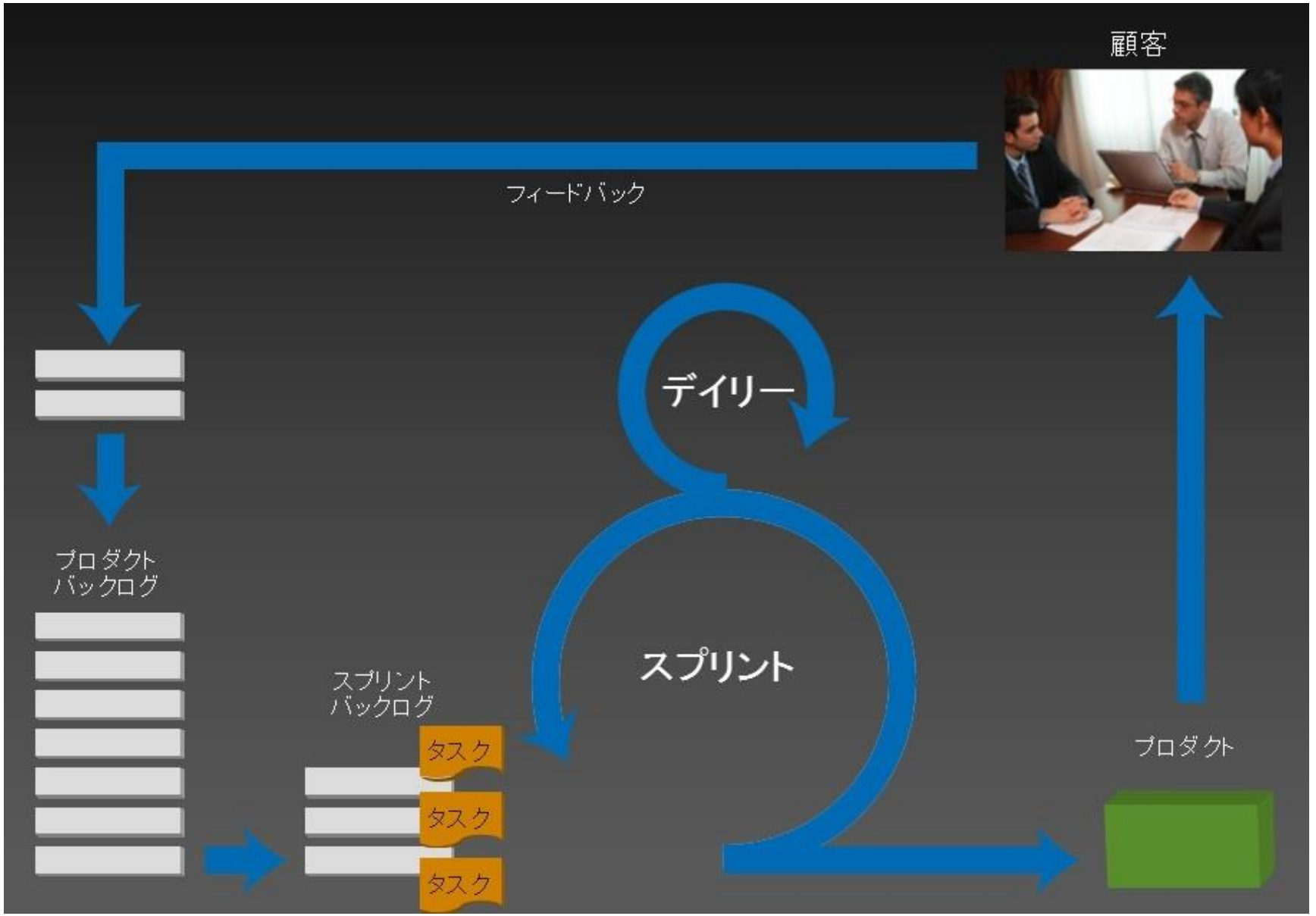
NTT DATA
Global IT Innovator

Scrumを効果的に定着させるためのプラクティス

株式会社 NTTデータ
技術革新統括本部 技術開発本部 Ａｇｉｌｅプロフェッショナルセンタ
篠崎 悦郎

Scrumの概要



Scrumの概要

Scrum ロール

- プロダクトオーナー (PO)
- 開発チーム (DEV)
- スクラムマスター (SM)

Scrum イベント

	月	火	水	木	金	月	火	水	木	金
09:00～12:00	DailyScrum スプリント計画 第1部 スプリント計画 第２部	DailyScrum	DailyScrum	DailyScrum	DailyScrum 作業時間	DailyScrum	DailyScrum	DailyScrum	DailyScrum	DailyScrum
13:00～15:00										スプリントレビュー
15:00～17:00			バックロググルーミング					バックロググルーミング		振り返り

SMとして何度か遭遇するScrum導入時の課題・原因・解決

課題

Scrum 開発が効果的に定着出来ない

- 形骸化したロール定義
- 誤った責任分担
- 経験的知見が生かされない

Scrum導入で期待されている
開発の速度を上げる事について
期待通りにならない

原因

Scrum 自体が抽象が高いプロセス

実案件では開発プロセスを補完
特に従来型のやり方を用いる

補完したやり方がAgilityを
意識したやり方であったか？

従来型の良い知見であっても
Agility が考慮されて
いなければ障害になる

解決

Scrumの推進者のSMがAgilityを意識した
適切なプラクティスを導入・率先垂範

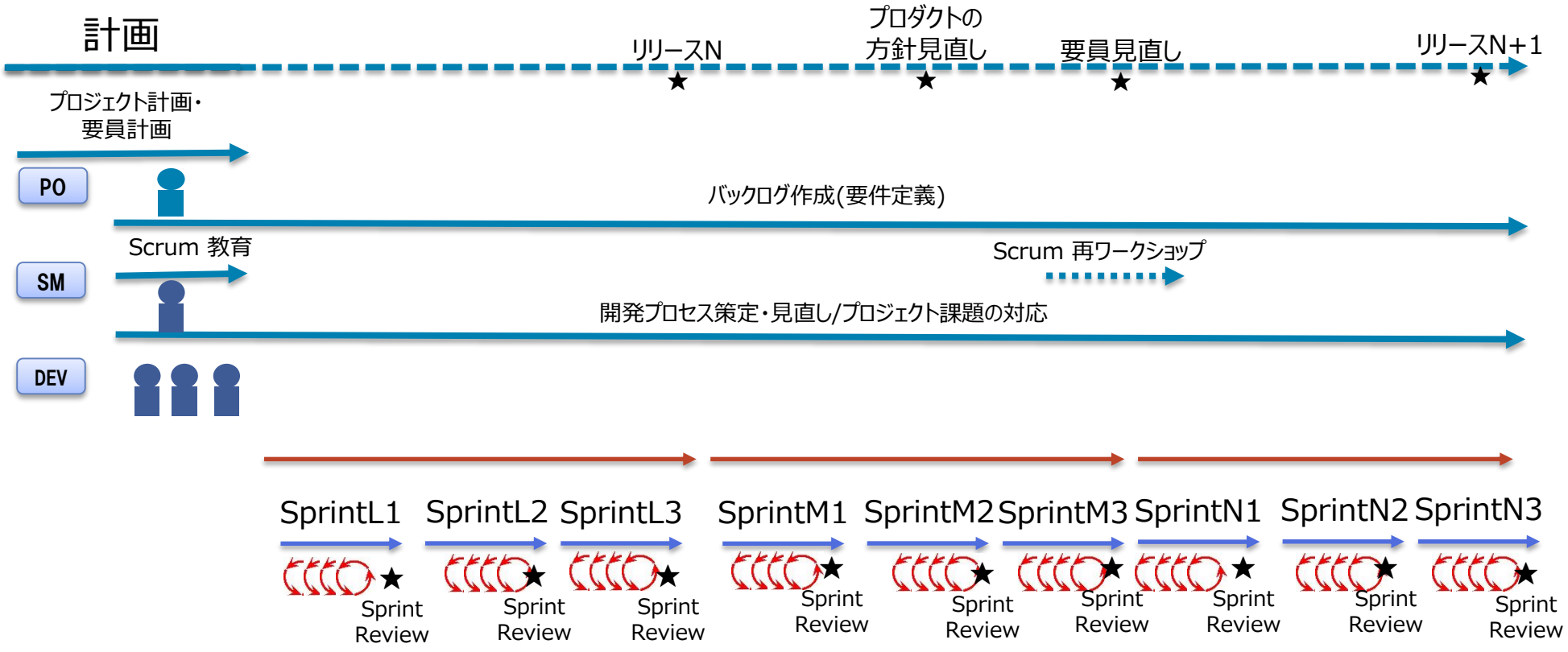
プラクティス

①：計画策定
②：Sprint計画会議
③：ディリースクラム
④：Sprintレビュー
⑤：振り返り
⑥：プロダクトバックログ
⑦：ツール・アイテム

①：計画策定

- Scrumの各イベントのスケジュールを計画書に記載する.
- 体制図を明記し役割を明確にする. 特に Scrumチームとチーム外の周囲関係者の役割分担・コミュニケーションを明確にする.
- 計画初期においてリリース計画を立てる. 特にリリース対してテーマを決める. ただし機能を固定化しない.
- 正式なScrumのトレーニングを実施し共通認識を持たせる.

①：計画策定：Scrumがチームに定着するまでの3段階



導入初期 (守)

- **初期は Scrum の原則を変えずに実施する.**
- 新しいやり方に戸惑いScrumの原則に反したカスタマイズしたくなるが，まずは原則を守る.
- 初めて Scrum に取り組む際には，今までのやり方を変える意識を持たせる.

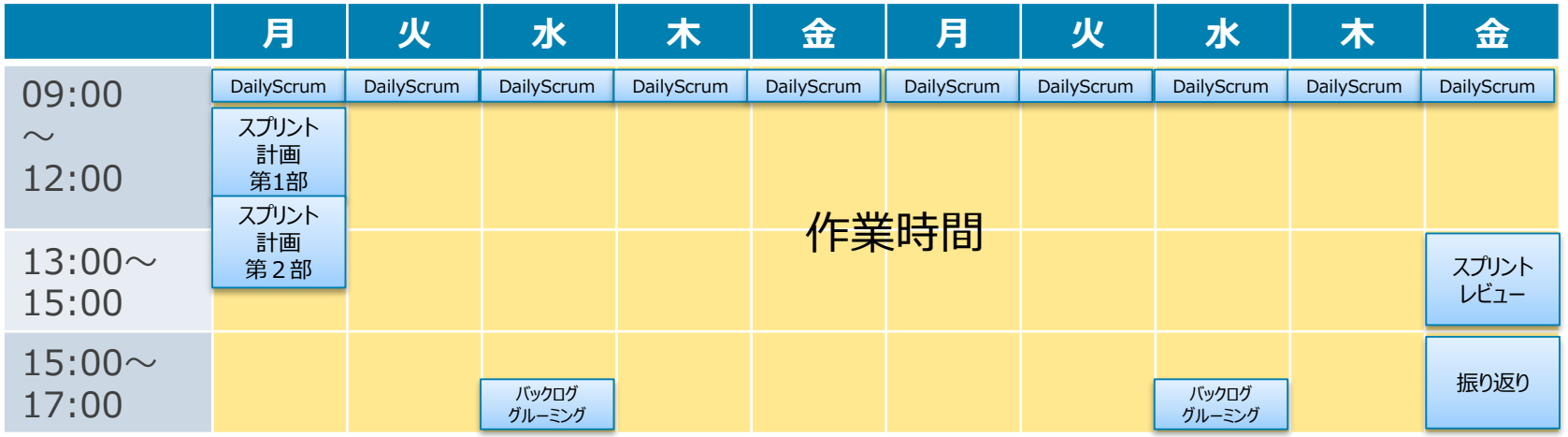
導入中期(破)

- **Scrum の原則を定着させる.**
- Scrum の原則の意味を理解し，プロジェクトの状況を分析出来る能力を身につける.
- 分析結果を元に改良・改善を行う方法を身につける.

導入後期 (離)

- ようやく**自分達の状況に合わせた見直しが可能.**
- 原則に反していたとしても，意味を理解している事で適切なやり方が出来るようになる
- 自己組織化により本質的な問題に気づき始めるとき

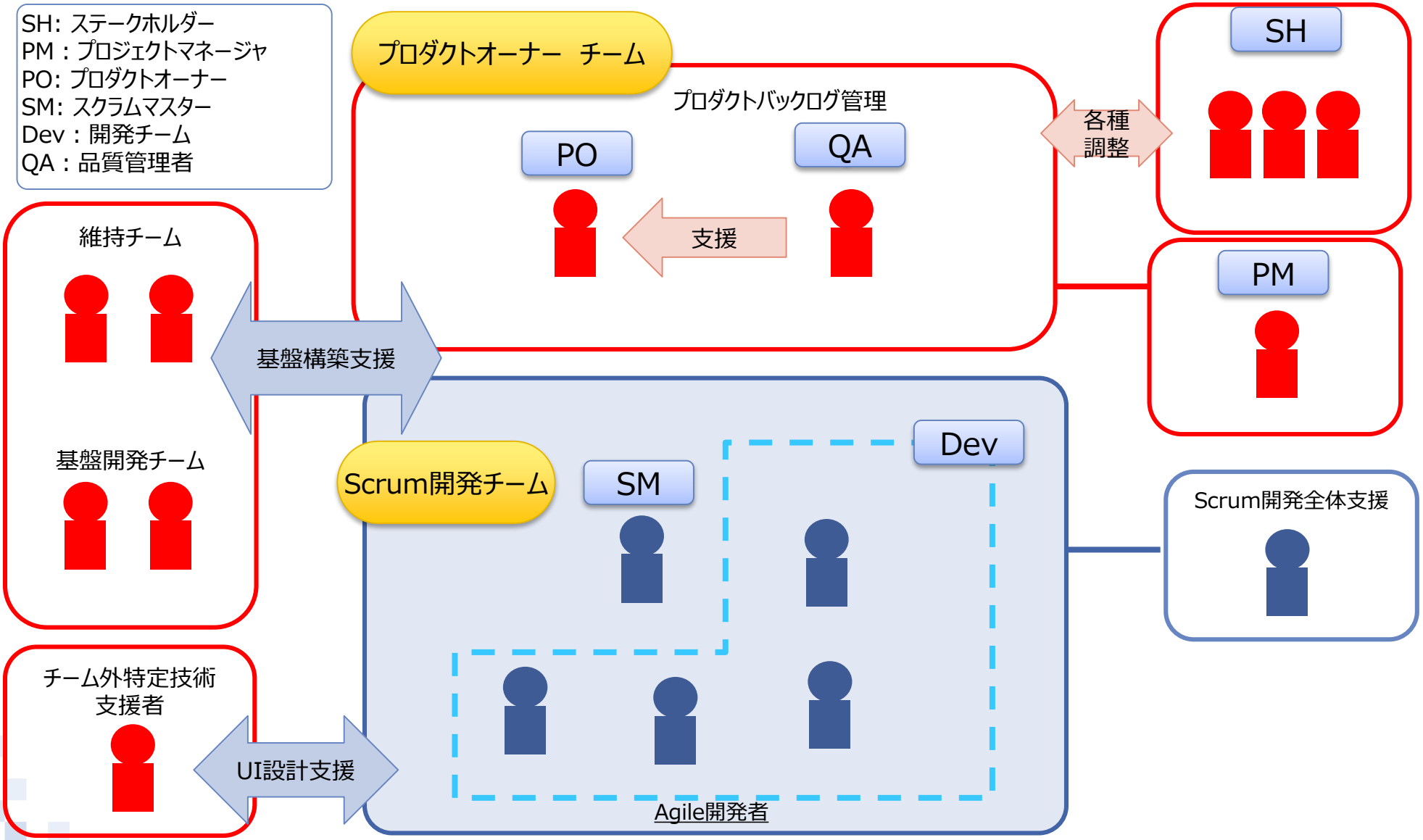
①：計画策定：各イベント



イベント	タイムボックス	プロダクトオーナー	スクラムマスター	開発チーム	ステークホルダー
Daily Scrum	15分	△	△	◎	×
スプリント計画第１部	2時間	◎	○	○	△
スプリント計画第２部	2時間	△	○	◎	△
バックロググルーミング	1時間	◎	○	○	△
スプリントレビュー	2時間	○	○	◎	△
振り返り	2時間	△	◎	○	×

◎：主催　　△：出席するかどうか状況による
○：必須出席　　×：参加可能(制限あり)

①：計画策定：体制



②：Sprint計画会議

- 会議の最初に,説明対象となりえるプロダクトバックログの一覧を示し,会議での目途時間を決めてタイムボックスを意識させる.

- プロダクトバックログの見積時には完了させるまでの作業タスク(スプリントバックログ)を共有する.

- プランニングポーカーでは状況により，些末なストーリーポイントの議論をさせない.

②：Sprint計画会議:Agenda(例)

- バックログ一覧の確認及び会議の目途時間の決定
- 優先順位上位のプロダクトバックログから具体化
  - POよりプロダクトバックログの説明.
  - 質疑応答. および具体化したタスクの洗い出し
  - プランニングポーカーによる見積. 差異の確認
  - 上記を繰り返し実施.差異がなくなる若しくは規定回数まで実施しストーリーポイントを確定
- ストーリーポイントの総和がチームベロシティまで達したら具体化完了
- プロダクトバックログ毎のタスクをスプリントバックログとして抽出
- スプリントバックログを時間単位まで見積る
  - Sprintでの作業時間内に完了するか確認しSprintでのプロダクトバックログを確定

③：ディリースクラム

- 以下の事を各自発言する.
  - **昨日の事**
  - **今日の予定**
  - **今抱えている問題**
- 発言の内容がタスクボード上のスプリントバックログ/タスクを示しているか確認する.
- チームイベント，勤務予定等のメンバーの予定について共有する.

- 単なる情報共有の場ではない.計画の確認と見直しの場.
  - **感想や所感を共有する場から各自がスプリントバックログ/タスク化を短時間で判断できるようにする.**
  - **各自が他者の発言に対して検査出来るようにする.**
  - **各自が自然と報告の問題の掘り下げを意識出来るようにする.**

④：Sprintレビュー

- バックログ一覧の確認及び会議の目途時間の決定
- 動くソフトウェアを用いてプロダクトバックログのデモする.
- Sprintレビューでプロダクトバックログの受け入れ条件と照らし合わせる.
- 参加者が利用者となって仮説を持ってレビューする.レビューの結果改善を促す機能要望はプロダクトバックログとして抽出する.これは不具合としない.

- 最後にプロダクトバックログ毎の Done/Not Done を読み上げる.

⑤：振り返り

- 会議の最初に会議内でのタイムスケジュールを示しタイムボックスを意識させる.
- 振り返りではKPTを実施する.特に初期は振り返り方法を定着させる.
- KPTのProblemについて,問題と事実を分けて整理出来るチームかどうかで KPTのファシリテーションをコントロールする.
- プロダクトバックログの実績ストリーポイントを算出させて,解離原因の改善を促す.
- Agenda
  - レトロスペクティブのやり方説明（5分）
  - 実績ベロシティの確認(5分)
  - 前回 Try の確認 および KPT の棚卸（5分）
  - Keep, Problemの抽出（各自：10分）
  - 各自 Keep, Problem の説明（10分）
  - 問題の掘り下げ（20分）
  - Try の確定（10分）

⑥：プロダクトバックログ

プロダクトバックログ本体

[ペルソナ]は
[機能]が欲しい
そうすれば
[ペルソナ]は
[効果]が出来るようになる

- ビジネス側の言葉で記載してペルソナのシナリオを明確にする. 最低粒度のシナリオ.
- [ペルソナ]が利用する[機能]に加えt[効果]を記載する.
- [効果]は[機能]によって解決したい事を記載する. 特に[機能]が実現できた事によってペルソナが出来るようになる"活動"を示す.
- <主語> want to ～ so that, <主語> can(will, may) <効果> が本来のテンプレート

受入条件

- 空文字のとき○○である事
- XXを入力し忘れたら○○となる
- 5秒以内に○○が表示
- 1000件までのデータを表示する

- プロダクトバックログを充足するための条件を記載する.
- POが拘っている仕様を記載する. その内容は受入時にPOが確認すべき仕様になる.
- 「～のとき」「～たら」「○○まで」等の表現で仕様に対する条件・範囲を示す.
- 非機能要件は[受入条件] として表現する.具体的な数字, 条件を明確にする.
- 最低3～5個の[受入条件]を記載する.過度に[受入条件]が多い場合には, プロダクトバックログに対して価値が多数混在していないか確認する.

⑦：開発ツール・アイテム

開発ツール

- タスクボード:手書き, JIRA, Redmine等. プロジェクト事情に合わせたバックログ状態管理が出来るものを選ぶ.
- Git：Scrumでは NOT DONEになった場合に,特定の機能をリリースから外す事がある. Git Flow による Featureブランチ戦略によりリリースから外れたコード管理が容易になる.
- チャットツール:特に複数拠点で開発する場合の有効. サンプルコード提示やファイルパスの共有など

アイテム

- プランニングポーカー
- 付箋, ペン, ホワイトボード, イーゼルパッド
- プロジェクタ, 差し棒

実施結果

案件概要

- 頻繁に要件の優先度が大きく変わる案件であったので,より変化に適応しやすいScrumを採用した.
- Scrum導入前までに基本機能の開発を実施. Sprint 1～Sprint7までは基本機能に対しての追加開発, Sprint8以降を別の新規サブシステム開発として開発を実施した.
- Sprint は2週間で固定して実施した.

実施結果(品質評価)

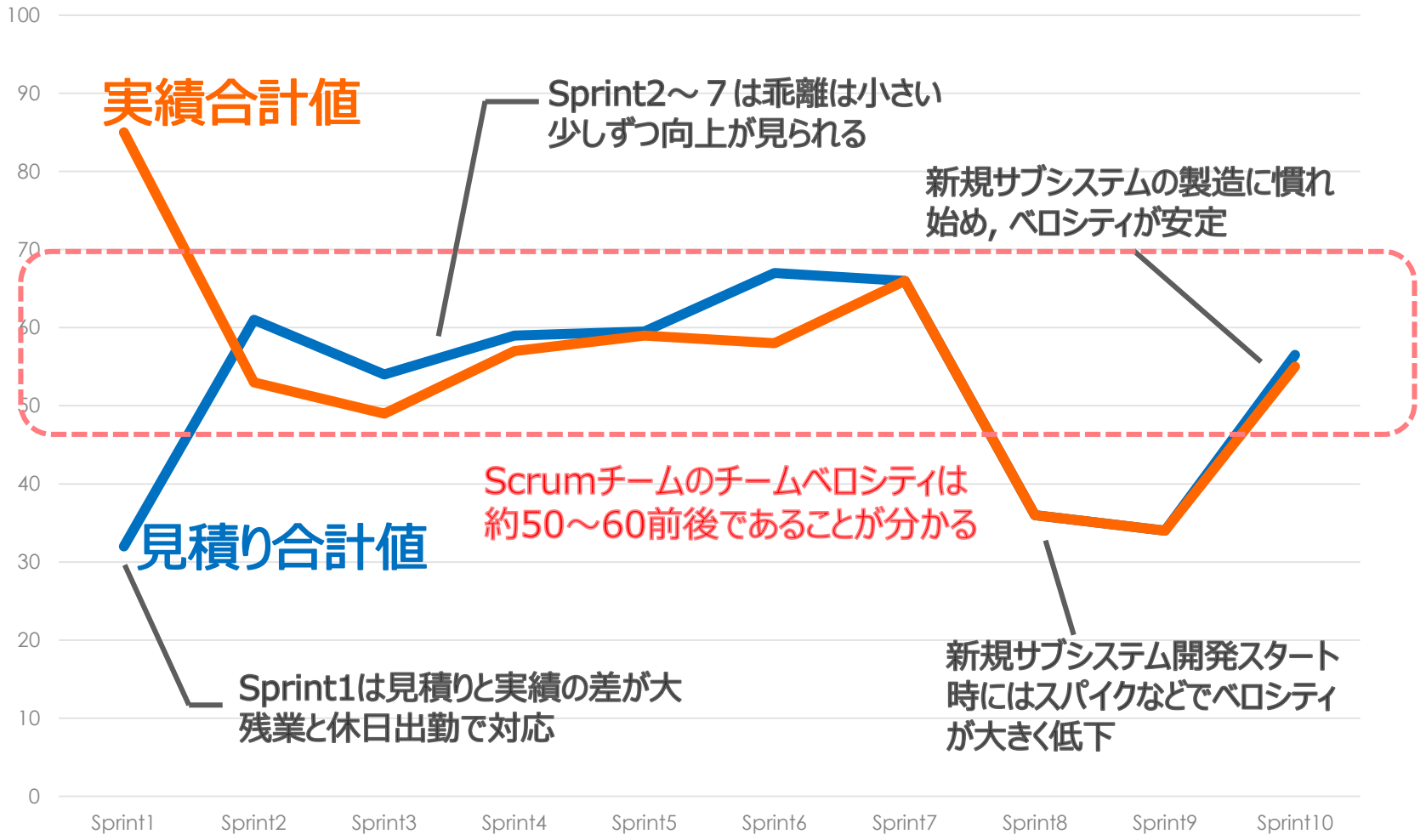
	対象	期間	母体	改造	UT		IT		ST	
					試験密度	バグ密度	試験密度	バグ密度	試験密度	バグ密度
2015	サイクル1	2ヶ月	6.7	0.8	277.2	5.0	65.6	2.0	14.4	1.2
	サイクル2	2ヶ月	9.4	1.9	177.6	3.6	56.6	1.0	36.8	0.0
	サイクル3	2ヶ月	16.4	2.3	117.0	0.9	25.1	0.9	10.3	0.9
	サイクル4	2ヶ月	4.6	0.5	193.1	0.0	51.1	0.1	27.2	0.0
	サイクル5	2ヶ月	9.9	0.8	174.5	2.4	39.0	1.2	26.3	0.0
	合計	10ヶ月	47.0	6.5	187.8	2.3	47.5	1.0	23.0	0.4
2016	sprint1	2週間	8.3	1.7	119.0	1.7	49.4	0.0	11.4	0.1
	sprint2	2週間	9.8	0.7	75.8	5.0	23.7	0.0	17.5	0.0
	sprint3	2週間	5.8	1.0	137.4	2.8	40.4	0.9	9.4	0.1
	sprint4	2週間	3.5	0.6	183.0	1.4	73.7	0.0	27.7	0.0
	sprint5	2週間	7.3	0.8	109.1	2.6	30.1	0.0	16.6	0.0
	sprint6	2週間	4.6	1.7	193.4	1.7	61.9	0.1	55.7	0.0
	sprint7	2週間	2.4	0.4	67.5	0.0	28.7	0.0	28.7	0.0
	合計	3.5ヶ月	41.7	↑7.2	↓126.5	↓2.1	↓43.9	↓0.1	↑23.9	↓0.1

実施結果(品質評価)

結果(品質評価)

- 試験密度についてSTが増加,UTとITが減少した.原因は前半のSprintではUIの変更が多く,実機確認を優先した事に起因する.またUTとITの実施方法について観点が被っている部分の削減などをしたことで作業のムダを省いた.
- バグ密度について全ての工程で減少した. Scrumの実施に伴い試験環境へのリリース作業がScrumチームに移管されたことでリリースが頻繁にできるようになり,試験実施前にバグを潰すことが影響したと考えられる.
- 相対的にはM/UTの実施時間が増えた.Scrum導入によって仕様検討,設計と進めるのではなく,M完了後に動作するアプリケーションで細かい部分をチェックした後で,設計書作成との順番になった事による設計書等の手戻り修正等が減少した事が原因.

実施結果 (ベロシティ推移)



NTT DATA

実施結果(アンケート結果)

- タスクボードやディリースクラムの実施によりコミュニケーションが活発になった.
- 開発チームへの責務を増やすことで,自分達で考え解決する力が向上.
- 外部仕様のチェックはSprintレビューで実施し, それまでの実装方法や進捗管理はチームに任せたことで能動的な動きができるチームへ成長した.
- 会議の頻度が多くない方がよい. あまり多すぎると開発チームの負担になり本来実施すべき開発作業に時間が取れなくなる.

結論・課題

結論

- Scrumで効果的なプラクティスを導入し，Agileにおける開発の生産性に加えて開発者の主体性の向上も見られた．これらのやり方が適切である事が確認できた．

課題

- 従来型の開発との違いが幾つか課題として出ている．
  - **短期間で開発することで開発チームは疲弊しやすい．**
  - **DEVからの提案が受け入れられないとモチベーション低下が一部起きていた．**
  - **ある程度,一人称で作業が実施出来るメンバーが揃っていないと開発に貢献しづらい．**
  - **Agile, Scrum だけで全ての問題が解決出来てしまうという先入観がある．**
- Scrumはどの開発プロジェクトにも適合出来るわけでない.その一方で開発プロジェクト以外のビジネススタートアップ事業に向いているという意見もある．どのようなプロジェクト特性がScrumに向いているかを分析する事に価値があるのではないか．

NTT DATA

Global IT Innovator