ハイブリッドアジャイルへ至る道と、その【真】の姿

～日本流アジャイルを模索し続けた15年をふりかえる～

セッション概要（公式サイトから）

2002年からアジャイル開発に取り組み始めた英氏は、2008年頃までに様々なアジャイル開発手法を実践し、評価してきました。また、大規模かつ受託開発が主流の日本で、どうやってアジャイル開発手法を役立てるか、試行錯誤を繰り返し、研究を続けてきました。そして考案されたのが「ハイブリッドアジャイル」です。

ウォーターフォールとアジャイル、相反するように見える２つの手法を、様々な形で組み合わせることで、プロジェクトの課題に幅広く対応できる、実に奥深い考え方です。

当セッションでは、英氏の15年間の研究・実践でのエピソードを語っていただきながら、日本でアジャイル開発を幅広く活用するヒントを考えます。

Q1. １５年前をふりかえる

- アジャイルに取り組み始めた頃の状況
  - 2002～2003年頃
  - ご自身の周囲の状況
  - 世の中の動向

- アジャイルに取り組み始めたきっかけと苦労
  - 2000年頃まではどんなことをされていたのですか
  - 何をきっかけにアジャイルを始めたのですか
  - その時にどんな苦労がありましたか

HITACHI
Inspire the Next

Q1：15年前をふりかえる

HITACHI
Inspire the Next

株式会社日立ソリューションズ
業務革新統括本部　IT技術推進センタ

ハナブサ　シゲオ
英　繁雄

私のアジャイル取り組み履歴

- 2002年　社内システム2件をアジャイル開発し、国内事例発表
- 2003～　さまざまなアジャイル手法を評価
  2008年　委託開発にはアジャイルは避け、アジャイルプラクティスの良いところをウォーターフォールに取り入れるようにしていた
  製品開発には、アジャイルを推奨していた
- 2009年　大規模な委託開発向けアジャイル手法開発に取り組む
  海外のアジャイルコンサル企業視察（開発現場や委託開発現状）
- 2010年　大規模システム向け日本版アジリティ開発手法発表
- 2013年　「ハイブリッドアジャイルの実践」 出版
- 2014年　BRMSで自動生成するアジャイル手法　Agile BRMS on Cloud 開発
- 現在　社内システムをアジャイル手法で継続開発

1-2. アジャイルに取り組み始めた頃の状況

HITACHI
Inspire the Next

2002～2003年頃

- XP/Scrumで社内システム2件を開発
- 国内のアジャイル事例として社外発表

周囲の状況

- アジャイルの取り組みを呼びかけても、反応はあまりなかった
- アジャイル ＝ いいかげん　の認識

世の中の動向

- XPが話題、さまざまなアジャイル手法の中Scrumが注目されはじめていた

  技術的なプラクティス中心のXPに、Scrumは自己組織化するプラクティスが加わった
- アジャイルプロセスは勉強段階。アジャイル書籍出版や講演がよく開催されるようになってきた。国内事例はほとんど無し。

1-3. アジャイルに取り組み始めたきっかけと苦労

HITACHI
Inspire the Next

2000年頃までにやっていたこと

- 生産技術部で、主にWebアプリケーション開発の開発基盤整備
- 開発ツール導入、フレームワーク開発、ウォーターフォールプロセス整備、UML設計、見積もり手法(FunctionPoint法、UseCasePoint法)など

アジャイルのきっかけ

- 開発ツールや自動生成技術での生産性向上に限界を感じていた
- 全工程に影響するプロセス改善に、生産性向上の余地があると感じた

その時の苦労

- 生産技術部門であるため、アジャイル手法の評価として実施できたが、正式な開発案件には、会社標準の開発手順ではないため承認されない。
- 設計ドキュメントは、最小限で初期開発を行ったが、数年後のエンハンス作業委託開発時に設計書作成することになった。
- 品質保証部門の既存の役割では、参画できない。

Q2.各アジャイル手法の特徴

- 様々なアジャイル開発手法を試したとのこと
  - 部分的なプラクティス導入など
- 各手法をやってみて感じた特徴（良い/悪い）
  - 特に、現在主流であるXP/Scrumの印象は？
- 実践にあたっての日本文化との相性

※本セッションでは「契約」にはあまり触れません

HITACHI
Inspire the Next

Q2：各アジャイル手法の特徴

2-1. 様々なアジャイル開発手法

HITACHI
Inspire the Next

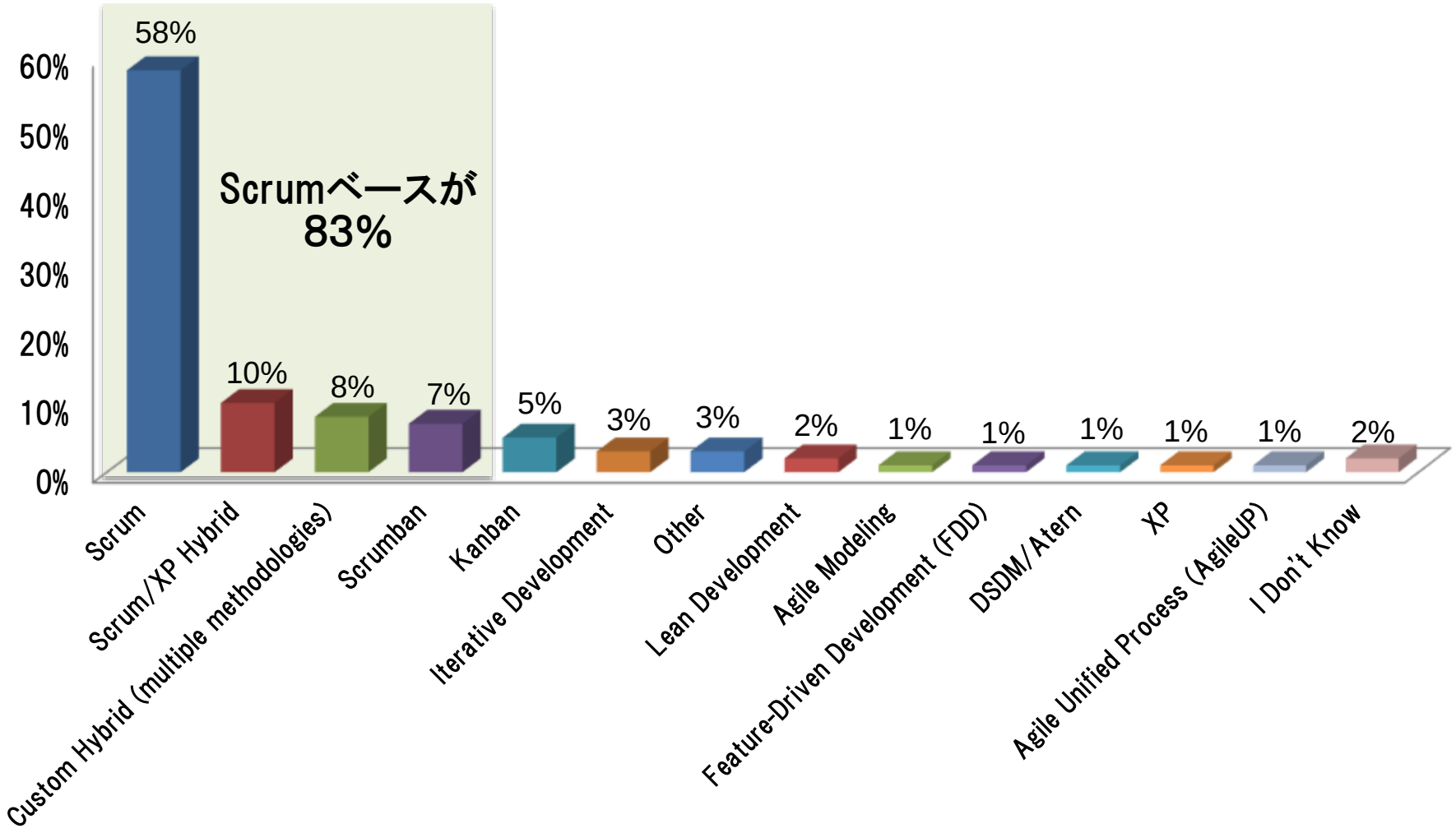
アジャイル開発に分類される手法

- XP(eXtreme Programming):Kent Beck, Ward Cunningham, Ron Jeffries, Martin Fowler, Robert C. Martin, James Grenning
- Scrum:Ken Schwaber, Jeff Sutherland, Mike Beedle
- FDD(Feature-Driven Development):Jeff De Luca, Peter Coad, M.A. Rajashima, Lim Bak Wee, Paul Szego, Jon Kern, Stephen Palmer
- Lean Software Development:Mary Poppendieck, Tom Poppendieck
- Kanban:David Anderson
- DSDM:Arie van Bennekum
- Crystal:Alistair Cockburn

2-2. 様々なアジャイル開発手法

HITACHI
Inspire the Next

アジャイルで最も利用されている手法はScrum

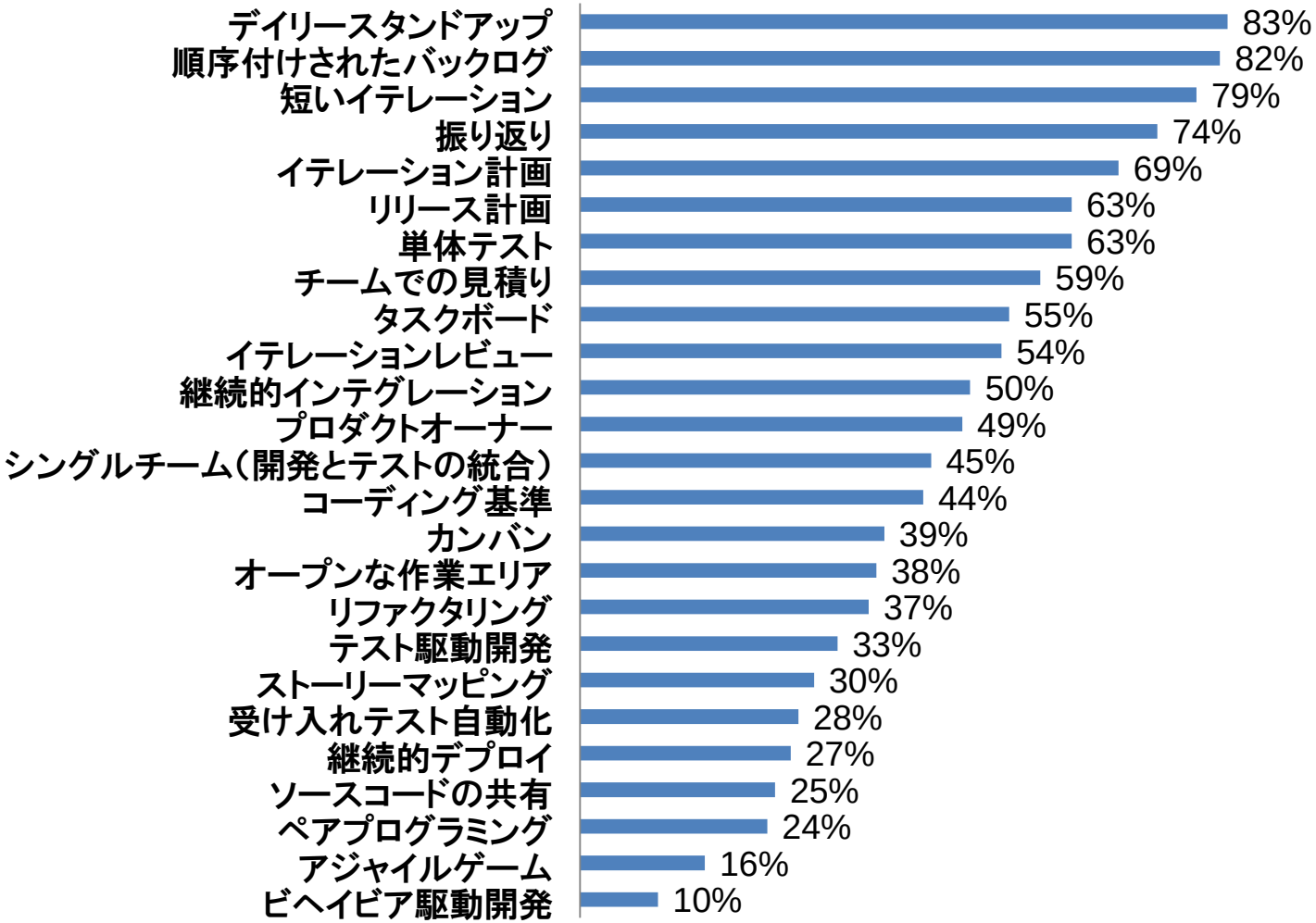


出典：VersionOne-10th-Annual-State-of-Agile-Report（2015年 世界3,880件） から加工（日本語訳、グラフ表示の変更）して作成
http://stateofagile.versionone.com/

2-3. 様々なアジャイル開発手法

HITACHI
Inspire the Next

アジャイルプラクティスの利用は、プロジェクトごとに異なる



出典：VersionOne-10th-Annual-State-of-Agile-Report（2015年 世界3,880件） から加工（日本語訳、グラフ表示の変更）して作成
http://stateofagile.versionone.com/

2-4. 様々なアジャイル開発手法

HITACHI
Inspire the Next

XPとScrum

XPのプラクティス
イテレーション(反復)
共通の用語
開けた作業空間
レトロスペクティブ(振り返り)
テスト駆動開発(TDD)
ペアプログラミング
リファクタリング
ソースコードの共同所有
継続的インテグレーション
YAGNI
責任の受入れ
援護
四半期ごとの見直し
ミラー
最適なペースの仕事
ストーリーの作成
リリース計画
受入れテスト
短期リリース

Scrumのロール
プロダクトオーナー
スクラム開発チーム
スクラムマスター
Scrumの会議
スプリント計画会議
ディリースクラム&スプリント実行
スプリントレビュー会議
レトロスペクティブ
バックログ調整会議

Scrumでは、自己組織化するためのルールが目立つ。

Scrumは、技術的なプラクティスはXPを推奨している。

朱書き:講演者自身が実践したことのあるプラクティス

2-5. 様々なアジャイル開発手法

HITACHI
Inspire the Next

FDDのプラクティス

FDDのベストプラクティス
領域オブジェクトモデリング
ユーザ機能による開発
クラス(コードの個別担当)
ユーザ機能チーム
インスペクション
定期的な構築
構成管理
進捗の報告と可視性

キーワード	内容
フィーチャ(機能)	2週間で開発が完了する程度で、粒度が一定となる
フィーチャ・セット	フィーチャをグループ化したもの
主要フィーチャ・セット	フィーチャ・セットの上位概念
プロジェクトマネージャ	財務面、レポート面を含め、プロジェクトの管理面すべてを担当
チーフアーキテクト	設計セッション、コードレビュー、テクノロジの決定を含め、システムの設計全体を担当
開発マネージャ	開発チームとその作業の調整、リソース問題の処理など、毎日の開発作業を引き受ける
チーフプログラマ	進行中の設計および開発アクティビティにかかわるシニア開発者。1つ以上の機能セットの担当が割り当てられる
クラスオーナー	チーフプログラマの指示下で機能の実装に従って機能の設計、コーディング、テスト、文書化を行う開発者
テスタ	各機能が定義どおりに実行しているかどうかの検証を担当
デプロイヤ	各種環境へのコードの実際のデプロイメントだけでなく、データの定義/形式間の変換も処理
テクニカルライタ	ユーザが必要とするすべてのオンライン文書および印刷文書の作成と保守を担当

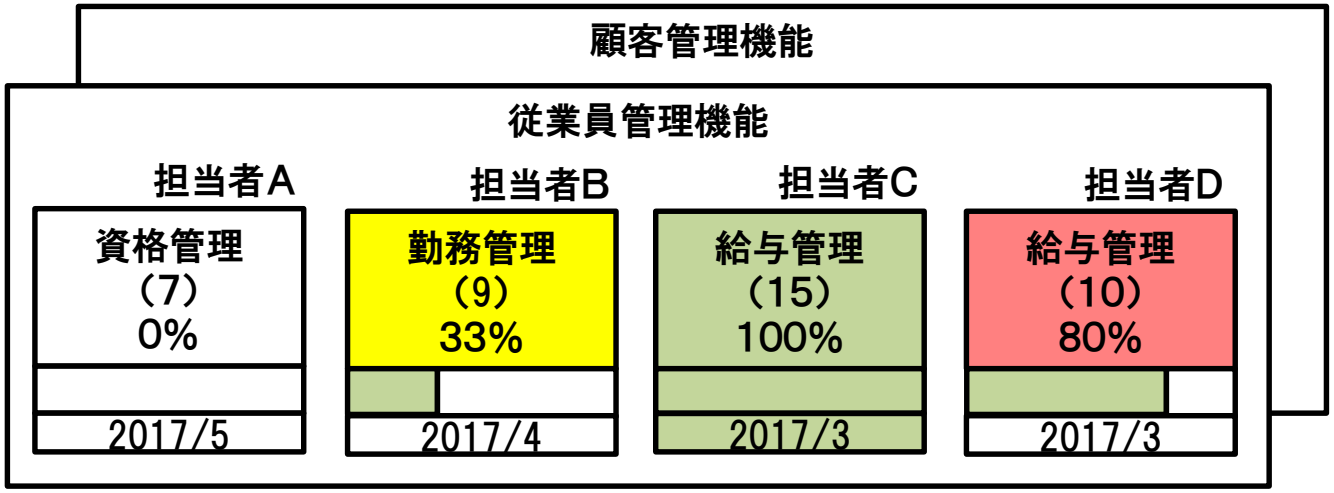
2-6. 様々なアジャイル開発手法

HITACHI
Inspire the Next

FDDの進捗の報告と可視性(パーキングロットチャート)

マイルストーン	重み(例)
領域ウォークスルー	1%
設計	40%
設計インスペクション	3%
コード	45%
コードインスペクション	10%
構築への統合	1%

機能が開発サイクルのどこに位置しているかを正確に把握するには、表の6つの具体的なマイルストーンを追跡する



担当するチーフプログラマの名前
プロジェクト内の各機能セット
各エリア内の機能の数
機能セットの完了率
各機能セットの完了目標年月

白:未着手
黄:進行中で予定どおり
緑:完了
赤:注意(遅延など)

2-7. 様々なアジャイル開発手法

HITACHI
Inspire the Next

Lean Software Development 7つの原則と22の思考ツール

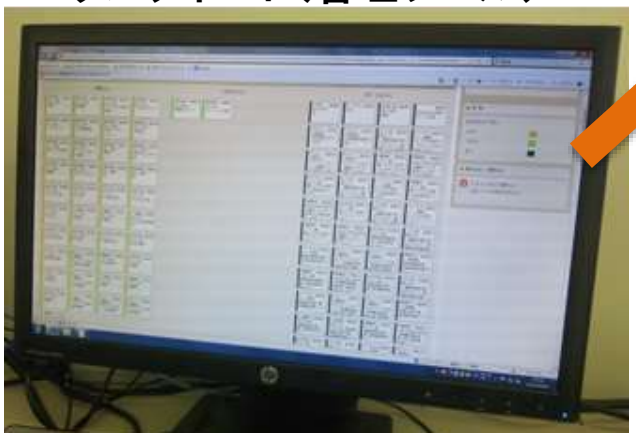
7つの原則	22の思考ツール
ムダを排除する	1:ムダを認識する(未完成作業、余分なプロセス、余分な機能、タスク切り替え、待ち、移動、欠陥) 2:バリューストリームマッピング(工程間の待ち、工程内の実作業時間を見える化する)
学習効果を高める	3:フィードバック 4:イテレーション 5:同期 6:集合ベース開発
決定をできるだけ遅らせる	7:オプション思考 8:最終責任時点 9:意思決定
できるだけ速く提供する	10:プルシステム 11:待ち行列理論 12:遅れのコスト
チームに権限を与える	13:自発的決定 14:モチベーション(所属、安全、能力、進捗) 15:リーダシップ 16:専門知識
統一性を作り込む	17:認知統一性(利用者に対するユーザビリティ、信頼性、経済性のバランス) 18:コンセプト統一性(アーキテクチャなどのシステムの基本概念) 19:リファクタリング 20:テスティング
全体を見る	21:計測 22:契約

HITACHI
Inspire the Next

Kanban はプル型システム

タスクボード(カードウォール)

タスクボード(管理ツール)



- Kanbanは、開発プロセスではない
  ソフトウェアやサービスの企画から開発、リリースするまでの作業を効率化する仕組み
- カードウォールやタスクボードは、進捗を見える化するためのツール
- Kanbanは、**プル型システム**(後工程引き取り方式)
  その結果、最適ペースを可能にする
- Kanbanによる進捗の管理は、タスクボードによって見える化することができる

プル型システムになっていれば、Kanbanシステムと言える

委託開発の契約やワークスタイル改革に、たいへん重要な考え方である

2-9. 様々なアジャイル開発手法

HITACHI
Inspire the Next

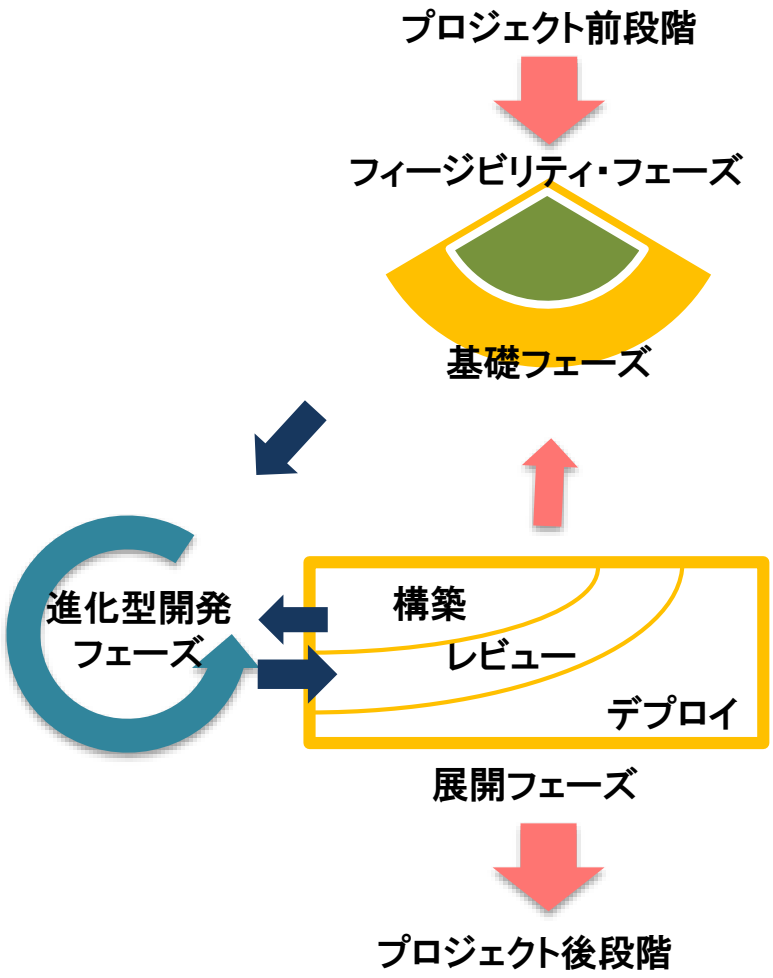
Kanban Systemの主要目的

主要目的	内容
既存プロセスの最適化	見える化や仕掛り制限の導入
高品質でのデリバリー	品質基準など次の作業段階で引き取る条件のポリシーを定義する
リードタイム実現比率の向上	仕掛り量とリードタイムの相関関係により、仕掛り上限を同意する
従業員の満足度向上	良いワークライフバランスの提供により、従業員の士気を高め、高いレベルのパフォーマンスを生み出す
改善を可能とする「ゆとり」の提供	「ゆとり」があれば、緊急要件への対応能力が高まり、プロセス改善を考える幅が生まれる
優先順位付けの簡易化	MoSCoW法などの優先順位付け手法を適用する
システム設計と運用への透明性の提供	仕掛り、デリバリー率(スループット)、品質に透明性をもたらす
「高成熟」組織の出現を可能とするプロセス設計	市場変化に、迅速に対応可能な組織の出現を可能にする

2-10. 様々なアジャイル開発手法

HITACHI
Inspire the Next

DSDMの6段階と4つの主要フェーズ（フィージビリティから展開フェーズ）



① プロジェクト前段階
プロジェクトを明確に定義する

② フィージビリティ・フェーズ
技術的な観点から実現可能か、ビジネス上の観点からコスト効果があるかを確認する

③ 基礎フェーズ
ビジネス上の根拠、作成される潜在的な解決策、およびソリューションの開発と提供がどのように管理されるかについての基本的な理解を確立する

④ 進化型開発フェーズ
反復開発,タイムボックス,MoSCoW法による優先順位付け,モデリング,ワークショップなどのプラクティスを適用して開発を進める

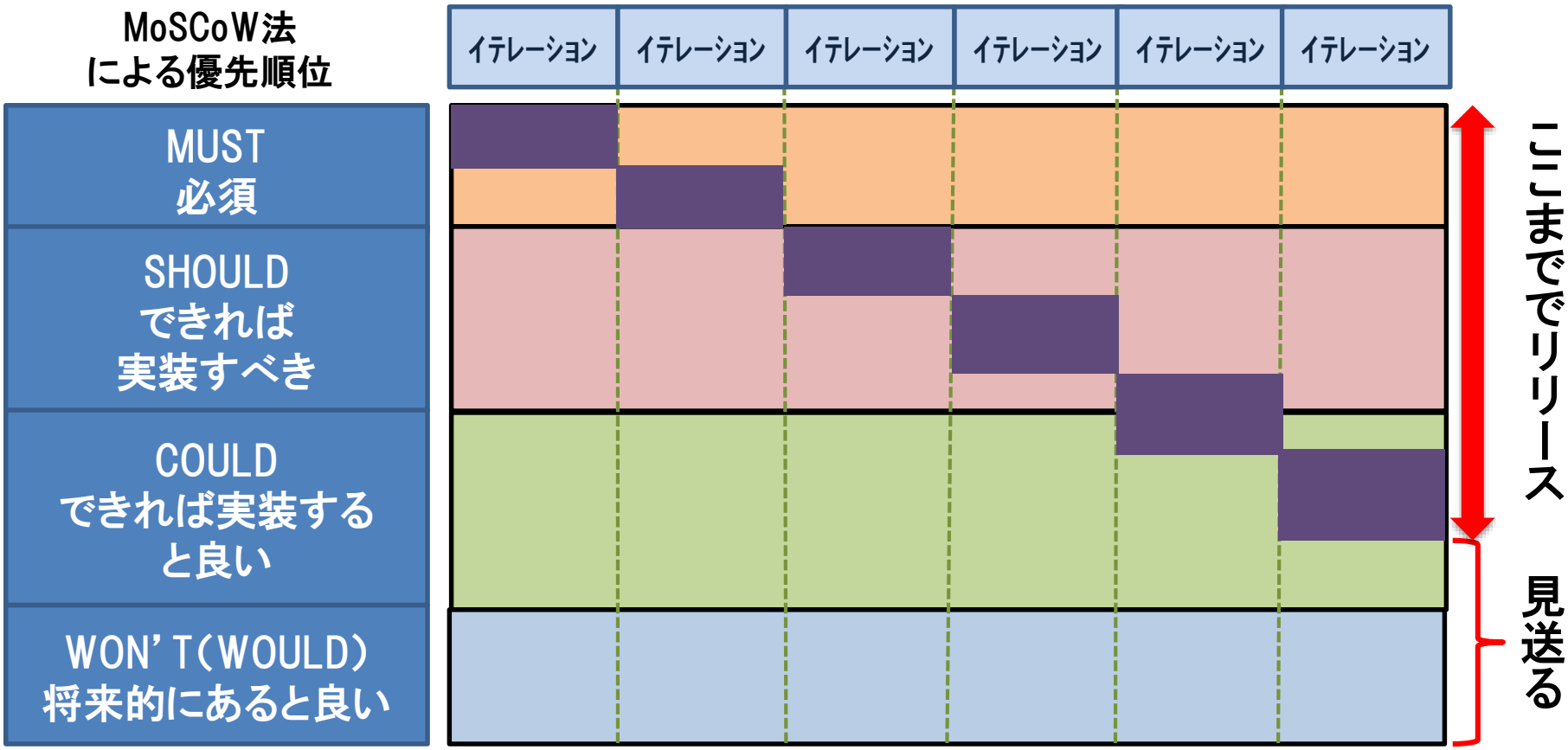
⑤ 展開フェーズ
リリースまでに構築、レビュー、デプロイの3つの主要なアクティビティがあり、最終リリースをもって完了する

⑥ プロジェクト後段階
予想されるビジネス上の利益がどれだけ満たされているかを確認する

2-11. 様々なアジャイル開発手法

HITACHI
Inspire the Next

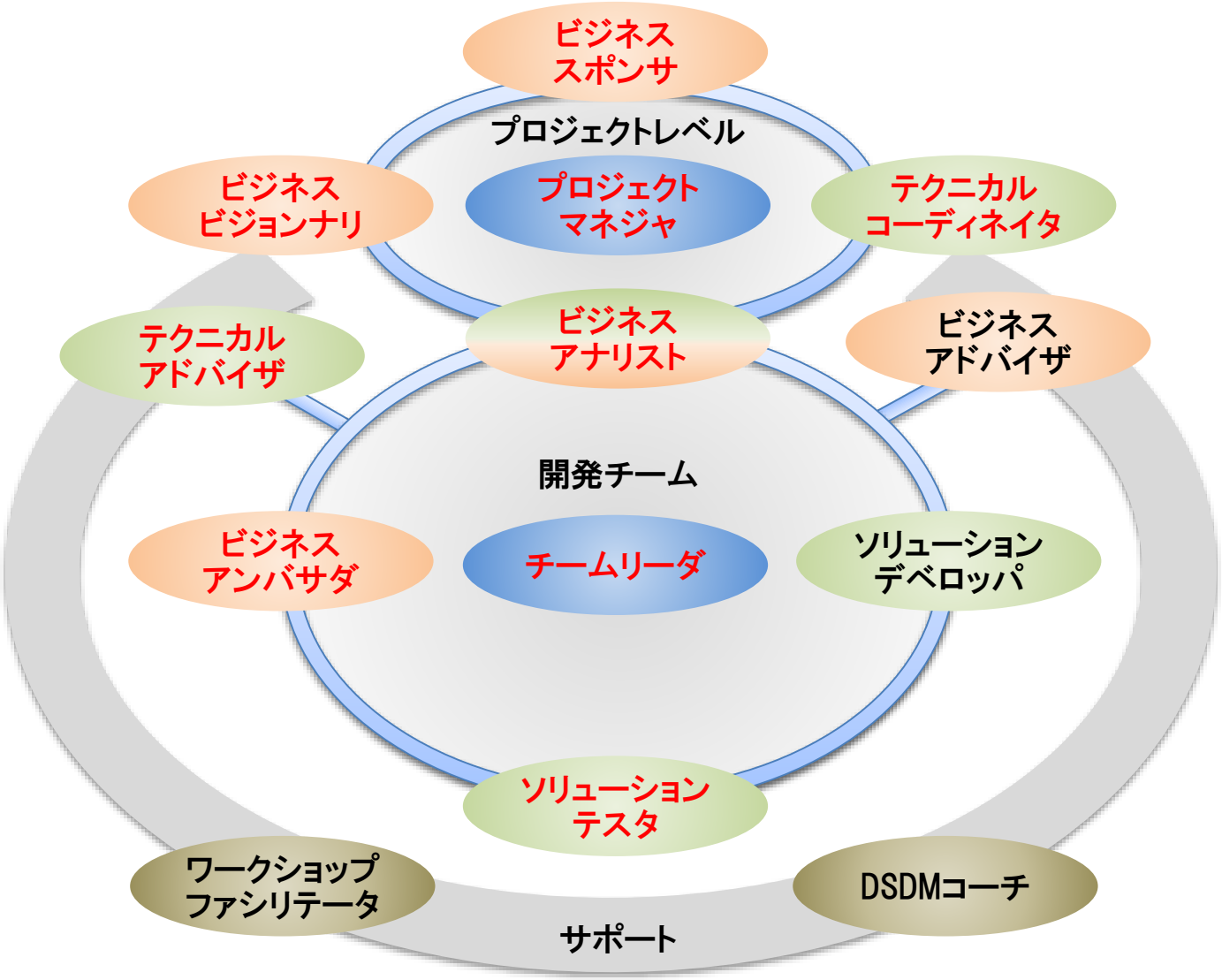
MoSCoW法で、4段階に優先順位を付ける（図は例）



- Backlog List　の優先順位付けに便利な手法
- 委託開発時のスコープ調整や予算超過防止に重要な考え方である

HITACHI
Inspire the Next

DSDMの体制



2-13. 各手法をやってみて感じた特徴

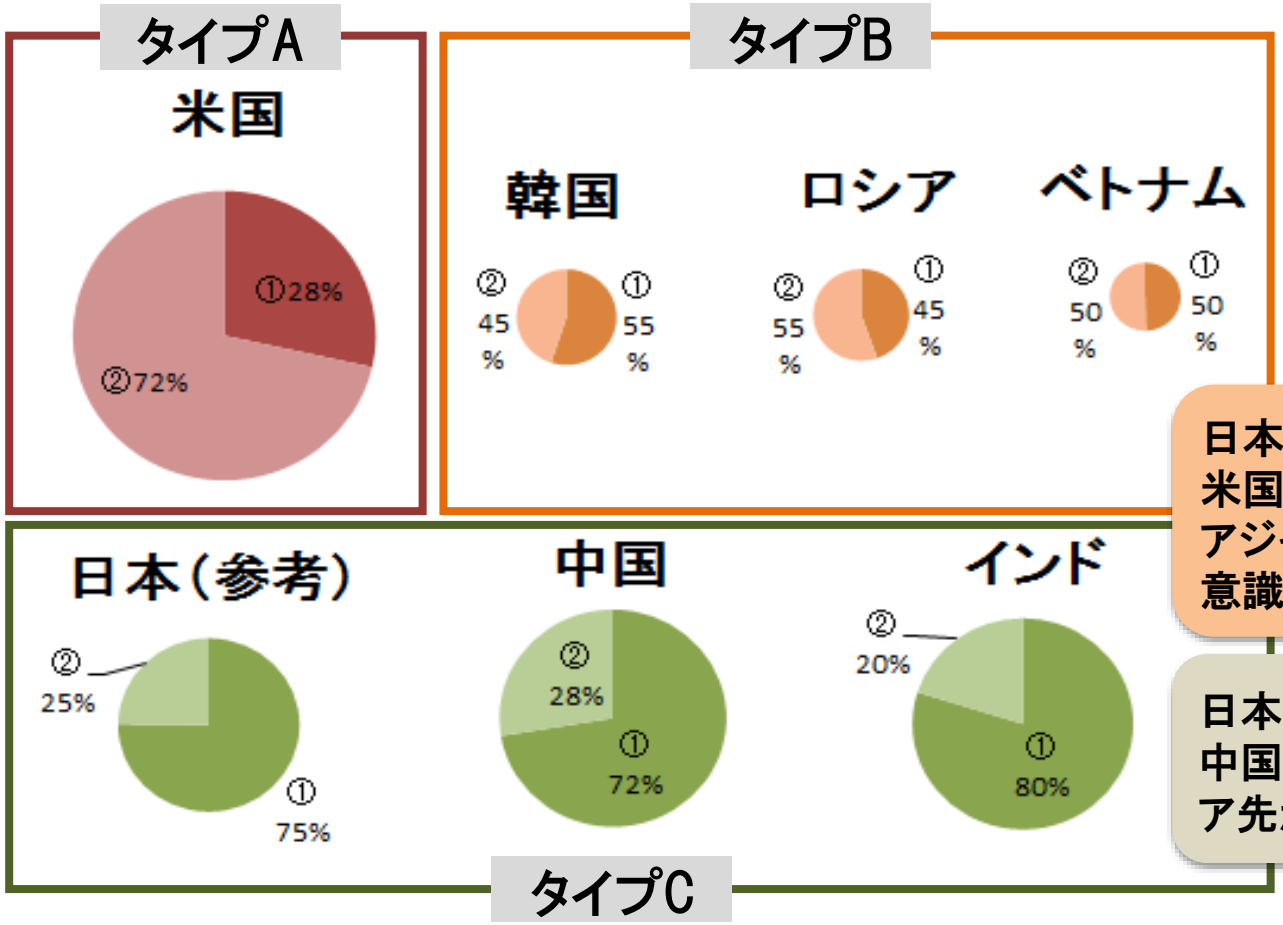
HITACHI
Inspire the Next

手法	感想
XP	・アジャイル技術の基本。小規模な内製向き
Scrum	・XPに自律的な組織化の考えが加わり、手法として理解しやすい ・内製向き
FDD	・製品開発向きと思う ・クラス単位の進捗管理方法は、管理しやすい
Lean	・具体的なプラクティスではないが、考え方として重要 ・委託開発も意識されている
Kanban	・単に見える化するための手法ではなく、プル型の考え方は、委託開発やワークスタイル改革に対して、重要な考え方
DSDM	・委託開発を意識している ・プロジェクト体制やMoSCow法を推奨とする優先付けは参考になる

2-14. 実践にあたっての日本文化との相性

HITACHI
Inspire the Next

日本はITサービス企業に75%、米国はユーザ企業に72%



凡例
① ITサービス企業に勤務
② ユーザ企業に勤務

日本は、極端な委託開発傾向
米国は、極端な内製傾向
アジャイル手法には、委託開発を意識したものが少ない

日本は、大手SIベンダが多い
中国・インドのSIベンダはオフショア先が多い

出典:各国統計資料(米国労働省労働統計局等)
公知情報(NASCOMM、アジア情報化レポート、RUSSOFT、IPA IT人材白書2010)
その他:「ガートナー/Enterprise IT Spending by Vertical Industry Market, Worldwide, 2008-2014, 2Q10 Update」の内部サービスコスト、及び「平均給与単価(後述)」に基づく推計値から(独)情報処理推進機構:IT 人材白書 調査報告書 グローバル化を支えるIT 人材確保・育成施策に関する調査, p.44, 図表3-14, http://www.ipa.go.jp/files/000010609.pdf(2016年8月1日現在)に纏めたもの

2-15. 実践にあたっての日本文化との相性

HITACHI
Inspire the Next

法律は、国によって異なる。アジャイルで委託開発するために考慮が必要

請負契約

- 契約のための見積もりが必要
- 業務完成義務
- 瑕疵担保責任
- 指揮命令系統

準委任契約

- 契約のための見積もりが必要
- 善管注意義務
- 報告義務
- 指揮命令系統

指揮命令系統

・ユーザ企業からの委託先担当者へ直接指揮命令はできない

資産計上

・リリースすれば、資産となる。内製でも同じ。

2-16. 実践にあたっての日本文化との相性

HITACHI
Inspire the Next

予算化

- 年度または期単位に全体予算を申請する（随時予算追加困難）

契約
役割

- 委託開発では、法律上の作業指示系統の制限（全員同席困難）
- 請負契約の場合、成果物責任による要件追加・変更に対する契約が必要（変更受入れが困難）
- 工程別にマルチベンダ開発や製造工程のオフショア発注など縦割りとなるケースが多い（要件定義、設計、開発の同時進行が困難）
- 設計者とプログラマという役割分担のためドキュメントベースでの仕様伝達（ドキュメント省略の限界）

品質

- 「バグは付きもの」では済まされない。日本では、「バグはあってはならない」という感覚。テスト十分性の結果報告が求められる（テスト工程省略が困難）

Q3.教科書アジャイル

- 日経SYSTEMS記事「さらば教科書アジャイル」
  （2015年9月号）
  - これから初めてアジャイルを実践する現場にとって「教科書アジャイル」は役に立たない？
- 真意を教えてください
- 初めてアジャイルに取り組む際の心得とは
- お勧めの本

HITACHI
Inspire the Next

Q3：教科書アジャイル

3-1. 日経SYSTEMS記事「さらば教科書アジャイル」

HITACHI
Inspire the Next

教科書アジャイルがダメと言っているのではない

- 教科書アジャイル(義務教育)を習得して、プロジェクト条件(社会)に
  に合わせて工夫すると良い

基本はScrumを推奨

- 標準的なアジャイル手法のScrumをまず理解し経験する

3-2. 初めてアジャイルに取り組む際の心得と推薦図書

HITACHI
Inspire the Next

提言

- 既存のウォーターフォールを経験してから、アジャイルに取り組む
  何がムダか設計すべきことが分かる。
- 「とりあえずやってみる」では、効果は解らない。プロジェクトの目的と成果を明確にして行う。

推薦図書

- 具体的な書名ではないが、Scrumの手法を解説した書籍
  （まずは、アジャイルの技法を学ぶ）
- アジャイルの考え方を解説した書籍
  （リーンソフトウェア開発、カンバンなど） アジャイルの奥深さに気づく

Q4.ハイブリッドへの想い

- 10年以上ハイブリッドアジャイルを考え続けた、その想いを教えてください

- 心に残った英さんの言葉
  - 異なるものを掛け合わせるからハイブリッド
  - 同じ方向性のものならブレンド

- 異なるものを掛け合わせる → 無限の可能性

HITACHI
Inspire the Next

Q4：ハイブリッドへの想い

4-1. ハイブリッドアジャイルの想い

HITACHI
Inspire the Next

日本の法律に適合した大規模委託開発への対応

初期：アジャイルは、小規模な内製でしか通用しないと感じた

委託開発には、アジャイルのプラクティスのよいところを取り入れて改善できる（継続的インテグレーション、チケット管理、テスト駆動開発など）

海外で大規模開発の事例が出てきた。委託開発の事例もある。

日本の法律に対応した委託開発向けアジャイル手法を研究
海外のアジャイル現場の視察

大規模システム向け日本版アジリティ開発手法開発
（ハイブリッドアジャイルに日本の法律に対応することと独自の開発基盤（ツールなど）を活用した開発効率向上施策を組込んだ

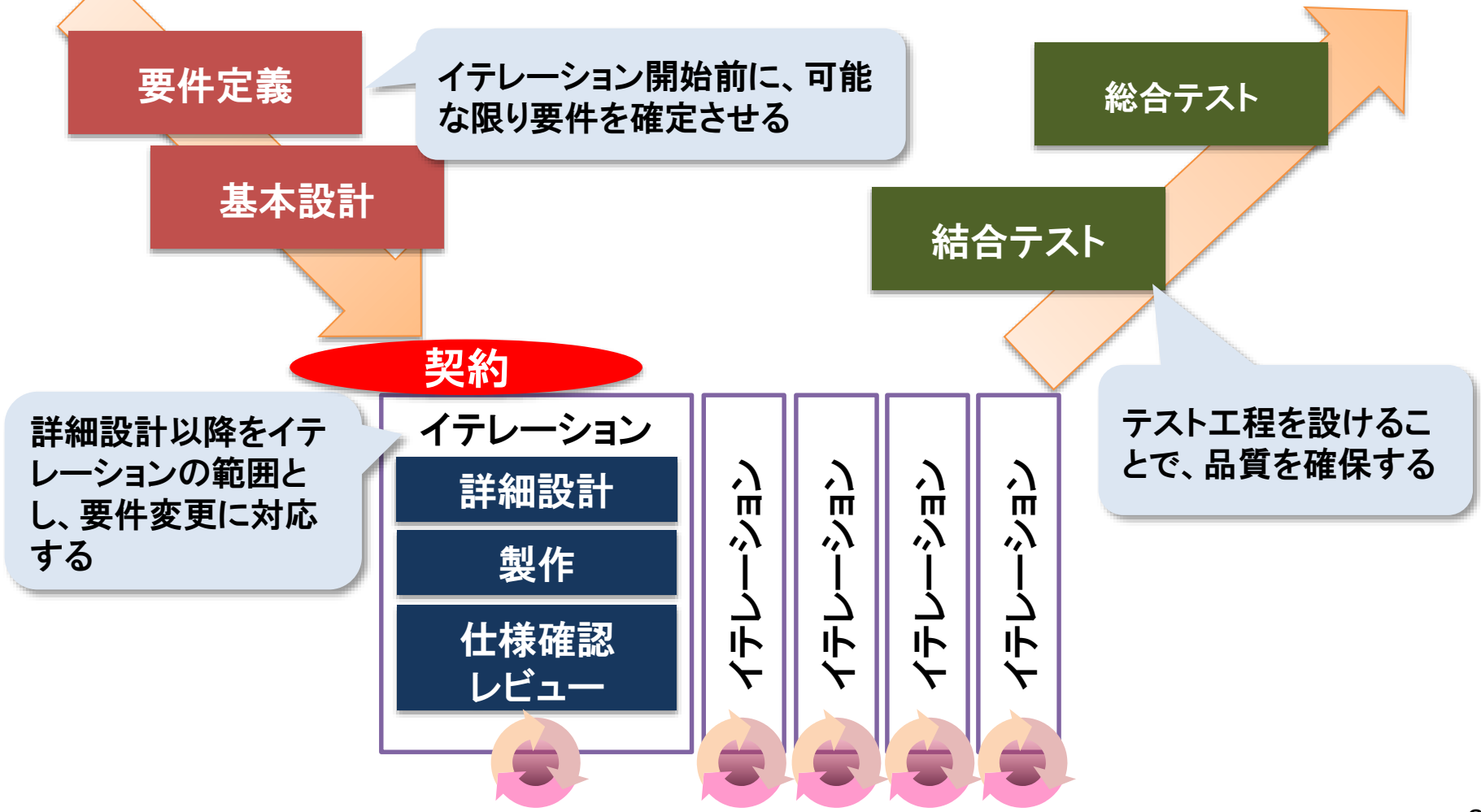
ハイブリッドアジャイル実践（委託開発で変更を受け入れながら行う）

4-2. ハイブリッドアジャイルの想い

HITACHI
Inspire the Next

受託開発へのアジャイル適用を想定したプロセス

■ ウォーターフォールに、アジャイルによる反復開発を部分的に適用する
（詳細設計～単体テストを適用した場合の例）



4-3. ハイブリッドとブレンド

HITACHI
Inspire the Next

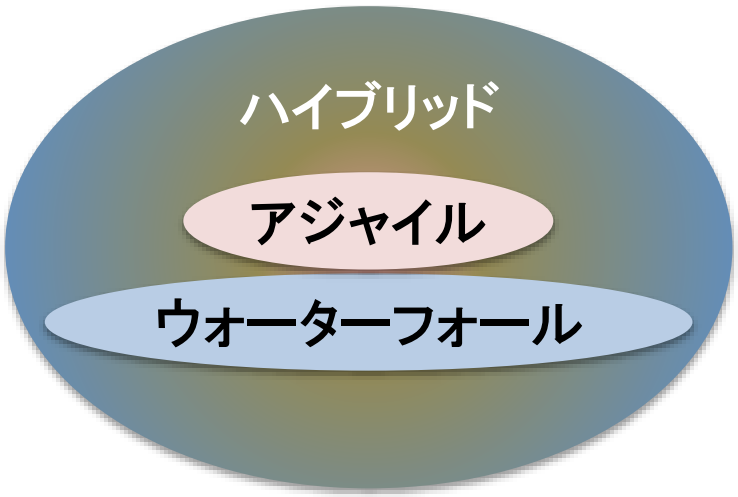
ハイブリッドアジャイルの定義はない

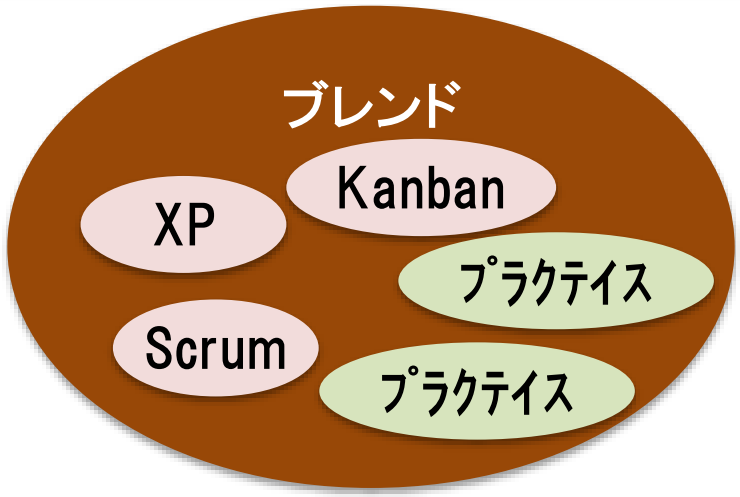
P11のスライドにあるCustom Hybrid(multiple methodologies) は、特に定義されたアジャイル手法ではなく、さまざまなプラクティスを混ぜ合わせたり、他の開発手法と組み合わせたものなどがある。

開発手法をウォーターフォールとアジャイルに分けるのであれば、次のような表現にした方が良いと思う。

・ハイブリッドアジャイル：ウォーターフォール　＋　アジャイル

・ブレンドアジャイル：アジャイル同士やアジャイルプラクティスの組み合わせ





注） ブレンドアジャイルは、今回初めて使用した言葉であり、一般的な用語ではない

4-3. 異なるものを掛け合わせる

HITACHI
Inspire the Next

ハイブリッドアジャイルの主なパターン

Q5.プロセスの組み合わせ方

- ウォーターフォールプロセスでの21世紀での歪み
  - CPU使用料の変化（昔：高価、今：格安）
  - プログラム設計の意義（アセンブラと高級言語の差）

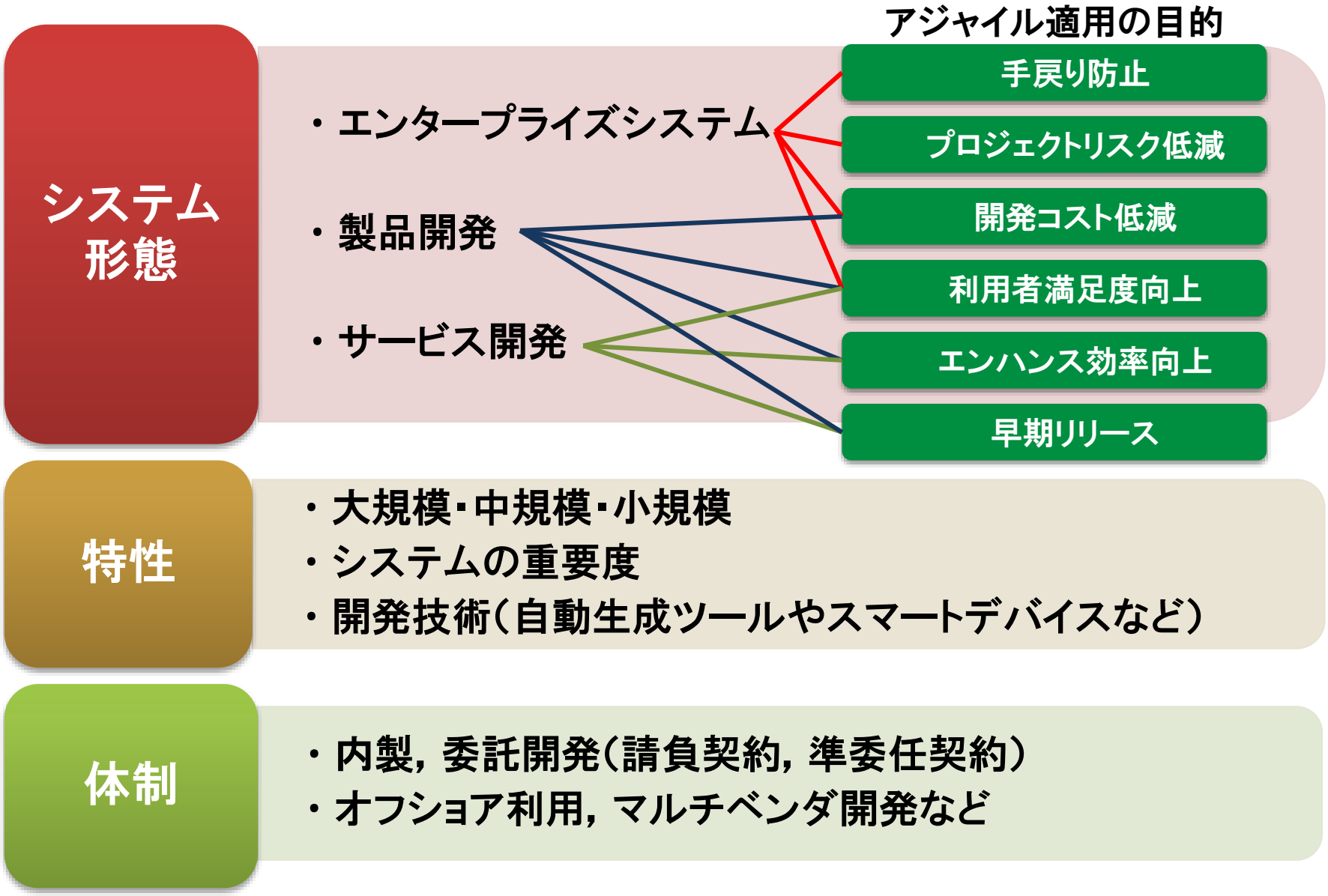
- ウォーターフォールの良いところを活用しつつ
  時代の変遷により生じた弱点を補完する
  - アジャイルプロセスへの期待

HITACHI
Inspire the Next

Q5：プロセスの組み合わせ方

5-1. プロジェクト条件は常に異なる

HITACHI
Inspire the Next



5-2. ハイブリッドアジャイル事例1

HITACHI
Inspire the Next

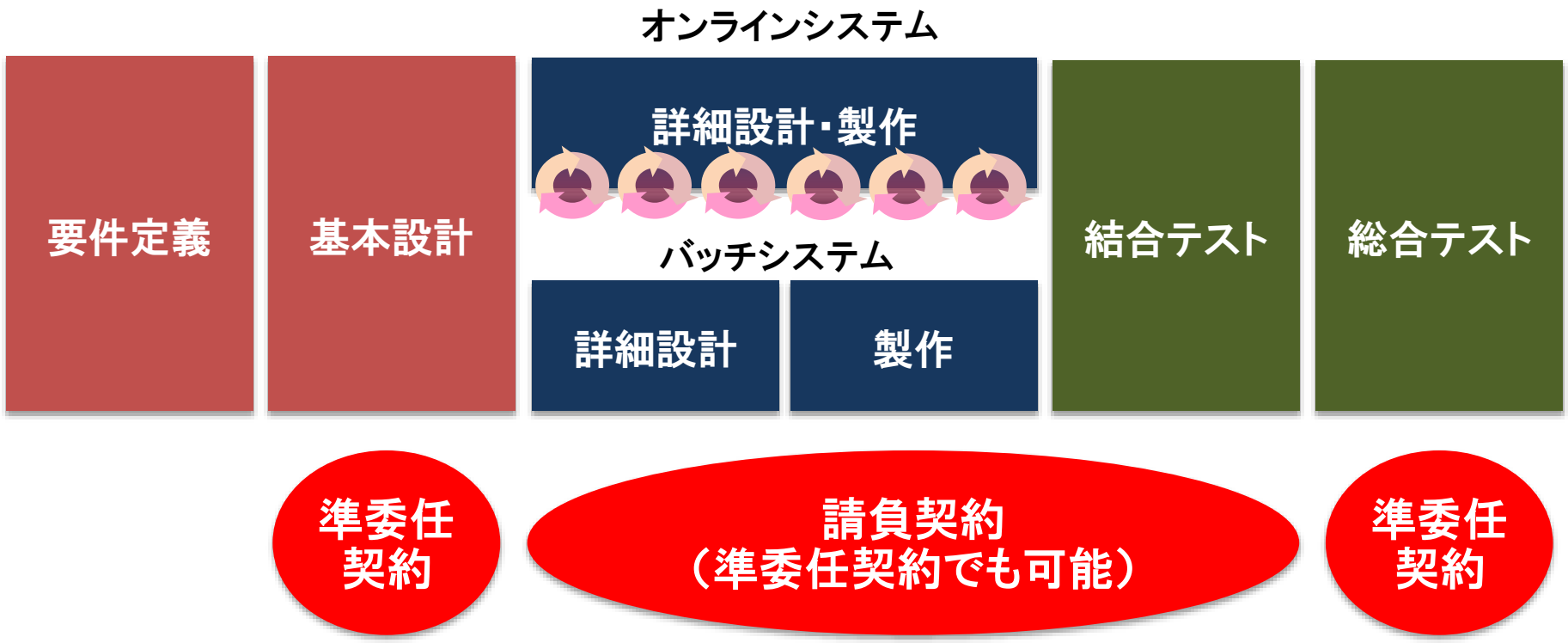
中規模システムの委託開発

項目	説明
契約形態	請負契約(見積もり価格提示、品質保証、納品日確定)
システム形態	Webアプリケーションとバッチプログラム
規模	中規模
目的	作り直しリスク低減:開発中に動作確認でき、修正対応可能にする コスト削減:作業のムダ取り、自動化による生産性向上
ユーザー事情	・他プロジェクトや多くのSIベンダがいるオープンフロア(プロジェクトルームではない) ・ユーザは常時同席はできない(定期的な打合わせで行う) ・プロジェクト状況の定期報告、仕様変更ルール

5-3. ハイブリッドアジャイル事例1

HITACHI
Inspire the Next

オンラインは、詳細設計・製作工程をアジャイル、バッチはウォーターフォール



5-4. ハイブリッドアジャイル事例1

HITACHI
Inspire the Next

採用したプラクティス

ハイブリッドアジャイル

委託開発契約
品質確保

Lean
（ﾌﾟﾛｼﾞｪｸﾄﾏﾈｼﾞｬ,ﾃｽﾀ,目標コスト）

見積もり
（FunctionPoint法,ﾀｽｸ法）

Kanban, チケット管理

要件・課題整理
開発優先順位付け
開発コスト調整

DSDM
（MoSCoW法）

XP
（ﾚﾄﾛｽﾍﾟｸﾃｨﾌﾞ,TDD,ﾍﾟｱﾌﾟﾛ,
ﾘﾌｧｸﾀﾘﾝｸﾞ,ﾐﾗｰ,ｽﾄｰﾘｰの作成,
CI,ﾘﾘｰｽ計画,受入れﾃｽﾄ）

利用者満足度向上
コスト削減
手戻り作業防止

Scrum
（ﾌﾟﾛﾀﾞｸﾄｵｰﾅｰ,ﾊﾞｯｸﾛｸﾞ作成,
ｽﾌﾟﾘﾝﾄ,ｽｸﾗﾑ会議,ｽﾌﾟﾘﾝﾄﾚﾋﾞｭｰ）

5-5. ハイブリッドアジャイル事例1

HITACHI
Inspire the Next

考慮したこと

追加予算確保

- 仕様変更は追加契約と言われてもユーザ企業も簡単に社内で承認されるわけではない
- 変更に対する工学的見積もり手法の適用は、無駄に管理コストを増やす

主管部門は、通常業務で多忙

- 主管部門は、誰もが通常業務に追われている
- 委託開発では、ユーザ企業が直接委託先の担当者へ指示できない

実装を諦める機能もあるという認識を共有する

- 変更を受入れ、期限までに契約コスト内で実装できるとは思ってはいけない

5-6. ハイブリッドアジャイル事例2

HITACHI
Inspire the Next

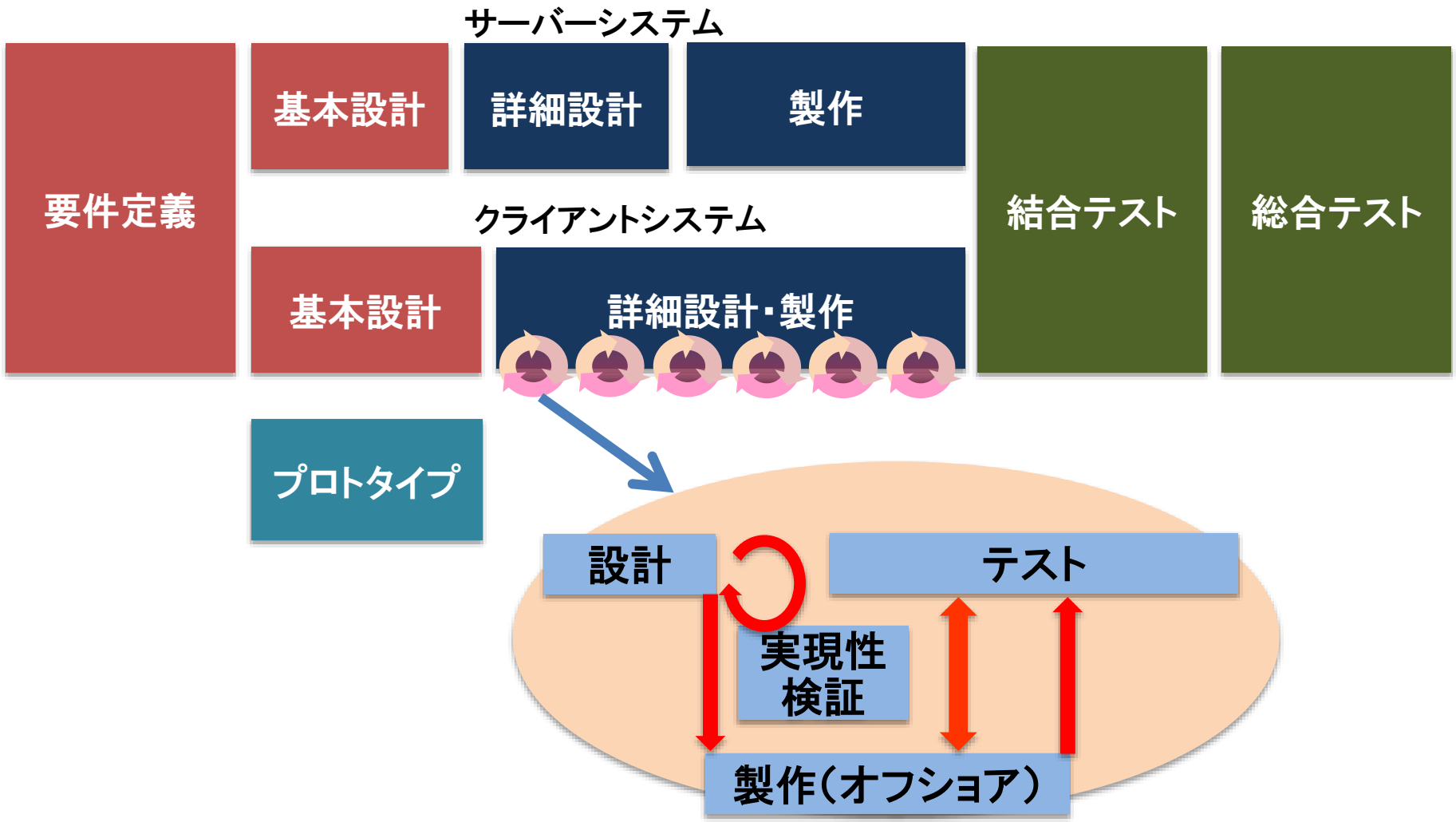
大規模製品開発

項目	説明
契約形態	内製による製品開発(オフショア活用)
システム形態	タブレットアプリとサーバシステム
規模	大規模
目的	新製品開発(市場確保しながらの開発) ・開発中から営業活動が始まる(展示博覧会出展、デモなど) ・プロトタイプ開発 ・実現性を確認しながら開発 コスト削減 ・オフショア活用 ・作業のムダ取り
プロジェクト事情	マルチステークホルダー ・仕様確定するまでの調整が難しい 企業ルール ・事業化認可プロセス、プロジェクト状況報告

5-7. ハイブリッドアジャイル事例2

HITACHI
Inspire the Next

オンラインは、詳細設計・製作工程をアジャイル、バッチはウォーターフォール



5-8. ハイブリッドアジャイル事例2

HITACHI
Inspire the Next

採用したプラクティス

- Scrum
  （ﾌﾟﾛﾀﾞｸﾄｵｰﾅｰ,ﾊﾞｯｸﾛｸﾞ作成,
  ｽｸﾗﾑﾏｽﾀ,ｽﾌﾟﾘﾝﾄ,ｽｸﾗﾑ会議,
  ｽﾌﾟﾘﾝﾄﾚﾋﾞｭｰ）
- オフショア活用
- CI（継続的インテグレーション）

開発期間短縮
コスト削減

- MoSCoW法

要件・課題整理
開発優先順位付け
開発コスト調整

- プロトタイプ

実現性確認
マーケティング

- ハイブリッドアジャイル

予算見積もり
品質確保

5-9. ハイブリッドアジャイル事例2

HITACHI
Inspire the Next

考慮したこと

混在する開発手法

- プロトタイプ：営業活動や実現性検証
- ウォーターフォール：サーバ機能やバッチの開発
- アジャイル：複雑な機能を持つクライアントアプリ開発
- 工程別、アプリ別のハイブリッドアジャイル

スクラムマスタの不足

- 複数のスクラムチームの体制で行った場合、スクラムマスタを担当できる人材は不足したため兼任させた

ステークホルダーの調整

- 意見が纏まらず仕様確定に苦労する

5-10. ハイブリッドアジャイル事例3

HITACHI
Inspire the Next

小規模社内システム

項目	説明
契約形態	内製による社内システム開発
システム形態	Webアプリケーションとバッチプログラム
規模	小規模
目的	早期リリース ・必要な機能から段階的にリリース ・リリース間隔は3～6ヵ月 ユーザ満足度向上 ・人間中心設計、テスティングラボで検証 コスト削減 ・作業のムダ取り ・自動化 ワークスタイル改革 ・担当者は週2日自宅勤務 ・分散拠点のメンバ1名も時々参加
プロジェクト事情	企業ルール ・予算確保、資産計上

5-11. ハイブリッドアジャイル事例3

HITACHI
Inspire the Next

プロトタイプでユーザビリティを確保、テスト観点別にテスト実施

5-12. ハイブリッドアジャイル事例3

HITACHI
Inspire the Next

採用したプラクティス（全員同席は古い、ワークスタイルも改革）

- リーンスタートアップ
- DevOps

リリーススピードアップ

- CI（継続的インテグレーション）
- Kanban, チケット管理
- MoSCoW法

要件・課題整理
開発優先順位付け
開発コスト調整

- ワークスタイル改革
（タイム&ロケーション　フリー）

- Scrum
（ﾌﾟﾛﾀﾞｸﾄｵｰﾅｰ,ﾊﾞｯｸﾛｸﾞ作成,
ｽﾌﾟﾘﾝﾄ,ｽｸﾗﾑ会議,ｽﾌﾟﾘﾝﾄﾚﾋﾞｭｰ）

- 人間中心設計
- プロトタイプ

利用者満足度向上

- デザイナーによる画面デザイン
- テスティングラボ

5-13. ハイブリッドアジャイル事例3

HITACHI
Inspire the Next

考慮したこと

予算確保

- 予算の考え方にKanban （予算内でできたところまでをリリース）
- 予算と実績に差が出ない

開発標準（内部統制）

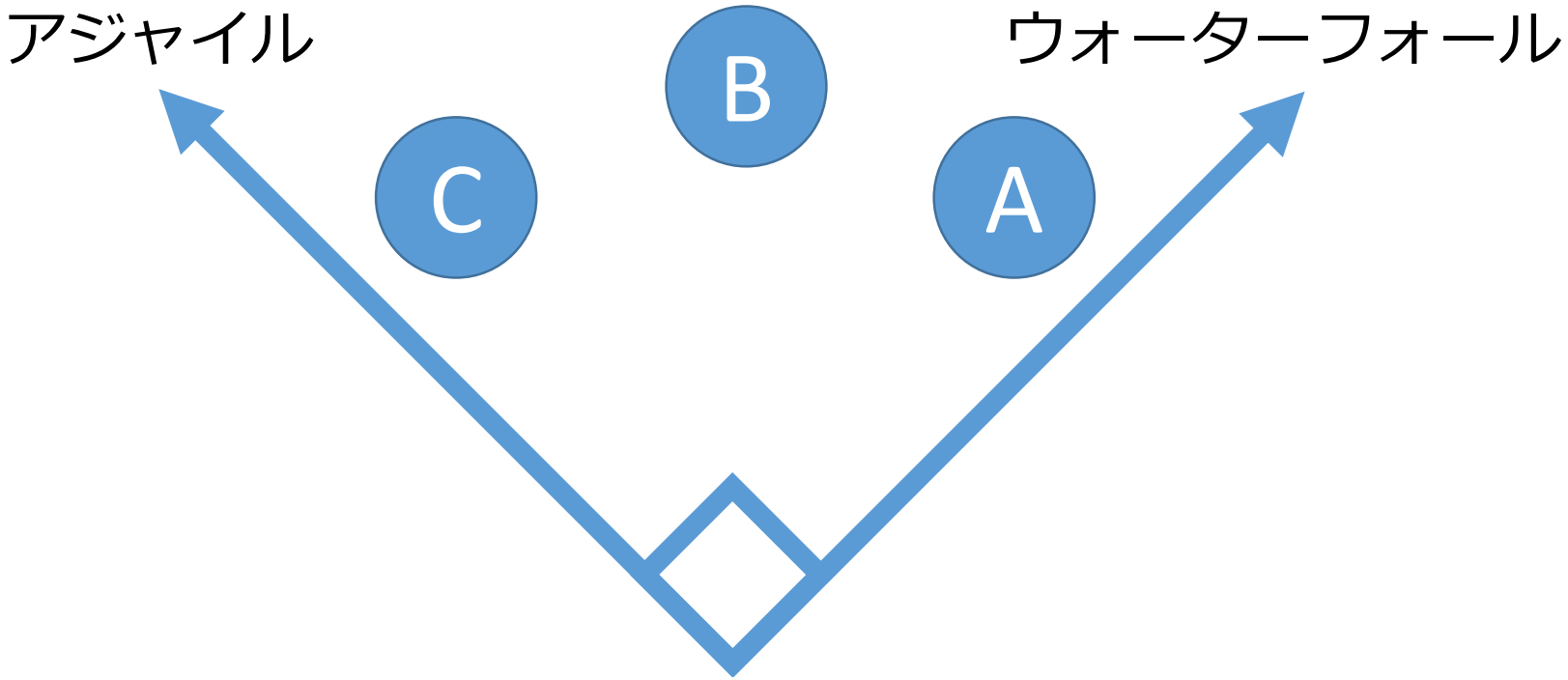
- ウォーターフォール型の標準をアジャイルにテーラリング
- 設計書は、ストーリーカード，操作マニュアル，FAQで代用
- UT，CT，STの観点でテストケースを作成し、テスト実施

資産計上

- 資産として登録はフェーズ単位（数ヵ月）
- アジャイルのように数週間単位で資産計上をするようでは、財務部門が堪らない

Q5.プロセスの組み合わせ方

- ウォーターフォールとアジャイルは、
  どちらか一方しか選択できないものではない
- プロジェクトの課題に応じて、双方を様々に
  組み合わせることでプロジェクトを成功させたい



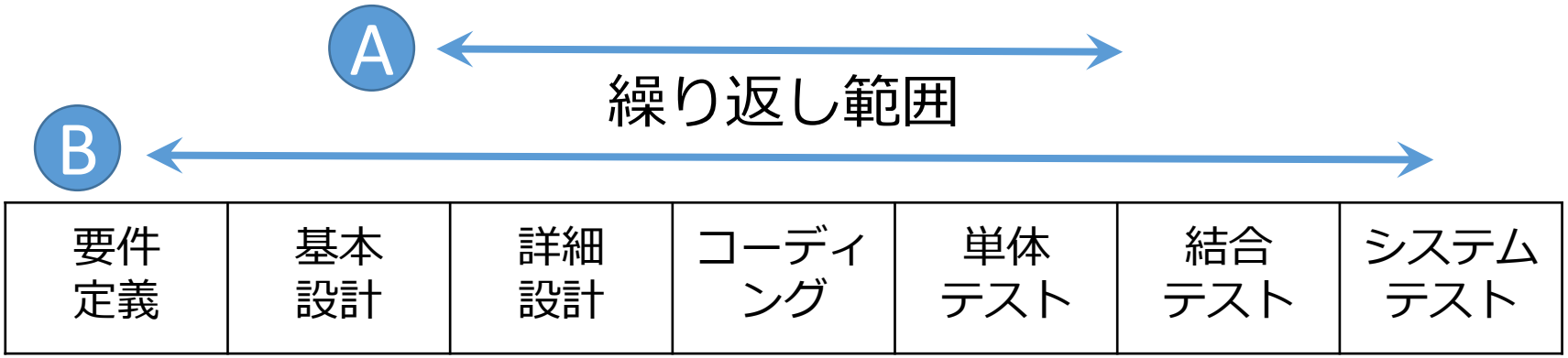
Q5.組み合わせ方

A 実現時の異常を検知することを目的

- 要件と基本設計は最初に決める
- 実装工程での問題を早く見つける

B 要件の一部調整も許容することを目的

- 主要な要件は最初に決め、詳細な要件は調整対象
- 主に概要設計～結合テストの繰り返し

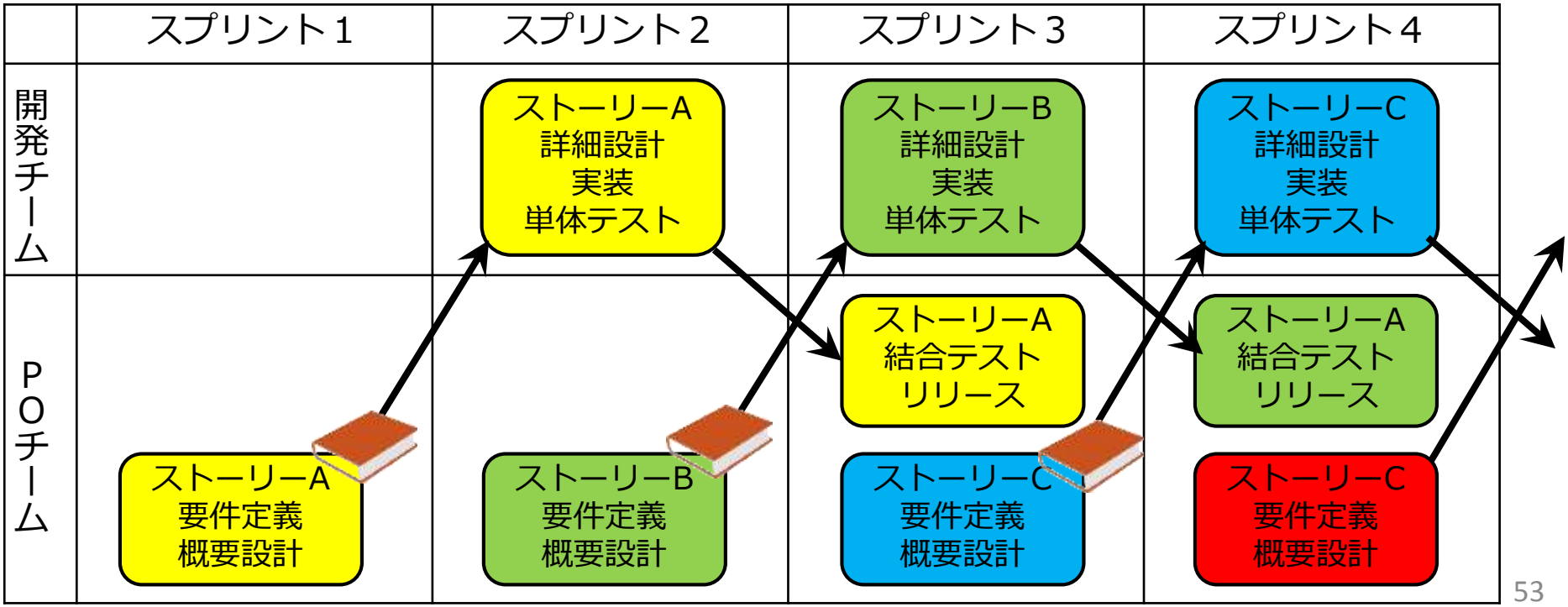


Q5.組み合わせ方

C スプリントの中をミニウォーターフォール

- 開発チームはウォーターフォール工程で開発する
- POチームは設計書を作成して渡す 
- 手待ちが出ないように複数ストーリーを並行させる

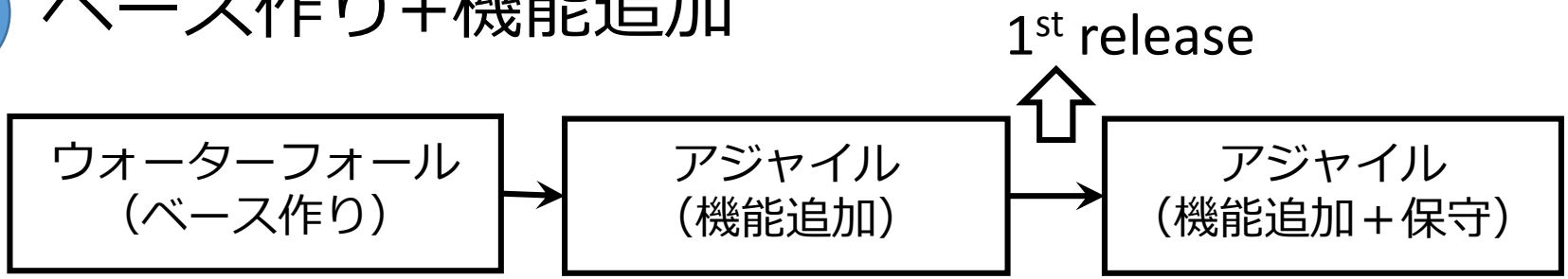
スプリントをまたいだストーリーの動き



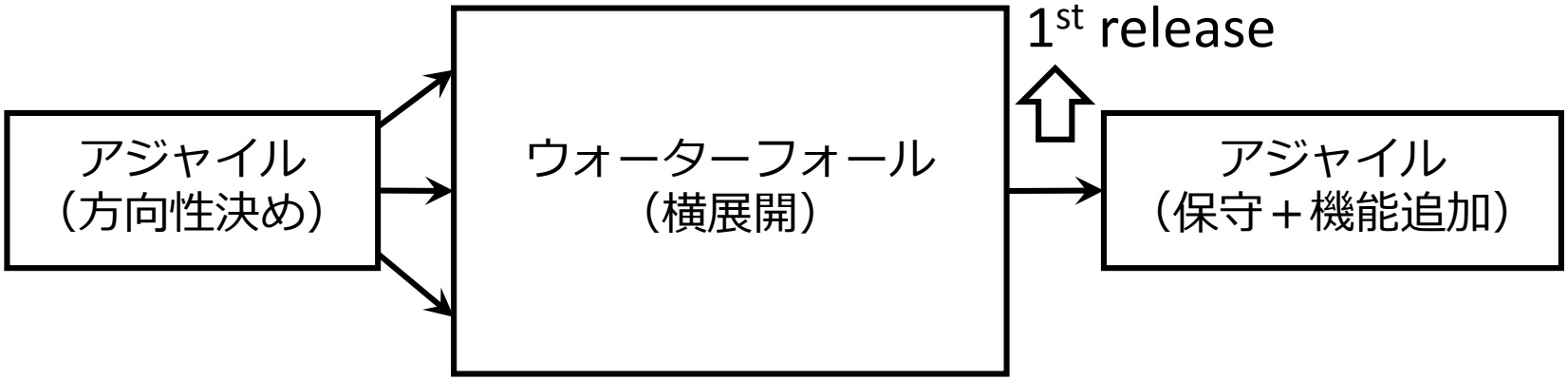
Q5.組み合わせ方

- ハイブリッドカーを参考に
  - エンジンとモーターを状況に応じて切り換える

D ベース作り＋機能追加



E 方向性決め＋横展開＋保守（機能改善）



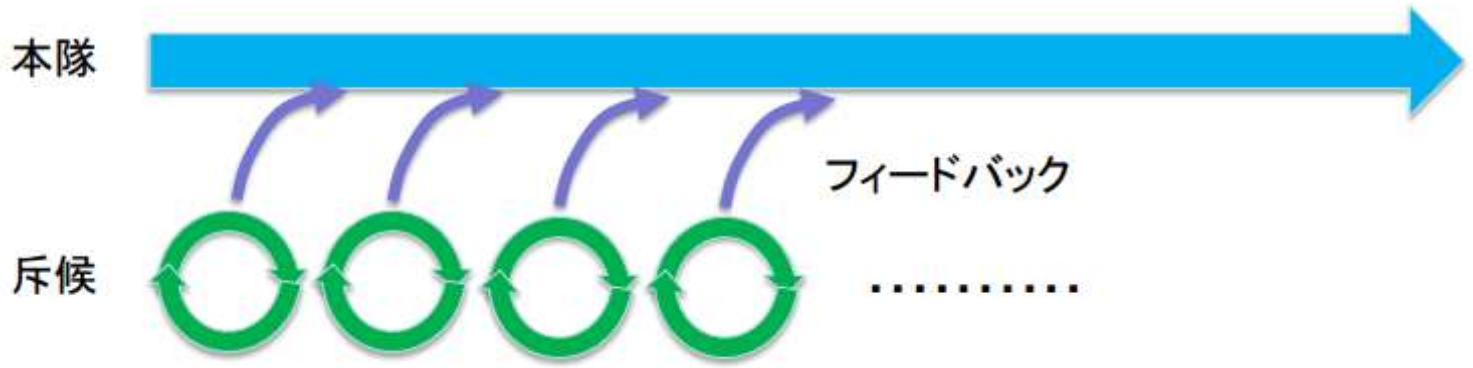
Q5.組み合わせ方

F ウォーターフォールとアジャイルを並行させる

→　例：斥候としてのアジャイル開発（細谷氏）

MITSUBISHI
Changes for the Better

(1)斥候としてのアジャイル開発



斥候としてアジャイル開発が先行し、本隊に対して継続的なフィードバックを与える。

引用：「斥候としてのアジャイルプロセス活用の提案（p.8）」細谷 泰夫 氏（三菱電機）SPI Japan 2012

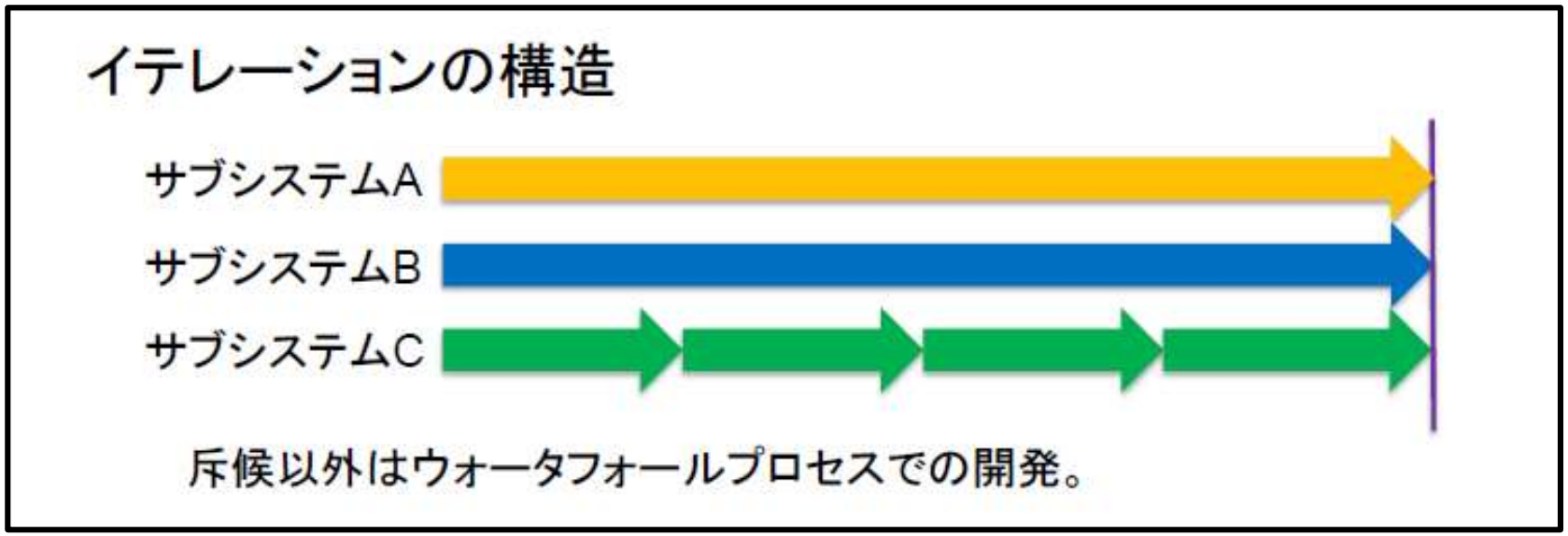
Q5.組み合わせ方

【課題】

大規模な開発はリスクが高く、初期段階で精度の高い仕様を決めたり、仕様の整合性を確保するのが難しい

→　大規模開発にはアジャイル開発をする動機がある

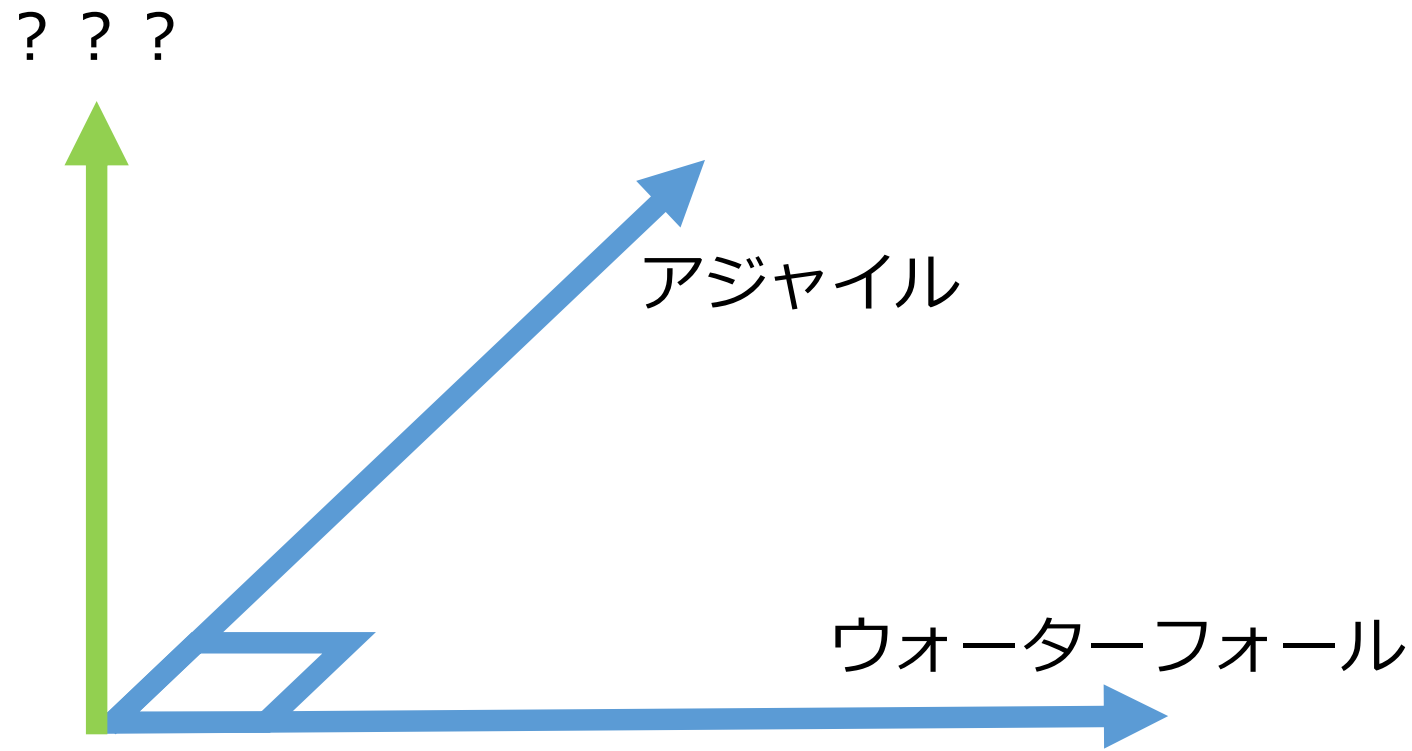
→　特定部分にアジャイル開発を採用する
「斥候としてのアジャイル開発」を実施



引用：「斥候としてのアジャイルプロセス活用の提案（p.5,6,12）」細谷 泰夫 氏（三菱電機）SPI Japan 2012

Q5.組み合わせ方

- 第３軸を追加してみる



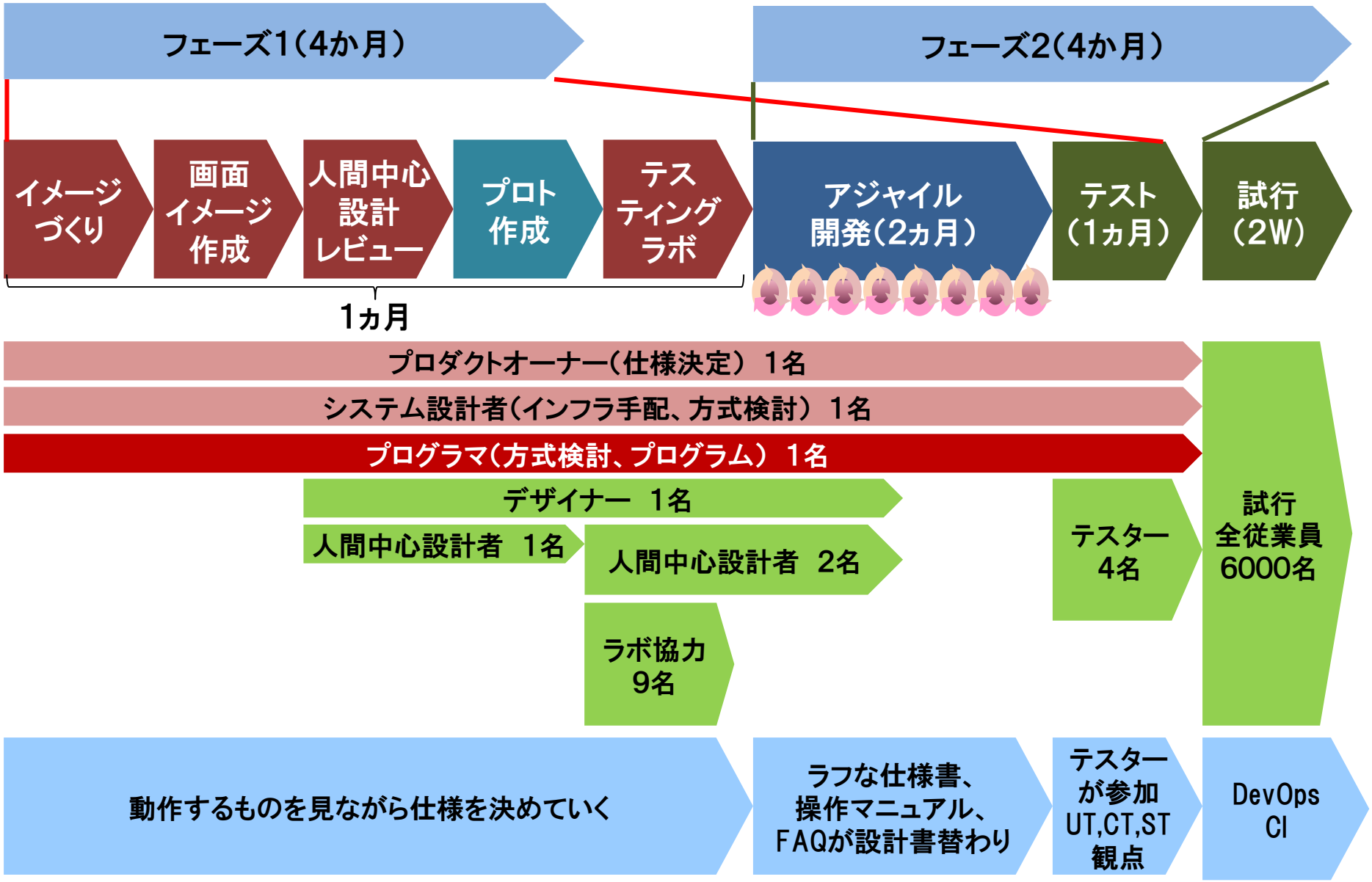
Q6.そしてDevOpsへ

- ハイブリッドアジャイルからDevOpsへ
- 社内実践の感想を教えてください

HITACHI
Inspire the Next

Q6：そしてDevOpsへ

6-1. ハイブリッドアジャイルからDevOpsへ



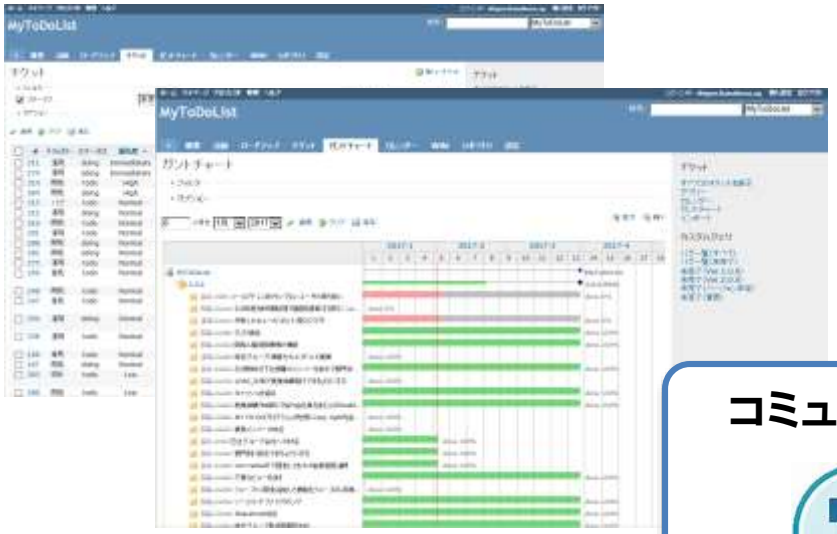
6-2. ハイブリッドアジャイルからDevOpsへ

リリースサイクル	3～6ヵ月 あまりに頻繁にリリースすると資産計上作業がたいへん
スケール	フェーズ1は最低限必要な機能のみ、フェーズ2以降順次機能拡張
人材	開発者と運用者は同一人物

6-3. ハイブリッドアジャイルからDevOpsへ

HITACHI
Inspire the Next

チケット管理(タスク、仕様、バグなど)



REDMINE

開発言語・フレームワーク

Ruby

RAILS

Bootstrap

コミュニケーション

実行環境

Linux

MySQL

CI環境

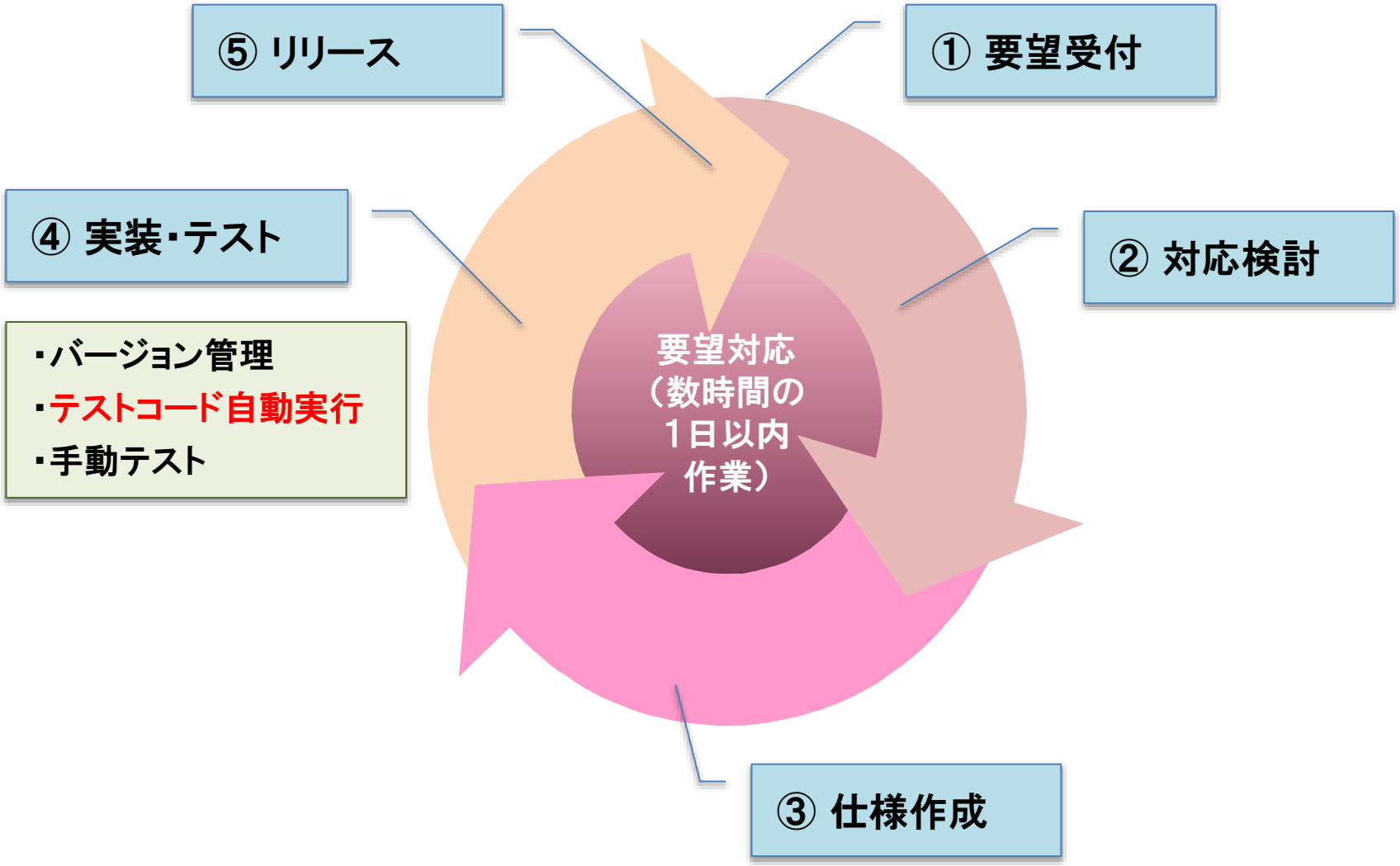
GIT

Jenkins

6-4. ハイブリッドアジャイルからDevOpsへ

HITACHI
Inspire the Next

設計・開発・運用が同じチームのため決断が速い、CI環境よる作業効率化



6-5. 社内実践の感想

HITACHI
Inspire the Next

利用者に驚きを与え、不満が1日で感謝に変わった

10月某日

10:08
管理機能に□ □ □ □できるよう改善してください。

11:31
ご要望は、□ □ □ □ □ □ □ □ □ □ で、よろしいでしょうか？
□ □ □ □ □ の仕様としたいと思います

12:52
早々の回答ありがとうございます。
□ □ □ □ □ で結構です。

17:34
□ □ □ □ □ようにいたしました。ご確認よろしくお願いいたします。

17:44
早い！！ありがとうございます。多謝です。感激です。

2月某日

9:19
画面遷移先を変更できませんか？

10:06
さまざまなご意見がありますので、総合的に判断します。

10:34
今の仕様ですと管理者側からみて不便です。

17:11
検討の結果、ご意見を反映させていただきます。

17:44
はやい対応で驚きました。ありがとうございます。

Q7.会場のみなさんへ

- 最後に一言お願いします。

HITACHI
Inspire the Next

Q7：会場のみなさんへ

7-1. 会場のみなさんへ

HITACHI
Inspire the Next

1. 開発手法

- Scrumは、基本知識として習得することを推奨
- プロジェクトに合わせた開発手法を選択する
- さらに改善すること(オリジナルプラクティス)を考えよう
- ハイブリッドアジャイルは、委託開発に対応するために採用した
  目的は、変更受け入れ／リスク対策／利用者満足度向上のため
- アジャイルの具体的な手法だけではなく、本質の理解が重要

2. 日本の事情・企業の事情

- 外国生まれのアジャイル手法には、日本の事情は考慮されていない
- 日本では、委託が主流。アジャイルで委託契約の場合は、法遵守に注意する
- 企業のルールがアジャイル導入の支障となる場合があるが、
  これを乗り越えなければ、定着はできない
- Kanbanの考え方が、IT技術者のワークスタイル改革の鍵
  無理な残業をしてまで、全てを完成させなくてもよい。健康的な業務時間内にできるところまでにするという考えを発注者と受注者間で理解が必要。
  できれば完成させる、できなければ見送る機能を共有しておくことが必要。

おわり

ご参加ありがとうございました！