

アジャイル領域へのスキル変革の指針

アジャイル開発の進め方

2020年2月

IPA 独立行政法人
情報処理推進機構

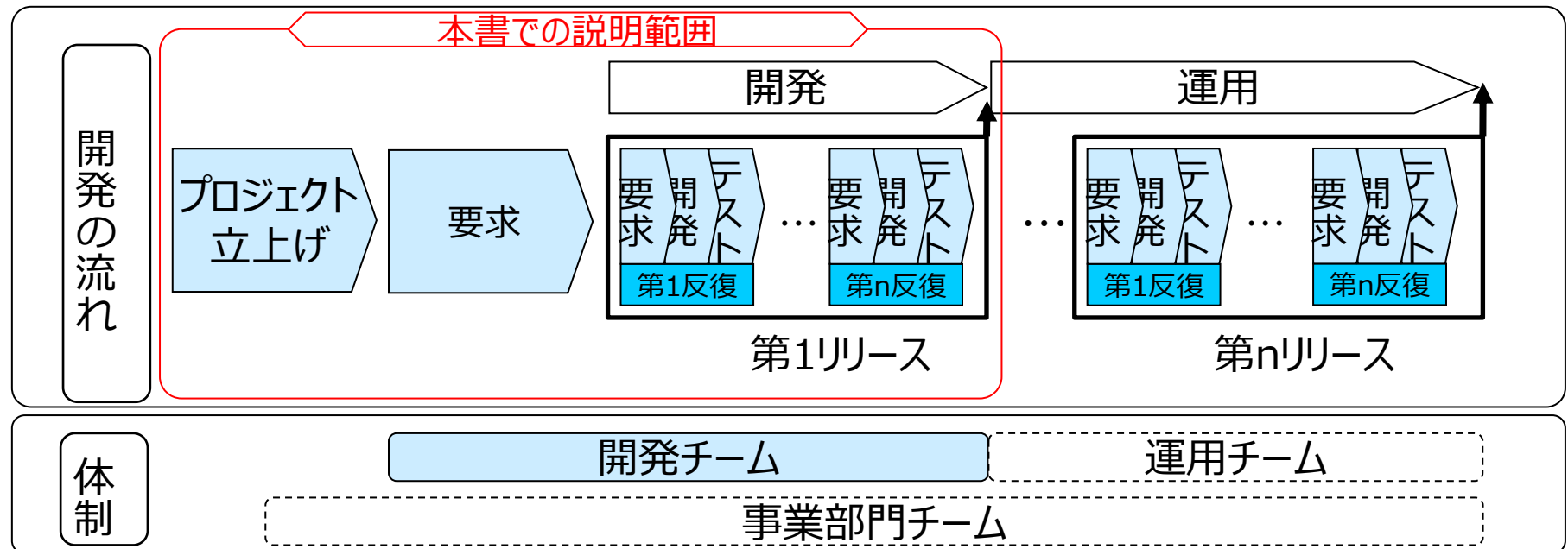
はじめに

- 本書は、アジャイル開発のプロセス、アジャイル開発チームにおけるメンバーの役割、および必要なスキルについて解説しています。
- アジャイル開発には複数のアプローチ（スクラムやXPなど）があります。本書では、代表的な手法であるスクラムを例にして、その特徴を解説しています。
- アジャイル開発の進め方には厳格な決まりごとや規範はありません。本書で説明（例示）する進め方、メンバーの役割（ロール）など、実際のソフトウェア開発プロジェクトでそのまま適用するものではありません。実際のプロジェクトや組織に適したやり方を取捨選択し、カスタマイズすることが必要となります。
- 「唯一の正しい」アジャイル開発というものはありません。自分のいる組織に合ったやり方が、その組織のビジネスや活動、文化から自然と育っていくのがアジャイル開発の本質です。基本的なことを書籍や外部の人を通じて学んだ後、組織内で自律的に推進できるようにすることが必要です。

本書におけるアジャイル開発のスコープと体制について（前提）

■ 本書での前提

ソフトウェア開発の進め方には、規模や開発方針の違いにより、いろいろなバリエーションがあります。本書では、アジャイル開発の基本的な進め方を理解するため、開発モデルとしてシンプルなるものを前提に考え、下図における「開発の流れ」中における「プロジェクト立上げ」～「開発」の範囲を説明対象としています。



プロジェクト立上げ時に開発チームを編成し、ユーザー側の事業部門内のチームと連携を図りながら、開発を進めていきます。比較的大規模な開発では、機能を設計開発するチーム（複数）と基盤・共通部を設計開発する共通的なチームを設置する場合があります。

目次

■ アジャイル開発のプロセス

- アジャイル開発のプロセス（スクラムの例）
- アジャイル開発の進め方の特徴（スクラムの例）
- 役割（ロール）の特徴（スクラムの例）
- 開発プロセスと役割（ロール）の関連（スクラムの例）

■ アジャイル開発チームのつくり方

- アジャイル開発チームのもつべきスキル
- スキル一覧

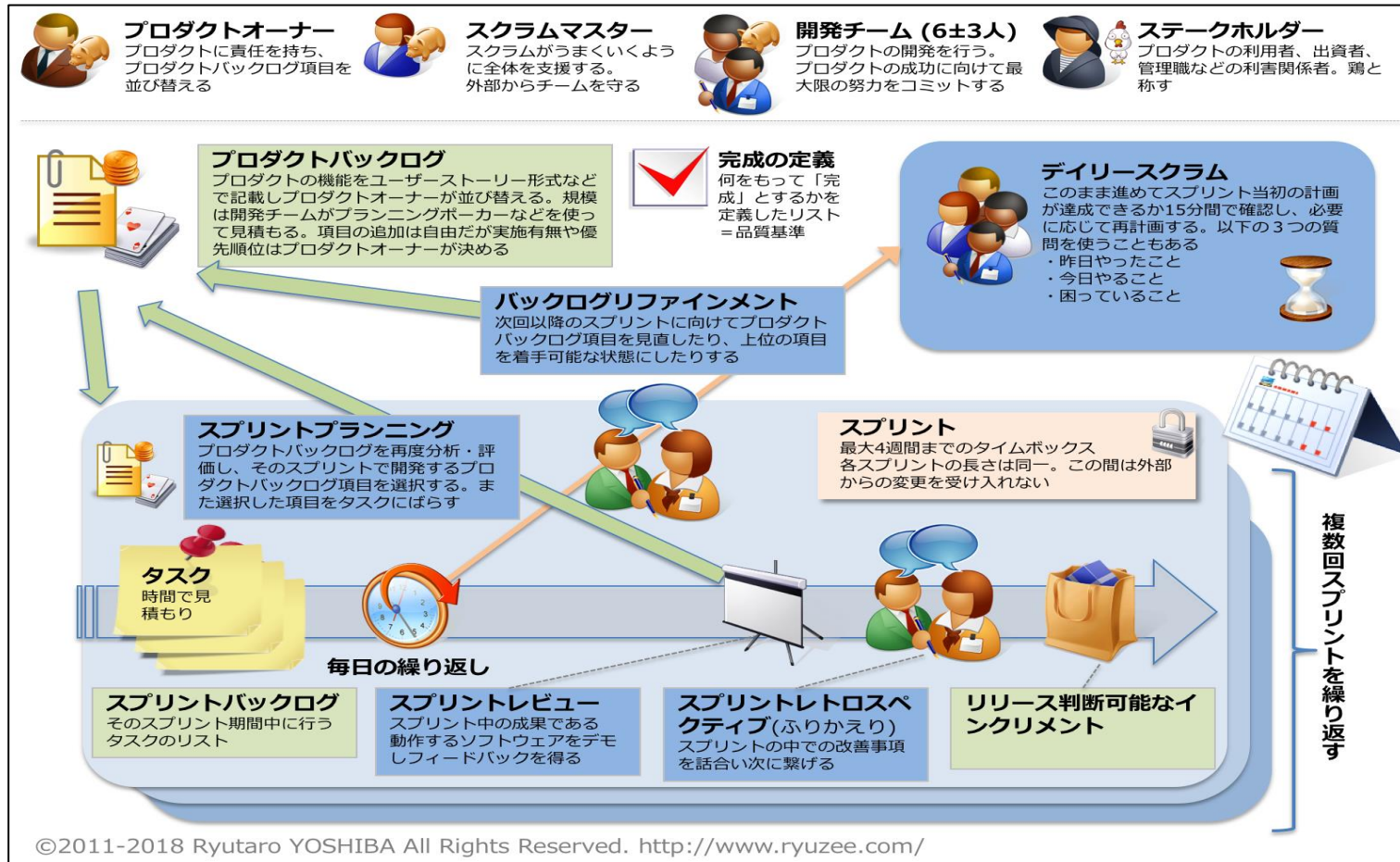
■ 参考資料

- <参考1>従来型ロールとアジャイル型ロールの比較表
- <参考2>アジャイル開発の概念整理

アジャイル開発のプロセス

アジャイル開発のプロセス（スクラムの例）

アジャイル開発には、複数の進め方があります。本書では、その代表格であるスクラムを例にしてアジャイル開発のプロセスを概説します。



スクラムの基本的な開発の流れ（プロセス）

出典:<http://www.ryuzee.com>（許可を得て転載）

アジャイル開発のプロセス（スクラムの例）

- スクラムは反復（イテレーション）を繰り返す開発プロセスです。この反復の単位を「スプリント」と呼びます。スプリントの中身は、「スプリントプランニング」「デイリースクラム」「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）」、そして実際の「開発作業」です。
- 「スプリント」は1～4週間の時間枠（タイムボックス）であり、予定されている機能が完成できなくても延長されることはありません。この期間内で開発チームはスプリントバックログの開発に集中し、リリース可能なインクリメント（プロダクト）を作り出します。
- 「スプリントバックログ」は、プロダクトバックログから抜き出された、今回のスプリントで追加する機能のリストを言います。スプリントプランニングでプロダクトオーナーの決めた順位と開発チームが決めた見積りの両方の情報を合わせて抜き出されます。このリストは一回のスプリントにだけ使用されます。
- 「リリース判断可能なインクリメント」とは、一回のスプリントにおける成果を指します。スプリント終了時にはプロダクトが動く状態であることが必要となり、これをレビューして、プロダクトオーナーが実際にリリースするかどうかを決定します。すなわち、スプリント終了時には「リリース判断可能」になっている必要があります。
- プロセス中における各イベントの特徴を次ページ以降に説明します。

アジャイル開発の進め方の特徴（スクラムの例）

名称	特徴
<p>プロダクトバックログ</p>	<p>プロダクトバックログは、プロダクト（製品）へ追加する要求（ストーリー）のリストをいう。ユーザー（顧客）の分かる言葉で書かれている必要がある。このリストはプロダクトオーナー（PO）が管理する。このリストは、順位づけされて並んでいることが重要である。また、プロダクトの開発が続く間、変化し続け、維持される。</p> <p>チームの中で作業の「完了」についての共通見解を合意して定義（完了の定義（Doneの定義））しておく。この定義は開発者だけでなく、プロダクトオーナーや顧客との間で合意しておき、可能な限り、定義を全員が常に認識できるようにしておくことが望ましい。</p> <p>プロダクトバックログの決定は、単なるプロジェクトやタイムマネジメントのための計画行為とは異なることに注意する。提供すべきユースケースとプロダクトとしてのインフラ・共通部分（非機能要求などのバックログ項目）を見極めつつ、ユーザーにとっての価値とビジネスとしての成功、開発のスピードや品質をプロジェクトゴールに照らしてバランスさせる必要がある。その前提で各スプリントで必要なストーリーの実現と、それらのストーリー（およびそこに含まれるインフラ機能）の組合せとしての複数回のスプリントでアウトプットするリリースとを決めていく。ユーザー価値とビジネス（経営）ニーズと開発チーム能力とのマッチングの総合判断となる。</p> <p>プロダクトバックログに含まれる項目に対して、詳細の追加、見積り、並び替えをすることを、プロダクトバックログのリファインメントと呼ぶ。これはプロダクトオーナーと開発チームが協力して行う継続的なプロセスである。プロダクトバックログのリファインメントによって、バックログ項目のレビューと改訂が行われる。</p> <div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;"> <p>ユーザー ストーリー</p> <div style="border: 1px solid black; padding: 5px; width: 150px;"> <p>ストーリー項目1</p> <p>ストーリー項目2</p> <p>ストーリー項目3</p> <p>⋮</p> </div> </div> <div style="text-align: center;"> <p>プロダクト バックログ</p> <div style="border: 1px solid black; padding: 5px; width: 150px;"> <p>バックログ項目1 <small>優先1</small></p> <p>バックログ項目2 <small>優先2</small></p> <p>バックログ項目3</p> <p>⋮</p> </div> </div> </div> <p>プロダクトバックログは、作りたいプロダクトの提供すべき価値（ユーザーストーリー：機能要求、ユーザーエクスペリエンスなど）のリストである。各ストーリー項目に優先度をつけて、開発対象のバックログ項目を決める</p>

アジャイル開発の進め方の特徴（スクラムの例）

名称	特徴											
<p>スプリント プランニング</p>	<p>スプリントプランニングは、スプリント（直近の1反復）の開始に先立って行われるミーティングをいう。プロダクトバックログから今回のスプリントで扱うスプリントバックログを抜き出して決定する。プロダクトオーナーが順位にしたがって、今回扱うバックログを選び出し、スクラムチーム全体でそれらを理解する。それらを開発チームが見積もり、前回のスプリントで測定された開発実績に照らして、順位の上からどこまでを今回のスプリントに入れるかを決める。さらに、その後、開発チームが計画を詳細化し、タスクにまで分割する。通常、タスクは時間単位（2～8時間程度）で見積もられる。1つのタスクは1人に割り当てられる。</p> <p>スプリントプランニングは、直近のスプリントでやるべきタスクを単に並べるだけでなく、ユーザー視点の意味を意識しながら（適切な境界で適切な粒度で）実現のためのタスクを切り出すことが必要となる。そのスプリントのゴールを意識しながら、そのストーリーを実現するためのユーザー視点と開発者視点の両方でのアクションの切り出しが必要となる。その際、それを支えるアーキテクチャを構想し、ストーリーの各ステップの具体的なアルゴリズムやデータ構造、インターフェース、実装技術を想定して作業時間の見積りを行う（あくまで仮説ベース。技術的に不確かな要素があれば、その調査タスクを別途切り出して追加する）。その切り出しの単位は同時に他のタスクとの接続性やテストやすさを考慮したものにする必要がある。つまり、計画しながら設計と見積りと自主的な要員アサインとを同時にやっており、総合的な判断作業といえる。</p> <div style="display: flex; align-items: center;"> <div style="margin-right: 20px;"> <p>プロダクトバックログ</p> <div style="border: 1px solid black; padding: 5px; width: 150px;"> <p>バックログ項目1</p> <p>バックログ項目2</p> <p>⋮</p> <p>⋅</p> </div> <p style="text-align: right;">直近のスプリントで扱うバックログ項目を抜き出す</p> </div> <div style="margin-left: 20px;"> <p>↓</p> <p>スプリントバックログ</p> <table border="1" style="border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="width: 20%;">スプリントで扱うバックログ項目</th> <th colspan="3">タスク</th> </tr> </thead> <tbody> <tr> <td rowspan="2">バックログ項目1</td> <td style="background-color: yellow;">要求</td> <td style="background-color: lightblue;">設計実装</td> <td style="background-color: lightgrey;">テスト</td> </tr> <tr> <td style="background-color: yellow;">要求</td> <td style="background-color: lightblue;">設計実装</td> <td style="background-color: lightgrey;">テスト</td> </tr> </tbody> </table> </div> </div> <p>スプリントプランニングでは、ユーザー視点と開発者視点の両方で実装するタスクを切り出す</p>	スプリントで扱うバックログ項目	タスク			バックログ項目1	要求	設計実装	テスト	要求	設計実装	テスト
スプリントで扱うバックログ項目	タスク											
バックログ項目1	要求	設計実装	テスト									
	要求	設計実装	テスト									

アジャイル開発の進め方の特徴（スクラムの例）

名称	特徴	
	<p>スプリント内の開発作業は、コーディング作業だけではなく、要求の確認とテスト計画、詳細設計（場合によっては外部設計へのフィードバックも発生）、コーディングと単体テスト、ビルドと部分的なシステムテスト、UIの確認といった有機的に繋がった非常に幅広い作業の集合体である。このことから開発チームが協働しなければならないこと、（各人の得意分野の違いを互いに補い合えるよう）それぞれが得意分野を持ちつつも広い範囲の仕事がこなせる多能工でなければならないことになる。</p>	<p>開発チームが協働して、要求～設計～テストまでの作業を繰り返す</p>
<p>スプリント内の開発作業</p>	<p>開発の進め方は、「テスト駆動」に基づくことが基本である。まず、そのタスクが完了した際に満たすはずのテストプログラムをコーディングし、現時点ではそれが失敗することを確認する。次に、そのテストを満たす最もシンプルな設計のコードを完成させ、テストが成功することを確認する。ここで、他のタスクの成果物とビルドしたうえでのテストが実行される。そのうえで、コードの構造や設計をさらに望ましい形にリファクタリングし、テストが成功することを確認する。このとき、要求の見直しや、ストーリーの一部省略、変更も発生する。</p> <p>最初に詳細レベルのテストを仕様化するという事は、テストで確認すべき機能仕様を定義する活動でもあり、前もってこれらのテスト内容を書くということは、ソフトウェアの設計について考えるということになる。この繰り返しのサイクルを速くするため、自動化できるものは全て自動化することが肝心である。</p>	<p>テスト駆動開発の流れ</p> <p>開発の進め方は、「テスト駆動」に基づくことが基本。繰り返しのサイクルを速くするために、自動化できるものは全て自動化する</p>

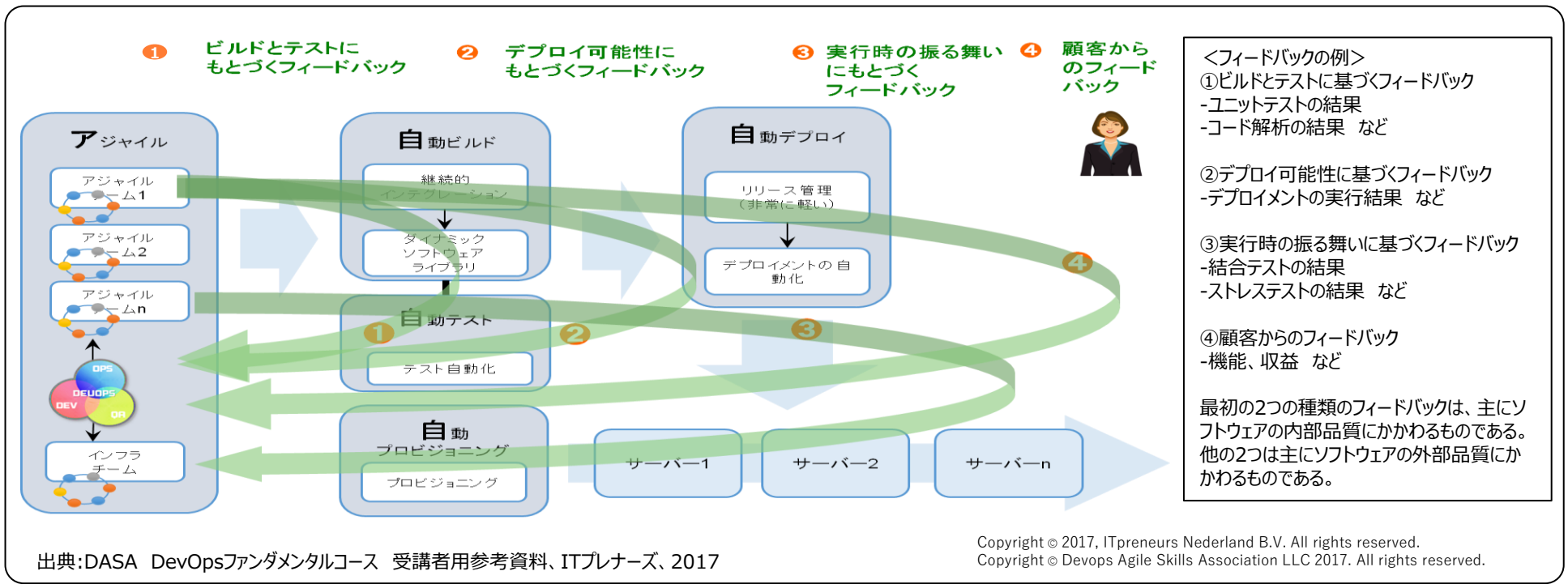
アジャイル開発の進め方の特徴 (スクラムの例)

名称	特徴																					
<p>デイリースクラム</p>	<p>開発チームが全員の活動状況を共有し、前回のデイリースクラム以降に行った作業と、次回のデイリースクラムまでに行う作業を確認する。「スタンドアップミーティング」とも言われ、立ったまま、毎日決まった時間に決まった場所で、15分の短い時間で行う。朝行うことが多いため、日本では「朝会（あさかい）」という呼び名で知られる。</p> <p>チームは1人ずつ、「昨日やったこと」「今日やること」「障害になっていること」の3つを順に話す。開発チームが解決できない障害を取り除くことが、スクラムマスターの仕事になる。また、デイリースクラムの状況はプロダクトオーナーに報告する。</p>	<table border="1"> <thead> <tr> <th></th> <th>担当</th> <th>To Do</th> <th>Doing</th> <th>Done</th> </tr> </thead> <tbody> <tr> <td>バックログ項目#1</td> <td>●●</td> <td>xxx xxx xxx</td> <td>xxx</td> <td>xxx xxx</td> </tr> <tr> <td>バックログ項目#2</td> <td>▲▲</td> <td>xxx xxx xxx</td> <td>xxx xxx</td> <td>xxx xxx</td> </tr> <tr> <td>バックログ項目#3</td> <td>■■</td> <td>xxx xxx xxx</td> <td>xxx xxx</td> <td>xxx</td> </tr> </tbody> </table> <p>開発チームが全員の活動状況を共有する</p>		担当	To Do	Doing	Done	バックログ項目#1	●●	xxx xxx xxx	xxx	xxx xxx	バックログ項目#2	▲▲	xxx xxx xxx	xxx xxx	xxx xxx	バックログ項目#3	■■	xxx xxx xxx	xxx xxx	xxx
	担当	To Do	Doing	Done																		
バックログ項目#1	●●	xxx xxx xxx	xxx	xxx xxx																		
バックログ項目#2	▲▲	xxx xxx xxx	xxx xxx	xxx xxx																		
バックログ項目#3	■■	xxx xxx xxx	xxx xxx	xxx																		
<p>スプリントレビュー</p>	<p>スプリントの終了時、関係者を呼び集めてできあがったプロダクトのデモンストレーションを行う。開発チームにとっては、自分たちが作ったバックログの項目が動いていることをアピールする機会であるうえに、他の関係者にとっては、スプリントが上手くいってプロダクトが徐々に成長していることを確認する機会でもある。</p>	<p>スクラムチームと関係者が、各スプリントの終了時にスプリントの成果をレビューする</p>																				
<p>スプリントレトロスペクティブ (ふりかえり)</p>	<p>スプリントレビューの後に行われる、今回のスプリントを振り返る機会をいう。レトロスペクティブともいわれる。ここでは、このスプリントでうまくいったこと、うまくいかなかったこと、どうやったら次のスプリントでよりうまくできるか、が話し合われる。これが成長の機会となり、チーム学習やチーム改善の活動となる。</p>	<table border="1"> <tr> <td>Keep うまく行ったこと</td> <td>Try 改善法</td> </tr> <tr> <td>Problem うまく行かなかったこと</td> <td>解決法</td> </tr> </table> <p>新しい問題 新しいアイデア</p> <p>スプリントごとに「ふりかえり」を繰り返すことでチームが成長していく</p>	Keep うまく行ったこと	Try 改善法	Problem うまく行かなかったこと	解決法																
Keep うまく行ったこと	Try 改善法																					
Problem うまく行かなかったこと	解決法																					

アジャイル開発の進め方の特徴（スクラムの例）

前述のような総合的な作業を、日々チームで会話を交わしながら毎日行うので、初心者でも計画、見積り、設計、個々のフレームワークや環境、言語の使い方の実践的な練習が繰り返されることとなります。いわば、トレーニングをごく自然に受けている状況になり、自分たちの判断した「ストーリーの定義」・「環境の選択」・「設計」・「コーディング」の成功／失敗は、それぞれ「1スプリント」、「1～2スプリント」、「1スプリント～数日」、「数日～数週間」後には結果としてフィードバックされる環境で作業を進めることができます。失敗も成功も、自分たちチームの経験としてスキルアップに直接つながっていくことが実感できます。おのずと技術向上が効率的に行われる開発プロセスです。

上記に関連して、開発中に活用することのできるフィードバック例を下図に示します。

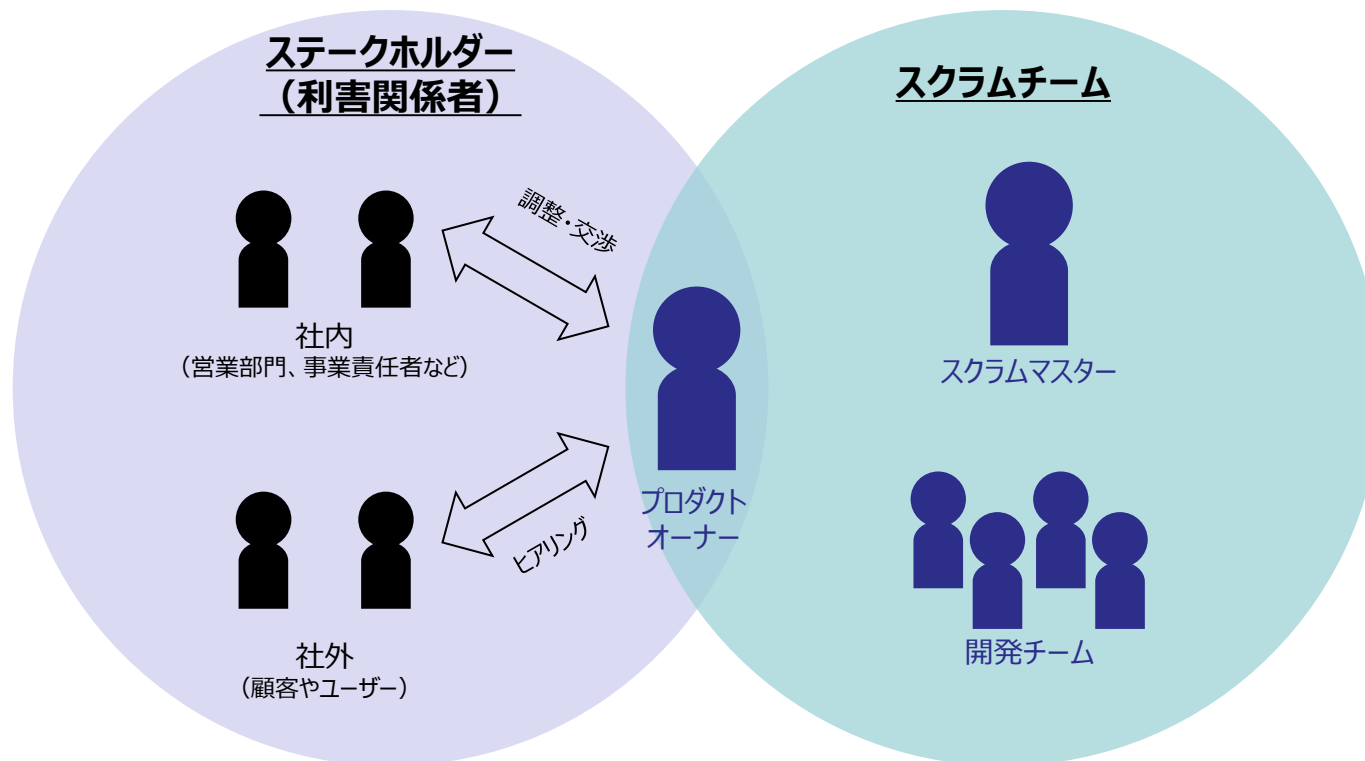


(参考) ソフトウェアデリバリープロセスのフローのうえで発生するフィードバック例

役割（ロール）の特徴（スクラムの例）

スクラムで決められている役割（ロール）は、「プロダクトオーナー」「開発チーム」「スクラムマスター」の3種類です。これら全体を「スクラムチーム」と呼び、3つの役割が協調することで、大きな効果を創出します。

開発チームは、プロダクトの開発プロセス全体に責任を負い、開発プロセスを通して完全に自律的である必要があります。スクラムではこの自律したチームのことを「自己組織化された」チームと呼びます。チームがプロダクトを開発するために必要なスキルを全て備えている必要があります。従来型では、特定の専門性をもったメンバーが役割分担して開発することが一般的でしたが、スクラムの開発チームは、一人が複数のタスクを担う多能工である必要があります。



役割（ロール）の特徴（スクラムの例）

ロール	説明	詳細
<p>プロダクト オーナー</p>	<p>何を 開発するか 決める人</p>	<p>開発への投資に対する効果（ROI）を最大にすることに責任をもつ。チームに最も価値の高いソフトウェアを開発してもらうために、プロダクトに必要な機能を定義し、その機能を順位づけする。機能がプロダクトバックログというリストになっている。バックログへの追加、削除、順位づけは、プロダクトオーナーに最終的な責任がある。また、プロダクトオーナーには、開発チームに機能を説明して理解してもらう責任がある。もちろん、プロダクトのビジョンを示すことも大切な仕事である。</p>
<p>開発 チーム</p>	<p>実際に 開発作業に 携わる人々</p>	<p>実際に開発を行うチームのことで、開発者たちを指す。従来型では、役割ごとに、タスクや役割が決まっているのが一般的である。スクラムでは、ビジネスアナリスト、プログラマー、テスター、アーキテクト、デザイナーなどの明示的な区分けはない。個人の専門分野はあってよく、むしろ強みを持ち寄り、その垣根を超えて貢献しあう。機能横断的な様々な技能を持った人がプロダクトを中心に集まり、自律的に行動する。開発チームはバックログに入っている項目を完了状態にし、プロダクトの価値を高めていくことに責任を持つ。</p>
<p>スクラム マスター</p>	<p>全体を支援・ マネジメント する人</p>	<p>スクラムマスターはスクラム全体をうまく回すことに責任を持つ、キーパーソンといえる。スクラムチーム全体が自律的に協働できるように、場作りをするファシリテーター的な役割を担う。ときにはコーチとなってメンバーの相談に乗ったり、開発チームが抱えている問題を取り除いたりする。</p> <p>スクラムチーム全体をマネジメントするが、コントロール型の管理を行うのではなく、チームを支援する役割を担う。サーバントリーダー（奉仕型のリーダー）といえ、開発チームを外部の割り込みから守り、チームの障害を取り除くために外部との交渉を行う。</p> <p>開発のやり方に関する決定はスクラムマスターではなく、開発チームが行う。スクラムマスターが細かい指示を出すのではなく、自分たちで決めながら動く自律したチームを作ることが、生産性をあげる鍵となる。</p>

開発プロセスと役割（ロール）の関連（スクラムの例）

アジャイル開発のプロセスと役割（ロール）の対応関係について、次表に示します。

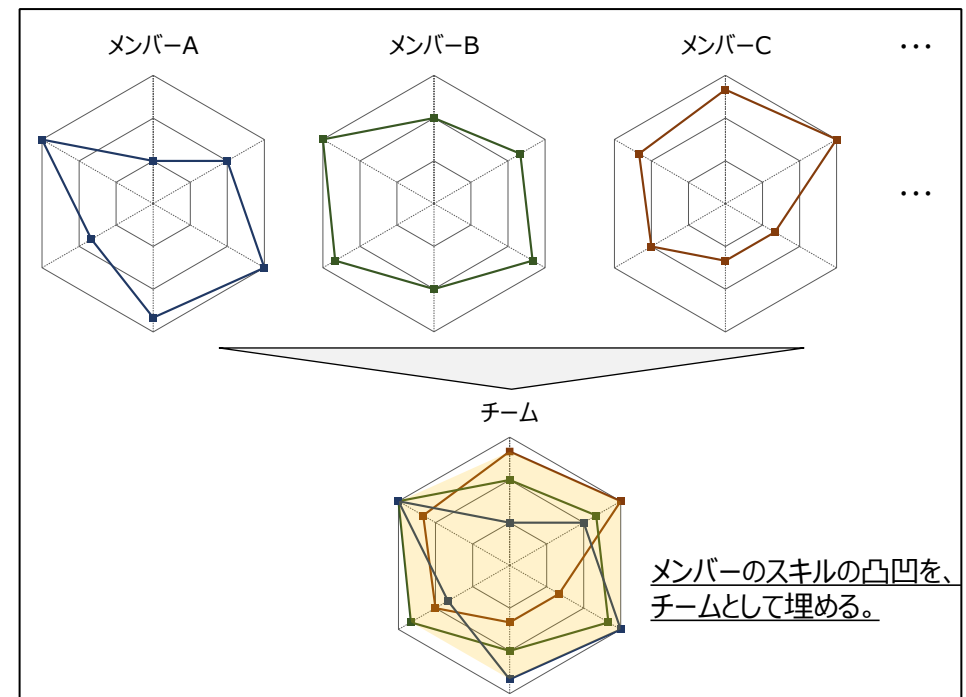
プロセス				役割（ロール）			
大分類	中分類	小分類	評価項目	プロダクト オーナー	開発チーム	スクラム マスター	
スクラム	プロジェクト 立ち上げ	プロジェクト方針の初期検討	ビジネスのビジョン、戦略を共有する	◎/○	-	-	
			プロジェクトとしての目標、あるべき姿、基本的価値観の共有を図る				
		プロジェクトチームの編成	プロダクトオーナー、スクラムマスターの役割を決定する 開発メンバーを決定する	◎/○	-	-	
		事業部門との体制構築	事業部門と体制、役割について合意する。	◎/○	-	-	
	プロダクト バックログの 決定	リリース計画	システムの目的の合意	システムの目的やゴールの共有を行う。	◎	◎	○
			リリース計画	ユーザー要望を受け、ユーザーストーリーを決定する	◎	◎	○
				スプリント期間を決定する			
				スプリント計画を準備する			
				最初のプロダクトバックログのグルーピングを行う プロダクトバックログアイテムを見積もる			
	スプリント ※繰り返し	スプリント計画 (イテレーション計画)	次のスプリントの目標を定義する。	◎	○	○	
			仕事量の見積りを行い、スプリントで扱うストーリーの数を決定する。				
			実施するタスクをスプリントバックログに追加する。				
		スプリント (イテレーション)	タスクを実施する。	△	◎	○	
			(毎日) デイリースクラムでチームの状況を共有する				
			(毎日) デイリースクラムの状況をプロダクトオーナーと共有する				
スプリントレビュー (デモ)	スプリントの成果物をプロダクトオーナーにデモする	◎	◎	△			
ふりかえり (レトロスペクティブ)	スプリント中の改善事項を検討し、次につなげる	○	◎	◎			
プロダクトバックログリファインメント	更新されたストーリーをプロダクトバックログに追加する	◎	○	○			

記号	意味
◎	主体となって実施するタスク。
○	他のロールが主体となって実施し、補佐的にかかわるタスク
△	実施にはかかわらないが、タスクの情報を共有する
-	何も行わない（実施にかかわらず、タスクの情報も共有しない）
×	アジャイル開発ではどのロールも実施しないタスク

アジャイル開発チームの作り方

アジャイル開発チームのもつべきスキル

- アジャイル開発へとパラダイムシフトする際の最も重要な側面の1つは、開発チーム内の役割の違いを理解することです。
- アジャイル開発におけるチームの特徴は機能横断（クロスファンクション）型のチーム体制です。チームがプロダクトのライフサイクル（設計、ビルド、テスト、デプロイ、実行）を通じて完全に自律的であるためには、チームとしてバランスのとれたスキルセットを備えることが必要です。チームメンバーは仕事をこなすための深い開発スキルを持つと同時に、チームとしてパフォーマンスを最大化するためのスキルが必要です。
- アジャイル開発に必要な全てのスキル・知識を持つ人材を育てることは必須の条件ではありません。一人の個人だけでは、スキル領域ごとのレベルに凸凹がありますが、その凸凹をチームとして埋めていきます。一人の個人だけでは不足している知識・スキルを、チームとして補っていくのです。

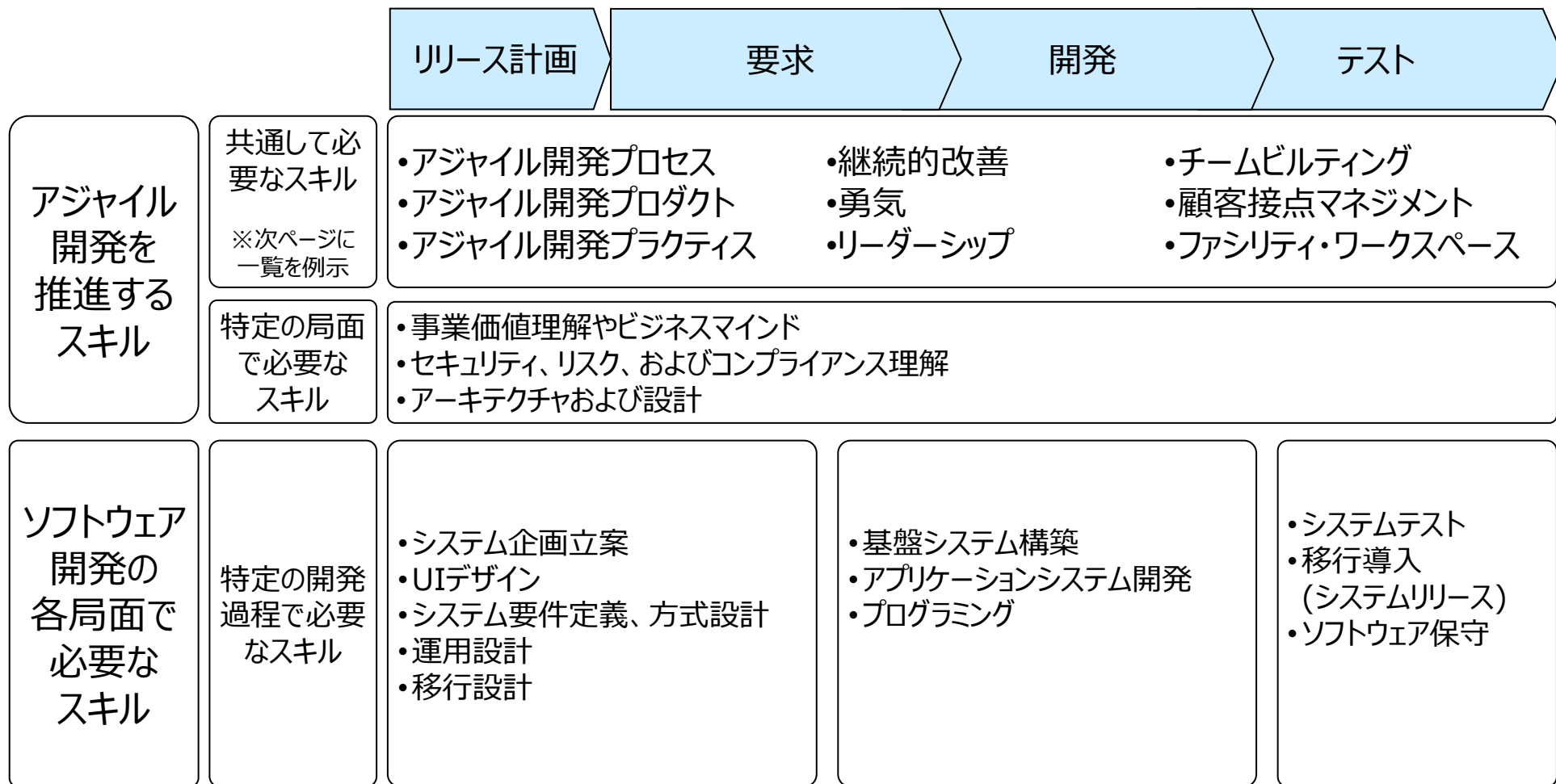


アジャイル開発チームのもつべきスキル

- 従来型の開発では、要件定義、設計、開発からテストを経て運用へと、明確な役割分担のもとにプロセスが進みます。初期工程で仕様をきっちりと決めるため、誰が何をすべきかを明確にすることができます。一般に、SE(設計者)、プログラマー、テスターなどを専任の役割としておくということは珍しくありません。このような役割分担で開発を進めると、タスクの切れ目に、人のアサインや引継ぎなどのオーバーヘッドが生じます。
- アジャイル開発では、チーム員同士で教えあい、チーム一丸となってプロダクトを開発していきます。チームで仕事をする事により、個人が得意とする分野だけでなく、より広範な分野の業務を実施できるようになり、個人の成長につなげることができます。このため、専門領域以外のスキルを埋めるために、チーム内の他の人材や他ステークホルダーと連携する能力も備わっていることが重要となります。
- アジャイル開発では、最終プロダクトをリリースするために必要となる全てのスキルが1つのチーム内に備わっているため、タスク間の引継ぎが最小限に抑えられ、プロセス全体のスループットが最適化されます。アジャイル開発では、スピードが重要であり、できるだけ早くユーザー機能を顧客に提供することが重要です。組織は、価値あるフィードバックループを活用でき、自分たちの進んでいる方向が正しいかどうかを判断できることにもつながります。
- あるタスクの高度な専門家が活動できない場合でも、プロダクトやサービスを継続的に提供するためには、各開発者が自分の能力に、さらに多くのスキルや知識を追加していくことが重要となります。アジャイルなチームにおいては、リソースは通常は複数のスキルを持ち、必要な場合には積極的にスキルを向上させようと努めます。

スキル一覧

アジャイル開発に必要なスキルを分類して示します。下図では、「アジャイル開発を推進するスキル」と「ソフトウェア開発の各局面に必要なスキル」とに分類して示します。前者は、アジャイル開発に特化したものですが、後者は、従来型においても必要な開発スキルになります。



■ アジャイル開発を推進するスキル例 (1/4)

スキル カテゴリ	スキル項目	知識項目	別名
技術スキル	アジャイル開発プロセス	リリース計画ミーティング	
		イテレーション計画ミーティング	計画ゲーム、スプリント計画ミーティング、反復型計画
		イテレーション	タイムボックス、スプリント、反復
		プランニングポーカー	見積りポーカー
		ベロシティ計測	
		日次ミーティング	朝会、朝礼、デイリースクラム、スタンドアップミーティング
		ふりかえり	レトロスペクティブ、リフレクション、内省、反省会
		かんばん	Kanban、フィーチャーパイプライン
		スプリントレビュー	デモ
		タスクボード	スクラムボード、タスクカード
		バーンダウンチャート	
	アジャイル開発プロダクト	ユーザーストーリー	ストーリーカード
		スプリントバックログ	
		インセプションデッキ	
		プロダクトバックログ	マスターストーリーリスト

■ アジャイル開発を推進するスキル例 (2/4)

スキル カテゴリ	スキル項目	知識項目	別名
技術スキル	アジャイル開発プラクティス	ペアプログラミング	ペアワーク、ペアリング
		自動化された回帰テスト	リグレッションテスト
		テスト駆動開発	
		ユニットテストの自動化	デベロッパーテスト
		受入テスト	顧客テスト、機能テスト、ストーリーテスト
		システムメタファ	
		スパイク・ソリューション	実験、曳光弾
		リファクタリング	
		シンプルデザイン	YAGNI
		逐次の統合	
		継続的インテグレーション	常時結合、CI
		集団によるオーナーシップ	共同所有
		コーディング規約	コーディング標準

■ アジャイル開発を推進するスキル例 (3/4)

スキル カテゴリ	スキル項目	知識項目	別名
ヒューマンスキル	継続的改善	私たちは今日、昨日よりうまく仕事をする	
		カイゼンのマインドセット	
		源流管理	
		最初から正しく	
		知識の共有	
		順応性	
		総合	
	勇気	伝える情熱	
		コーチング	
		自信	
		自発性	
		反省	
		信頼	
		オープンな議論	
		実験	
		早く失敗すること	
		変更する勇気	
	リーダーシップ	チームのハイ・パフォーマンス化の促進	
		謙虚さ	
サービスライフサイクルのマインドセット			
利害関係者のマネジメント			

■ アジャイル開発を推進するスキル例 (4/4)

スキル カテゴリ	スキル項目	知識項目	別名
ヒューマンスキル	チームビルディング	顧客プロキシ	
		オンサイト顧客	
		プロダクトオーナー	
		ファシリテータ	スクラムマスター
		アジャイルコーチ	
		自己組織化チーム	
		ニコニコカレンダー	
		他者の視点の理解	
		コラボレーション	
		相互の説明責任	
		共通の目的	
		サービス/プロダクトを総合的にサポートする能力	
	顧客接点マネジメント	対面コミュニケーション	
		顧客起点	
		価値起点	
		場づくり	
	ファシリティ・ワークスペース	共通の部屋	
		チーム全体が一つに	
		人材のローテーション	
		インテグレーション専用マシン	

参考資料

<参考1>

従来型ロールとアジャイル型ロールの比較表

従来型ロールとアジャイル型ロールの比較表について

従来型ロールとアジャイル型ロールの比較表は、従来型（ウォーターフォール型）開発に従事してきた人材が、アジャイル開発について学ぶ時、従来型ロールとアジャイル型ロールの実施するタスクの違いを比較するための参考資料です。本表のタスクは、iCD2017を参照して、SI型アプリケーションシステム開発に典型的なタスクを抜き出しています。従来型ロールは、企画・開発を職務とするロール（iCDではタスクプロフィールと呼ぶ）を抽出し、タスクとの関連を示しています。今回、このタスクに対して、アジャイル型ロールではどう対応するかを例示しています。従来型の各ロールが実施して各タスクをアジャイル型ロールがどういうフォーメーションで実施しているのかを確認することができます。

タスク構成図（iCD2017より抜粋）

戦略	ST01 事業戦略策定
戦略	ST02 事業戦略把握・策定支援
戦略	ST03 IT製品・サービス戦略策定
企画	PL01 IT戦略策定・実行推進
企画	PL02 システム企画立案
ライフサイクル	DV01 システム要件定義・方式設計
	DV02 運用設計
	DV03 移行設計
	DV04 基盤システム構築
	DV05 アプリケーションシステム開発
	DV06 ソフトウェア製品開発
	DV07 組込みソフトウェア開発
	DV08 Webサイト開発
	DV09 システムテスト
	DV10 セキュリティテスト
	DV11 移行・導入（システムリリース）
	DV12 ソフトウェア保守
	DV13 ハードウェア・ソフトウェア製品導入
	DV14 ファシリティ設計・構築
活用	US01 サービスデスク
活用	US02 IT運用コントロール
活用	US03 システム運用管理
活用	US04 Webサイト運用管理
活用	US05 ファシリティ運用管理
評価・改善	EV01 システム評価・改善
	EV02 IT戦略評価・改善
	EV03 IT製品・サービス戦略評価・改善
	EV04 事業戦略評価支援・改善支援
	EV05 事業戦略評価・改善
	資産管理・評価

役割別タスクプロフィール（ロール） （iCD2017より抜粋）

戦略的職務	PL01	PL02	PL03	PL04	PL05	PL06	PL07	PL08	PL09	PL10	PL11	PL12	PL13	PL14	PL15	PL16	PL17	PL18	PL19	PL20
企画・開発	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
運用																				

タスクとロールとの対応

各タスクに対して、従来型ロール（下図の赤枠）とアジャイル型ロールの対応（下図の緑枠）を示している。記号の意味は次のとおり。

記号	意味
◎	主体となって実施するタスク。
○	他のロールが主体となって実施し、補的にかかわるタスク
△	実施にはかわらないが、タスクの情報を共有する
-	何も行わない（実施にかかわらず、タスクの情報も共有しない）
×	アジャイルではどのロールも実施しないタスク

従来のタスクとアジャイル型ロールの比較表

タスク大分類コード	タスク大分類	タスク中分類コード	タスク中分類	タスク小分類コード	タスク小分類	評価項目コード	評価項目	ウォーターフォール型ロール		アジャイル型ロール（例）									
								プロダクトオーナー	スクラムマスター	開発チーム	開発チーム	開発チーム	開発チーム	開発チーム	開発チーム				
PL02	システム企画立案	PL02.1	システム化構想の立案	PL02.1.1	システム化構想基本方針の策定	PL02.1.1.1	システム化構想によるビジネス課題解決の達成目標を確認する	○	○	○	○	○	○	○					
							事業環境、業務環境の情報を収集し、事業課題を分析する	○	○	○	○	○	○	○					
							システム化構想の前提となるIT戦略を把握する	○	○	○	○	○	○	○					
							事業環境および業務環境の分析結果と情報システム化目標の関係をIT化方針として文書化する	○	○	○	○	○	○	○					
							PL02.1.2	現行業務、システムの調査分析	現行業務を把握する	○	○	○	○	○	○	○	○	○	○
							現行システムの状況を調査し、現行システムの稼働状況を把握する		○	○	○	○	○	○	○	○	○	○	
							PL02.1.2	現行業務やシステムの状況から、開発、改善、改革対象の範囲を把握する	現行業務を調査し、業務実態を把握する	○	○	○	○	○	○	○	○	○	○
							現行業務やシステムの状況から、開発、改善、改革対象の範囲を把握する		○	○	○	○	○	○	○	○	○	○	
							PL02.1.3	新業務の全体像把握と評価指標の	システム化で対応する業務機能のあるべき姿を描く	○	○	○	○	○	○	○	○	○	○
							あるべき業務機能に求められる主要機能を明らかにする		○	○	○	○	○	○	○	○	○	○	
							企画するシステムにおける業務運用の定量的評価指標を設定する		○	○	○	○	○	○	○	○	○	○	
							PL02.1.3	企画するシステムにおける業務運用の定性的評価指標を設定する	企画するシステムにおける業務運用の定性的評価指標を設定する	○	○	○	○	○	○	○	○	○	○
							投資規模の策定		○	○	○	○	○	○	○	○	○	○	
							PL02.1.4	システム開発（一次費用）に関する期間、体制、工数、設備費の概算を見積もる	システム開発（一次費用）に関する期間、体制、工数、設備費の概算を見積もる	○	○	○	○	○	○	○	○	○	○
システム開発（ランニングコスト）に関する期間、体制、工数、設備費の概算を見積もる	○	○	○	○	○	○	○		○	○	○								
PL02.1.4	ビジネスモデルとシステムアーキテクチャによる企業目標、経営戦略およびIT戦略の実現性を評価する	ビジネスモデルとシステムアーキテクチャによる企業目標、経営戦略およびIT戦略の実現性を評価する	○	○	○	○	○	○	○	○	○	○							

※比較表では青色のタスクを抽出

■注意：本表は従来型開発のタスクの違いを比較するために、スクラムのフレームワークで登場するロールを参考に示しています。そのため、従来型のロールとスクラムのロールとで比較するタスクをあえて同一項目にして表しています。本表は参考資料として示すものであり、全てこのとおりに実施すること、もしくはこれだけ実施すればよいことを示すものではありません。

従来型ロールとアジャイル型ロールの比較表の見方

本表を次に示す観点で比較することにより、従来型ロールとアジャイル型ロールを比較してください。

観点①

アジャイル型ロールは、従来型ロールとは概念が大きく変わり、従来型より広範な能力が求められる。
アジャイル型ロールは多様なタスクをこなす（多能工化している）ことを理解する。

観点①
アジャイル型では多様なタスクを実施（多能工化）している。

観点②

従来型では単独で行っていたタスクをアジャイル開発では、複数のロールが協働して行う。

観点②
従来型では単独で行っていたタスクをアジャイル型では複数のロールが協働して行う。

観点③

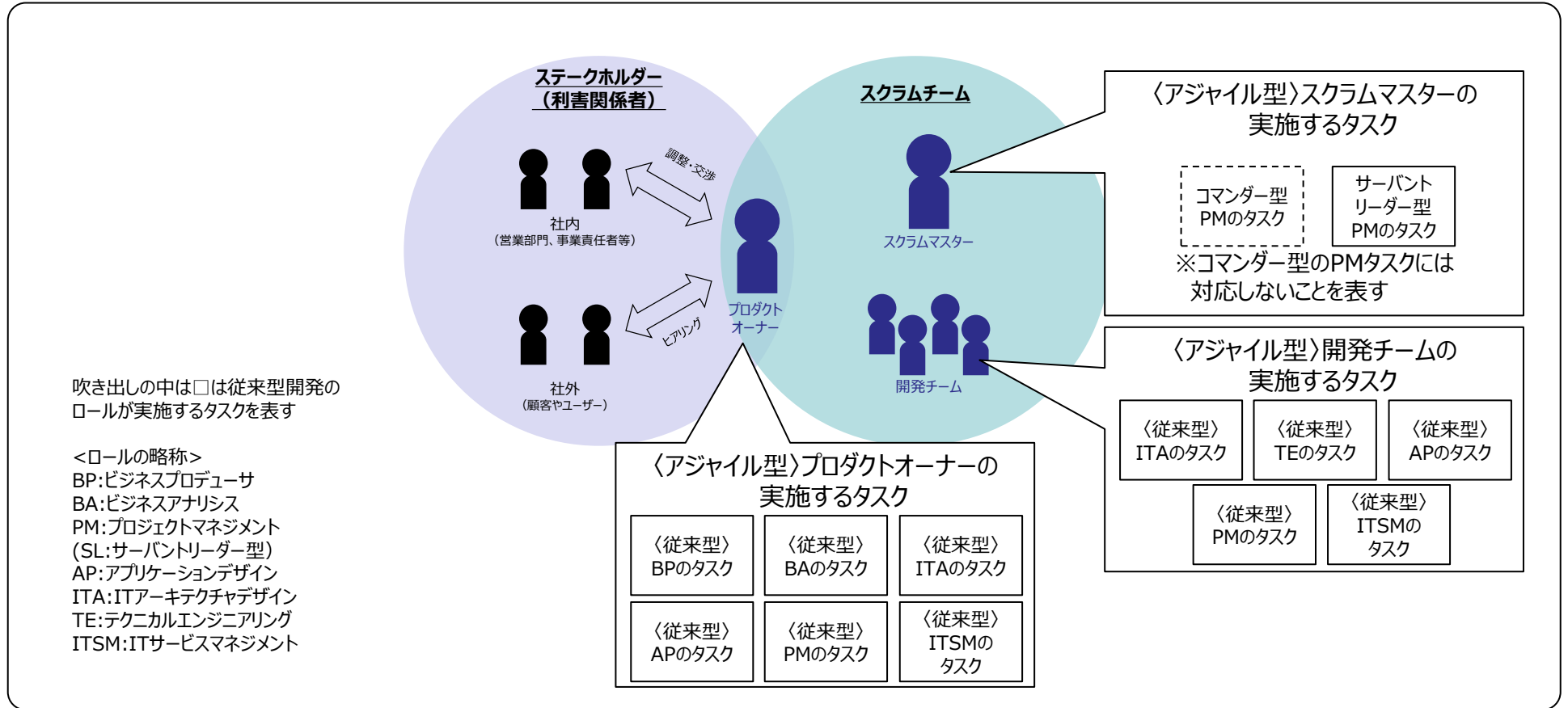
従来型のPMの仕事の多くの部分はチームメンバー各人が自律的に行うことになる。プロジェクトマネジメントのタスクのうち、アジャイル型のチームメンバーが担わない仕事がスクラムマスターの担う役割となる。逆にいうと、スクラムマスターの果たす役割は、従来型のPMの仕事に比べて、「サーバントリーダー」の位置づけとなるため、「コマンダー型」のタスクには対応しなくなる。

観点③-1
従来型ではプロジェクトマネージャーのみが担っていたタスクをアジャイル型では各ロールが行っている。

観点③-2
アジャイル型のPM（スクラムマスター）は、従来型のPMのタスクがカバーしない要素が多い。

従来型ロールとアジャイル型ロールの対応イメージ

- ・従来型のロールでは、実施するタスクの括りが異なる。
- ・アジャイル開発のロールは、従来型開発では複数のロールが実施していたタスクを実施する場合がある。



■ 従来型ロールとアジャイル型ロールとの対応 (イメージ)

<参考2> アジャイル開発の概念整理

本ドキュメントは、「アジャイル開発とは」を端的に説明することを狙いとして整理を試みました。
アジャイル開発の全体像を理解するための参考にしてください。

アジャイル開発の概念整理

• アジャイル開発の目的

アジャイル開発とは、ビジネス価値の最大化に向けて、顧客に価値のあるソフトウェアを早く、継続的に提供するためのアプローチです。これを実現するためにアジャイル開発で重要となる事項について説明します。

• アジャイル開発の本質

アジャイル開発での最終的な目標は、ビジネスの価値の最大化です。開発者も常に顧客と同じ目線で、顧客にとっての価値が最大となるよう取り組む姿勢が必要です。

一方で、アジャイル開発を作業効率化の1つの手法として考えている方がいるかもしれません。開発者としてなるべくムダな作業を行わないことは、アジャイル開発では基本的な考え方ではありますが、いくら作業を効率化できたとしても、価値あるものを創出できなければ意味がありません。

• 顧客にとっての価値を知るには

顧客にとっての価値とは何かを知るために、実際に作ったものを使ってもらって、顧客が満足しているかどうかを確認します。

• 常に状況は変化すると考える

昨今のビジネス環境は絶えず変化します。それに俊敏に対応するため、ベストではないものの、ベターなビジネス解を徐々に改善していく傾向が強くなっています。こうした状況から生み出されるシステム要求も、ビジネス解の継続的な変化に対応し、変更し続けることとなります。アジャイル開発では変化に対応して、価値のあるソフトウェアを早く、継続的に提供していくことが求められます。

• 変化に柔軟に対応するためには

顧客の要求も絶えず変化しますので、顧客からのフィードバックを短いサイクルで得ながら、提供したものに価値があるかどうかを継続して確認します。

アジャイル開発の概念整理

• ビジネス価値のある動くソフトウェアとは

アジャイル開発では、ソフトウェアを提供するため、タイムボックスを使用した反復型のアプローチをとります。顧客が評価できるソフトウェアを提示し、顧客からのフィードバックを短いサイクルで得ながら、提供したものに価値があるかどうかを確認します。（現場現物現実、高速仮説検証サイクル）

• 顧客とのWin-Winの関係を構築する

顧客からのフィードバックを効率的、効果的に得るためには、開発者と顧客が直接対話しながら、Win-Winの関係を築いていることが肝心です。

• 自律的なチームで、人の能力を最大限に発揮する

アジャイル開発に限った話ではありませんが、組織やプロジェクトにおいて一番大切なものは“人”です。

自動化が進んでも、やはり人間でなければできないことはたくさんあります。特にアジャイル開発では、ムダな作業を極力減らし、空いた時間で人間の能力を最大限に活用できるようにすることが求められます。

開発チームは開発プロセスのライフサイクル全体を通して完全に自律している（自己完結している）必要があります。そのため、チームは顧客とエンドツーエンドでコミュニケーションするとともに、開発プロセス全体を遂行するために必要な全ての専門知識やスキルをチームとして備えている必要があります。これにより、要員アサインや承認のための待ちなどを排除してムダをなくすことができます。

• 技術・プログラミングの重要性

人間の能力を最大限に活用するために、開発者は何を身に付けておくべきでしょうか？

アジャイルソフトウェア開発宣言の背後にある原則の1つにあるように、「技術的卓越性と優れた設計に対する不断の注意が機敏さを高める」ため、常に最新の技術を身に着ける努力が必要となってきます。アジャイル開発では動くソフトウェアに価値を求めるため、とりわけプログラミングに関する知識やスキルは必須とも言えるでしょう。

アジャイル開発の概念整理

アジャイル開発の活動を支える柱（大原則）として、「人間中心」と「技術の尊重」をあげることができます。

• 人間中心

人間中心は、従来の顧客起点ではなく、社会を構成する一人一人（顧客だけでなく、経営者も従業員も）の生きる意味を考えると社会の価値につながることを示しています。また、提供する人間の能力を最大限に活かすことが重要です。個人個人の能力が社会の中で創造的かつ健全に開花し、多様なチーム、組織、コミュニティに価値を提供し、その中で生きがいを持って協働できる働きやすい社会を目指すことが重要です。ITはそれをエンパワーするものであるべきです。

• 技術の尊重

技術力をもって生産性と品質・信頼性を担保するとともに、常に適切な技術とスキルを学習し、それを社会に還元し、次世代に継承する努力を怠らず、学習する組織・社会を導くことも重要です。

技術活用は、プログラミング的思考、よい技術、よい設計によりよい品質を追求すること、できるだけ自動化し、人の無用な負担を排除することなどが該当します。

• 人間中心と技術活用のバランスが重要

人間中心と技術活用という2本の柱のバランスが重要です。2本の柱でイノベティブな社会変革を人間中心で仮説設定・検証を繰り返しながら進めていきます。特に技術中心で考えてきた日本の企業に対して人間中心なイノベーションを個人・組織に植えつけることが必要です。本質を理解せずに形だけ真似しても成果は創出することはできません。

アジャイル開発の概念整理

- **継続的な改善、成長を目指すマインドセット**

決められたプロセスに沿って作業を進めることも重要ですが、アジャイル開発では状況の変化に応じてプロセスも柔軟に変更して対応することが求められます。同じやり方を続けていては、そこに改善や成長は生まれません。たとえ期待した効果が得られなかったとしても、その経験から学ぶこともあります。早く実践（あるいは失敗）すれば、その分早く改善することもできます。また改善は開発プロセスに限った話ではありません。アジャイル開発では、自分自身も常に成長を求め続けるマインドセットを持つことが重要です。

- **あるべき姿にむけて改善しつづける人材に**

アジャイル開発は一度やり方を決めたら、そのままやり続けることを善しとするものではありません。アジャイル開発では、常にあるべき姿にむけて改善し続けるにはどうしたらよいかを考えます。例えば、過去の技術が足かせにならないように、常に技術動向を追い続け、ソフトウェアの提供をより早く行うためにはどうしたらよいかを常に考え続けます。アジャイル開発の実践の場は、継続的に改善しつづける人材になるための学びの場ともいえます。

アジャイル開発の概念整理

これまでの説明を総合して、アジャイル開発全体の概念構造を「アジャイル開発の家」として表現してみました。家をモチーフに、アジャイル開発の目指すもの（屋根、梁）、開発活動を支える大原則（柱、土台）、そして目的を達成するための活動（家の中）を表しています。アジャイル開発とは何かを整理する上での参考としてください。

アジャイル開発とは、ビジネス価値の最大化に向けて、顧客に価値のあるソフトウェアを早く、継続的に提供するためのアプローチです。

• 活動の目的（屋根／梁）：

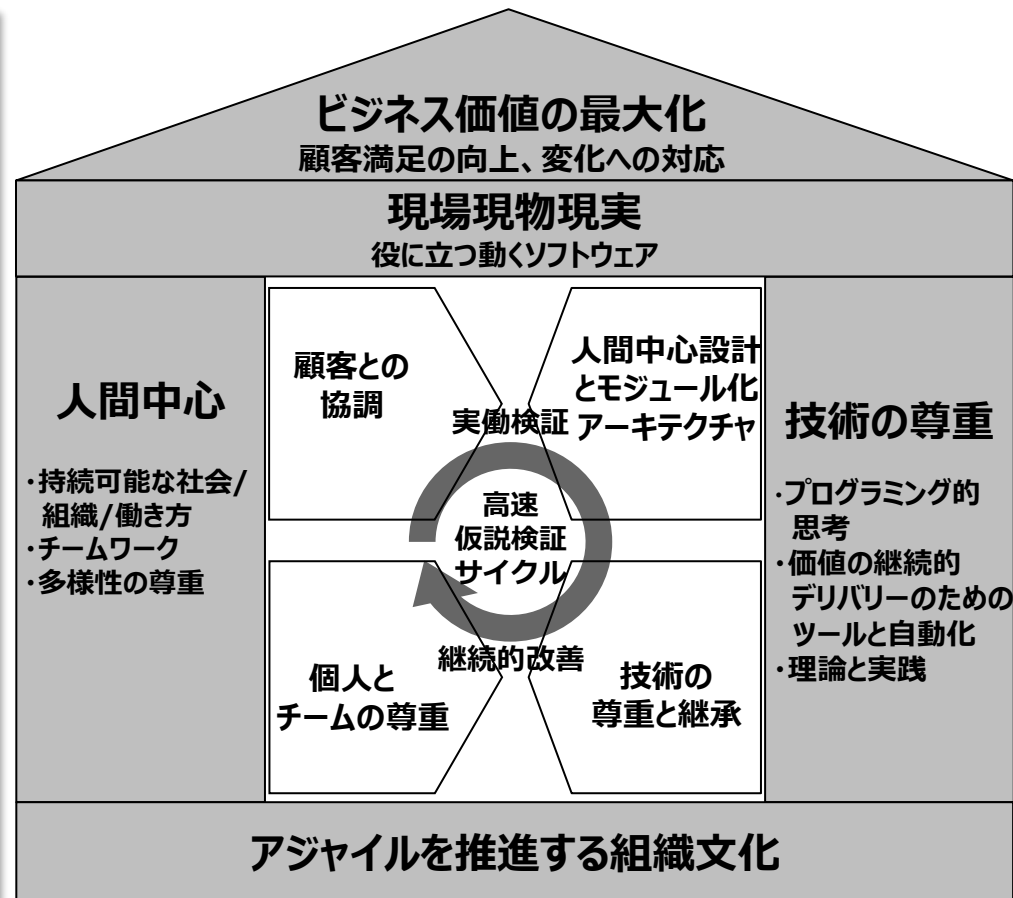
ビジネス価値の最大化を実現するため、顧客満足の上昇（何に価値があるかを見極めること）、変化への対応（素早く提供しつづけること）を意識する

現場現物現実で、実際に役に立つ、動くソフトウェアを提供し、顧客からのフィードバックにより継続的に改善する

• 活動を支える原則（柱/土台）：

- 人間中心：持続可能な社会/組織/働き方、チームワーク、多様性の尊重
- 技術の尊重：プログラミング思考、価値の継続的デリバリーのためのツールと自動化、理論と実践
- アジャイルを推進する組織文化

• 活動(家の中)：ビジネス価値の最大化を実現するための活動
 高速仮説検証サイクル（実働検証と継続的改善）、顧客との協調、個人とチームの尊重、人間中心設計とモジュール化アーキテクチャ、技術の尊重と継承



アジャイル開発の全体像

アンケートにご協力ください

アクセス
はこちら



本冊子をお読みいただき、ありがとうございます。
Webアンケートへのご協力をよろしくお願いいたします。