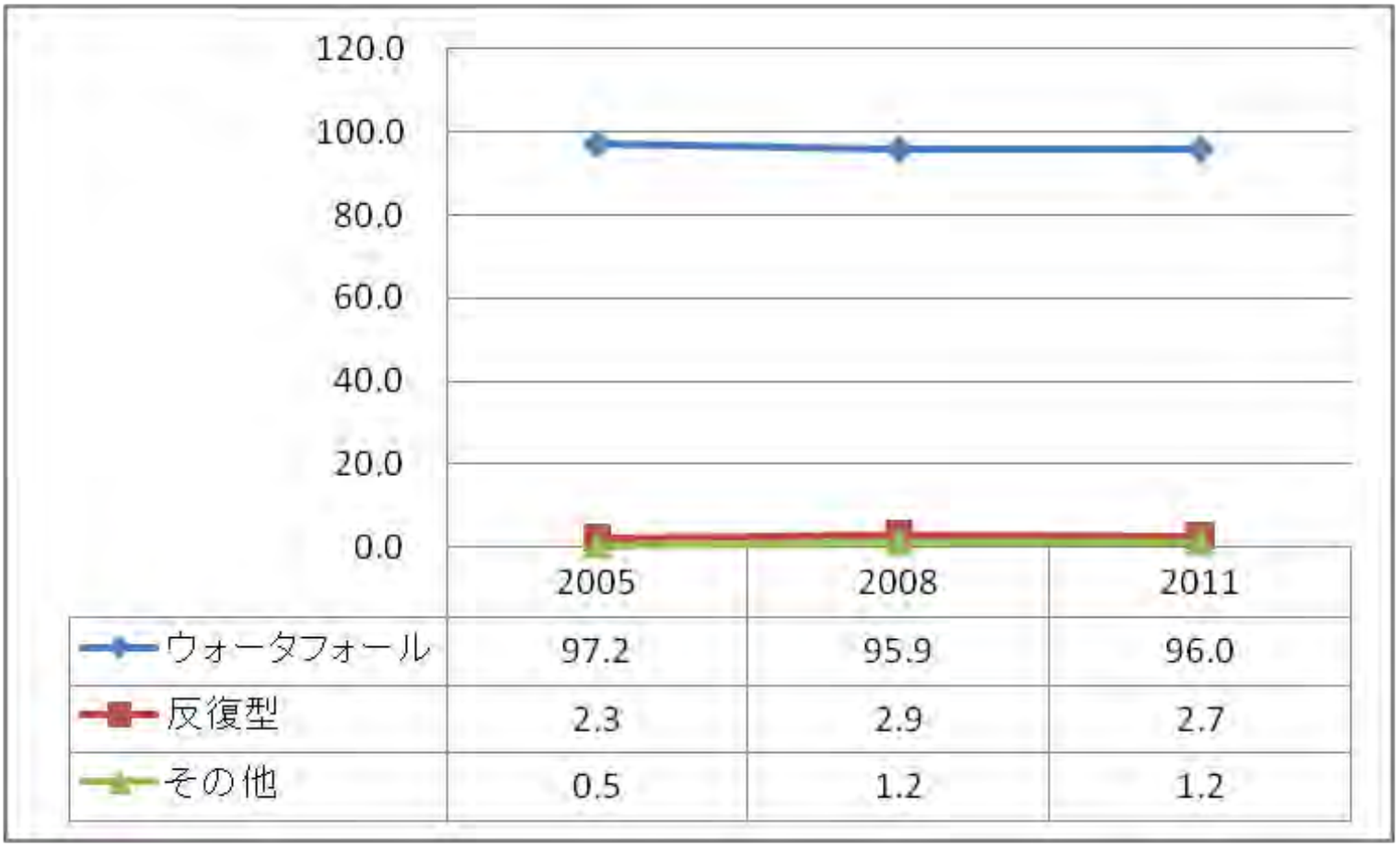
タスクボード

認定スクラムマスター研修の様子(余談)

- 2011/6/16-17 東京開催
  - 14名の認定スクラムマスターが生まれた



それでもウォーターフォール



	2005	2008	2011
ウォータフォール	97.2	95.9	96.0
反復型	2.3	2.9	2.7
その他	0.5	1.2	1.2

データ引用：ソフトウェア開発データ白書（IPA/SEC）

非ウォーターフォール事件簿

- 事例１：過剰な期待にUPは応えられるか
- 事例２：UNDONE地獄から抜けられない
- 事例３：Scrumが向いている小規模ゲーム開発

事例1：プロジェクト概要

項目	概要
ドメイン	セミナー運営システム
規模・期間	100人月、10ヶ月
技術	WebベースのJavaアプリケーション
関係組織	顧客企業の業務3部門、情報システム運用など
開発体制	顧客側：責任者（オーナー）、リーダー、業務担当 開発側：PM、アーキテクト、アナリスト（要件定義）担当、開発エンジニア 5名スタート、ピーク時15名体制
採用プロセス	UP：統一プロセス

事例1：UP採用の経緯

- システム企画の反故
- イベントへの出展・デモ
- 技術的チャレンジ

事例1：アプローチ

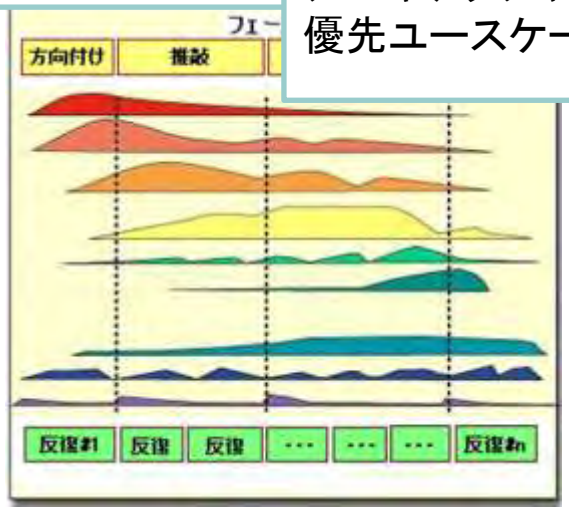
企画

開発（UPフェーズ）

1ヶ月	2ヶ月	4ヶ月	1ヶ月
方向付け	推敲	作成	移行

概念モデリング
ユースケース定義
プランニング

アーキテクチャ構築
優先ユースケース実現



（ひたすら）
ユースケース実現

データ移行
システム移行

事例1：方向付けフェーズの様子

- 要件定義の真剣さ欠如
  - 過剰な方法論への期待
  - 「いま、ここで決めなくてもいいんでしょう」
- ユースケース駆動
  - エンジニアリングの明確化
  - 「慣れないなあ。このドキュメント」
- 方向付け定義書？
  - 成果物に対するレビューと承認の意味
  - 「要件定義書はどれ？」

事例1：推敲フェーズの様子

- アーキテクチャへの過剰な要求
  - 他のシステムでも利用可能な共通基盤
  - 「アーキテクチャを作りながら開発するんですか」
- 初回の反復だから仕方ない
  - 反復1回目でアーキテクチャ要素の大きいUCを
  - 「反復期間をもう少し延ばしてもらえませんか」
- 推敲が終わらない
  - 推敲の終了基準
  - 「推敲と作成フェーズの違いは何でしょうか」

オープンソースとサードバーティ製品の互換性問題発覚！

事例1：作成フェーズの様子

- W/Fで残作業を一掃しゴールへ一直線
  - 2週間で仕様曖昧性を排除し要件FIX
  - 1ヶ月でアーキテクチャ仕様をスリム化し必要最低限を開発
  - 実現すべき機能要求のスリム化
  - 残りの作業をプランニング
  - 開発要員の増員

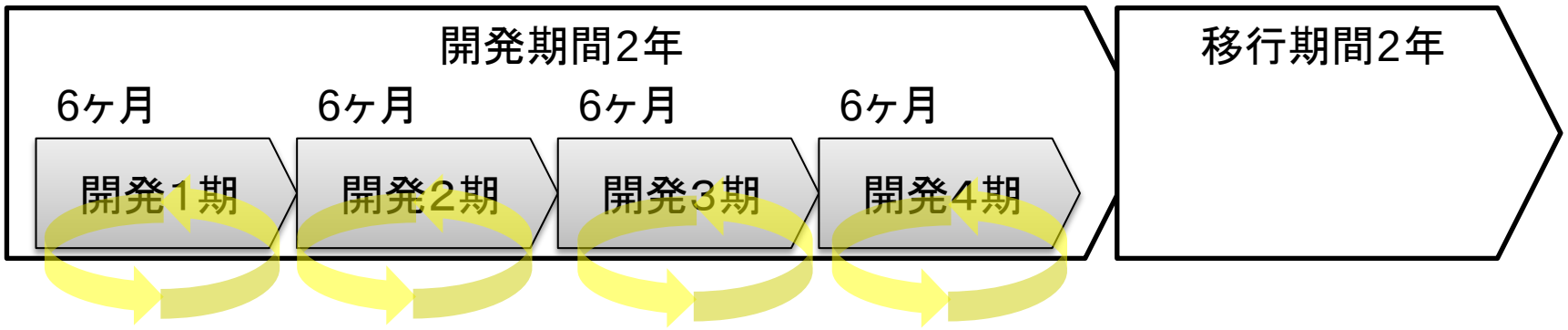
事例１：５つの教訓

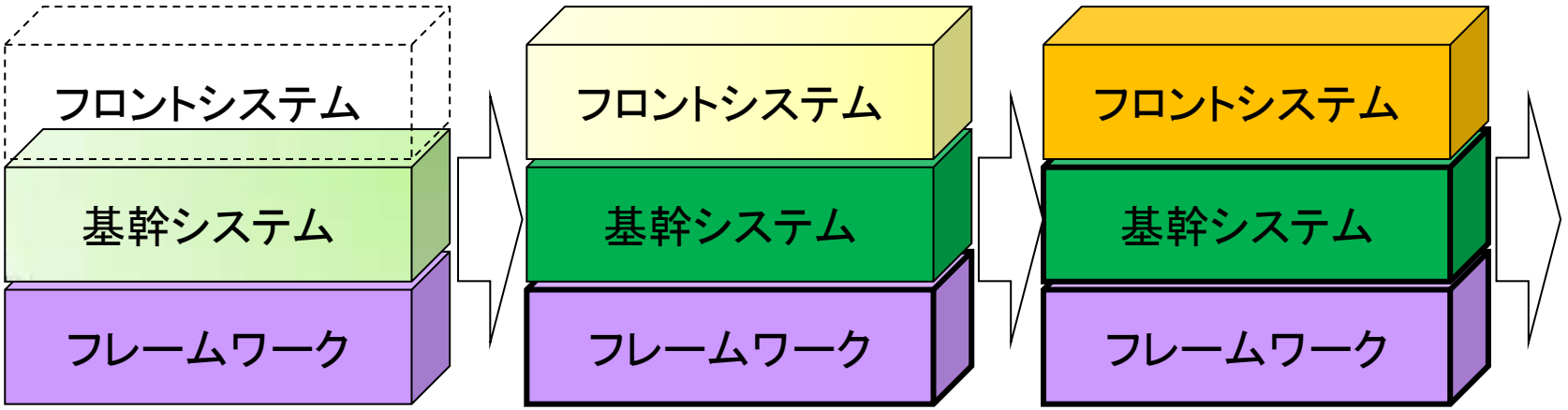
- プロセス理解は顧客を含めて
- ビジネスモデリングは企画フェーズへ
- 要求フィックスは推敲で
- 推敲の負荷を軽減する工夫
- 途中からW/Fへ切り替える柔軟性

事例2：プロジェクト概要

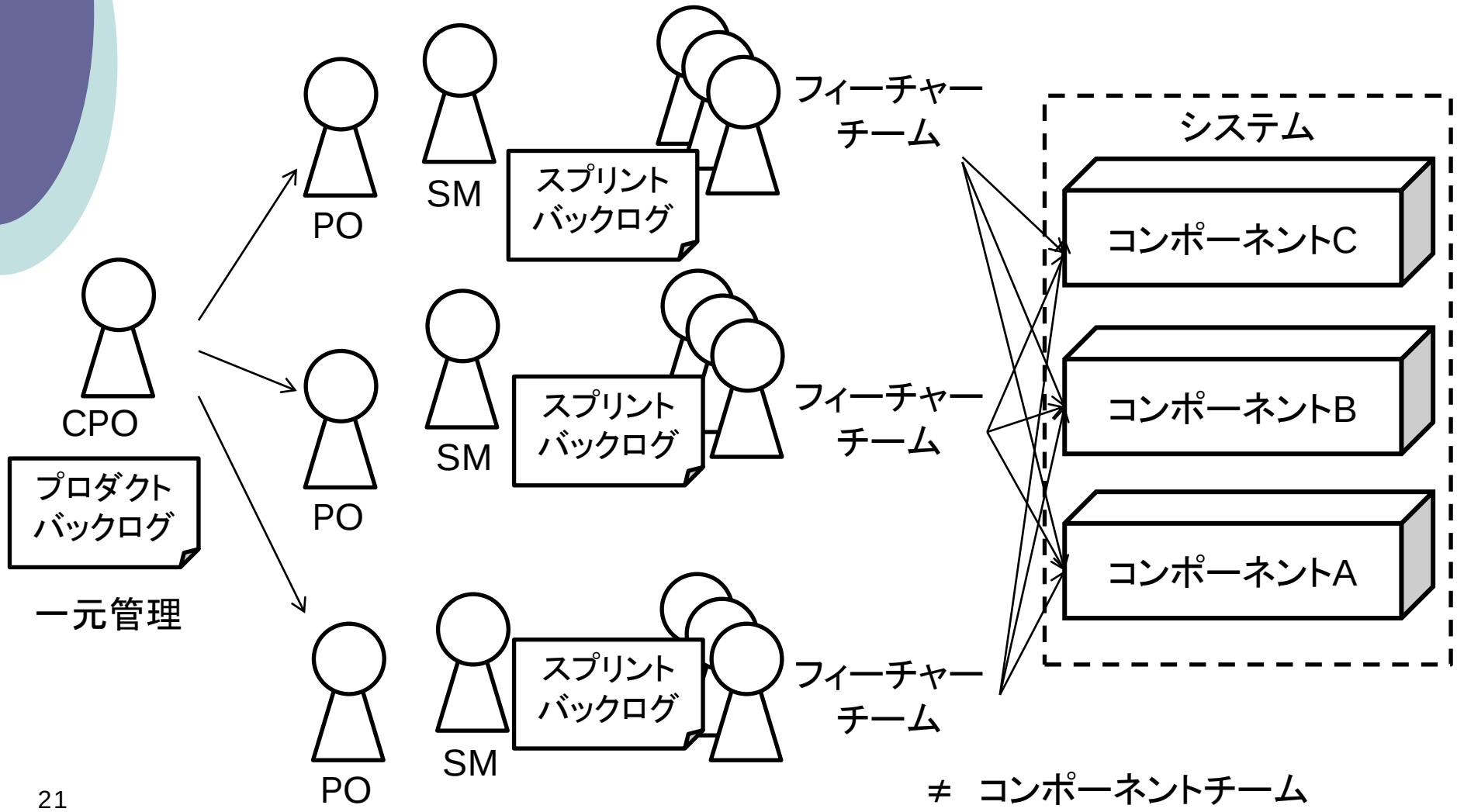
項目	概要
ドメイン	金融系基幹システム再構築
規模・期間	数万人月、4年
技術	WebベースのJavaアプリケーション
関係組織	顧客企業の全部門
開発体制	顧客側：業務改革、情報システム部門 開発側：約10社ベンダー、数百人体制
採用プロセス	アジャイル(Scrum)、W/F　混在

事例2：プロセスアプローチ

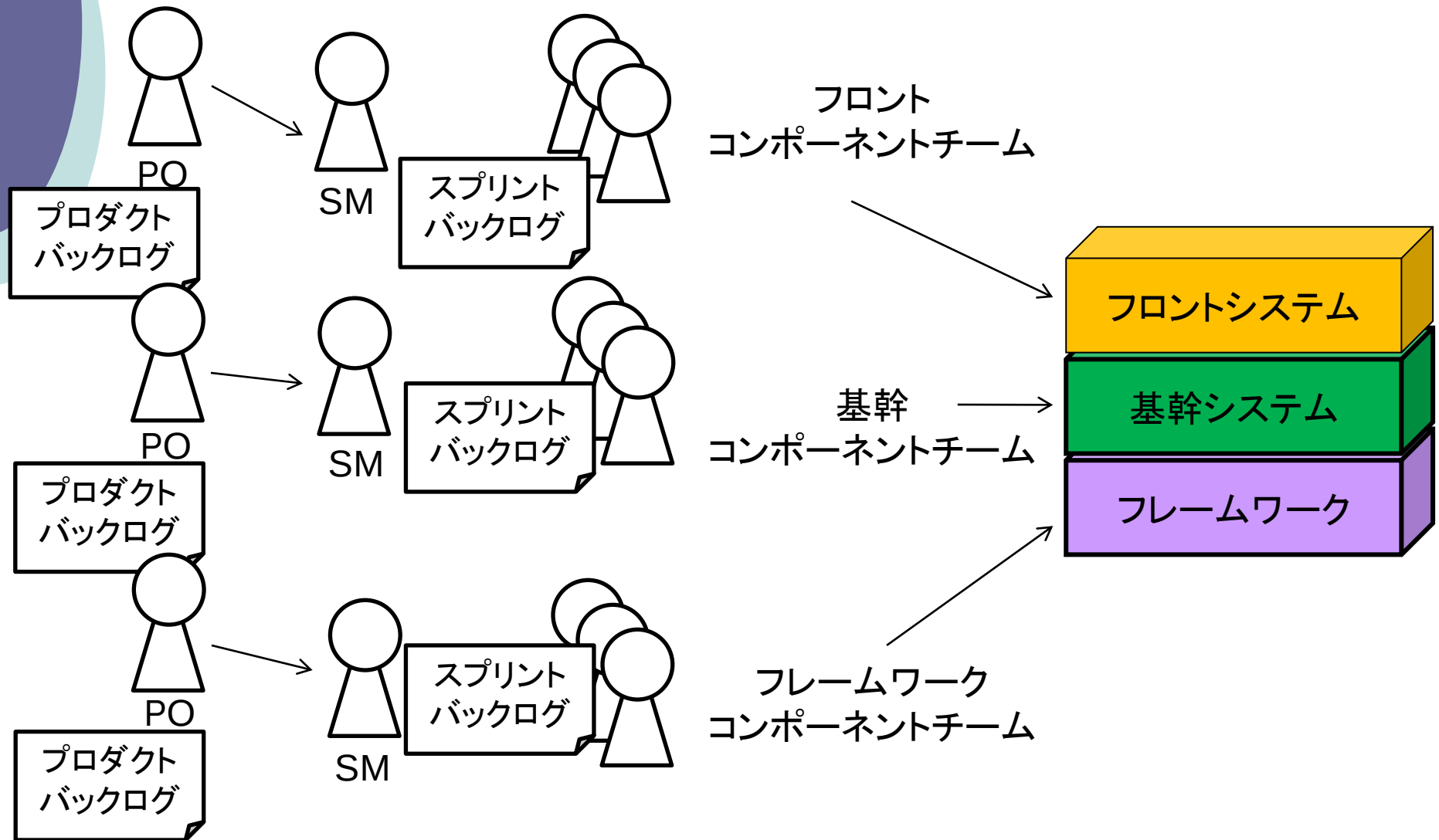




事例2：Scrumのスケーリング（理想）



事例2：Scrumのスケーリング（実態）



事例2：UNDONE地獄

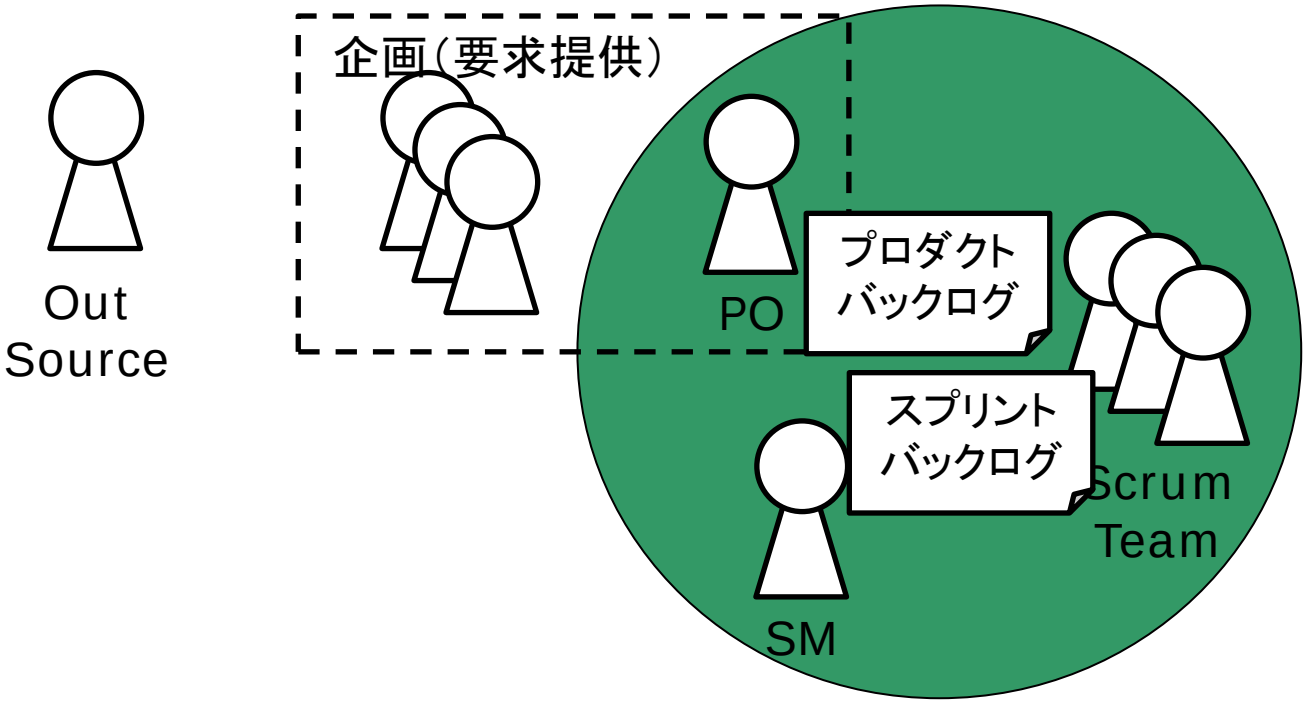
- 仕様調整のオーバーヘッド
  - みんなアジャイル：コンポーネント間の調整に苦戦
- エンジニアリングの不整合
  - 会話重視ですから：成果物があったりなかったり
- 要求の根拠が見えない
  - POが正しい要求を判断できるとは限らない

事例2：5つの教訓

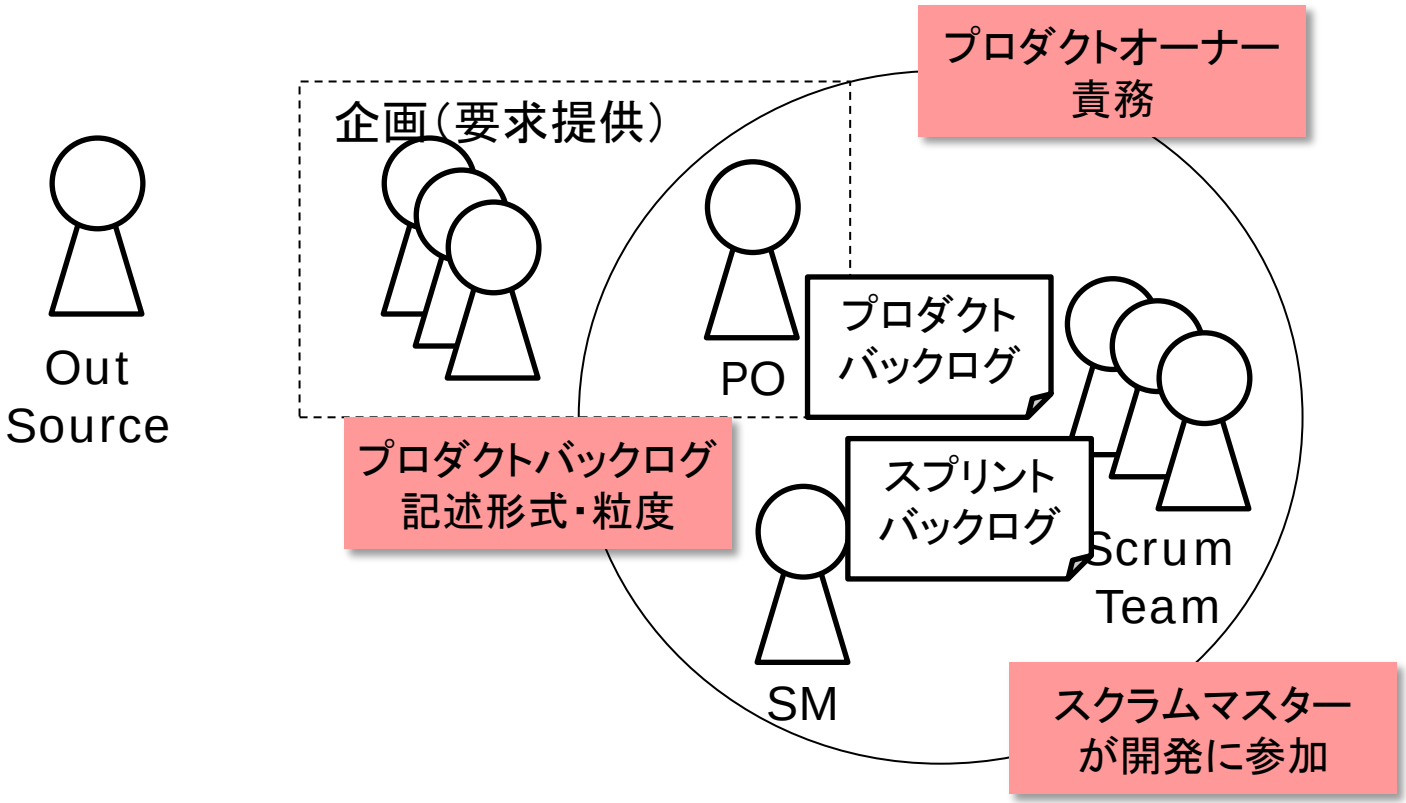
- エンジニアリングを決めて場当たり開発を防止
- 要求スプリントを柔軟に取り入れる
- POが最重要（ハイパフォーマーは開発チームだけではない）
- チーム編成を柔軟に見直す
- 必要に応じてスプリント間に休息を

事例3：比較的良好なScrum導入例

- プロバイダ事業者：携帯ゲームコンテンツ開発
- 1スプリント=2W　×　9スプリント（約3ヶ月）



事例3：比較的良好なScrum導入例



非ウォーターフォール成功の秘訣

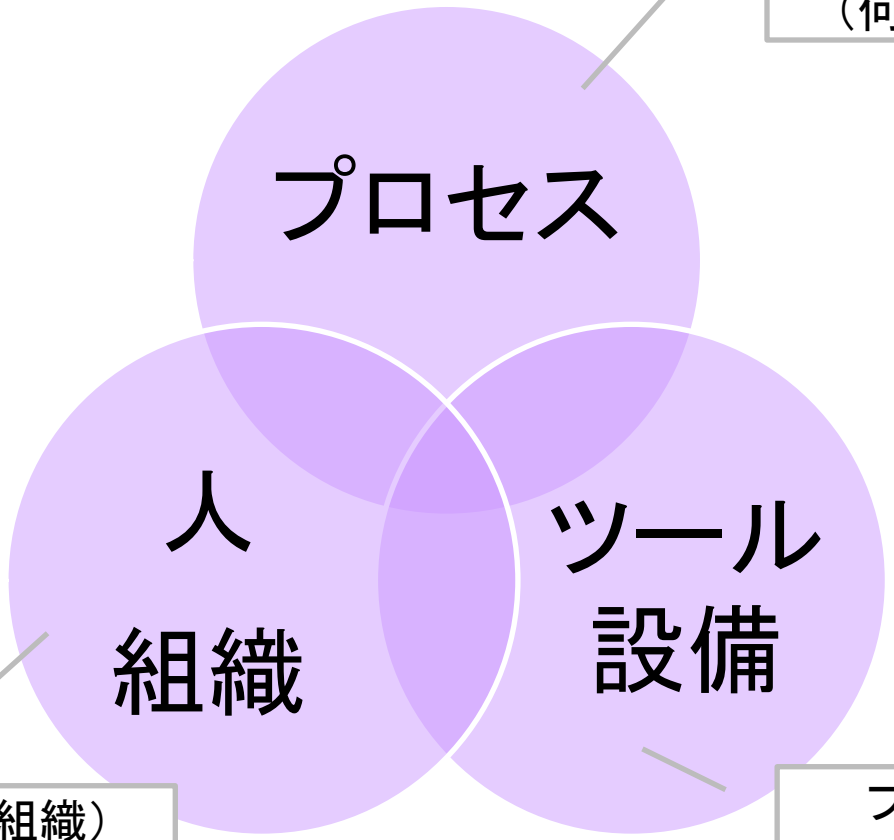
- 非ウォーターフォール適用のポイント
- 開発プロセスの基本

非ウォーターフォール適用のポイント

- 要求の収束と固定
  - ユースケースでもプロダクトバックログでも、収束させ固定化する時期を予め合意しておくこと（特に予算制約、期間制約が大きい場合）
- 無計画は重厚計画より悪い
  - 計画＆コマンドコントロールの行き過ぎはモティベーションダウンを引き起こすが、無計画はプロジェクトのメルトダウンを引き起こす
- エンジニアリングは最低知識
  - エンジニアリングを知らない人にアジャイル開発を行わせてはならない。最悪のシステムが出来上がる
- もちは餅屋の役割分担
  - プロダクト責任と開発責任は明確に分ける。自由と責任を与え、開発者には作ることに専念させること
- 整理整頓・行き届いた開発環境
  - 構成管理、変更管理、継続的インテグレーション・・・これらを支える開発環境は開発前に整えておくこと

開発プロセスの基本

どのような順序で何を行うか
（何を割愛してもよいか）



スキルをもった人（組織）
やる気のある人（組織）

プロセス効率を支える
ツールや設備

ご静聴ありがとうございました

ご質問、ご意見につきましては
horie@mamezou.com へ