特集号
招待論文

アジャイル開発プロジェクトマネジメントに対応する人材育成
―改善によって成長を期待するマネジメント―

松浦 豪一[†1]

[†1] (株) 富士通マーケティング

近年，さまざまな変化により，アジャイル開発の適用が求められている．しかし，現場ではアジャイル開発に適した適応型プロジェクト・マネジメントが理解されないために，アジャイル開発に失敗する．その失敗談などから，アジャイル開発を敬遠する事態になっている．アジャイル開発に合った計画，適応型プロジェクトマネジメントを理解した上で，アジャイル開発の価値観を体感できる人材を育成する必要がある．富士通では実践者の経験を活かした人材育成により，アジャイル開発を展開し，リードタイムの短縮やコスト削減を実現した．そして，ワークショップ型教育にてアジャイル開発を疑似体験させることによって理解を深めている．本稿では，適応型プロジェクトマネジメント，適用時の効果，および，富士通での展開方法について説明する．

1. はじめに

近年，社会環境の変化や顧客のリテラシの変化により，ITシステムには要求に対する早い解決が求められている．アジャイル開発とは，利用者が検証可能なプログラムを定期的に提供し，フィードバックを得ることによって改善するソフトウェア開発手法である．この手法によりプロダクト開発のリスクを早期に低減させ，不確実性や変動性への対応を向上させる．アジャイル開発は変化に対応する1つの解になると考えられている．

しかし，日本国内ではウォーターフォール開発のマネジメントは定着しているが，アジャイル開発のプロジェクトマネジメントは定着していない．Scrum Allianceが公表している2012年3月時点の情報では，日本のScrum認定者数は米国の160分の1であり，Scrum Allianceの拠点がある米国や英国以外の調査対象国と比較しても10分の1前後であり，国内にアジャイルが普及していないことが分かる[1]．コスト削減などの効果を期待してアジャイル開発を適用しているが，プロジェクトマネジメントがうまくいかず実践した人が納得する結果を得られていない．非ウォーターフォール型開発の普及要因と適用領域の拡大に関する調査 調査概要報告書[2]によると，実践した人の63%がうまくいった面もあるがうまくいってない面もあると答えている．このような事態から誤解が広まり，現場がアジャイル開発の適用を敬遠する一因となっている．

富士通では社内システムの開発やパッケージ製品の開発をアジャイル開発で実施している．また，アジャイル開発実践者の経験を利用し，アジャイル開発に適応したプロジェクトマネジメントを教育プログラムとして展開している．アジャイル開発は期間とコストを固定し，スコープを変更しながら開発を行う．計画に従うのではなく，自律的に改善する行動が開発チームに求められる．自律的に改善するためには，適応型プロジェクトマネジメントに合わせたチームビルディングが必要になる．そこで，模擬プロジェクトを実施するワークショップ型教育を実施する．その結果，アジャイル開発を実施すると直面する問題を体験し，改善することを学ぶことができる．

本稿では，アジャイル開発を実践するための適応型プロジェクトマネジメントに関してチームビルディングを中心に，適用時の効果，および，富士通での展開方法について説明する．なお，本稿の主題ではない以下の内容は含まれていない．

- ステークホルダ管理
- バーンダウンチャートによる進捗管理
- 振り返り

2. 現状と課題

本章では，アジャイル開発を適用する上での課題について説明する．ウォーターフォール開発と異なりアジャイル開発は，計画に従うのではなく，動作するプログラ

ムを提供し，フィードバックに対応することを優先する．提供とフィードバックを繰り返すことにより，利用者と開発者が学習し，プロダクト品質を向上させる改善型のマネジメントである．2週間から4週間に1度の頻度で動作するプログラムを提供することを重視する．つまり，プロジェクトマネジメントやプロセスが異なる（表1）．ウォーターフォール開発の常識が通用しないため，経験が阻害要因になる．従来の常識を捨て，アジャイル開発を実践するためには，以下の3つの課題がある．

- アジャイル開発に合わせた計画
- 適応型プロジェクトマネジメント
- アジャイルに合わせた人材育成

ウォーターフォールのプロジェクトマネジメントはアジャイル開発を阻害する

2.1　アジャイル開発に合わせた計画

不確実性や変動性に対応するためには，アジャイル開発に合わせた計画および計画立案の方法を理解しなくてはならない．アジャイル開発に合わせた計画を実践する方法を以下のとおり説明する．

- ユーザーストーリーを使った開発計画
- 開発期間に合わせた開発項目の分割
- 相対見積りと理想時間見積り
- 不確実性に対応するためのバッファ算出
- 優先順位設定（価値とリスクの4象限）

2.1.1　ユーザーストーリーを使った開発計画

アジャイル開発では要件と開発を直結して開発するため，要件をユーザーストーリー[3]という形式にまとめる．

「私は○○○として，○○○がほしい，
　なぜなら，○○○だから」

この形式にまとめることで，利用者の明確化，要求の明確化，必要性の明確化を行う．さらに，ユーザーストーリーを大目標，中目標，および，小目標を作り階層化する．たとえば，パッケージ開発の場合，運用レベルのストーリー，機能レベルのストーリー，操作レベルのストーリーを図1のように作成する．図1のような計画を立てた場合，4週間の開発期間で機能レベルのインクリメンタル開発を行う．その過程で操作レベルのストーリーをどこまで作成するかを実装レベルとコストを加味しながら決定していく．4週間に1回計画を行い，今回の経験を活かして次回の計画を立案することを繰り返す．

ユーザーストーリーは，INVEST[3]に作成することを心構えとしている．INVESTとは5つの単語の頭文字をとった造語で以下の6つを表す．

- Independent（独立している）
- Negotiable（交渉可能）
- Valuable（価値を提供する）
- Estimatable（見積り可能）
- Small（小さい）
- Testable（テスト可能）

INVESTなユーザーストーリーを作成することで，要求を端的に小さく表現し，実物との比較でフィードバックしやすい状態を作り，開発チームが後からでも顧客にとって価値のある仕様を決める余地を残すことができる．

ユーザーストーリーに分割する場合，以下の2つの点に注意する．注意点は矛盾しているように見えるが，スプリント中に達成すべき目標を設定し，誰にいつ提供するといった運用方法についても合わせて検討する．

1）プロトタイプではないため，スプリント期間中に，実際にユーザが利用できる品質を確保する．たとえ

表1　ウォーターフォールとアジャイルの比較

	ウォーターフォール	アジャイル
事前予測	予見可能	不確実性と変動性が存在する
計画	最初に最後までをすべて計画	全体計画した上で実施直前に詳細化
見積り	• 要件定義・外部仕様設計，システム設計の段階で再見積り • 理想時間見積り	• 初回と繰り返し単位で見直し・相対見積りと理想時間見積りの併用
マネジメント	• 計画駆動型 • 稼働率を上げる • 変化は手戻り • 予定維持していることを管理する	• 適応型 • 仕掛かりを減らす • 変化を活かす • 異常を検知する
重点	予定どおりを維持することでコスト増加を防ぐ	提供を優先し，コストの範囲で内容を見直す
リーダ	トップダウン型	サーバントリーダ型
開発項目	優先度で決定しすべて実施する	優先順位で随時決定する．開発しない項目もあるし，計画にない項目も開発する

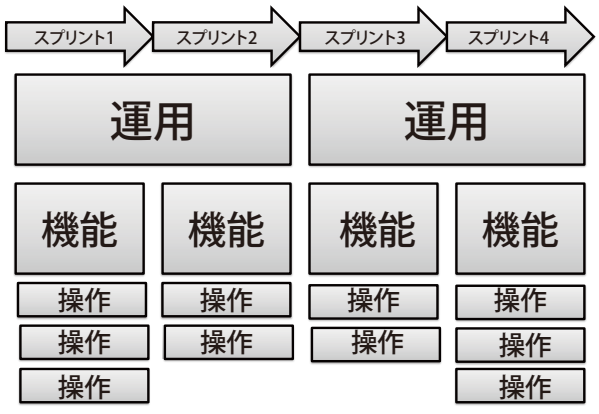

図1　階層化したユーザーストーリー

ば，操作だけ体験できるという状態ではいけない．

2）スプリントの区切りはウォーターフォールのようにテストが完了し稼働保証がなくてもよい．たとえば，一時的に機能が閉塞されデグレードしてもよい．

2.1.2　開発期間に合わせた開発項目の分割

アジャイル開発では，2週間から4週間を1サイクルとし，動作するプログラムを提供する．このサイクルをスクラム用語でスプリント[3]と呼ぶ．スプリントをまたがった開発は原則として行わないため，開発項目はスプリントの長さ以下になるように分割する．タスクに分解するときも同様で，1日の作業時間より短くするため，2時間から6時間くらいの長さに分割する．分割した例を図2に示す．

開発項目を分割すると，サイクルタイムを短くする効果がある．待ち行列の理論からも分かるように，サイクルタイムが短くなれば変化に対応しやすくなり，遅れの影響を局所化することができる[4]．

2.1.3　相対見積りと理想時間見積り

アジャイル開発では，相対見積りと理想時間見積りを併用する．プロダクトバックログは，相対見積りにて，ストーリーポイントを見積りする．スプリントバックログはタスクごとに理想時間を見積りする．

アジャイル開発ではチーム全員で計画し開発をするために，プランニングポーカ[5]を使ってメンバの認識をすり合わせながら見積りをする．

プロダクトバックログは．開発期間の全体を見据えたものである．開発当初にすべてを洗い出すことはできない上に，変動する可能性がある．確定させて見積りをすることができないため，ほどほどの情報で見積りができる相対見積りを利用する．図3のように基準に対して大きさが約2倍と見積りできる．個人の能力を基準にしていないので，メンバが変わっても使用でき，有効期間が長い[5]．

スプリントバックログは直近のスプリントの計画であり，タスクレベルまで洗い出すため，理想時間見積りを利用する．スプリントに実施する作業のため，時間レベルの精度で見積りすることが可能である．追加作業が発生し，スプリント完了までに目標を達成できるかをすばやく判断することができる．

図2　開発項目のタスク分割の例

結果をもとに繰り返し計画を行うが正確性を追求せず

2.1.4　バッファ算出

不確実性に対応するために，プロジェクトバッファを算出する．アジャイル開発の場合，概算見積りした状態から開発を始めるため，コスト管理にはリスクを伴う．そのため，ウォーターフォール開発よりも多めのバッファを見積りする．たとえば，概算見積りを1.5倍にし，30%を予備として保持する．このほかに50%の見積りと90%の見積りの標準偏差をとる方法がある[5]．これらをプロジェクトバッファとして管理し，開発項目の変更に伴う不足コストを補う形で利用する．

バッファを算出するのは，マネジメントを行うためであるが，バッファが小さい場合により計画重視の傾向が強くなるのを防ぐためである．顧客が変更を言いやすい環境を整える効果がある．

バッファは失敗の補填が目的ではない．要求の変化や価値の追求のために用意する

2.1.5　優先順位設定（価値とリスクの4象限）

準備期間に開発項目を洗い出し，図4にある価値とリスクの4象限[5]で優先順位を設定する．スプリントごとに見直しを行い，開発項目を変動させる．リスクが高いものから優先度を高めることによって，プロダクト全体のリスクを低下させようとしている．ウォーターフォール開発では．優先順位を初期段階で確定させるためにリスクや価値などを絶対値として数値化する場合がある．

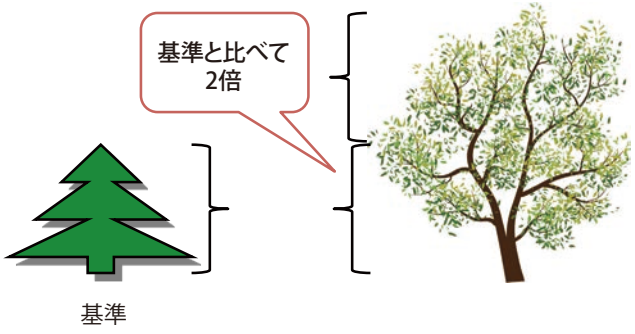

図3　相対見積りの考え方

最初に決めた順位は変更できないため，変更しないための根拠が必要だからである．アジャイル開発の場合，優先順位は変化することを前提としている．計画した時点での順位を決めるために利用するので，複数のリスクを比較してどちらが高いかを決めればよい．短期間に繰り返し計画をするやり方では相対関係で決定したほうがコストも少なくできる．

アジャイル開発は，リスクの高いものから開発し，早期検知する

2.2　適応型プロジェクトマネジメント

アジャイル開発を適用するためには，適応型のプロジェクトマネジメントに変更する．

以下の方法に取り組む必要がある．それぞれについて説明する．

- 管理機会を多くする
- 開発項目の単位を変更する

アジャイル開発は，提供とフィードバックを短期間で繰り返し行う．異常はできるだけ早く検知する必要がある．また，小さい作業単位で依存関係の少ないマネジメントができれば，追加や変更に対応しやすくなる．

2.2.1　管理機会を多くする

リスクを早期に検出することを目的とし，管理機会を多くするために図5のように計画ミーティング，朝会，ペアプログラミング，スプリントレビュー，および，振り返りを行う．会話する機会が多くなり，リスクを早期に検出できる．

適応型プロジェクトマネジメントでは，早期に異常が分かるほうがよく，そのような状態を保ち続けるようにする．具体的には，多くのプロジェクトメンバが一つひとつの開発項目にかかわるようにマネジメントする．参加する人数が増えれば会話する機会が多くなる．共有のタイミングが増え，問題の早期発見につながる．

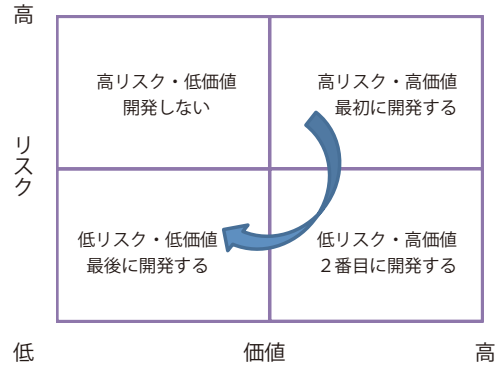

図4　価値とリスクの4象限 [5]

2.2.2　開発項目の単位を変更する

変動性に対応するために開発項目の単位分割し，割当て方法を変更する．限られた範囲内で異常を早期に検知することによりリスク対応が早まる．また，間隔が短いため，変更をするまでの時間が短くなり，変化に対応しやすくなる．施策は以下の2つであり，図6のように変化する．

① 開発項目を期間に分割し，優先順位どおりに割り当てる．

② 仕掛中の作業（Work In Progressといい，WIPと略す．以降WIPと記載する）を減らし，タスクレベルでの

	1日目	2日目	3日目	4日目	5日目
スプリント1	朝会	朝会	朝会	朝会	朝会
	振り返り	開発	開発	開発	開発
	スプリント計画	開発	開発	開発	スプリントレビュー

図5　一週間の流れ

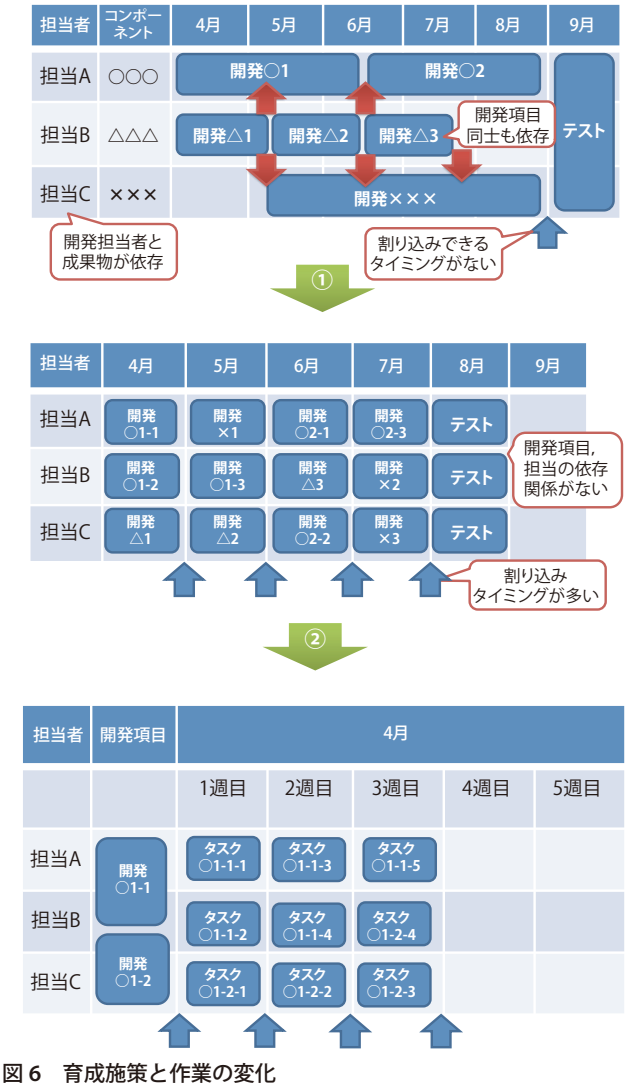

図6　育成施策と作業の変化

分割を行う（図6の例では4月のWIPを3個から2個に変更している）.

【効果】

- 実施する単位を小さくすることで割込みが入るタイミングが多くなり，優先順位の変更までの時間が短くなる.
- 定期的に提供を行うことにより，1個の処理時間が短くなるため，遅れの影響が小さくなる．逆に1個の処理時間が長く，1列待ちになっていない場合，待ち行列で待たされる時間が長くなる.
- 人が変わることにより，相互の作業をチェックするようになる.

【適用時の工夫点】

やり方を変更したときにいろいろな人がかかわるように運営し，人が変わることによって異常を検知しやすくする.

- 連続している作業は別な人の方がよい.
- 割込みのタイミングで優先順位を変更する.
- 区切りのタイミングやそれより早い段階で問題がないかをチェックする.

コストより提供を優先するため，
人の稼働を増やすより仕掛かりを減らす

2.3　アジャイル開発に合わせた人材育成

適応型プロジェクトマネジメントを実施するためには，アジャイル開発に合わせたチームビルディングをする必要があり，合わせて人材育成することが課題になる．従来どおりのプロジェクトマネジメントを実施すると，アジャイル開発に合わせたチームビルディングができない．理由は以下のとおりである.

- アジャイル開発を始めるために，どのプラクティスが必要か判断できない.
- ウォーターフォールの常識で考えるため，アジャイル開発の価値観を理解できない.
- アジャイル開発を実施すると，計画を変更する問題が頻繁に発生するが，状況を受け入れられないため，プロセスが悪いと考えてしまう.
- 開発者が稼働することを優先するため，仕掛中の開発項目が多くなり，優先順位どおりに開発できない.
- 期間に合わせて機能を減らすような考え方をしたことがないため，期間に合わせてスコープを調整できない.
- 再帰的にテストできないため，デグレードを防止できない.
- 仕様を決定し仕様どおりに作ることが習慣になっており，利用者を想定した開発ができない.

上記のようなことを理解していないため，開発者全員で計画し，全員で自律的に開発することができない．アジャイル開発のメリットを説明されても必要性が理解できない．そのため，模擬プロジェクトで実施し，失敗して改善することで，適応型プロジェクトマネジメントを理解させ，人材育成をする.

3.　富士通での実践と展開方法

富士通内でアジャイル開発を実践したプロジェクトのノウハウをワークショップ型教育で展開している．アジャイル開発に適したマネジメントを適用した事例，効果，および，教育での展開方法について説明する.

3.1　プロジェクトへの適用事例と効果

製品保守開発と新機能開発を同時に行うパッケージ開発プロジェクトに適応型プロジェクトマネジメントを適用した．プロジェクトマネジメントの実例について説明する.

3.1.1　プロジェクトへの適用事例

【ケース1】要求の価値やリスクが不明確な状態

複数の開発項目の価値やリスクが不明確な場合は，いったん，開発項目の一部を並行に開発する．たとえば，WIPを3にして作業する．部分的に価値が判明し，優先順位をつけられる状態になったら，WIPを1にして，1つずつ開発する．図7のように変化し，開発Aは第3週に，開発Cは第4週に開発Bは第5週に提供できる.

【ケース2】期限が短い開発が追加になった場合

緊急の割込保守開発が発生した．短納期であり，保守担当の人数では納期までに提供できない場合，一次的に保守枠を超えて開発し，超えた分に関しては別の開発でリカバリをする．図8のように変化し，具体的には，1人で開発すると1カ月かかる保守開発を1週間で提供した.

【ケース3】開発項目を中断して遅らせる場合

パッケージ製品の開発項目は，リリースに間に合わない開発は後回しにして，保守開発を先行させることがある．提供可能な機能の開発に開発項目を切り替える．図9のように変化する.

3.1.2　プロジェクト適用への効果

適応型プロジェクトマネジメントを適用したことによ

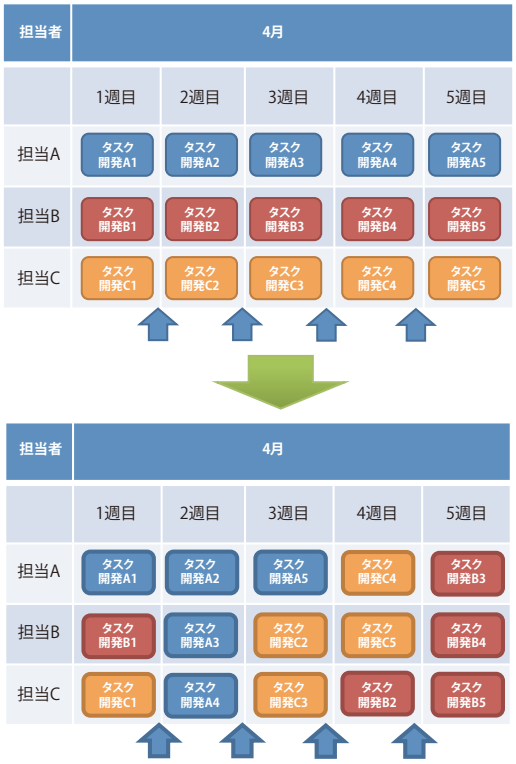

図 7　アジャイル開発に合わせた順序の変更

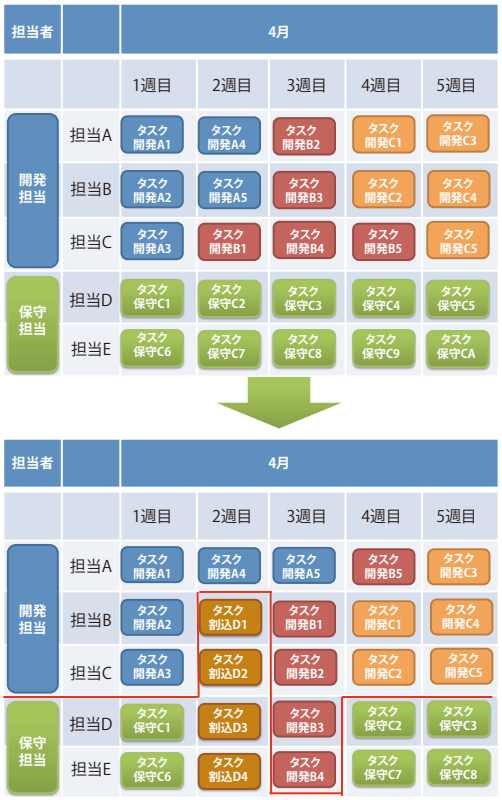

図 8　優先順位の変化による対応

図 9　優先順位および開発項目の変化

タイムボックスで異常は検知できるので，
マネジメントが流れを変える．
変化に対応してチームは成長する

り，すばやく価値を提供することができた．具体的な成果は以下のとおりである．

- 優先度の高い保守開発案件が希望納期で完了した．
- 新規開発についても予定納期前に完了した．
- 全体予算に対してコストを 10% 削減した．

3.2　富士通展開方式

富士通ではアジャイル適用を実践したノウハウを活用し，アジャイル開発実践経験者によるアジャイル教育を進めている．アジャイル開発を適用したプロジェクトは初期段階に課題が発生するため，開発の初期段階に経験者による支援を行っている．アジャイル開発を疑似体験できるワークショップを行い，適応型プロジェクトマネジメントの経験を体験する．実プロジェクトで発生した事象を擬似プロジェクトで発生させる．本教育は，月に 1 回から 2 回程度実施しており，1 回あたりの受講者は 6 名から 18 名程度である．2014 年度の教育受講者が 100 人で，年間の受講者を 200 人を目標に活動している．

3.2.1　ワークショップ研修での体験

スクラムを利用したワークショップ研修の中で，開発中に発生しがちな状況を再現させる．プロジェクトマネジメントを理解することができ，コミュニケーションが活発になり，チームの成長を促す方法を理解することができる．研修による期待効果について状況別に説明する．

【状況 1】優先順位どおり開発できない

ウォーターフォール開発のように複数の開発項目を同時並行で行うため，優先順位の低いものが先に完了する（優先順位の低い項目も同じ納期を設定し，両方を同時に開発するように促す）．

優先順位の高いものを提供するために，マネジメントの方法を変更する必要があることを理解できる．

【状況2】繰り返し開発でのデグレード

開発項目同士に関連性を持たせ，既存の資産を壊す開発項目を仕込んでいる．

アジャイル開発は繰り返しプログラムを修正するため，修正によってデグレードを引き起こす可能性が高いことを理解できる．

【状況3】利用者への考慮が足りない

開発者は要件を満たしていると判断したものに対して，利用者が実際利用した場合に発生する指摘をする．

スプリント開発は，直近の目標に対して開発をするため，視野が狭くなりやすい．タスクに集中してユーザーストーリーを見落とすことがある．このような体験をすることで，アジャイル開発の目的を見失うことなく開発を進めることができる．

【状況4】計画を変更する

スプリントごとに開発項目の優先順位を変更する．もしくは，追加の開発項目を入れる．

開発項目が計画どおりであったら，アジャイル開発を適用する必要はない．スプリント計画時に計画を見直すことを実践することで適応型プロジェクトマネジメントを理解する．

【状況5】すべては予測できないことを体験

スプリントの最中に計画にない作業が必要なことに気づかせる．

問題点を見つけて改善する方法を考えるきっかけを与える．問題をチームの問題として捉え，行動を変える施策に取り組む意識を高める．

【状況6】ペアプログラミングの価値を知る

開発者を複数人にして，1人作業とペアプログラミングの2つを体験させる．

スクラムを利用して運営するためには，管理機会を多くすることで異常が発生しやすい状況を保つ必要がある．ペアプログラミングもその機会の1つである．すべてのメンバがペアプログラミングをしてしまうとメリットが理解できないが，1人作業を与えることにより，ペアプログラミングの重要性を理解できる．

参加者の人数や作業の状況に合わせて，問題を発生させ，体験してもらう．発生した課題を解決するところまでを体験する．

3.2.2　教育から展開の考察

経験者のノウハウをワークショップ型教育で体験することにより，効果的にアジャイル開発を適用できる．考察内容は以下のとおりである．

- 必要最小限のプラクティスが明確になり，準備が容易になる．
- チームメンバに用語の齟齬や価値観のずれが少なくなる．
- 問題が発生したときに自責に考えることができ，最初から異常を改善することができる．

その結果，全員受講しているプロジェクトは受講していないプロジェクトに比べ自律的に改善できるようになるまでの期間が短い．教育を全員が受講していないプロジェクトでは，プラクティスの検討で1カ月間かかった．また，1通りのプラクティスを実施し，自律的に作業できるようになるのにさらに2カ月かかった．しかし，全員が受講しているチームは，翌日から開始することができ，1カ月程度で自律的に活動できるようになった．

自律的に改善することを期待して，アジャイル開発を始められるようにする

4.　おわりに

富士通のアジャイル開発を実践している現場では，社内，社外での事例紹介に力を入れている．事例紹介をすることにより，自分たちの活動を振り返るとともに成果を体系的にまとめる．その結果，チームのコンセプトが明確になり，暗黙知の表出化が行われる．また，社内外の交流により，実践者が成長し，高いレベルを目指すきっかけを与えている．

SI部門，サービス部門，プロダクト部門，およびパッケージ開発部門など教育を受講する部隊は多岐にわたっている．開発物の内容や目的が違うことから，同じアジャイル開発になることはない．すべてのプロダクトに対応した教育を展開することは難しい．そのため，すべての部門に対応した教育を展開することを目指さず，共通の教育で基本動作を教え，開発現場に出向き支援で改善していく方法をとっている．部門をまたがった交流や事例紹介をすすめている．集まった情報をもとに，教育プログラムや提供するコンテンツを充実させていく必要がある．

上記のような取り組みを通じて，アジャイルの知識を持っている人を増やすのではなく，アジャイル開発を体験している実践者を増やしていく．

参考文献

1) (独) 情報処理推進機構 技術本部 ソフトウェア・エンジニアリング・センター：非ウォーターフォール型開発の普及要因と適用領域の拡大に関する調査～非ウォーターフォール型開発の普及要因の調査～調査報告書，平成 24 年 6 月 11 日（2013）．
2) (独) 情報処理推進機構 技術本部 ソフトウェア・エンジニアリング・センター：非ウォーターフォール型開発の普及要因と適用領域の拡大に関する調査 調査概要報告書，平成 24 年 3 月 28 日（2013）．
3) ケネス・ルービン（著），岡澤裕二，角　征典，高木正弘，和智右桂（翻訳）：エッセンシャルスクラム，（株）翔泳社（2014）．
4) メアリー・ポッペンディーク，トム・ポッペンディーク（著），平鍋健児，高嶋優子，佐野建樹（翻訳）：リーンソフトウェア開発，アジャイル開発を実践する 22 の方法，日経 BP 社（2004）．
5) Mike Cohn（著），安井　力，角谷信太郎（翻訳）：アジャイルな見積りと計画，価値あるソフトウェアを育てる概念と技法，（株）毎日コミュニケーションズ（2009）．

松浦 豪一（非会員）m.hidekazu@jp.fujitsu.com
1986 年富士通静岡エンジニアリングに入社し，基本ソフトウェアの開発に従事する．2007 年 TPS（トヨタ生産方式）の KAIZEN 伝道師の教育を受け KAIZEN 活動の伝道を行う．富士通の ERP パッケージ GLOVIA の製品開発をアジャイル開発で行う．現在，（株）富士通マーケティングにて ERP パッケージを利用した SI 構築をしながら，富士通アジャイル実践センターからの依頼のもと，アジャイル開発の教育講師および，アジャイル開発現場の支援を行っている．

採録決定：2016 年 4 月 11 日
編集担当：藤原陽子（レノボ・エンタープライズ・ソリューションズ（株））