

アジャイルプラクティス実践記録

～新人がアジャイルに携わるうえで何が必要か～

東京海上日動システムズ（株）

■ 執筆者Profile ■



2017 年 東京海上日動システムズ（株）入社
アプリケーション開発部配属

2018 年 現在 クリエイティブデザイン部所属

平原 みちる

■ 論文要旨 ■

筆者は、2017 年 10 月から 5 ヶ月間、アジャイル開発手法を適用したシステム開発プロジェクトに開発者として参加した。会社全体でも、アジャイルで開発した実績は少なく、手探り状態でのスタートだった。プロジェクト初期に生じた問題を解決するため、筆者のチームは 14 のアジャイルプラクティスを導入した。なかでも、「ペアプログラミング」「プランニングポーカー」が開発効率の向上という観点で特に効果的であった。また、筆者がプロジェクトに参加したのは入社 1 年目であり、実業務のシステム開発は初めてだった。新人社員が業務に携わるうえで重要なことは様々だが、アジャイル開発では「チームをより良いものにしていくという意識」「会話・発言する力」が特に必要であると体感した。本稿では、アジャイルプラクティスの実践記録、導入時のポイントと、新人社員がアジャイル開発に携わるうえで重要と感じたポイントについて論じる。

■ 論文目次 ■

1. はじめに	《 3》
1. 1 当社の概要と部のミッション	
1. 2 AOSORA開発プロジェクトの概要	
1. 3 プロジェクト体制	
2. アジャイルプラクティスの実践記録	《 4》
2. 1 ペアプログラミング	
2. 2 プランニングボーカー	
3. 導入効果と新人に必要なこと	《 7》
3. 1 導入効果	
3. 2 新人に必要なこと	
4. まとめ	《 9》
5. 所感	《 10》

■ 図表一覧 ■

図1 AOSORAビジネスモデル図	《 3》
図2 プロジェクト体制	《 4》
図3 導入したアジャイルプラクティスとねらい（抜粋）	《 5》
図4 バックログの重みづけと選別	《 7》
図5 バーンダウンチャート	《 8》

1. 1 当社の概要と部のミッション

東京海上日動システムズ株式会社（以下、弊社とする）は1983年9月に設立され、東京海上グループの情報システムの開発・保守・運用を担う。筆者が昨年度まで所属していたアプリケーション開発部、現在所属しているクリエイティブデザイン部は、デジタル時代の変化に柔軟に対応するため、アジャイル、DevOps などを取り入れ、東京海上グループシステムの内製開発に取り組んでいる。

1. 2 AOSORA 開発プロジェクトの概要

AOSORA 開発プロジェクト（以下、本プロジェクトとする）とは、IoT デバイス補償サービス構築を支援するシステム開発である。「AOSORA」の利用対象は、IoT デバイスを販売している事業者であり、デバイス補償サービスとは、販売しているデバイスが故障・破損した際に修理や交換を補償するサービスである。事業者がシステムを構築するには多大なコストがかかるため、簡単にデバイス補償サービスを開始できる「AOSORA」システムを開発するのが本プロジェクトである〔図1〕。AOSORA システムに類似するビジネスモデルは前例がないため、プロジェクト開始時点でビジネス部門・開発チームともに具体的にどんなものを作ればよいかイメージできなかった。そのため、短いスパンで要件定義から開発ができるアジャイルを取り入れた。

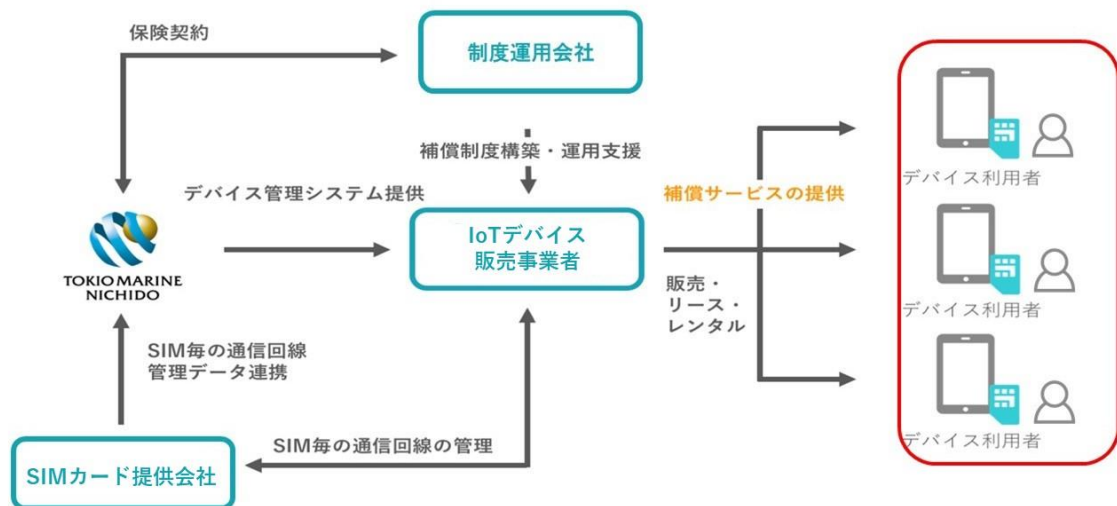


図1 AOSORA ビジネスモデル図

1. 3 プロジェクト体制

アジャイルプラクティスの導入効果を述べるうえで、本プロジェクトの体制を説明する。

「開発チーム」、「スクラムマスター¹」、「プロダクトオーナー²」でスクラムチームを組み〔図2〕、2週間を1スプリントとして全8スプリントの開発を行った。

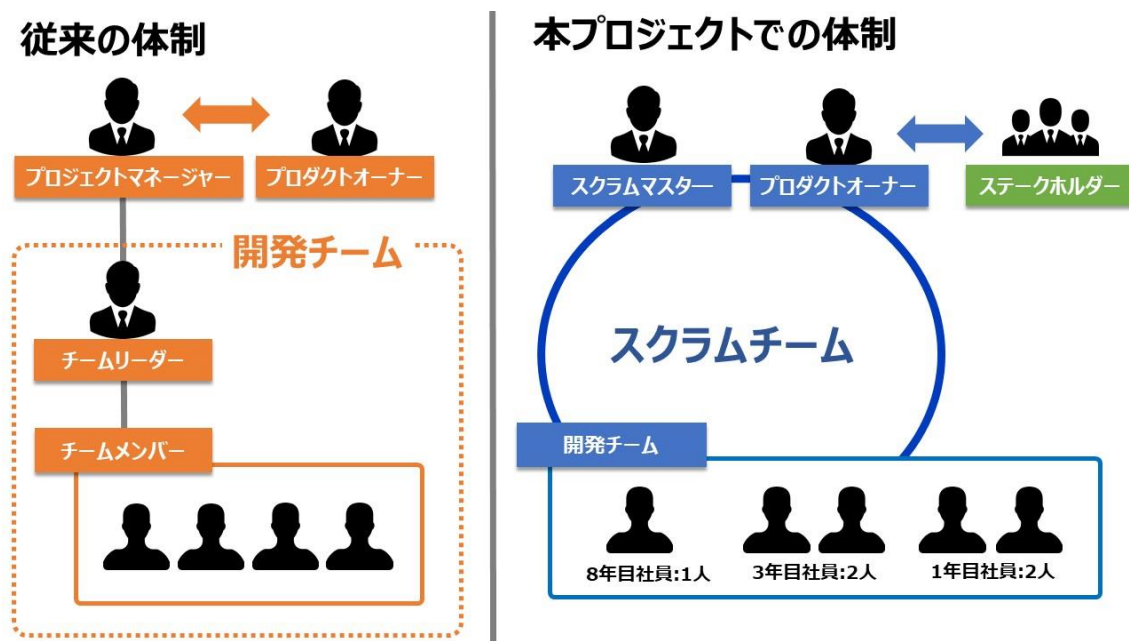


図2 プロジェクト体制

2. アジャイルプラクティスの実践記録

プロジェクトを短いスパンで効率よく進めるために14のアジャイルプラクティスを実践した〔図3〕。下記プラクティスのうち、特に有効と感じた「ペアプログラミング」「プランニングポーカー」をピックアップして効果を述べる。

¹ スクラムマスターとは、チームが目標を達成できるように障害物を取り除く（技術的な成長を促すような活動を出したり、そのリソースを確保するように動いたりする）役割のことである。

² プロダクトオーナーとは、チームが生み出す成果物の方向性を決め、成果物の受け入れ可否の基準を決める役割のことである。

プラクティス名	チームのねらい
デイリースクラム	朝会でその日実施するタスクを確認する
スプリントバックログ	1 スプリント（2 週間）で着手するタスクが何なのか明確にする
プロダクトバックログ	「何を優先して開発するか」を決める
バーンダウンチャート	日ごと、スプリントごとの進捗状況を把握する
レトロスペクティブ	スプリントを振り返り、次スプリントの目標や決まりを作る
ペアプログラミング	チームの技術力を向上させる
プランニングポーカー	見積もり精度を向上させる

図3 導入したアジャイルプラクティスとねらい（抜粋）

2. 1 ペアプログラミング

（1）導入の経緯とねらい

今まで導入実績のない言語やツールを使うため、開発チームのメンバーは未経験の技術を学習しながら開発を進めていく形となった。入社 1 年目のメンバーはプログラミングの経験も浅く、開発タスクを与えられても独力で完了させることは難しいため、技術力の差を少しでも埋めるべくペアプログラミングを採用した。

（2）用意したもの

40 型のディスプレイと、向かい合って座れる席を用意した。コーディングする人の画面をディスプレイに投影し、3 年目社員が 1 年目社員に教えながら交代でコーディングを進めた。

（3）成果

ペアの先輩社員にアドバイスを受けながらコードを書くことで、アルゴリズムの組み立て方やデバック方法をその場で教わることができた。プロジェクトの終盤には、導入実績のなかった言語やツールも扱えるようになり、独力で開発タスクを遂行できるまで成長し、「チームの技術力の底上げ」につながったと感じた。

またお互いがコードを確認しながらコーディングするので、レビューと同等の効果が得られ、レビュー時間が削減された。開発効率が上がるとともに、チームの生産性が向上した

と考える。

さらに、先輩社員のコーディング手法を観察することで、「数年後にはこのレベルになりたい」という具体的な目標が立てられた。「自分が3年目になるころには、先輩と同じくらい成長できるよう頑張ろう」と思えた。新人が組織の中で明確な目標を立て、成長していくための助けになるという面でもペアプログラミングは有効だった。

一般的に、ペアプログラミングは万人向けのプラクティスではないと言われている。しかしメンバー間の技術力に差がある場合、ペアプログラミングの効果は大きいと考える。特に、入社1年目のメンバーにとって、技術力が高い先輩社員とコードを書くことは非常にいい経験になり、自信にもつながる。チームの条件が当てはまる場合、ぜひ導入をお勧めしたい。

2. 2 プランニングポーカー

(1) 導入の経緯とねらい

スプリント開始時にユーザーストーリーを洗い出し、バックログ（要件）に落とし込み優先度をつけた。スプリント計画のなかで、1スプリント内で開発完了可能なバックログを正確に見積もり、優先順位をつけるために、プランニングポーカーを採用した。

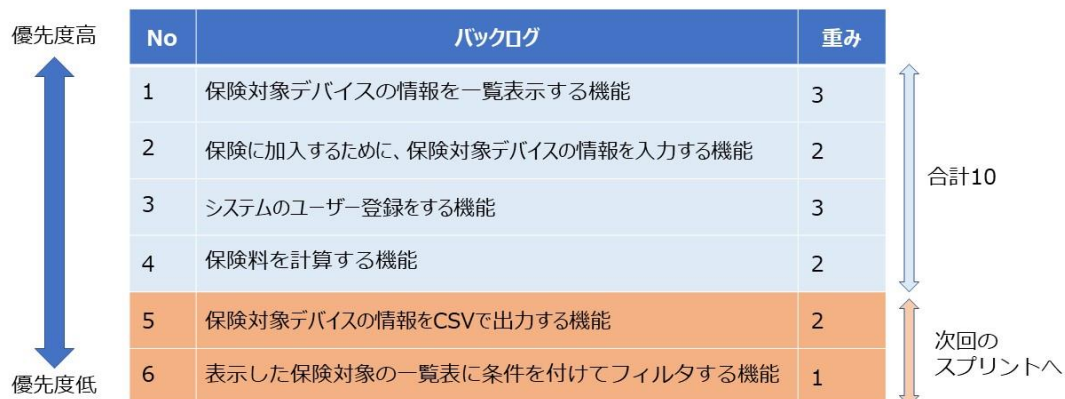
(2) 用意したもの

一般的にはトランプなどのカードを使うが、シンプルに「とりあえずやってみよう」という考えから、指を挙げる方式にした。

(3) 成果

プランニングポーカーでは各々が提示した数値に差が出ると、見積もり理由と根拠を説明する。例えば、あるバックログに対して筆者が「1」を提示し、「前回のスプリントで実施したものと似ているので、少なめに見積もった」と説明する。他のメンバーが「3」を提示し、「このバックログは、前回と違って〇〇という点も考慮しなければならないのでこの数値がよい」という説明をする。このように理由を共有することで「バックログに対して考慮漏れがあった」という気づきを得られ、さらに繰り返すことによって、認識齟齬の発生が減っていった。

導入前はスプリント内で完了できなかったものが、前述した話し合いによって、次スプリントの適正量が明確になった。導入後はバックログの重みが数値化され、見積もりと開発を繰り返すことで、「このチームが1スプリント内で開発完了可能なのは、重みの合計「10」である」ことも判明し、スケジュールが遅れなくなった〔図4〕。



バックログそれぞれを見積もり、「1スプリントで実施するもの」を優先度の高いものから決定していく。
重みの合計が10を超えたら、次回のスプリントに回す。

図4 バックログの重みづけと選別

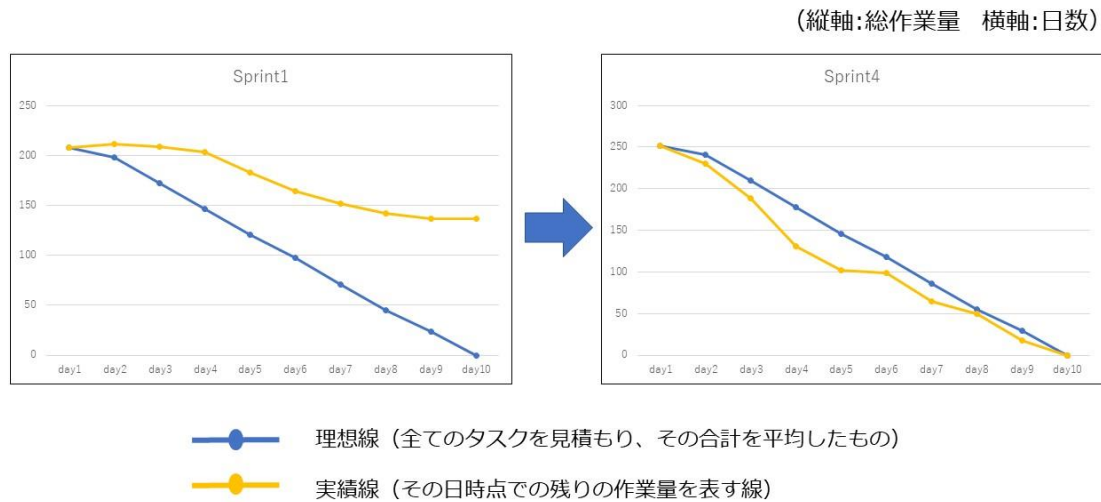
見積もり理由の中には、「自分は〇〇の知識が乏しく、この時間で完了できるか不安である」という意見も発言できた。余裕をもったスケジュールに見直したり、先輩がフォローするという形で進め方を変えられた。新人ならではの不安を打ち明け、現状の知識レベルをチームに周知する場としてもプランニングポーカーは有効であったと考える。

3. 導入効果と新人に必要なこと

本プロジェクトにアジャイルを導入した総合的な結果と、併せて新人がアジャイル開発のプロジェクトに携わるうえで必要と感じたことを説明する。

3. 1 導入効果

プロジェクト最初のスプリントと、アジャイルプラクティスを繰り返した4回目のスプリントのバーンダウンチャート [図5] を比較すると、バックログを消化するスピードへの効果がわかる。



スプリント1では進捗に遅れが出ていたが、スプリント4では計画通りにタスクが消化できていることがわかる

図5 バーンダウンチャート

チームの技術力が底上げされ、新人もタスクを消化できるようになった「ペアプログラミング」と、バックログに対する認識齟齬をなくし、手戻りを減らした「プランニングポーカー」を導入した。さらに、スプリント終了時に振り返りを実施し、次スプリントの目標を立てた。スプリントを重ねるにつれて効果が高まっている点については、こうした「繰り返し」の要素が功を奏していると考ええる。アジャイル開発では、要件定義から受け入れテスト、振り返りを繰り返す中で、アジャイルプラクティスの手法も少しずつ改善していく。長い開発期間で絶えず実践と改善を積み重ねるのは、アジャイルならではのことである。振り返りとアジャイルプラクティスの実践を繰り返し、改善し続けたことが高速開発の実現につながったのである。

3. 2 新人に必要なこと

(1) チームをより良いものにしていくという意識

アジャイル開発は、短いサイクルの中で改善を繰り返し、積み重ねることによって高速開発を実現する。アジャイルプラクティスは、改善を繰り返すための手段であり、メンバー全員が「チームをより良いものにしていこう」という意識を持つことがさらに効果を高める。新人だからといって「誰かがなんとかしてくれる」という受け身の考え方では、チームはいい方向に向かない。受け身の姿勢を捨てて、今のチームはどんな状況なのか、問題点は何か、改善するためにはどうしたらよいかを新人がアジャイル開発のプロジェクトに携わるときには常に意識する必要がある。

（２）会話・発言する力

アジャイルプラクティスは、すべてにおいて「会話」が重要であると筆者は考える。前述したプランニングポーカーは、自分の見積もり理由を話し、ペアプログラミングは、ペアのプログラマと会話しながらコードを理解し、より良いものにしていく。自分の考えを相手に伝え、また相手の考えを理解するという「会話する力」、「発言する力」も重要であると考ええる。チームをより良いものにしていこうという意識をもって、振り返りのミーティングや要件検討の場でも積極的に発言することが重要だ。アジャイルというスピードが最優先の環境の中で、「自信がない」「恥ずかしい」などといった感情によって発言しない、会話しなないことは、まったくもって時間の無駄である。

4. まとめ

入社１年目の新入社員には、身につけるスキルや大事な心構えがたくさんある。その中でも、アジャイルの「実践と改善を繰り返す」特性上、「チームをより良いものにしていくという意識」「会話・発言する力」が特に重要であるという学びを得た。同時に、「発言しやすい環境を作る」「努力を評価するチームメンバー」をチームとして用意することが重要である点に気づいた。

（１）発言しやすい環境

新人に必要なこととして「会話・発言する力」と述べたが、チームは発言しやすい環境を作ることも必要だと考える。チーム全員が発言しやすい環境である、先輩社員が発言の機会を作る、意見を否定せずにはまず聞いてくれる、このような環境がなければ、プラクティスを導入しても効果があげられなかっただろう。

（２）努力を評価するチームメンバー

新人とチームメンバーでは経験と知識に差があるため、「自分はプロジェクトの足を引っ張っていないか」と新入社員は不安になるだろう。筆者も実際に感じる事が多々あったが、努力を評価してくれるメンバーがいるとモチベーションが回復した。スクラムマスターが開発チームのメンバー一人ひとりと面談し、それぞれの不安などを聞き出し、その面談の場でねぎらいの言葉まで掛けられ、とても嬉しくなった。新人の努力を客観的に評価する人物がいるとチーム力は格段に上がるであろう。

今回新人でプロジェクトに参加し、自分自身そしてチームに必要なことの両方を学んだ。今後先輩社員としてアジャイル開発に携わるとき、今回の学びを活かして新人をサポートできるようにしたい。また、今回紹介した２つのプラクティスは、あくまで「新人である筆者が、導入してみて最も成長につながったもの」である。今後、有効活用することはもちろん

ん、新たなプラクティスも試し続けていきたい。

5. 所感

今回はじめてアジャイルに取り組んでみて、アジャイルとは「楽しく、あらゆる業務の改善を繰り返すものである」と理解した。アジャイルプラクティスを真剣に取り組むとき、チームには会話が増え、楽しさや明るさが生まれる。また、アジャイルやアジャイルプラクティスは「素早く改善すること」が基本であると考え。この考えは、システム開発だけでなくあらゆる業務においても導入できる。今後の業務で問題に直面した時、気軽に素早くアジャイルの考え方を応用し、改善を繰り返していきたい。

参考文献

[1]Rasmusson J.（著），近藤修平，角掛拓未（訳）：“アジャイルサムライ 達人開発者への道”，オーム社（2011）