FUJITSU
shaping tomorrow with you

規律正しさと柔軟さの両立、学び続けるチーム、を実現するためには

～プラクティスは適用して満足してちゃ「ダメよぉ～、ダメダメ」～

富士通株式会社　SI技術本部　システム技術統括部
アジャイル実践センター
山下　勝
和田憲明

@JASPIC 第9回 SPI トワイライトフォーラム2015/2/25

目次

FUJITSU

- 1.アジャイル実践センターの活動紹介
- 2.ワールド・カフェ
- 3.『高品質ソフトウェア開発チームアセスメントシート』とは？
- 4. 超ざっくり　アジャイル開発とは
- 5. 事例概要
- 6. アセスメントシートを事例で評価してみました
- 7. 自律的に機能するチームの特徴とふるまい
- 8. プロセス改善におけるアジャイル開発からのヒント
- 9. 最後に

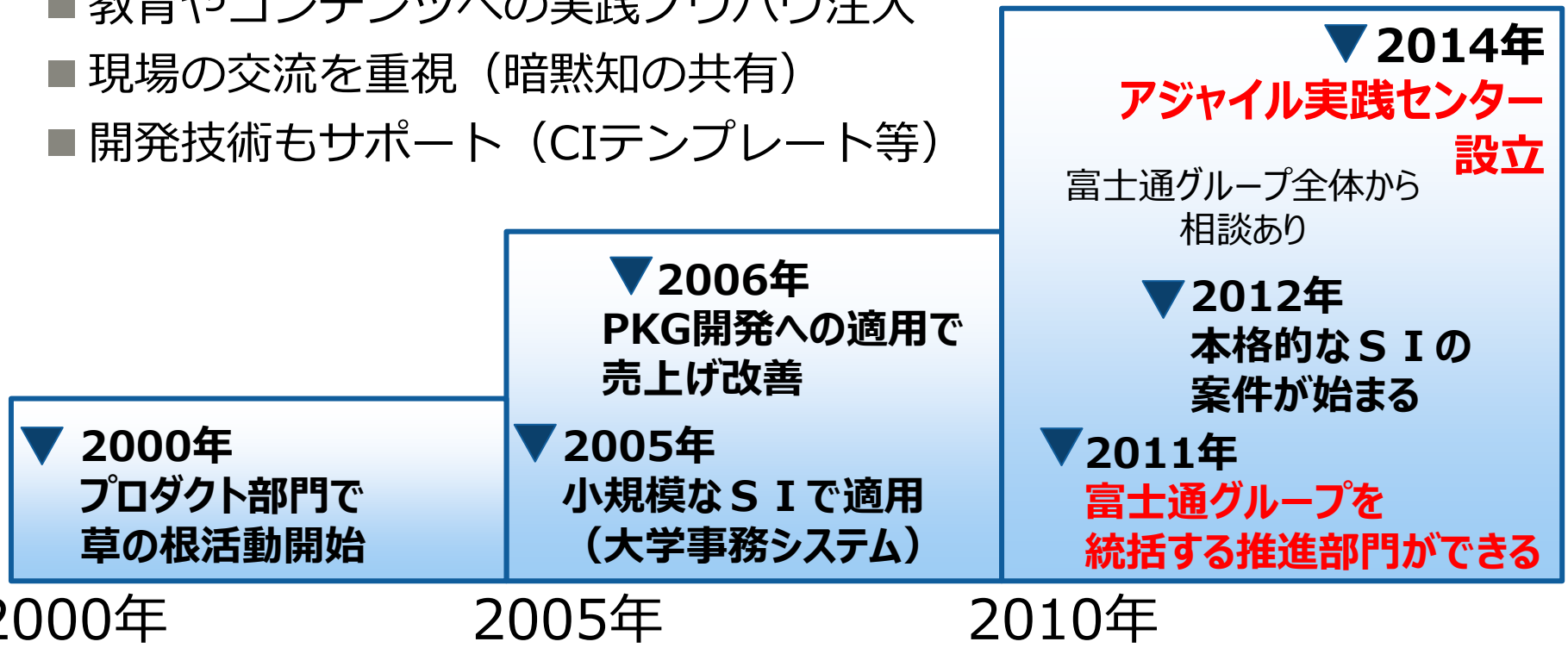
FUJITSU

１．アジャイル実践センターの活動紹介

富士通におけるアジャイル開発の取り組み

FUJITSU

- 長年の現場実践ノウハウを有効活用し、普及展開へ
  - 業種、地域、分野（ＳＩ、組込み、など）を越えた活動
  - 教育やコンテンツへの実践ノウハウ注入
  - 現場の交流を重視（暗黙知の共有）
  - 開発技術もサポート（CIテンプレート等）



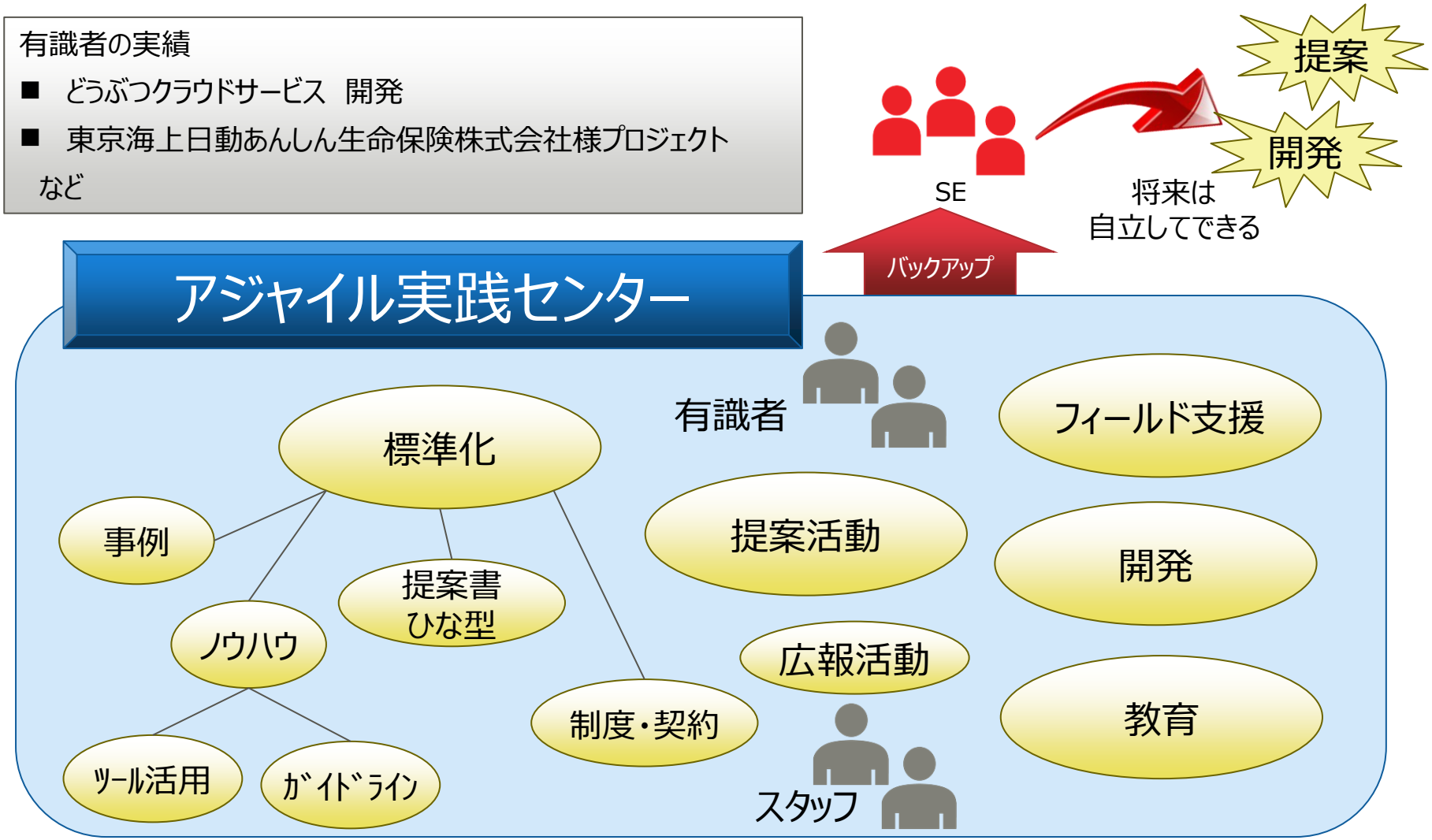
アジャイル実践センター

FUJITSU

■ アジャイル開発の有識者が、プロジェクトをバックアップします。

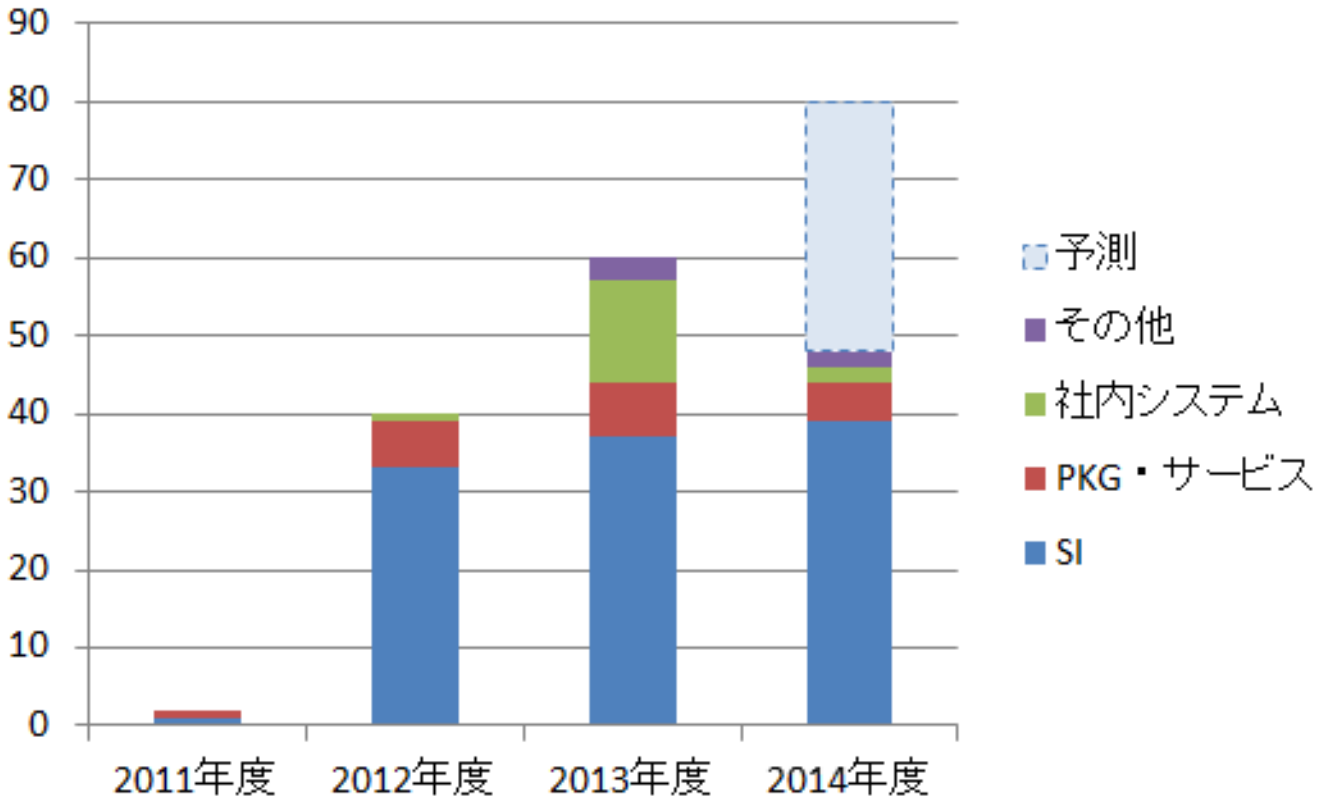
有識者の実績
- ■ どうぶつクラウドサービス　開発
- ■ 東京海上日動あんしん生命保険株式会社様プロジェクト

など



アジャイル開発への関心の高まり

- 様々な企業でアジャイル開発の試行・実践が進んでいる
- 推進部門への問い合わせが年々増加している
  - お客様から（情報提供依頼）
  - 社内現場から（相談・支援依頼）



アジャイル開発に関する社外発表

FUJITSU

- 前頁の事例を中心に、社外イベントで積極的に事例等を発表
  - Agile Japan 2009, 2010, 2011, 2012, 2013, 2015
    - モチベーション駆動開発～パッケージ開発の現場から～（2009）
  - SPI Japan 2012, 2013, 2014
    - パッケージ開発プロセス改善による品質向上と生産性向上 品質データからのアジャイルに関する考察（2013）
  - IPA/SECセミナー 2012, 2013, 2014
    - IT新市場開拓プロジェクトにおけるアジャイル開発（2012年）
    - 東京海上日動あんしん生命システム構築におけるエンタープライズアジャイル事例紹介（2014年）
  - JaSST Tokyo 2005, 2006, 2013, 2014
    - テスティングフレームワーク「JUnit」が開発現場にもたらした８つの効果と３つの戦い（2005）
  - PROMAC 2013
  - SPES 2014
    - スクラムマスター奮闘記 waterfallな若手SEのアジャイルチームビルディング（2014）
  - Developers Summit 2008, 2010
  - 日本OSS推進フォーラム2015
    - 浪江町タブレットを利用したきずな再生・強化事業への取組み

タイトルをいくつか
ご紹介します

※日本のITベンダーの中では最も多いと思われる

FUJITSU

2．ワールド・カフェ

- 少人数のグループ（４，５人）に分かれます
- 自己紹介をしましょう
- グループで気軽に次の話題について会話してみてください
  - 話題：「あなたの企業のアジャイル事情」

FUJITSU

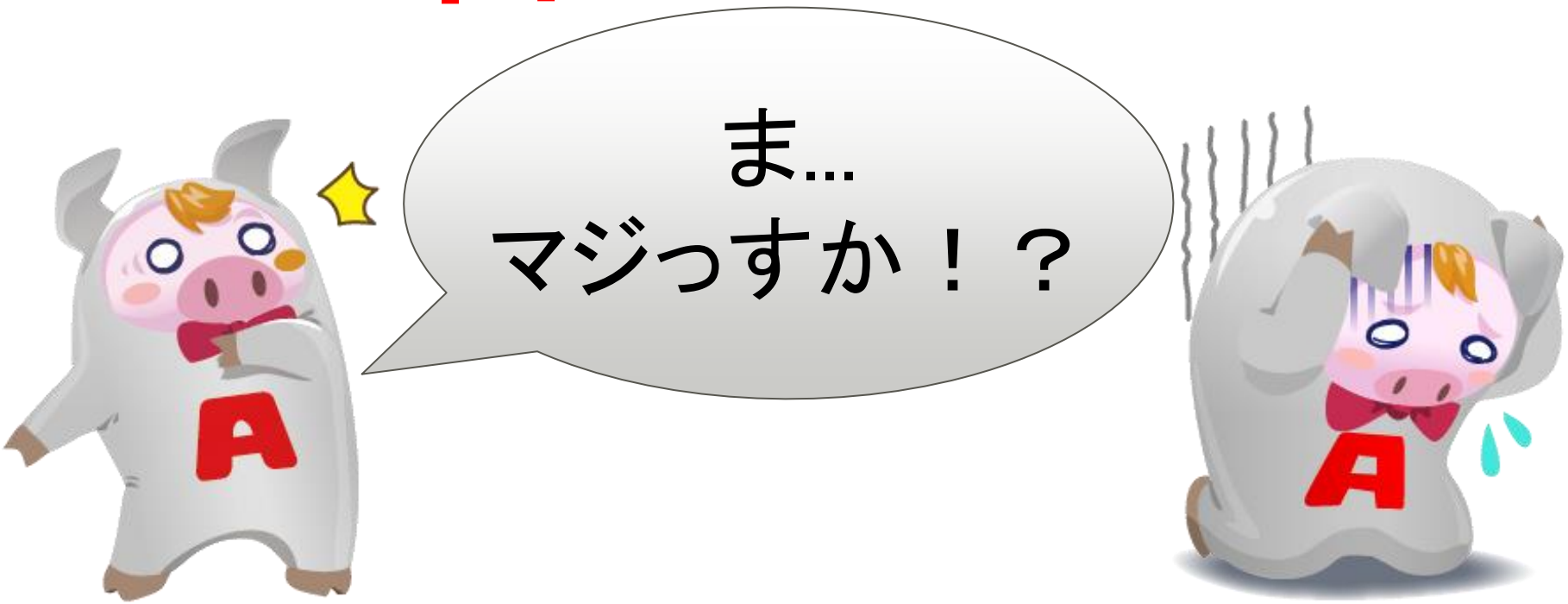
3．『高品質ソフトウェア開発チームアセスメントシート』とは？

事の発端は・・・

FUJITSU

- JASPIC分科会における研究成果である「高品質ソフトウェア開発チームアセスメントシート」について、参加企業の研究員で調査したところ全25項目のうち、20項目以上をマークしたのが、たったの・

・・・ **2名** という衝撃の事実が発覚！



高品質ソフトウェア開発チームアセスメントシート 1/3

FUJITSU

リーダー

- リーダーの仕事は、自分で作業するのではなく、作業する人をリードすることであると知っている
- リーダーは明確なチームの最終的なゴールと短期的なゴールを持っている
- リーダーはチームメンバの手本となる行動をしている
- リーダーはチームが成功することを信じている
- リーダーはチームメンバーの成功を認めている

チームメンバー

- メンバーはチームへの帰属意識を持っている
- メンバーは共通のチームゴールにコミットしている
- メンバーは開発プロセスと計画を自分のものであると感じている
- メンバーは計画を作るスキルを持っている
- メンバーは計画に従う規律を持っている

2014年度　JASPIC
分科会での研究報告より

高品質ソフトウェア開発チームアセスメントシート 2/3

FUJITSU

動機付け

- リーダー・メンバーは自分の自由意思で計画にコミットしている
- リーダー・メンバーはコミットする前に交渉ができる
- リーダー・メンバーが何にコミットしたかが明確になっている
- リーダー・メンバーがコミットメントを果たすための計画を自分で作っている
- コミットメントが確実に守られているかフォローされ続けている

チーム構築

- 計画に基づいてリソース（人、環境、ツール）が割り当てられている
- 開発において必要とされるスキルが明確になっている
- スキルアップのためのトレーニングが実施されている
- メンバー同士が他人の意見を聞き、論理的な決断をしている
- 見積もりと計画がチームメンバー全員によって行われている

2014年度　JASPIC
分科会での研究報告より

高品質ソフトウェア開発チームアセスメントシート 3/3

FUJITSU

チームワーク

- 開発作業を行っている時間とその他の時間がそれぞれ計測されている
- プロジェクトの進捗状況が定量的に評価されている
- 要件の変更に同意する前に、すべての変更を評価し、計画に与える影響を理解している
- コミットメントしたスケジュールに遅れそうな場合、計画通りに戻すための行動が取られている
- 助けが必要な場合、すぐに対策の計画を作り、マネジメントに問題を報告している

2014年度　JASPIC
分科会での研究報告より

率直な感想

FUJITSU

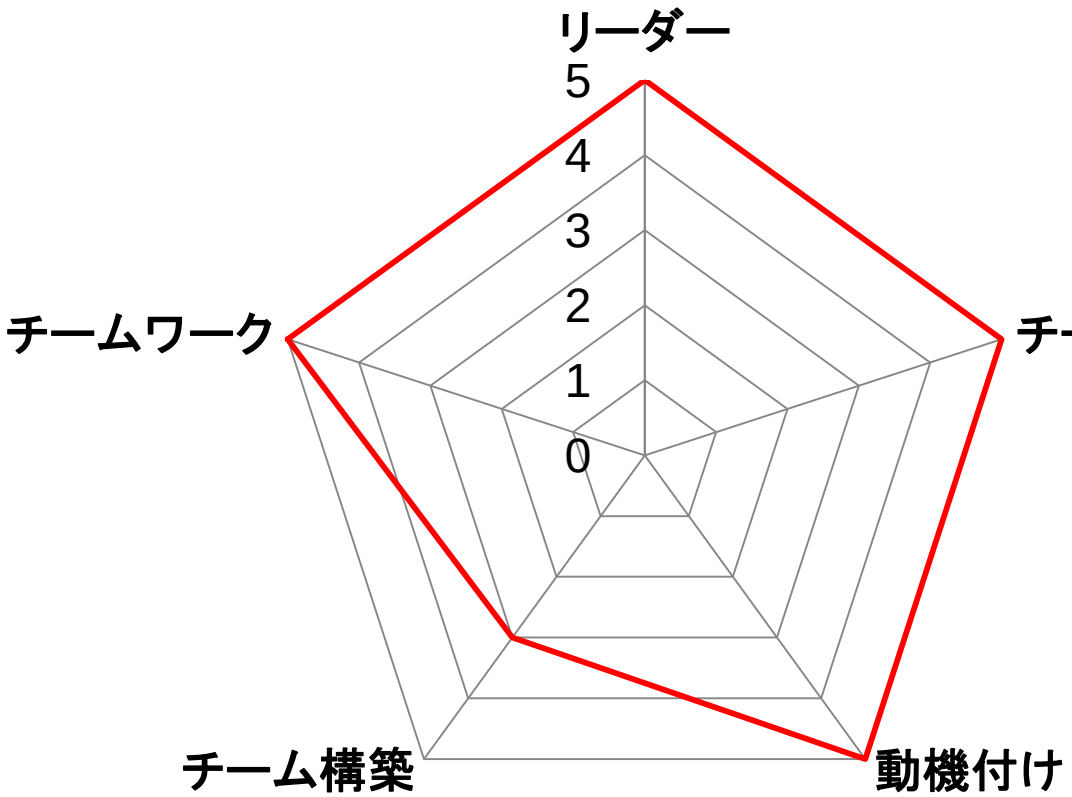
ちゃんとした
アジャイル開発なら・・・、

普通に満足
できますよねぇ？

念のためにアジャイル事例で評価してみました

FUJITSU

アセスメントシートを事例で評価



FUJITSU

４．超ざっくり
アジャイル開発とは？

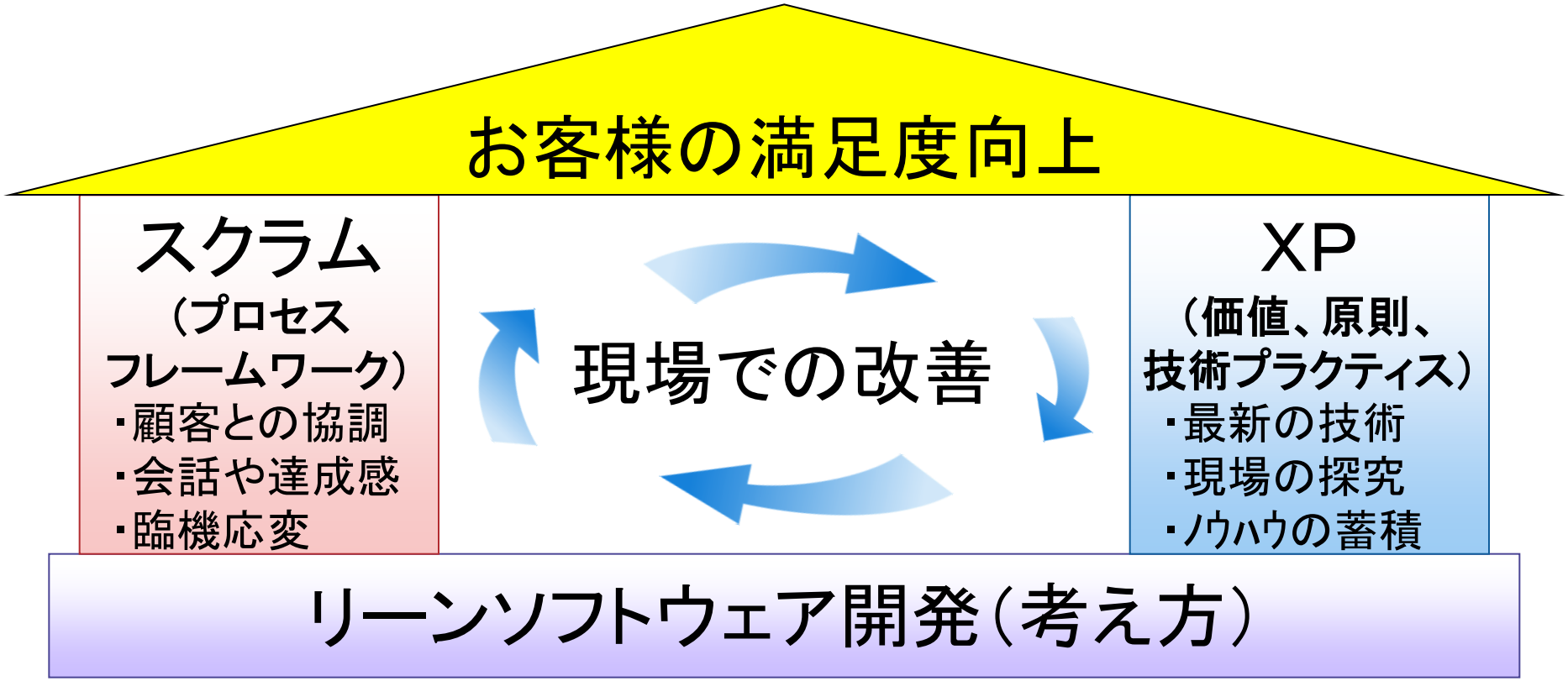
お客様の満足度を最優先にする

FUJITSU

■アジャイル開発手法の中では、リーン、スクラム、XPが主流です。

■リーンをベースにして、スクラムとXPを現場に合わせてカスタマイズし適用します。

■また、現場で継続した改善を行うことで、お客様の満足度向上を目指します。



※リーンソフトウェア開発：トヨタ生産方式（TPS）等の優れた生産現場の手法をソフトウェア開発に応用したもの

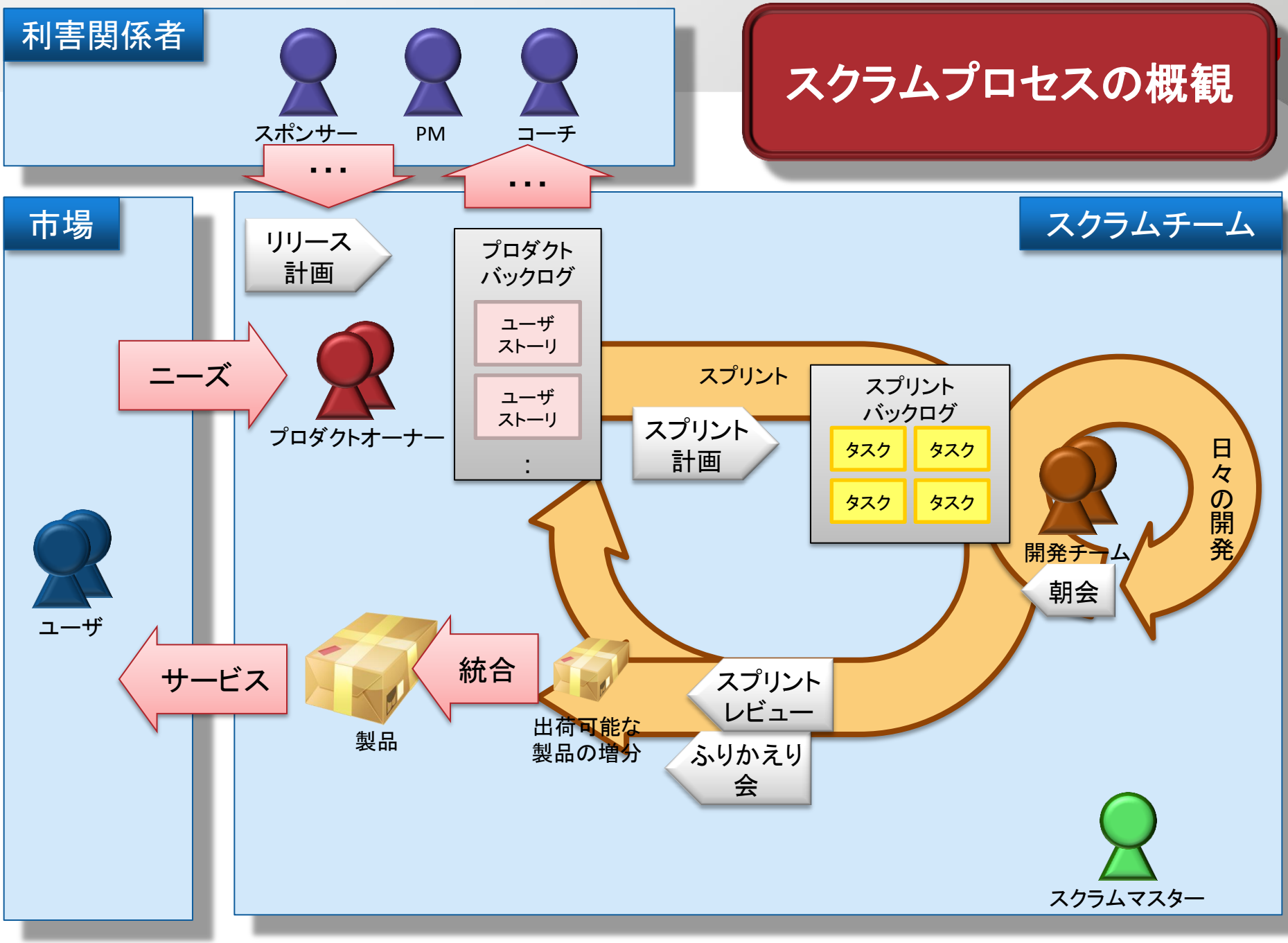
（参考）リーン、スクラム、XPについて

FUJITSU

アジャイル開発現場の状況により、3つの手法を組み合わせて活用します。
（注：3つの手法は考え方が重なる部分がありますが、ここでは分けて説明しています）

手法名	観点	主なキーワード	内容
リーンソフトウェア開発	価値と原則	7つの原則	アジャイル開発を実践する際の思考や行動の基礎をまとめたもの。全員で共有する。
		7つのムダ	トヨタ生産方式で有名な「7つのムダ」をソフトウェア開発の現場に置き換えたもの。
スクラム	プロセスフレームワーク	役割の明確化	プロダクトオーナー、チーム、スクラムマスターを配置する。
		スプリント（または、イテレーション）	一定期間（1週～4週）の中で要件定義～設計～開発～テストを繰り返し、動くものをリリースする。
		プロダクトバックログ	お客様の案件を必要とする順番に上から1列に並べる。状況の変化に応じて逐次並び替える。
エクストリームプログラミング（通称はXP）	価値、原則技術	ペアプログラミング	不確実で困難な作業に対して2人の知恵を結集して克服する。その過程でお互いの知恵も共有する。
		テスト駆動開発（TDD）	テストコードの作成と実行からフィードバックを繰り返し得ることで設計と開発を洗練する。
		継続的インテグレーション（CI）	ビルドとテストを随時自動実行することにより早期に異常を検知して常に障害のない状態を維持する。

スクラムプロセスの概観



FUJITSU

5. 事例概要

プロジェクト・プロフィール

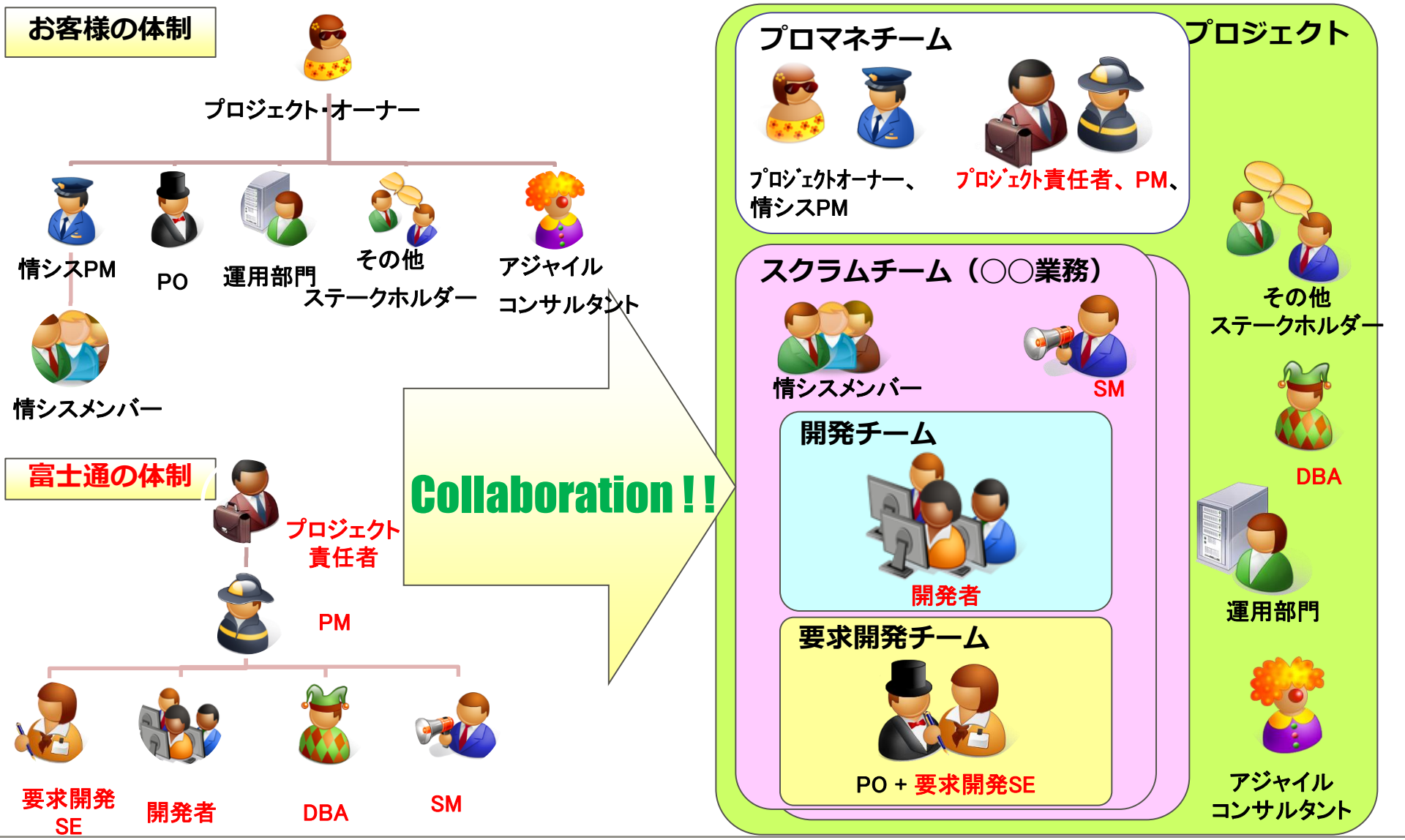
FUJITSU

プロジェクト名／業種	営業支援システム　／　製造業A社様
システムの概要	新たな営業支援機能をクラウド（AWS）上に構築、利用はiPadから
システムの目的	iPadを活用した営業のワークスタイル変革
プロジェクトの特徴	・「要求開発」と「アジャイル開発」を並行して実施 ・アジャイル開発の経験豊富な他社ソフトウェアハウスとの協働
開発体制	・開発ピーク時：2チーム15名（他社は含まず）　うち富士通グループ：10名 内訳（PM：1名、要求開発SE：2名、DB：2名、開発者：7名、SM：2名、デザイナ：1名） ※SM：スクラムマスタ ・保守：　開発者 2名　うち富士通グループ：1名
開発期間	開発：2012/07 － 2013/12（18ヶ月）　／　保守：2014/01 －
開発規模	約165人月／約100画面
開発言語／フレームワーク／構築環境	Ruby／Ruby on Rails／Amazon EC2　※左記はお客様の指定 その他言語：Haml, JavaScript, CoffeeScript, 他
契約形態	準委任にて契約。
総評	・エンドユーザーから、使い易い良いシステムができたと賛辞をいただけた。 ・お客様において初めての本格的アジャイル開発であったが、お客様の協力もあり、エンドユーザーが満足するシステムをリリースできた。また、先進的な開発支援ツールを活用したプロセスの効率化、コード品質向上等のノウハウが蓄積できた。

体制（モデル）

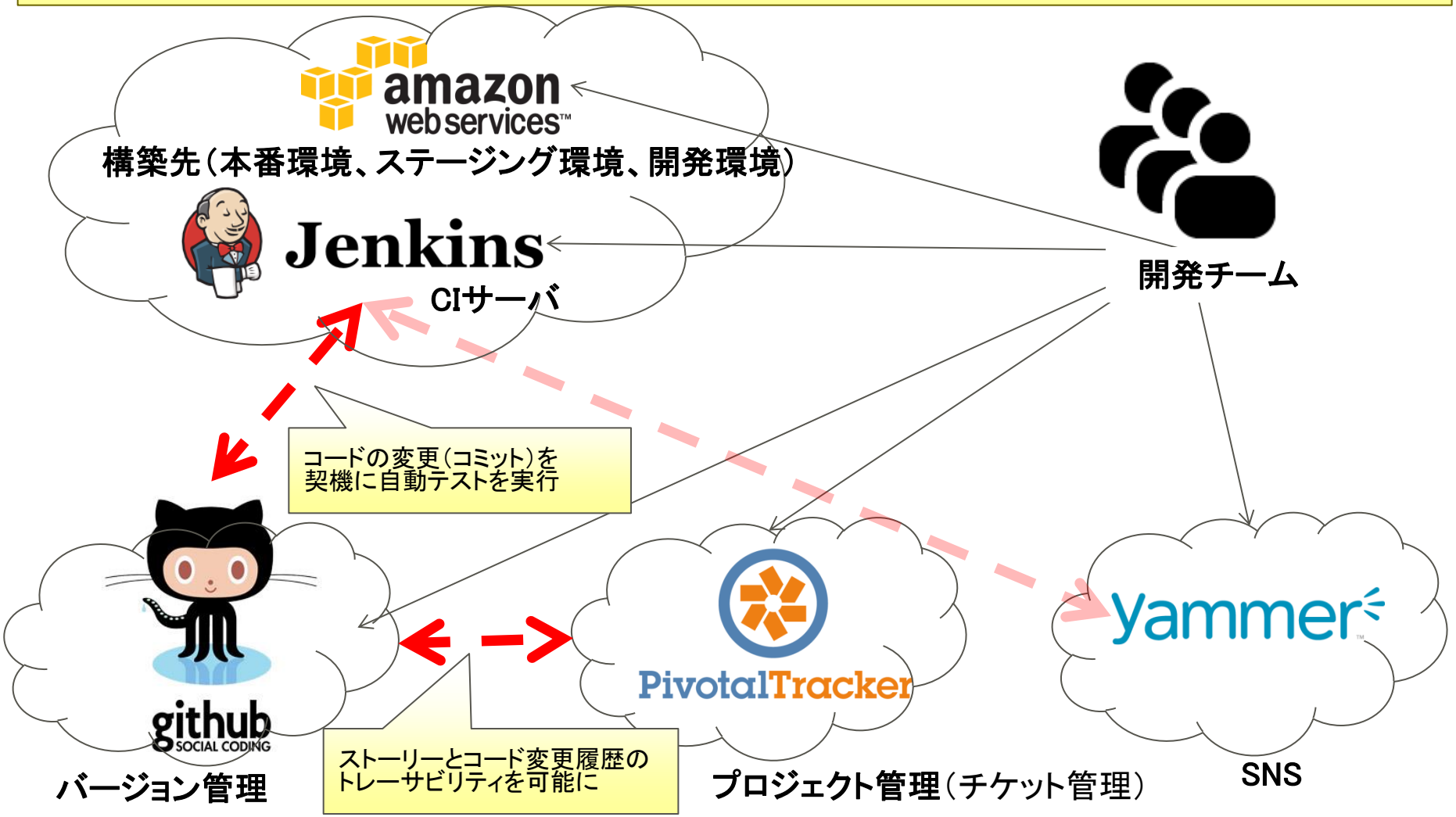
FUJITSU

- 各階層のチームをお客様と富士通のコラボレーションで構成し、受発注関係の組織を横断したチームとして共通のゴールを目指した。



開発環境

- agility を促進するためには開発環境（主に開発支援ツール類）の選択も重要。これらは協働していた他社のノウハウを吸収
- バージョン管理ツール：github と CIサーバ：Jenkis を連携し、githubへのソースコードのコミットを契機に自動テストを実施
- github と プロジェクト管理ツール：Pivotal Tracker を連携し、ソースの変更履歴とユーザーストーリーとのトレーサビリティを実現
- Jenkis とSNS：yammer! を連携し、自動テストの結果などをメッセージとして通知
  ⇒ Jenkinsが異常終了した際に、不要なメッセージが増加するため途中で連携は解除



６．アセスメントシートを事例で評価してみました

事例評価(リーダー)

FUJITSU

リーダー

リーダーの仕事は、自分で作業するのではなく、作業する人をリードすることであると知っている

- リーダー像の違い。当事例では、リードプログラマとしてPOとの仕様調整をする程度。
- その他はフラットで、メンバはリーダの指示待ちもしないし、リーダも管理しない
- 自律、自己組織化

リーダーは明確なチームの最終的なゴールと短期的なゴールを持っている

- リーダというより、チーム全員が同じゴールを共有
- インセプションデッキ、リリース計画、スプリント計画、朝会など、各レベルでゴールを共有

リーダーはチームメンバの手本となる行動をしている

- リーダが手本となる特別な何かは無い。リーダー、メンバーの区別なくチームとしての説明責任にしたがって行動している。

リーダーはチームが成功することを信じている

- リーダはもちろん、メンバ、スクラムマスターも成功を信じているというより、成功のために諦めない。
- 問題はすぐに露見するので、それに対する対策をひとつひとつ迅速に打っていく

リーダーはチームメンバーの成功を認めている

- 成果については、リーダー、メンバの区別はなく、チームとして成果に責任を持つ。
- コードの担当は存在しない。コードシェアや、アジャイルインスペクション、日々のリファクタリングなどで共有している。自発的、積極的に負荷の高いメンバを相互に支援することが日常。

事例評価（チームメンバー）

FUJITSU

チームメンバー

メンバーはチームへの帰属意識を持っている

- プロダクトの品質はチームとして保証するため、帰属意識は必然的にある。

メンバーは共通のチームゴールにコミットしている

- ゴールのコミットはリーダーのスライド参照
- コミット内容は、プロダクトバックログにつまれたストーリー、受入条件、プロダクトバックログの優先順位、リリース計画とリリース内容の調整

メンバーは開発プロセスと計画を自分のものであると感じている

メンバーは計画を作るスキルを持っている

- 全員で計画を行うため、計画スキルは育つ。ただ、最初からあるわけではなく、繰返し計測、見直しを行うことで身につけ徐々に精度も上がる。

  例１）最初は１週間のスプリントの計画に半日程度-> ３～４スプリントで１～２時間に

  例２）普通、２～３時間単位のタスクに分割できるようになるまで2ヶ月かかる

メンバーは計画に従う規律を持っている

- 規律は「ちゃんとした」アジャイル開発のキモ！ 自由なイメージがあるが、実は他のどんな開発手法よりも規則正しくプラクティスを実践することが要求される。そして、それらは誰かからやらされるのではなく、チームの中でプロセスが改善され、プラクティスの改善や新しいプラクティスが生まれる。

  例）アジャイルインスペクション＋リファクタリング、日次レビュー＋日次マージ

事例評価（動機付け）

FUJITSU

動機付け

リーダー・メンバーは自分の自由意思で計画にコミットしている

- リリース計画、スプリント計画は、全員参加で自由意志でコミット

リーダー・メンバーはコミットする前に交渉ができる

- リーダーとメンバの間に上下関係はなくフラット。
- 計画時にはコミット前に見積で共有、交渉可能。朝会でのタスク取りも基本的にサインアップ方式なので、何かを指示されるということはない。
- 権限、裁量がある程度認められていると同時に、個人ではなくチームとして成果、品質に責任説明を持つ。

リーダー・メンバーが何にコミットしたかが明確になっている

- 何にコミットしたかは、リリース計画、スプリント計画で明確化。
- 見積もり後にPOとコミットした内容は、SNSやプロジェクトマネジメントツール（チケット管理）上でチームだけでなくお客様のステークホルダーにも透明性をもってシェア

リーダー・メンバーがコミットメントを果たすための計画を自分で作っている

- 計画はチームで行い、合意し、POにコミット

コミットメントが確実に守られているかフォローされ続けている

- コミットの遵守状況は、常にプロジェクトマネジメントツール（チケット管理）、バーンダウンチャートで見える化された状態。
- フォローは、CIによるテスト結果、常時コミュニケーション、日次の朝会、週次のスプリントレビューで等でフォローされ続ける。

事例評価(チーム構築)

FUJITSU

チーム構築

計画に基づいてリソース(人、環境、ツール)が割り当てられている

- 計画の全体像が完全には明確でなく、リソースを計画通りに揃えることは難しい。想定可能な範囲内のリソースで万全を期す。状況の変化に応じてリソース追加の必要性がみえてきたら別途検討する。
- **アジャイルでは、個人個人がオールマイティである必要はないが、チームとしてすべての作業が出来るように最適化する。**知識の共有・創造によるチームの成長にも期待するが過度な期待は失敗のもと。

開発において必要とされるスキルが明確になっている

- 必要なスキルは可能な範囲内で揃える。
- 当事例では、契約前にお客様によるメンバの面接を行い、スキルレベルを試された。

スキルアップのためのトレーニングが実施されている

- スキルアップ計画は事前教育＋OJT。**自発的に勉強会や書籍の回し読み**も行った。
- **立ち上げ時には、支援チームの実践経験者によるプロセスのチェックとアドバイス(ダメだし)を定期的に(週1×2ヶ月)実施**し、メンバーが自分達で回せるようになるまで支援を受けた。
- **本開発の前にスパイクを実施し、プロセスの試行と生産性を計測し、本開発の見積に活かした。**

メンバー同士が他人の意見を聞き、論理的な決断をしている

- コミュニケーションコストはとても高い。フォーマルだけでなく、インフォーマルなコミュニケーションを多用。会話だけでなく、SNS、チャット、pull requestによる レビュー＆リファクタリング他

見積もりと計画がチームメンバー全員によって行われている

- 見積と計画はスプリント計画ミーティングで一同参加

事例評価(チームワーク)

FUJITSU

チームワーク

開発作業を行っている時間とその他の時間がそれぞれ計測されている

- 作業は総てタスク化され、付箋のタスクカードに見積／実績時間を記録。

プロジェクトの進捗状況が定量的に評価されている

- コミットした計画(リリース計画、スプリント計画)に対し、それぞれの進捗状況を2種類のバーンダウンチャート(リリースバーンダウンチャート、スプリントバーンダウンチャート)で見える化。定量はステップ数ではなく、受入されたユーザーストーリー(要求)のストーリーポイント数

要件の変更に同意する前に、すべての変更を評価し計画に与える影響を理解している

- 要求の追加・削除、優先順位の入れ替えは頻繁に発生する。スプリントレビュー後に毎週チームとPOで見直しを行い、同時にリリース計画への影響も合意をする

コミットしたスケジュールに遅れそうな場合、計画通りに戻すための行動が取られている

- 小さな単位でコミットしスプリント毎にスケジュールへの影響も確認していくため、突然大きなスケジュール遅延は出にくい。それでも優先順位の入れ替えなどによりリリース計画でコミットしたスコープが変更になることはある。これもスプリント毎にPOと合意・調整しながら進むため、大きな問題になることは少ない。開発終盤のスプリントで、プロダクトバックログの残項目に関する取り扱いについては、契約時に合意しておく必要がある。最終スプリントにおいてプロダクトバックログに残っている優先順位が低いユーザーストーリー関しては、実装しないことを合意する。

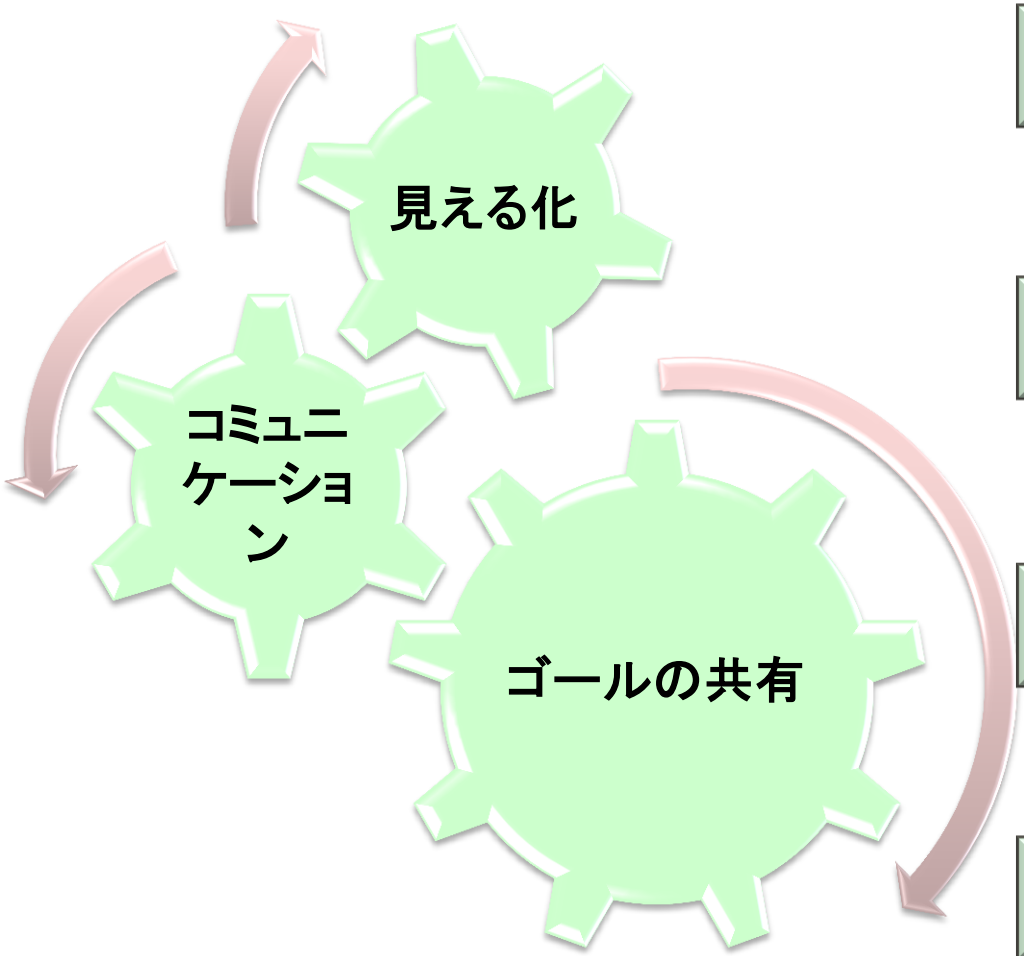
助けが必要な場合、すぐに対策の計画を作り、マネジメントに問題を報告している

- 助けが必要な状況は、朝会で最初に(緊急時は随時)ピックアップする。チームだけで解決できない問題は即座にプロマネにエスカレーションする。お客様を巻き込む必要がある場合は、緊急度によりインフォーマル、フォーマルなコミュニケーションを取る。

FUJITSU

７．自律的に機能するチームの特徴とふるまい

チームの特徴



信頼感の醸成

権限移譲と説明責任

自律、自己組織化

コラボレーションの促進

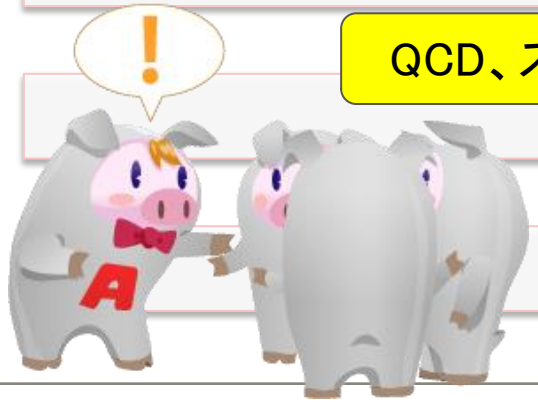
異常検出・カイゼンの仕組み

ゴールの共有

FUJITSU

インセプションデッキ　(出典:「アジャイルサムライ」- 達人開発者への道　オーム社)

1. 我々はなぜここにいるのか？ — プロジェクトの目的
2. エレベーターピッチ — プロジェクト、プロダクトの概要
3. パッケージデザイン — プロダクトの価値
4. 「ご近所さん」を探せ — ステークホルダー
5. やらないことリスト — スコープ
6. 技術的な解決案を描く — アーキテクチャー
7. 夜も眠れなくなる問題 — リスク
8. 期間を見極める — デリバリー/リリース
9. 何を諦めるのか — QCD、スコープの優先順位
10. 何がどれだけ必要なのか — リソース



インセプションデッキの威力（実践者の声）

FUJITSU

- プロジェクトの目的や背景について、お客様と開発チームが腹を割った議論ができた。
- 直接質問や会話をすることにより、従来は上流工程のドキュメントに「文字」として書かれた情報でしか理解できなかった部分が、ドキュメントには表現しきれないお客様のビジネスに対する「ニーズ」、「課題」、「思い」を共有することができ、同じゴールを目指すひとつのスクラムチームになることができた。
- 開発対象のプロダクトの本質的な魅力を掘り下げることで、何が重要であるか、本当に欲しいものは何かが明確になった。
- やらないことを明文化することで、スコープを明確にすることができ、スコープの開発に集中することができた。スコープを議論することで、お客様とのスコープの認識のズレを早期に発見できた。
- 技術的な解決案（アーキテクチャ）を議論することで、アーキテクチャが実現手段として最適かを検討することができ、潜む技術的なリスクを炙りだすことができた。
- 想定しうるリスクを洗い出し、対応可否を決定しておくことで、課題を早期に把握できた。
- 品質、時間、コスト、スコープの優先順位を明確にすることにより問題発生時の舵取りが容易になった。

コミュニケーション

FUJITSU

- Face to Face の腹を割った会話
- インフォーマルな会話の重要性（雑談、グチ、相談、仕事以外等）
- スピードと透明性（SNS、チャット等の積極的な活用）
- ペアプログラミング、etc

ペアプログラミング

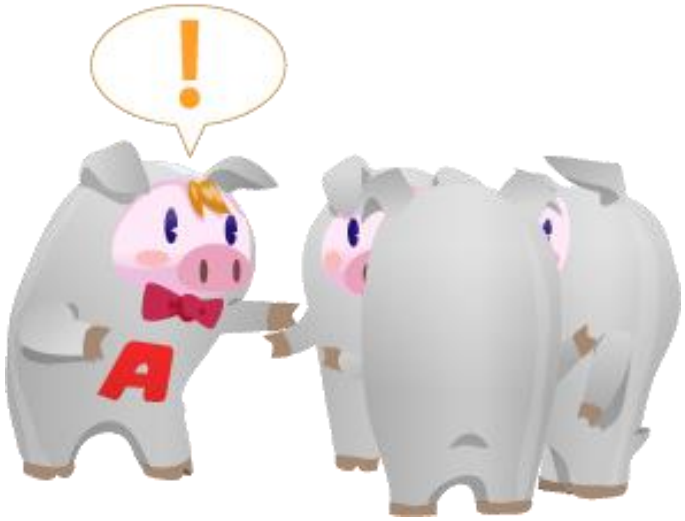
FUJITSU

- よくある誤解
  - 詳細設計書からコードに起こす作業を2人で行うのはムダでは？
    - プログラミングは本来「設計・実装・テストを同時に進めていく」ことを指す。コーディングのみを2人でやることではない。難しい作業だからこそ2人で行う価値がある。
  - 1人がコーディングし、もう1人がテスト仕様書を書く形態は？
    - 2人が別々のことを行うのはペアプログラミングではない。2人で同じ画面を見ながら知恵を出し合うことが大切。
- 生産性検証
  - 2人で同じ作業をすることから、工数2倍？（生産性1/2？）との質問をよく受けるが、1人で行うより速く終わるため工数が2倍になることはない。社内のパッケージ適用事例では、ウォータフォール開発（ソロ）と同等の生産性（工数）という測定データがある。
- スキルアップ効果
  - 業務知識、開発技術のスキル移転や高いレベルの平準化、説明能力や改善能力の向上、ショートカットキーやコマンドの簡略化など、小さな暗黙知、ノウハウの共有が進み、チームとしての能力が向上する。

ペアプログラミング

FUJITSU

ペアプロの効果（例）



- 集中力、問題の整理能力の向上。
- コラボレーションによる発想力向上。
- 暗黙知の共有、脱常識な改善案の創出。
- 単純ミスの軽減。
- スキル移転。
- 頻繁なペア替えによる属人性排除。
- 問題発生瞬間の共有と「自責」のふりかえり
  - 例えば誰かが何か失敗をした時、「あーっ！！」という声を発する。その瞬間、失敗を犯したメンバーのペアだけでなく、別のペアからも「どうしたの？」と声がかかり、その瞬間とその時に交わした会話は、映像とともに強く記憶されチームで共有される。

  ふりかえりの際には、失敗したメンバーが挙げた問題に対し、他のメンバーも、失敗の瞬間がその映像と共に生々しく思い出され、他人ごとではなく、問題を「自責」として考え、失敗した個人の問題ではなく、チーム対問題として捉えることができる。

お客様を含めた見える化

FUJITSU

- 貼りものによる進捗の見える化（スプリントバーンダウンチャート、リリースバーンダウンチャート、タスクボード）

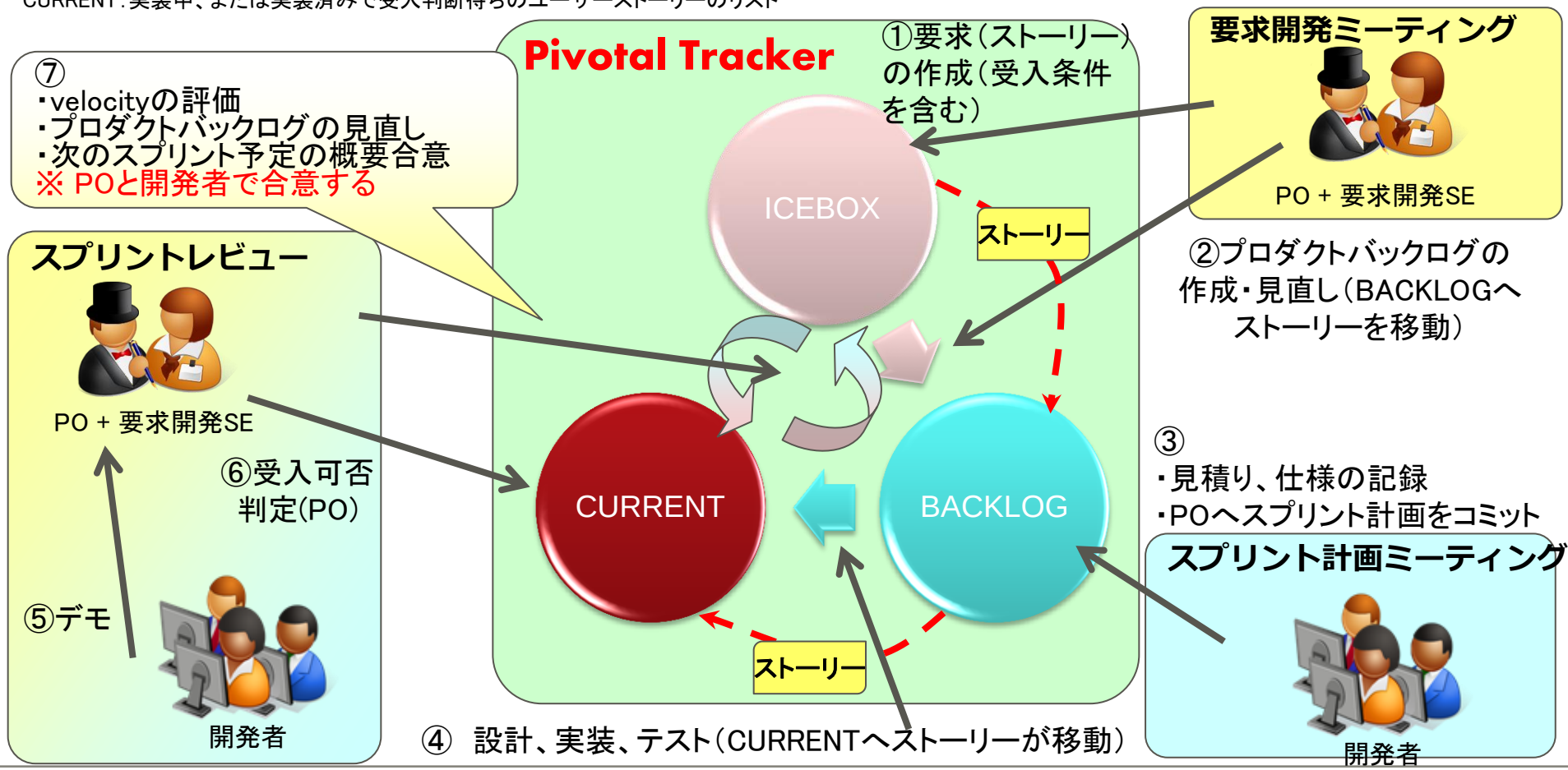
- コミュニケーションツールによる見える化（SNS、構成管理ツール（Github）、プロジェクト管理ツール（Pivotal Tracker）、wiki 等

Pivotal Tracker を活用したスクラムサイクル

FUJITSU

- シンプルな機能と簡単な操作の高生産性プロジェクト管理ツール、Pivotal Trackerを活用し、以下の効果があった。
- 要求に関する情報(優先度、着手状況、仕様など)の見える化と、全関係者(お客様含め)との真実の情報・状況の共有
- 要求の変更経緯や仕様の変化をユーザーストーリーに一元管理することで、不要な中間ドキュメント作成のムダを排除
- ユーザーストーリーは「チケット」としてスクラムのプロセスに沿ってPivotal上をICEBOX->BACKLOG-> CURRENT と遷移し、Pivotalの直感的なユーザーインターフェースで進捗が容易に把握できる
- スプリントレビューにおける受入プロセスを簡略化し、受入判断の明確化

※註
ICEBOX: とりあえず欲しいレベルのユーザーストーリーのリスト。最終的には実装しないものも含んでいる。
BACKLOG:実装すべき優先順位付けされたユーザーストーリーのリスト(プロダクトバックログ)。ICEBOXから選抜
CURRENT:実装中、または実装済みで受入判断待ちのユーザーストーリーのリスト



（ご参考）Pivotal Tracker 画面イメージ

FUJITSU

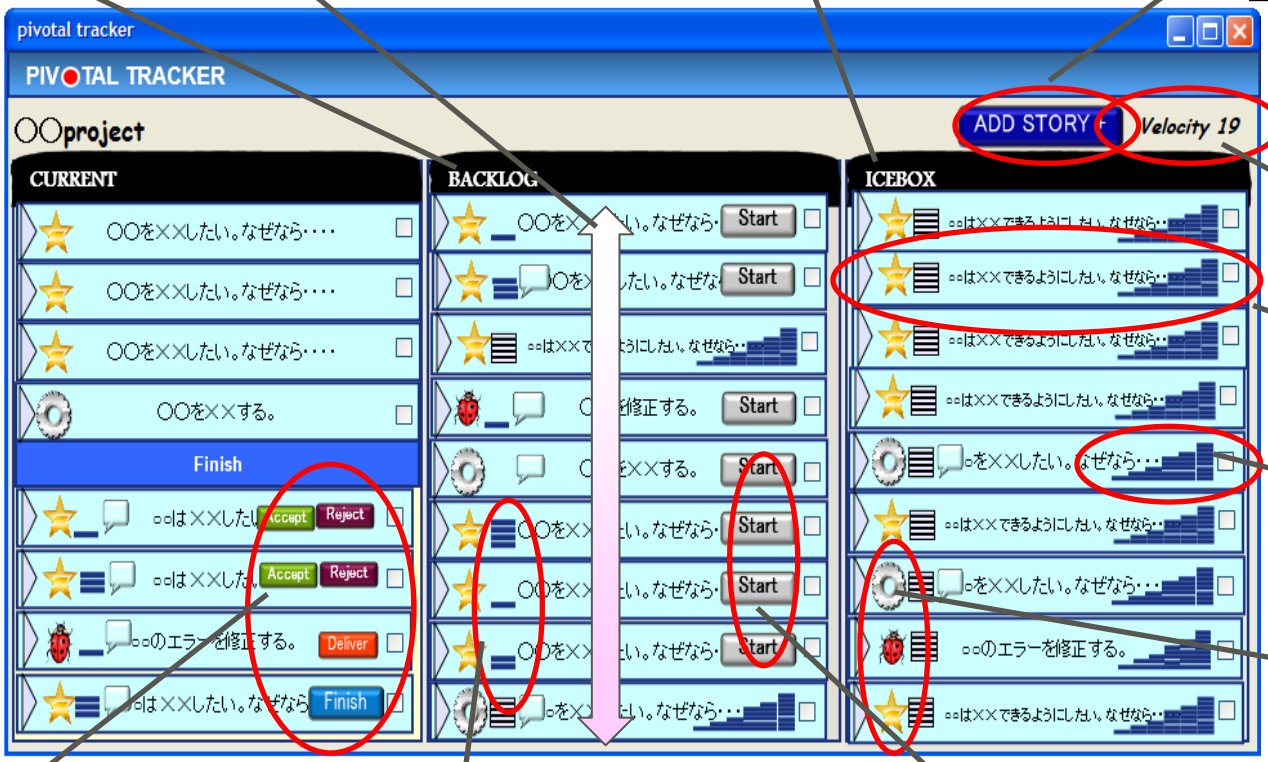
BACKLOG

ユーザーストーリの優先順位は
drag&dropで移動可能

ICEBOX

ADD STORY ボタン

押下すると新たなユーザーストーリーを
ICEBOXに作成する



velocity

Velocityを自動で計測し、前スプリントの実績を表示する

ユーザーストーリー

クリックするとダイアログが開き、詳細な情報を入力できる

見積り選択

レベルメーターを選択し見積り値（ストーリーポイント）を指定する

ストーリー種別

種別は3種：直感的アイコンでシンプルユーザーストーリ（星）
開発以外の作業（歯車）、
バグ（てんとう虫）

状態表示ボタン

「Finish」：開発中
↓　押下（開発終了）
「Deliver」：提供可能
↓　押下（受入可能）
「Accept」／「Reject」
↓
受入可否判断をし、何れかを押下し、ユーザーストーリーを完了する。

ユーザーストーリーは
drag&dropで移動可能

見積り値

見積り選択でストーリーポイントを選択すると、このアイコンが見積り値分着色される。

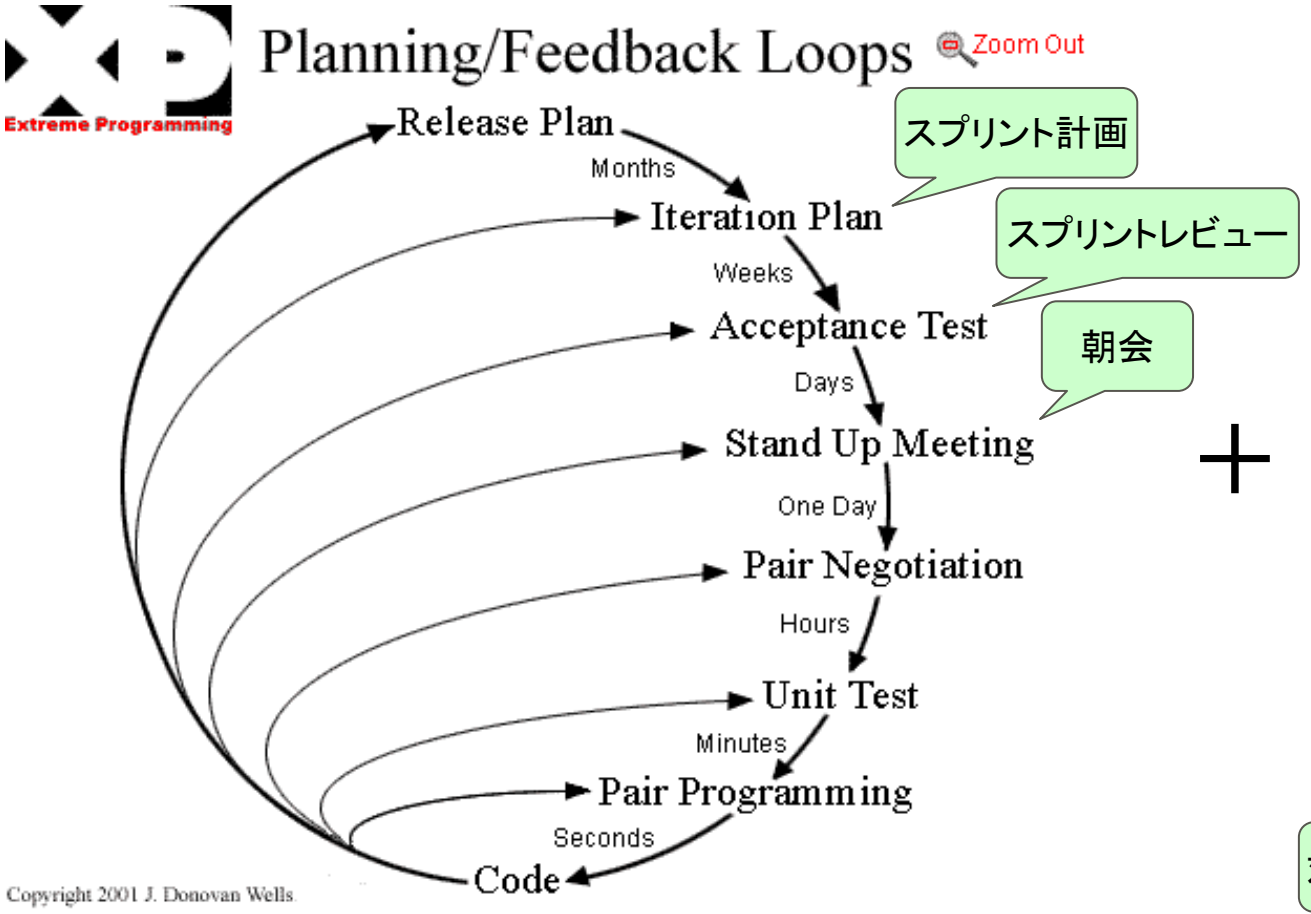
Startボタン

見積り選択でストーリーポイントを選択すると表示される。Start可能な状態を示す。Startボタンを押下すると、自動的にCURRENTへ移動し開発中を意味するFinishボタンが表示される。

異常検出・カイゼンの仕組

FUJITSU

頻繁な多重のフィードバック



スプリント計画

スプリントレビュー

朝会

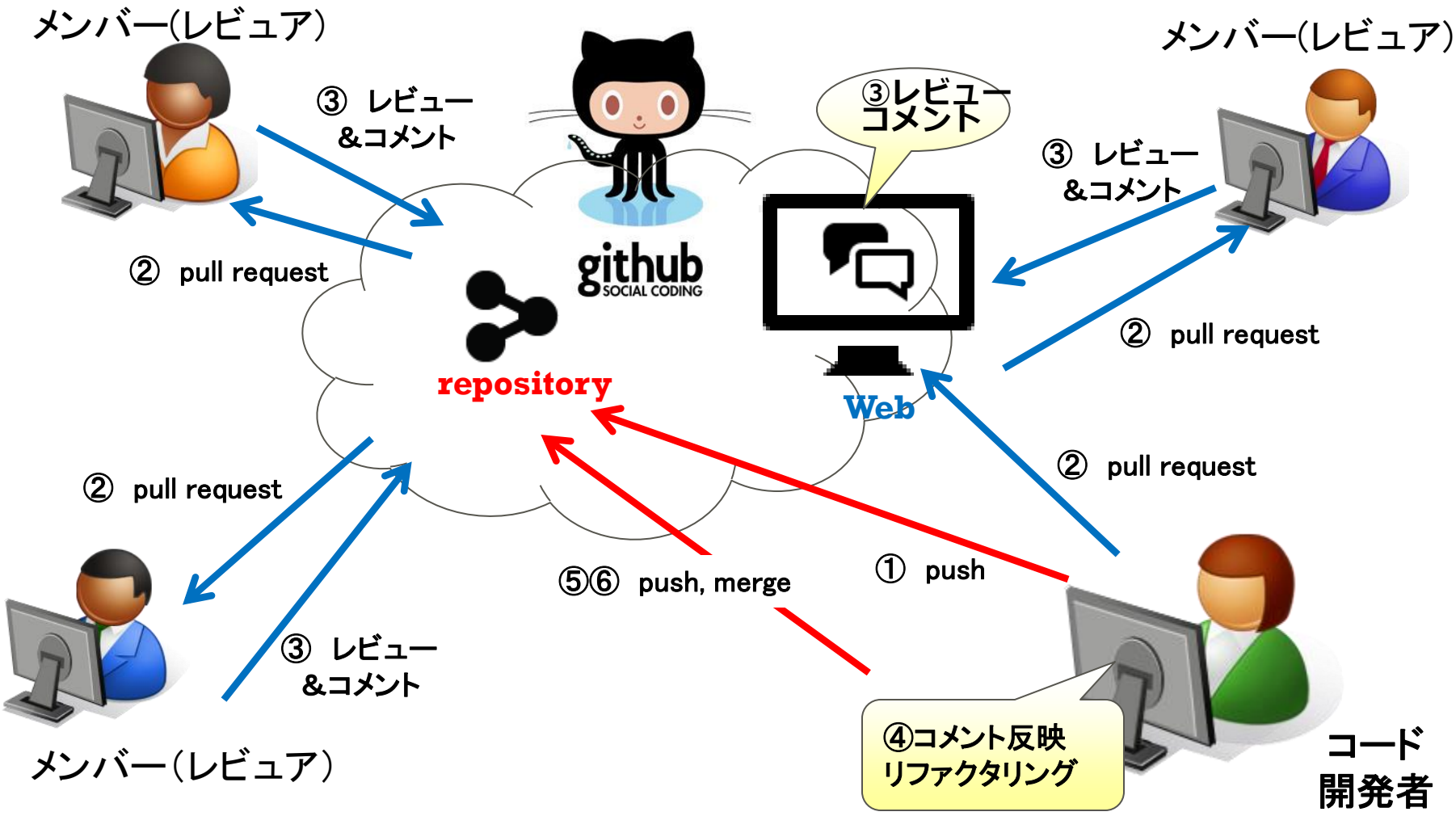
＋　ふりかえり

対応するスクラムのイベント名

コラボレーションの促進、リファクタリング、品質向上

FUJITSU

- github の pull request 機能を利用し、ソースコミット時のレビュー、リファクタリングを習慣化
- ①コードを反映してほしい開発者は、反映させたいリポジトリのブランチにpull request（マージの依頼）を発行する。
- ②ブランチの関係者にpull request が通知され、ブランチへのコードマージ依頼が通知される。
- ③pull request を通知された開発者は、マージ依頼のあったコードをレビューし、可否をコメントする。（request の拒否も可能）
- ④依頼者はレビュアのコメント反映、リファクタリングを行い、レビュアのOKが揃ったら⑤コードをコミットし⑥マージする。



FUJITSU

８．プロセス改善における アジャイル開発からのヒント

アジャイルソフトウェア開発宣言

私たちは、ソフトウェア開発の実践
あるいは実践を手助けをする活動を通じて、
よりよい開発方法を見つけだそうとしている。
この活動を通して、私たちは以下の価値に至った。

プロセスやツールよりも個人と対話を、
包括的なドキュメントよりも動くソフトウェアを、
契約交渉よりも顧客との協調を、
計画に従うことよりも変化への対応を、
価値とする。

すなわち、左記のことがらに価値があることを
認めながらも、私たちは右記のことがらにより価値をおく。

アジャイルソフトウェア開発宣言：http://www.agilemanifesto.org/iso/ja/

リーンソフトウェア開発の本質

FUJITSU

ソフトウェア開発の7つの原則

1. ムダをなくす
2. 品質を作り込む
3. 知識を作り出す
4. 決定を遅らせる
5. 速く提供する
6. 人を尊重する
7. 全体を最適化する

参考:「リーン開発の本質」 日経BP社

スクラム理論の実現

FUJITSU

透明性

- 重要なのは、**結果責任を持つ者に対して見える化がされている**こと。

  透明性とは、この点が透明化され、**関係者全員が共通理解を持つ**こと

検査

- **好ましくない変化を検知できるように、成果物や進捗がゴールに向かっているかを頻繁に検査しなければならない**。ただし、検査を頻繁にやりすぎて作業の妨げになってはいけない。

適応

- プロセスに不備があり、成果物であるプロダクトを受け入れられないと検査人が判断した場合、**プロセスやその構成要素を調整しなければいけない。調整はできるだけ早く行い**、これ以上の逸脱を防がねばならない。

検査と適応を行う4つのイベントを定義：

スプリント計画ミーティング、デイリースクラム（朝会）、スプリントレビュー、スプリントレトロスペクティブ（ふりかえり）

出典：「スクラムガイド」
https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum%20Guide%20-%20JA.pdf

XP（eXtreme Programming）のこころ

FUJITSU

価値

- コミュニケーション
- シンプルさ
- フィードバック
- 勇気
- 尊重

何を考え
何を行うのかを
判断するのに使う
大まかな基準

原則

- 人間性
- 改善
- 反省
- 失敗
- 品質
- 小さなステップ
- 責任の受入
- 多様性
- 経済性
- 相互利益
- 自己相似性
- フロー
- 機会
- 冗長性

プラクティス

- チーム全体
- ペアプログラミング
- ストーリー
- 常時結合
- テストファースト
- ・・・・etc.

価値がないと、
プラクティスは
それ自体のために
機械的に実施され、
目的や方向性を
欠けてしまう。

参考：「XPエクストリーム・プログラミング入門　変化を受け入れる第2版」 株式会社ピアソン・エデュケーション

9. 最後に

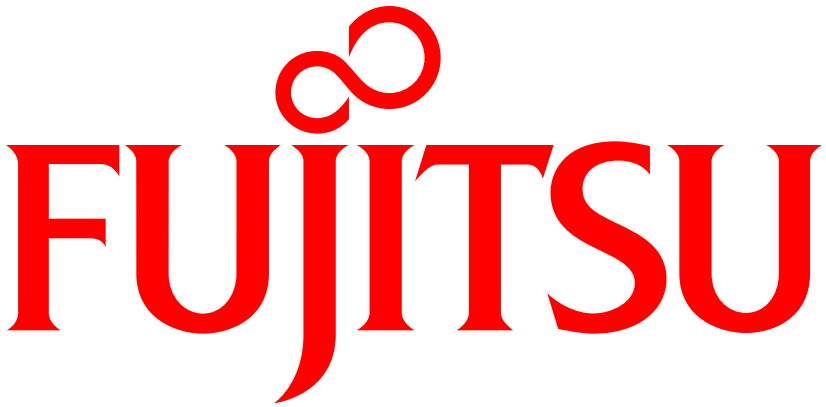
FUJITSU

- プロセス改善、チームの成長にはアジャイル開発にヒントがたくさん転がっており、アジャイル以外にも応用できる
- 目的を持たないプラクティスは機能しない。「何のためにそれをやるのか」、「それをやることで何を達成するのか」が重要
- プラクティス実践のキモは「規律正しさ」と「柔軟さ」

  それを分ける価値基準は「目的・ゴールの共有」から
- ゴールの共有、コミュニケーション、見える化、

  異常検出・カイゼンの仕組みがカギ
- コラボレーションにより、チームとしての

  知識創造が加速



shaping tomorrow with you