# スクラムマスターと
# プロダクトオーナーのための
# EXIN ハンドブック

*著者：ヨハン・ボタ*



**タイトル:** The EXIN Handbook for Scrum Masters and Product Owners
（スクラムマスターとプロダクトオーナーのための EXIN ハンドブック）
**著者:** ヨハン・ボタ
**出版社:** EXIN Holding BV
**ISBN:** 9789076531175
**発行:** 2025 年 3 月
**版権:** © EXIN Holding BV, 2025

All rights reserved. No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by the publisher. Although this publication has been composed with much care, neither author, nor editor, nor publisher can accept any liability for damage caused by possible errors and/or incompleteness in this publication.

All product names, logos, brands, trademarks and registered trademarks are property of their respective owners. Use of these names, trademarks and brands does not imply endorsement.

# 目次

はじめに ..... 10
序文 ..... 11
日本語版製作にあたって ..... 12
1　アジャイル シーンの設定 ..... 13
1.1　アジャイル思考 ..... 13
1.1.1　「プロジェクト」アプローチとしてのアジャイル（大文字 A のアジャイル（名詞）） ..... 13
1.1.2　考え方や行動の方法としてのアジャイル（小文字 a のアジャイル（形容詞）） ..... 15
1.2　アジャイルの論証 ..... 17
1.3　アジャイルを正しくするための重要成功要因 ..... 20
1.4　リーダーシップと行動、カルチャー、倫理観、信頼感 ..... 20
1.4.1　アジャイルとその組織構造への影響 ..... 23
2　アジャイル シーンの採用 ..... 24
2.1　アジャイルを採用する上での課題 ..... 24
2.2　変化へのマネジメント ..... 25
2.3　ADKAR®と ADAPT ..... 26
2.4　プロダクトの意味するところ ..... 29
3　リーン・マネジメント ..... 30
3.1　戦略・マネジメントアプローチとしてのリーン ..... 30
4　スクラムと継続的改善 ..... 33
5　スクラムの基本 ..... 34
5.1　スクラムの簡単な概要 ..... 34
5.2　異なる働き方 ..... 35
5.3　スクラムの背後にある本質的な理論 ..... 35
5.4　スクラムの 3 つの柱 ..... 36
5.4.1　スクラムイベント ..... 37
5.5　アジャイルスクラムの価値観 ..... 37
5.6　スクラムの説明責任の概要 ..... 38
5.6.1　スクラムチーム ..... 38
5.6.2　開発者 ..... 40
5.6.3　プロダクトオーナー ..... 40
5.6.4　スクラムマスター ..... 42
5.7　スクラムイベントの概要 ..... 44
5.8　スクラムイベント ..... 47
5.8.1　スプリント ..... 47
5.8.2　スプリントプランニング ..... 48
5.8.3　デイリースクラム ..... 48
5.8.4　スプリントレビュー ..... 49
5.8.5　スプリントレトロスペクティブ ..... 49
5.9　スクラムのアーティファクト ..... 50
6　スクラムチームのその他の活動 ..... 52
6.1　ポートフォリオ、プロダクト、ロードマップ ..... 52
6.2　ポートフォリオ計画 ..... 52

6.3 プロダクトを思い描く ..... 53
6.4 プロダクトとプロダクトゴール ..... 53
6.5 プロダクトゴールとビジネスバリュー ..... 54
6.6 実質価値の測定 ..... 55
6.7 プロダクトバックログの管理について ..... 57
6.7.1 適切な詳細さ ..... 58
6.7.2 見積もり ..... 59
6.7.3 創発的 ..... 59
6.7.4 順序付け ..... 59
6.8 プロダクトバックログのリファインメント ..... 59
6.9 プロダクトバックログアイテムの作成 ..... 61
6.9.1 非機能要求を分解する ..... 62
6.10 要求事項収集 – アウトプットと成果 ..... 62
6.11 ユーザーストーリーについて ..... 63
6.12 ユーザーストーリーとタスク分解 ..... 65
6.13 プロダクトバックログとロードマップの作成と維持 ..... 66
6.14 バックログアイテムの順序付けの基準は? ..... 68
6.15 ステークホルダーとのコミュニケーション ..... 70
6.16 プロダクトゴールの設定 ..... 71
6.17 要求事項の収集 ..... 73
7 アジャイル計画と見積もり ..... 75
7.1 アジャイルプランニングは何が違うのか? ..... 77
7.2 誰がプランニングに参加するのか? ..... 81
7.3 見積もり技法 ..... 82
7.3.1 再見積もり ..... 82
7.4 スプリントプランニングでは、他に何ができるのか? ..... 83
7.4.1 スプリントプランニング ..... 83
7.4.2 プロダクトバックログアイテムのサイジング ..... 88
7.4.3 ストーリーポイント ..... 88
7.4.4 プランニングポーカー(スクラムポーカー) ..... 89
7.4.5 理想日数、理想時間 ..... 90
7.4.6 付箋紙を使った高速見積もり ..... 91
7.4.7 その他、知っておくべきこと ..... 91
7.4.8 その他のストーリーは、他のステークホルダー ..... 92
7.5 スプリントの一環としての改善活動 ..... 92
7.5.1 障害物と制約理論(TOC)について ..... 92
7.5.2 5 つのステップに集中せよ ..... 93
7.5.3 継続的改善バックログ(CIB)について ..... 93
7.5.4 技術的負債 ..... 94
7.5.5 改善インクリメント ..... 94
8 スプリントでは他に何が行われるのか? ..... 95
8.1 デイリースクラムの詳細 ..... 95
8.2 レビューとレトロスペクティブの実施 ..... 96
8.2.1 スプリントレビューの詳細 ..... 96

8.2.2 スプリントレトロスペクティブの詳細 ..... 97
9 複雑で大規模なプロダクトバックログ ..... 98
9.1 スケールする方法 ..... 98
9.2 アジャイルスクラムのスケーリングに対する考え方 ..... 99
10 ビジュアルマネジメント、スクラムとカンバンボード ..... 100
10.1 スクラムボード ..... 100
10.2 スクラムボードを使う理由は何か? ..... 102
10.3 スクラムボードの使用 ..... 103
10.4 カンバンとスクラムボードの違いは何か? ..... 103
10.5 なぜフロー(流れ)が重要か? ..... 104
10.6 フローの理論を理解する ..... 105
10.6.1 リトルの法則 ..... 106
10.7 カンバンの手法で、スクラムボードはどう変わるのか? ..... 107
10.8 ブロック項目とは? ..... 108
10.9 スクラムボードとカンバンボードの比較 ..... 109
10.10 カンバン方式の活用 ..... 109
10.11 スクラムとカンバンの併用で、スプリントのタスクはどのように変わるのか? ..... 110
10.11.1 スプリント ..... 110
10.11.2 スプリントプランニング ..... 110
10.11.3 デイリースクラム ..... 110
10.11.4 スプリントレビュー ..... 111
10.11.5 スプリントレトロスペクティブ ..... 111
10.11.6 インクリメント ..... 111
10.12 良いスクラムボードには他に何が必要か? ..... 111
10.12.1 バーンダウン/バーンアップチャート ..... 112
10.12.2 ベロシティとは? ..... 113
10.12.3 ベロシティと SLE の違いは何か? ..... 114
10.13 情報ラジエータと集合場所 ..... 114
11 原則がパターンかアンチパターンを決める ..... 115
11.1 アンチパターンとは何か? ..... 115
11.2 「良いアイデア」…ではないもの! ..... 116
11.2.1 アジャイルの導入または変革を行うこと ..... 116
11.2.2 アジャイルスケーリング ..... 116
11.2.3 すべてに当てはまる唯一の方法を想定することの間違い ..... 117
11.2.4 指示と制御 ..... 118
11.2.5 制御できていないと感じることによる複雑な制御の導入 ..... 118
11.2.6 デリバリ—スピードに焦点を当てる ..... 118
12 NEXUS とスケーリング ..... 120
12.1 NEXUS とは? ..... 120
12.2 NEXUS とリリースアプローチの違い ..... 121
12.3 NEXUS フレームワークによるスケールアップの新手法 ..... 122
12.4 NEXUS のプロセス ..... 123
12.5 NEXUS の説明責任をより詳しく ..... 124
12.6 プロダクトオーナーシップ ..... 125

12.7 NEXUS 統合チームのスクラムマスター ..... 125
12.8 NEXUS 統合チームのメンバー ..... 125
12.9 NEXUS に導入された新しいイベント ..... 125
12.9.1 Nexus スプリントプランニング ..... 125
12.9.2 Nexus デイリースクラム ..... 126
12.9.3 Nexus スプリントレビュー ..... 126
12.10 NEXUS イベント ..... 127
12.10.1 リファインメント ..... 127
12.10.2 Nexus スプリントプランニング ..... 127
12.10.3 Nexus スプリントゴール ..... 128
12.10.4 Nexus デイリースクラム ..... 128
12.10.5 Nexus スプリントレビュー ..... 128
12.10.6 Nexus スプリントレトロスペクティブ ..... 128
12.11 NEXUS アーティファクト ..... 129
12.11.1 プロダクトバックログ ..... 129
12.11.2 Nexus スプリントバックログ ..... 129
12.11.3 統合されたインクリメント ..... 130
12.11.4 アーティファクトの透明性 ..... 130
12.11.5 完成の定義（DoD） ..... 130
12.12 NEXUS の核となる NEXUS 統合チーム ..... 130
12.13 NEXUS スプリントバックログの可視化とクロスチームリファインメント ..... 133
12.14 NEXUS のクロスチームリファインメント ..... 133
12.15 NEXUS スプリントプランニングと NEXUS デイリースクラムの詳細 ..... 136
12.15.1 Nexus スプリントバックログ ..... 137
13 アジャイル・スクラムの導入と成功のために ..... 138
13.1 すべては組織の変革のために ..... 138
13.2 変化をファシリテートする ..... 140
13.3 パイロットを使った実装方法 ..... 141
13.4 アジャイルスクラムを組織内に広める ..... 143
13.5 人への対応 ..... 144
13.5.1 スケプティック（懐疑者） ..... 145
13.5.2 サボタージュ（妨害者） ..... 146
13.5.3 ダイハード（頑固者） ..... 146
13.5.4 フォロワー（追従者） ..... 146
13.5.5 抵抗の種類に応じた対処法 ..... 146
13.6 適切な環境の構築 ..... 146
13.7 バーチャルチーム、リモートチームとしての活動 ..... 147
14 説明責任 – スクラムと NEXUS のイベントとプラクティス ..... 149
14.1 プロダクトオーナー ..... 149
14.1.1 スプリントプランニング ..... 149
14.1.2 デイリースクラム ..... 149
14.1.3 スプリントレビュー ..... 150
14.1.4 プロダクトバックログの作成と管理 ..... 150
14.1.5 スプリントレトロスペクティブ ..... 150

14.1.6 Nexus スプリントプランニング ..... 150
14.1.7 Nexus スプリントレビュー ..... 151
14.1.8 Nexus スプリントレロスペクティブ ..... 151
14.1.9 Nexus プロダクトバックログリファインメント ..... 151
14.2 スクラムマスター ..... 151
14.2.1 スプリントプランニング ..... 152
14.2.2 デイリースクラム ..... 152
14.2.3 スプリントレビューとレトロスペクティブ ..... 152
14.2.4 プロダクトバックログの作成と管理 ..... 153
14.2.5 Nexus ..... 153
14.3 開発者 ..... 153
14.3.1 スプリントプランニング ..... 153
14.3.2 デイリースクラム ..... 154
14.3.3 スプリントレビュー ..... 154
14.3.4 プロダクトバックログの作成と管理 ..... 154
14.3.5 スプリントレトロスペクティブ ..... 155
14.3.6 Nexus スプリントプランニング ..... 155
14.3.7 Nexus スプリントレビュー ..... 155
14.3.8 Nexus スプリントレトロスペクティブ ..... 155
14.3.9 Nexus プロダクトバックログリファインメント ..... 156
付録 A – その他のアジャイル手法 ..... 157
クリスタルの方法論 ..... 158
エクストリーム・プログラミング (XP) ..... 160
エクストリーム・プログラミング (XP) とは? ..... 160
エクストリーム・プログラミングのルール ..... 161
ペアプログラミング ..... 162
DSDM ..... 163
DSDM の原則 ..... 163
DSDM フレームワークのフェーズ ..... 163
フェーズ 1 – プレ・プロジェクト ..... 163
フェーズ 2 – 実現可能性 ..... 164
フェーズ 3 – 基礎 ..... 164
フェーズ 4 – 進化的開発 ..... 164
フェーズ 5 – デプロイメント ..... 164
フェーズ 6 – ポスト・プロジェクト ..... 164
LeSS ..... 165
LeSS 10 の原則 ..... 165
Large-scale Scrum (LeSS) はスクラム ..... 165
透明性 ..... 165
より少ないリソースでより多く ..... 166
プロダクト全体にフォーカス ..... 166
顧客中心 ..... 166
完璧を目指す継続的な改善 ..... 166
リーン思考 ..... 166

システム思考 ..... 166
経験的プロセス制御 ..... 166
待ち行列理論 ..... 166
スケールド・アジャイル・フレームワーク®（SAFE®） ..... 167
SAFe のリーン・アジャイルの 9 つの原則 ..... 167
SAFe のフレームワーク ..... 167
ディシプリンド・アジャイル・デリバリー（DAD） ..... 168
カンバン ..... 169
付録 B – アジャイル環境におけるリリースとよく使われるリリース管理ツールの詳細 ..... 170
リリースするかしないか、それが問題だ ..... 170
神話 1 ..... 170
神話 2 ..... 170
神話 3 ..... 171
神話 4 ..... 171
リリースガイダンス ..... 171
バーンダウンと見積もり ..... 172
ガントチャートを使ったリリース計画 ..... 173
書誌・参考文献 ..... 174

# 図の一覧

図 1 アジャイルは従来のプロジェクトマネジメントよりも早く、より多くの価値を生み出す ..... 20
図 2 リーンの原則 ..... 31
図 3 スクラムのすべてを示すハイレベルビュー ..... 45
図 4 スクラム活動の関係性を強調する（これはプロセス図ではありません。） ..... 51
図 5 プロダクトバックログの上位になるほど詳細度が増し、大きなストーリーは小さな（より実用的な）ストーリーに分解されます。 ..... 58
図 6 ストーリーとタスクチケットの例 ..... 64
図 7 プロダクトバックログ ..... 65
図 8 バケットシステムを適用したアクティビティへの見積もり工数の割り当て ..... 66
図 9 プロダクトロードマップの一例 ..... 67
図 10 プロダクトゴールとは？ ..... 71
図 11 トヨタ改善のカタのステップ ..... 72
図 12 詳細計画の信頼性 ..... 77
図 13 計画の層は、作業の実施に近づくにつれて、より詳細になります。アジャイルでは、すべての層で「ちょうどいい」計画を立てるという原則を採用しています。 ..... 80
図 14 エピックからタスクへ – 要求の分解 ..... 86
図 15 スクラムポーカーカード ..... 89
図 16 シンプルなスクラムボード（例：自動車のステアリングラックの製造）と、スイムレーンを使用してタスクを関連する PBI（ストーリー）にリンク ..... 101
図 17 ソフトウェア開発プロジェクトにおける典型的なスクラムボード（タスクが誰に割り当てられているかを示すために色付きの付箋を使用） ..... 102
図 18 WIP-limit によるボトルネックの回避とプルシステムの構築 ..... 107
図 19 KJ 法を用いたブロッカーチケットのグループ化 ..... 112
図 20 典型的なバーンダウンチャート ..... 113
図 21 典型的なスクラムチームのスペース ..... 114
図 22 Nexus vs リリースアプローチの比較 ..... 121
図 23 Nexus とスクラムの併用 ..... 123
図 24 Nexus 統合チームの作成 ..... 132
図 25 Nexus ボードの作成 ..... 134
図 26 Nexus スクラムチームに分散したタスクの依存関係への対応 ..... 135
図 27 チーム間の依存関係を利用する ..... 137
図 28 変化のイニシアティブにおける 3 種類のユーザーを表すスケール ..... 141
図 29 良いパイロットプログラムの選択 ..... 142
図 30 変化の抵抗に対処する方法を理解する ..... 145
図 31 一目でわかるエクストリーム・プログラミング（XP）の概要 ..... 160
図 32 XP でのループの計画 ..... 161
図 33 LeSS で使われているコンセプト ..... 165
図 34 DA Flex Workflow でのロールプレイヤーの連携方法 ..... 168
図 35 リリースバーンダウン・バー・チャート ..... 172
図 36 ガントチャートによるリリース計画 ..... 173

# はじめに

本書は、EXIN Agile Scrum Master と EXIN Agile Scrum Product Owner または Product Owner Bridge 認定スキームの公式なガイドブックです。

本書は、受験者が EXIN Agile Scrum Master、EXIN Agile Scrum Product Owner、および EXIN Agile Scrum Product Owner Bridge 資格の準備に役立てるという明確な目標を持って書かれています。

これらの認定は***アドバンスドレベルの資格***であるため、読者の方は、アジャイルとアジャイル手法としてのスクラムの基本的な概念をすでに理解されていることを前提としています。そうでない方は、Nader Rad 氏と Frank Turley 氏による ***Agile Scrum Handbook***、***Scrum Guide***、***Nexus Guide*** にも目を通されることをお勧めします。

EXIN Agile Scrum Foundation 資格は、本書の範囲内にある上級認定資格のための前提条件ではありませんが、スクラムの基本的な概念に精通していない場合には、Foundation レベルより進むことをお勧めします。

その他の EXIN アジャイル認定資格もございます。最新情報については、EXIN のウェブサイト https://www.exin.com/ja/certifications/ を参照してください。

# 序文

私がアジャイルに関わるようになったのはかなり昔に遡ります。20数年前に、もう一つのアジャイル手法であるエクストリーム・プログラミング（XP）を使い、ソフトウェア開発プロジェクトを担当したことから始まりました。私が大手 IT コングロマリットのインターネット スタートアップ企業を率いていたとき、異端のプログラマーの人達と一緒に XP を使っていました。それ以来、リーン、スクラム、ABC/DSDM、カンバン、DevOps の手法や概念を多くのプロジェクトで使い、また、コンサルティング、コーチングなどの仕事でも使ってきました。

適応型で経験的な方法（アジャイルスクラムなど）を使っていると、その自然な帰結として、自分の経験に基づいた独自の考え方や、それに対する先入観が生まれます。アジャイルであるためには、特定の手法を超えて、自分で考え自分で行動することが必要であると考えています。そうした経験をとおして独自の考え方を作り上げるのはとても素晴らしいことです。しかし、この書籍では、私の個人的な先入観（バイアス）や考え方を出さないように努めました。そのように完成できたと思っています。この書籍は、主にスクラムに関するものです。できる限りこの話題に集中するようにしています。

とはいえ、スクラム以外の手法や技法を使うことにより、スクラムが実にうまく機能することがよくあります。実例として2つの手法をここで紹介するとすれば、視覚的にイテレーションを管理することで、フローを実現するためのカンバンを使用することであり、そして、プロダクトバックログの順序付けを可能にするために、ABC/DSDM の中の MoSCoW を使用することでした。

この新しい第3版では、今まで含まれていたアンチパータンのいくつかを削除し、主にスケーリングの課題を扱っており、ここでは、Nexusを唯一のスケーリングとしています。

この書籍は、EXIN 試験の準備のためだけでなく、スクラムマスター、プロダクトオーナー、アジャイルコーチとして仕事をするためのリファレンスガイドとして、充分に実用的で実践的な内容になっています。あなたのお役に立てることを切に願っています。

ヨハン・ボタ、

2024年8月、ヨハネスブルグにて

# 日本語版製作にあたって

本書を翻訳するにあたり、多くの皆様のご尽力を得て、日本語版のリリースを迎えることができました。数か月に及ぶプロジェクトとなりましたが、無事リリースとなり、大変嬉しく思います。

アジャイル、スクラムにご経験豊富な実務者の皆様には、テクニカルレビューアとしてご参加いただきました。皆様の日頃の知見を活かして、技術用語を含めた英語文に対して、翻訳された文書をレビューいただきました。また、自発的に共有ポータルを立上げ、意見交換をされて取り組んでくださいました。

ウォーターフォールの上流からデリバリー、IT サービスマネジメント、アジャイル、スクラムを、広く深くご精通であり、尚且つローカライズのプロとしての教育者の方には、別の視点から、レビュー並びに最終的な日本語文章への落とし込みにご尽力をいただきました。

欧米のアジャイル、スクラムの在り方と、日本での取り組みにおける「カルチャー」のギャップ、そして、英語の持つ意味合いと日本語とのギャップの中で、特に監修者の方は悩まれ、ご苦労されました。

また、オランダ本社の開発チームの皆さん、特にプロジェクトマネージャの Ingrid Moleveld 氏には、本書の中国版製作との平行作業の大変な中、忍耐強く日本側の意向を汲みご協力を頂きました。深く感謝します。

こちらが、今回のプロジェクトでご支援を頂戴した皆様です。皆様の知見とご尽力に深く感謝いたします。

**テクニカル レビュー貢献者**

- 稲野 和秀（合同会社 JEI）
- 古野本 真之介（アクセンチュア株式会社）
- 佐田 裕輔（東京海上日動システムズ株式会社）
- 鈴木 隆文（株式会社クレスコ）
- 高田 紀子
- 竹内 徹（キンドリル合同会社）
- 中谷 和波（東京海上日動システムズ株式会社）
- 福田 朋紀（リコーIT ソリューションズ株式会社）
- 堀 孝夫（プロペラ&カンパニー）
- 宮本 ソアレスリノ(日本タタ・コンサルタンシー・サービシズ株式会社)

**監修者**

- 鈴木 寿夫（DIG2ネクスト株式会社）
- 原 清己（株式会社アイ・ラーニング）

（敬称省略、お名前の順番は「あいうえお順」にて）

EXIN JAPAN
代表 中川 悦子

# 1 アジャイル シーンの設定

## 1.1 アジャイル思考

**大文字「A」アジャイル**（名詞）:作業の短いフェーズをタスクに分割し、計画を頻繁に再評価させて適応させることを特徴とする、ソフトウェア開発によく使用されるプロジェクト管理の手法に関連するか、または意味する。

**小文字「a」アジャイル**（形容詞）:素早く簡単に動くことができる;素早く考え、理解し、対応することができる。

このシンプルな定義はオーストラリアの PM パートナーのウェブサイトに掲載されており、シーンを完璧に設定しています。

### 1.1.1 「プロジェクト」アプローチとしてのアジャイル（大文字 A のアジャイル（名詞））

アジャイルプロジェクト管理アプローチは、ソフトウェアの設計・開発のような経験的プロジェクトに非常に適しており、複雑なシステムを扱うどんなプロジェクトにも当てはまるでしょう。これらのプロジェクトでは、変動要素や未知の要因が多いため、従来のウォーターフォール型のアプローチは実用的ではありません。

アジャイルプロジェクト管理アプローチは、実験の結果としての発見の概念を重視し、進化するプロジェクトを実現するための手段と受け入れられています。

*アジャイル*の概念は、2001年2月11日、ユタ州の山中にあるスキーリゾート、Snowbird で初めて定義されました。この独創的な会議の成果は、今日では「アジャイル・マニフェスト」として知られる「アジャイル・ソフトウェア開発宣言」です。

この歴史的な会合には、現在も使用されている多くの一般的なアジャイル手法の代表者が参加しました。メンバー全員が、当時使われていたドキュメント駆動型の重厚なソフトウェア開発プロセスに代わるものが必要だと確信していました。

17 人の先見の明ある人達がアジャイル・マニフェストを策定した理由は、長いデリバリーサイクル、明らかに価値のない膨大な計画、周囲の状況変化によりプロジェクト期間を通して起こる継続的な要求変更、それらに対する失望でした。こうした失望から、マニフェストと12の原則が導き出されました。その中で、アジャイルが如何にして組織の価値を向上し、迅速化し、卓越したデリバリー品質を実現するのかを説明しています。

この内容は、程なく次のように理解されました。ソフトウェア開発プロジェクトで有効に機能するものは、要求が不明瞭で、徐々に大きく発展しているようなプロジェクトでも有効である、という事実です。

プロジェクトにおけるこのようなレベルの不確実性は広まっています。事実上すべてのプロジェクトは、多かれ少なかれ、曖昧さや要求変更に対処せざるを得ず、プロジェクトの進行に伴い、失敗から学ぶ必要があります。

したがって、先が予測できず、プロジェクトの初期に成果物を明確に定義できない環境、また、前に進みながら学ばなければならない環境に直面した場合、アジャイルとアジャイル手法が唯一の方法となります。

その理由は、組織やプロジェクトのような複雑適応系（complex adaptive systems: CAS）で起こる不確実性や曖昧さ、絶え間ない変化に対処するには、経験的なアプローチが最適な方法となるだからです。

em-pir-i-cal (ɛm-pîr-î-kl) adj. – 観察や実験を拠り所にしているか、または、そこから導き出され、あるいは観察や実験によって検証可能であること。理論ではなく実践的な経験によって導かれる。

従来の（いわゆるウォーターフォール型）プロジェクト管理の使用に限界があるのは、プロジェクトの開始前に詳細な計画を立てるべきであるという前提があるからです。

アジャイル手法は事前に計画を立てないという一般的な誤解があるようですが、実際はその逆です。事前計画に時間をかけなかったアジャイルプロジェクトを1つだけでも見つけ出すのは難しいでしょう。

アジャイルアプローチの違いは、知っていることに基づいて計画を立て、知らないことは記録しておき、実際にデリバリーを行う人が、その作業に従事するなかで、その質問に答えられるようにすることです。通常、その段階までには、前のステージで多くのことを学ぶことにより、それぞれの状況や文脈の中で何をしなければならないかを理解します。多くの場合、プロジェクトが進行しながら形が見えてくる中で、その全体的な状況や文脈に対する理解が深まっていきます。

プロジェクトの開始時に完璧で詳細な計画を立てることができると考えている人は、注意してください。すべてを予測しすべてを計画できるほど、頭がいいか、運のいい人はいません。

次に忘れてはならないのは、*プロジェクトは仕様よりも顧客の期待を重視しなければならない*ということです。事前に仕様をどれだけうまく定義しようとしても、顧客やユーザーの期待を捉えて定量化することはできません。

顧客、ユーザー、あるいは消費者は、受け入れるか否かを決める前に、具体的に見えるもので確認しなければならない場合があります。それは、期待通りのものか、そうでないものかの判断ができないからです。

この点について Henry Ford（ヘンリー・フォード）氏の言葉を借りると、もし私が顧客に何が欲しいですかと尋ねたら、彼らは「より速い馬がほしい。」と答えたでしょう。

顧客にとって重要なのは、アウトプット（何を提供するか）ではなく、「成果（提供したものを使って何ができるか、何を達成するか）である」とよく言われます。つまり、プロジェクトが提供する「何」（指定されたアウトプット）によって、顧客が望んでいることや必要としていること（成果）が実現できれば、顧客は満足するということです。

従って、価値は常にサプライヤと顧客の共創であることを忘れてはならないのです。

次の質問を考えてみましょう。

顧客と合意した仕様に非常に近い形で納品されたにもかかわらず、顧客に不満を抱かせてしまったプロジェクトに何度関わったことでしょうか？

この質問に対する答えは、おそらく、アジャイルのやり方を始める前では、殆どではないにしても、おそらく多くの場合はそうでしょう。

アジャイルを使うと顧客満足度が高くなる理由は2つあります。

- アジャイルでは、ユーザーや顧客をプロジェクトの一員として参加させることで、早い段階で頻繁にフィードバックをしてもらうことができる
- アジャイルでは、2年前の期待に応えるのではなく、2週間前の期待に応える能力を私達に与えてくれる

世界は変わり、顧客のニーズも変わります。

### 1.1.2 考え方や行動の方法としてのアジャイル（小文字 a のアジャイル（形容詞））

Snowbird での最初の議論の多くは、従来とは異なる仕事の進め方に関するもので、TQM（Plan Do Check Act（PDCA）サイクル）や Lean（あるいはトヨタ生産方式）の技法をソフトウェア開発にどのように適用するかに関するものだったことを覚えておくとよいでしょう。

アジャイルの12の原則を見ると、リーンの原則と酷似しているのも不思議ではありません。

意図したのは、ソフトウェア開発の1つの方法を開発することではありませんでした。それよりも、プロジェクトチームがどのような価値観を持つべきか（this OVER that）に焦点を当て、それをソフトウェア開発に適用するための一連の原則に変換しました。

実務者が原則から実践へと移行する必要があったため、その結果としてアジャイル手法が開発されたのは当然のことであり、現在のすべてのアジャイル手法は12の原則をガイダンスやモラル上の羅針盤として使用しています。

今こそ、アジャイルチームが大切にする 4 つの項目と、定義された12の原則を思い出す良い機会かもしれません。アジャイルの価値とは...

| | | |
|---|---|---|
| 価値1. | **個人と対話** | *プロセス*や*ツール*よりも |
| 価値2. | **動くソフトウェア** | *包括的なドキュメント*よりも |
| 価値3. | **顧客との協調** | *契約交渉よりも* |
| 価値4. | **変化への対応** | *計画に従うこと*よりも |

この価値を見ただけでも、顧客が価値を得られるように支援することが、プロジェクトで行う他のすべてのことよりも、重要であることは明らかです。これらの記述は、プロセス、ツール、文書、契約、計画が重要でないと言っているわけではないことに注意してください。価値は、人と話して仕事をすること、ニーズを満たすものを与えること、価値を提供する過程に人を巻き込むこと、彼らの状況やニーズがいつ変化したのかを知ることが、より重要であると述べています。

アジャイルの原則は、表明された価値観を基に、それを実行可能なものにします。それらは

原則1. **私たちの最優先事項は、価値のあるソフトウェアを早く継続的に提供することで、顧客を満足させることです。**なぜなら、顧客は、リリースの間に長い期間待たされるよりも、自分の成果を達成するために使用できるものを早く、頻繁に受け取る方が、より満足するからです。

原則2. **要求の変更はたとえ開発の後期であっても歓迎します。変化を味方につけることによって、お客様の競争力を引き上げます。**顧客の環境や状況が頻繁に変化すると、顧客のニーズも変化します。これは、スコープ・クリープではなく、現実そのものです。これに柔軟に対処しなければ、失敗します。ただ、新しい要求には大抵の場合、もはや有効でなくなった古い要求があることを覚えておいてください。

原則3. **動くソフトウェアを、2-3 週間から2-3 ヶ月というできるだけ短い時間間隔でリリースします。**これは原則1に触れていますが、全く同じではありません。ここでは、顧客が使用できるものをすぐに提供するだけでなく、予測可能な速度や周期で提供すべきだと述べています（原則 8 も参照）。

原則4. **ビジネス側の人と開発者は、プロジェクトを通して日々一緒に働かなければなりません。**利害関係者を早期かつ頻繁に巻き込みなさい。可能であれば、スプリントレビューだけでなく、すべてのデリバリーイテレーションに参加させなさい。私たちは、ソフトウェアのユー

ザーをスクラムチームの一員にすることもよくあります。その人以上に要求を知っている人はいません。

原則5. **意欲に満ちた人々を集めてプロジェクトを構成します。**彼らに環境と支援を与え仕事が無事終わるまで彼らを信頼します。これは、組織内の文化的変化なしにアジャイルを使おうとすると、おそらく失敗するだろうというヒントです。

原則6. **情報を伝えるもっとも効率的で効果的な方法はフェイス・トゥ・フェイスで話をすることです。**コミュニケーションの 80%は非言語的なものであることを忘れてはいけません。私たちは、物理的に繋がりの少ない世界に対応することを学びつつありますが、開発チームにおいては、フェイス・トゥ・フェイスの対話が基本です。注：ポスト Covid-19 の世界では、バーチャルチームで働くことが新しい現実です。しかし、可能な限りフェイス・トゥ・フェイスでのコミュニケーションを行うことが推奨されており、バーチャルチームのミーティングはビデオを使って行うべきです。

原則7. **動くソフトウェアこそが進捗の最も重要な尺度です。**それは仕事をすることができますか？それは使えますか？そして、意図した結果を生みますか？もしそうでなければ、失敗したことになります。

原則8. **アジャイルプロセスは、持続可能な開発を促進します。**スポンサー、開発者、ユーザーは、いつまでも一定のペースを継続的に維持できるようにしなければなりません。これは、原則 1 と 3 に密接に関連する原則です。アジャイルでは、プロジェクト全体の「いつまで」はありませんが、チームが今取り組んでいることは「いつまで」があるはずです。ビジネス環境が常に変化することに加えて、要求も変化します。つまり、アジャイルプロジェクトに終わりはないのです。

原則9. **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ます。**つまり、チームが適切なスキルを持ち、優れた設計のルールを守り、技術的負債を増やさないようにすることが重要です。

原則10. **シンプルさ（ムダなく作れる量を最大限にすること）が本質です。**この原則は「ゴルディロックスの法則」とも呼ばれ、すべてが丁度よくなければなりません。この原則は、品質と詳細にこだわる原則 9 とのバランスをとるためです。しかし、「多過ぎるのはよくない」ということを忘れてはなりません。我々は必要なことは何でもしなければなりませんが、過度すぎず不足すぎない必要なことをします。

原則11. **最良のアーキテクチャ・要求・設計は、自己組織的なチームから生み出されます。**この原則は、これからアジャイルを取り組み始めようとする組織にとって最も難しいものです。それは、マネジメントスタイル、パフォーマンス基準、そして時には役割と責任の再定義を含む組織構造の変更を意味するからです。ここでのポイントは、細部はその仕事に従事しているスペシャリスト、つまりチームメンバーに任せるということであり、マイクロマネジメントをしてはいけません。今日の従業員の多くは知識労働者であり、管理と監督は必要ありません。必要なのはモチベーションとサポートです。

原則12. **チームがもっと効率を高めることができるかを定期的に振り返り、それに基づいて自分たちのやり方を最適に調整します。**これは、チームが成熟することと、自分のことは自分で決めるという意思を示しています。これを実現させるためには、組織が組織改革に取り組むためのリーダーシップを持ち、お互いの高い信頼関係を構築しなければなりません。

小文字'a'のアジャイルとは、おそらく最も重要であると思える顧客中心の原則によって導かれたマインドセットのことです。と言える理由がもうお分かりいただけたでしょう。小文字の「a」をもつアジャイルは、文化的、組織的、行動的、振る舞いに関する包括的な変化を慎重に起こすことであり、組織のリーダーシップの階層から始めるべきものです。

本書に書かれている全ての技法を使っても、アーティファクト（作成物）を作成しても、役割の名前を変えたり、儀式を実践したりしても、大文字「A」のアジャイルを組織で成功させることはできません。なぜ多くのアジャイル・イニシアチブが失敗するのかを考えてみると、間違いなくその答えは、大文字「A」のアジャイルがあなたの組織で機能する前に、あなたが小文字の「a」のアジャイルになっていなければならないということです。

うまく機能しそうなモデルや手法に固執して、その手法を推奨し始めるのは自然なことです。これらの手法はすべて、大文字「A」をもつアジャイルです。しかし、12の原則を組織内に展開し、手法に関係なくビジネスアジリティを達成することを意図していれば、小文字「a」のアジャイルを実践していることになります。これは成功をもたらす可能性が高く、アジャイル・マニフェストの精神に最も近いものです。

では、なぜこの本が必要なのか、ということになります。

ここで紹介するアジャイルやスクラムの手法、説明責任、そして技法は非常に価値があり、あなたがそれらに使われない限り、組織に大きな結果をもたらすでしょう。それらは有用な補助手段であり、あなたの役に立つでしょう。しかし、どのアジャイル手法であれ、何をどのように行うかを指示してはいけません、そうなれば、真の組織のアジリティが失われてしまうからです。大文字「A」のアジャイルは、常に小文字の「a」のアジャイルに仕えるべきであり、その逆ではありません。

本書は、アジャイル手法としてのスクラムに関する実践方法とスクラムから最大の効果を得る方法に焦点を当てているため、組織の広い意味での「アジャイルな振る舞い」、「アジャイルな考え方」、「アジャイルな原則」に言及している特定の状況を除いて、ここから先は大文字「A」のアジャイルで表記します。

## 1.2 アジャイルの論証

アジャイルの便益を売り込むのは最初は難しく、個人レベルで行うのが最善です。この短い章では、一つの可能なアプローチを紹介しますので、検討してみてください。これはうまくいくことが証明されています。

懐疑派と擁護派の2人を主人公とした対話型のストーリーやロールプレイという形でアプローチしています。この物語は、IT プロジェクトの物語ですが、シナリオや使用するパラメータを変更することで、どのような環境にも簡単に適応することができます。多くの人がオンライン販売への取り組みの成功と失敗を目の当たりにしてきたので、この物語は適切な例と言えるでしょう。背景を説明します。

XYZ Inc.は、FMCG（動きの速い消費財）市場で成功を収めている小売企業です。過去10 年間で成功を収め、全米10 大都市圏に2店舗から15 店舗へと小売事業を拡大しています。

最近、XYZ はオンライン小売業者からのプレッシャーを受けており、同社は広範なオンラインマーケティングを行っていますが、経営陣は急速にオンライン小売能力（ウェブショップ）を開発する必要があると考えています。これを成功させることができれば、既存の市場での地位と関係を活用して、新たな競争相手を打ち負かすことができるでしょう。

望ましい目標を達成するために、2つのオプションを提示して検討することができます。

**オプション1：従来のウォーターフォール型プロジェクトアプローチ**

- 競合他社の動向を調べる （2ヶ月）
  - IT 部門が実施
- 利用可能な技術を調べる （2ヶ月）
  - IT 部門が実施
- オンライン小売機能の詳細仕様を定義し、ビジネスケースを作成する （4 ヶ月）
  - IT 部門のビジネスアナリストがビジネス要件を収集する
- オンライン小売機能の設計 （4 ヶ月）
  - IT 部門が実施
- オンライン小売機能の開発 （12ヶ月）
  - IT サイロで実施：開発、テスト、リリース管理、IT 運用
- **プロジェクトの総時間** **24 ヶ月**

ビジネスへの質問

1. 上記の内容は、あなたのビジネスにおけるこの種のプロジェクトを適切に表していると思いますか？
   - 通常、答えは**「イエス」**
2. 前述の競争の激しいシナリオの中で、解決策を得るために2年も待つことができるでしょうか？
   - 通常、上記の答えは**「ノー」**
3. 2年待たされたらどうなりますか？
   - 通常、その答えは悲観的なものになる
4. この新規参入者と競争できるようになるには、どのくらいの期間が妥当だと考えていますか？
   - 通常、4 ヶ月から 6 ヶ月の範囲が答え
5. 上のシナリオの 5 ヶ月目に定義され、合意された要件は、19 ヶ月後、プロジェクトがうまくいけば完了するときにも同じだと思いますか？
   - 通常、上記の答えは**「ノー」**
6. プロジェクトの 8 ヶ月目に承認のために提出されたビジネスケースが、16 ヶ月後にも有効だと思いますか？
   - 通常、上記の答えは**「ノー」**
7. 経験上、この種のプロジェクトは予算内で期限内に完了しますか？
   - 通常、上記の答えは**「ノー」**
8. どのような妥協案としたいですか？
   - ここでの回答は様々ですが、一般的には次のような案が挙げられます（重要度の順ではありません）
     a. より短い時間でより少ない範囲の製品を
     b. 他の業者にショップをアウトソーシング
     c. 既存のプラットフォームの活用
     d. より早く終わらせるために、より多くのお金を使う

b.と c.は有効な選択肢であり、検討すべきですが、どちらも明確なデメリットがあります。例えば、顧客に対するコントロールが失われたり、利益率が低下したりします。より多くの資金を使うことは、会社が現金を持っている場合には可能かもしれませんが、経験上、このオプションは実行できないか、少なくとも一般的でないことが多いのです。

**オプション 2 – アジャイルプロジェクトのアプローチ**

- 組織のニーズと競合他社の動向に基づいて、一般的なハイレベル要件を特定する
  （1ヶ月）
  - IT の助けを借りて、ビジネスで実行
- オンラインで最も売れる可能性の高い製品を 1 つか2つ特定する
  （2週間）
  - IT の助けを借りて、ビジネスで実行
- これらの製品をオンラインで提供するために必要な最低限の機能を特定し、調査する
  （2週間の設計スプリント）
  - ビジネスの助けを借りて IT が行う
- 基本的なショッピングカート機能の構築　（4 週間のスプリント）
  - IT で実施
- ユーザーインターフェースの構築とテスト　（4 週間のスプリント）
  - IT で実施
- ソリューションの改良　（2週間のスプリント）
  - IT で実施
- 基本機能の始動
- オンライン対応までの総時間　（4 ヶ月半）
  オンラインショップの機能を強化し、今後 1 年間でオンライン販売が可能と思われるすべての製品を最終的に追加する　（4 週間のスプリント）
  - 購入者/消費者の行動に関するビジネスからのフィードバックに基づいて IT が実施
- 洗練された豊富な機能を備えたオンライン能力が対応するまでの時間
  **18 ヶ月**

ビジネスへの質問

1. 発売日にオンラインショッピング機能がすべての要求を満たさなかったことを満足していますか？
   - 通常、上記の答えは**「ノー」**
2. あと19 ヶ月待つことになっても、すべての要件を満たしたいと思いますか？
   - 通常、上記の答えは**「ノー」**
3. 発売後 18 ヶ月の間に、要求や要求に対する理解が変わると思いますか？
   - 通常、上記の答えは**「イエス」**
4. 過去の典型的な2年間のプロジェクトで定義されたすべての初期要件は、必要または有効であることが証明されていますか？
   - 通常、上記の答えは**「ノー」**

過去のプロジェクトの経験を振り返り、プロジェクト定義からプロジェクト実施までの2年間の期間を経て

5. システムのアップデートを依頼しましたか、その時の理由は？
   - 通常、上記の答えは**「イエス」**。新しい要件の出現、市場、顧客、競争状況の変化などです。
6. 2年間でいくつの新しい要件が出てきましたか？
   - この答えは、業界によって異なるかもしれないが、平均すると**かなりの数**になります。
7. 現在の機能の中で、必須と思われるものはいくつありますか？
   - この答えは業界によって異なるかもしれないが、平均すると全てではないが、**ほとんどが**そうです。
8. 与えられた2つのシナリオのうち、もしあなたが XYZ 社の経営者だったら、どちらの選択肢を選んだでしょうか？
   - ほとんどの場合、答えは**オプション2です**。

ここで、実用的な図解として簡略化したイラストを紹介します。お金と同じように、時間にも価値があり、価値と時間の方程式を比較すれば、アジャイルは当然の選択です。

アジャイルは、可能性としてすべての組織に適しています。組織の考え方や仕事のやり方を変えるためには、他の組織よりも多くの課題に直面するところもあるでしょうが、誰でも可能です。そうしないと、この変化の激しいデジタル時代に生き残れない可能性が高いのです。

しかし、カルチャーの観点で、アジャイルはすべての環境でうまくいくわけではなく、**少なくともトップマネジメントがアジャイルの考え方を受け入れようとする意思がない限り**、アジャイルの導入はしないほうがよいでしょう。

*図 1 アジャイルは従来のプロジェクトマネジメントよりも早く、より多くの価値を生み出す*




図は EXIN 制作：Botha、J.（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

トップマネジメントが自分自身に問い掛けるべき質問とは、ビジネスアジリティの概念を取り入れないと、我々は失敗してしまうのか？それに「イエス」と答えることは、事実を直視する意思があることを意味します。アジャイルになることは簡単なことではありませんし、簡単な決断でもありません。それをやり遂げるには大胆さと勇気が必要です。しかし、幸運は勇気ある者に味方します。

## 1.3 アジャイルを正しくするための重要成功要因

アジャイルとは、短い反復型のデリバリーサイクルを用いて顧客に価値を提供するためのアプローチです。顧客の関与を高め、成果物の品質に責任を持つ機能横断的なチームに依ります。これ以上のことを説明すると、特定のアジャイル手法を掘り下げることになります。

一般的には、どのアジャイル手法を使っても、いくつかの重要成功要因を定義することができます。

## 1.4 リーダーシップと行動、カルチャー、倫理観、信頼感

どんな組織でもアジャイル手法を使うことができますが、適切な環境がリーダーシップによって作られたなら、アジャイルは最もうまく機能します。

アジャイルは、リーンと同様に、他の何よりもまず、カルチャーと行動（振る舞い）に関する現象のことであり、本質的には、アジャイルが提唱する行動（振る舞い）の変化は、組織のリーダーがそれを理解し、それを可能にすることをコミットした場合に、最も効果的に機能します。このようなカルチャーの変化は、組織内のリーダーシップのスタイルによってかなり大きな影響を受けます。

過去20 年間、人々はサーバントリーダーシップについて多くのことを語ってきましたが、マネジメントの行動、振る舞い、管理方法はほとんど変わっていません。アジャイルの導入を組織で成功させるのは、何よりも、組織のあらゆるレベルの管理者の姿勢や態度の変化です。

何故これが重要なのかというと、自律的に自己管理する機能横断的なチームがアジャイルプロジェクトを遂行するからです。そのためには、マネジメントやリーダーシップのスタイルを根本的に変える必要があります。

Michael Shota 氏の言葉、リーダーシップが理解すべきことは、「アジャイルを行う（Doing Agile）のではなく、アジャイルである（Being Agile）ことが重要であり、アジャイルである（Being Agile）ことは組織のリーダーシップから始まる」ということです。

アジャイルの導入に成功している組織のリーダーは、従業員がどのようにタスクを実行するかに真の関心を示し、従業員の成長を促し、監督者ではなくコーチ、教師、推進者（イネーブラー）として行動します。

アジャイルの導入を成功させるためには、**信頼性**と**オープンな**環境が必要です。つまり、マネジメントの全スタッフに対する揺るぎないコミットメント、組織で働く人々が可能な限り最高の力を発揮できることを可能にしたいという心からの願望が必要です。

また、コミュニケーションが**オープンで、率直で、公正な**ものであり、従業員が組織のリーダーシップや組織の関係者と関わりをもつ時、彼らを信頼できることが必要です。

信頼のあるオープンな環境をどのようにして作りますか?と尋ねられたら、私たちは次のように答えます。あなたの部下が正しく仕事をすると信じること。信頼することをマネジメントは学ばなければなりません。

従業員がマネジメントを信頼し、彼らが新しいことを試しても、真実を話しても、起こったことを正直に話しても、自分たちは罰せられないと思わない限り、アジャイルやリーンを機能させるために必要なオープンで率直なコミュニケーションを、行うことはできません。

Ernest Hemingway 氏の定義によると、「誰かを信頼できるかどうかを知る一番の方法は、その人を信頼することだ。」

信頼度の高い環境が、信頼度の低い環境に比べて、なぜ組織にとって有益なのか、その例を次に示します。

**低い信頼性**
マネジメントはスタッフを信頼せず、仕事をうまく行うとは思っていません。それゆえ、スタッフに仕事のやり方を指示します。多くの場合、マネジメントが指定した方法は非実用的であったり、時代遅れであったり、面倒であったりするので、スタッフは要求されたアウトプットを作成するためにやり易い方法で仕事をします。

このシナリオで危険なのは、規定の方法、プロセス、手順の中に組み込まれたセーフティネットやコントロールの機能が、抜け落ちてしまうことです。

もしも、何か問題が発生しても、誰もプロセスに従わなかったことを認めず、ミスに対する罰を「逃れよう」として、ミスを隠すことがしばしば起こります。

その結果、環境はさらに不安定になり、最終的には大きな破綻をきたします。原因究明の一環として、犯人を探し出して処罰するための犯人探しが行われます。

問題は、たとえ罪を犯した人が発見されて罰せられたとしても、組織やマネジメントが損害やダメージを受け、失うものだけ（得るものはない）ということです。驚くべきことに、多くの組織ではこのような悪い影響を及ぼす行為が行われています。

Richard DiPilla 氏は次のように述べています。チェスプレーヤーを雇って、彼らをチェスの駒のように扱うのは意味がありません。賢い人を雇って、愚かなルールに従わせることは意味がありません。

では、上記とは対照的に、アジャイル環境で期待される行動を探ってみましょう。

**高い信頼性**
マネジメントはプロフェッショナルな人材を採用し、彼らが組織の向上のために、そのスキルと才能を使うことを期待しています。3～9 人までの小さなチームのスタッフは、目標を与えられ、それをどのように達成するか、また設定された期間内に目標を達成するためにどのように仕事を構成するかを自分たちで決めます。

チームメンバーは現場に近いので、現実的にどのように動いているのか、何をしなければならないのかをよりよく理解しています。

彼らは、最適なパフォーマンスを達成しながらリスクを最小限に抑えるために、マネジメントが定義したヒューリスティックなコントロール[1]を利用して、自らの仕事を設計します。

何か問題が発生した場合、チームはオープンにコミュニケーションをとり、再発を防ぐために恒久的な解決策を見つけようとします。

マネジメントは、アドバイスをしたり、邪魔にならないように障害を取り除き、チーム間のファシリテーターや仲介者になったりして、チームをサポートします。

失敗は人生の一部として受け入れられるが、それが続くと容認されないため、問題は継続的に解決され、壊滅的な失敗にまで発展することはほとんどありません。このシナリオでは、常に組織的な学習が行われ、パフォーマンスが継続的に向上します。

- 2つの職場環境のうち、あなたが最も望ましいと考えるのはどちらですか？
- どちらで働きたいですか？

上記の例から、アジャイル型の組織で行わなければならない重要な文化的・行動的な変化の 1 つが、**学習する組織**になることであることに気づいたと思います。問題を見つけたら恐れずに取り組み、それが組織の中で問題になる前に解決するような、信頼の高い環境への転換が必要です。

学習する組織という考え方は、Peter Senge（ピーター・センゲ）氏が著書『*The Fifth Discipline*』（1990 年）（邦訳本：最強組織の法則）で広めたもので、この本を参考にする価値は十分にあります。彼は、学習する組織の特徴として次の 5 つを提唱しました。

1. **システム思考**とは、全員が組織をシステムとして理解することを意味します。全員がシステムを理解すればするほど、より多くのことを学ぶことができ、その結果、組織はより良くなっていくのです。

[1] ヒューリスティックコントロールは、一般的に、ある状況下での行動のルールを定義するもので、行動を定義するものではありません。どのように行動すべきか、どのような結果が期待されるかを知っていれば、最も適切な行動を決めることができます。しかし、コントロールが行動を定義している場合、適切かどうかに関わらず、その行動は常に取られることになります。

2. システム思考とは、**学習と個人の熟練**が組織の規範となるような活動に個人がコミットすることです。学習は単に知識習得すること以上のものです。それは、より生産性を高める能力に集中するようになります。個人の学習は個人の責任であり、組織だけの責任ではありません。また、個人の熟練は日々実践されるべきものです。
3. 人々が抱いている前提（**メンタルモデル**）を変えなければなりません。そのためには、対立的な態度ではなく、探求と信頼を重んじるオープンな文化が必要です。
4. **共有ビジョン**の策定は、学習に対する集中と活力を生む集団アイデンティティを作り出すので、スタッフにとって自己学習のための不可欠な動機となります。共有ビジョンを実践することで、組織内のコミュニケーションとコラボレーションが生まれ、信頼関係を構築するのに適した環境を作り上げ、スタッフが自らの経験や意見を共有することが促され、その結果、組織的な学習の効果を高めることができます。
5. **チーム学習**は、蓄積された個人の知識を共有することで生まれます。チーム学習や共有学習の利点は、スタッフがより早く成長し、（機能横断的な）知識や専門知識へのアクセスが良くなることで、組織の問題解決能力が向上することです。個々人は、共通の意味や理解をするために、対話や議論をしながら仕事に取り組みます。彼らは、知識を共有するための明確な方法を持たなければならなりません。

### 1.4.1 アジャイルとその組織構造への影響

理論的には、アジャイルはどんな組織でも使うことができます。うまくいけば、アジャイルは必然的に組織の仕組みを変え始め、構造に影響を与え、役割と責任の定義に最も影響を与えることになります。

なぜかと言うと、アジャイルは、自律的で機能横断的なチームがプロジェクトを遂行するため、必然的に組織がフラットになり、複数のスキルと複数の才能を持ち、管理されることを必要としない従業員が増えていきます。

これからの組織は、そのようなものになると考える強い理由があります。職場の多くの人が知識労働者になるにつれて、複雑な管理構造の必要性が低くなってきているからです。

# 2 アジャイル シーンの採用

現在、多くのアジャイル手法が存在します。スクラムは最もよく使われている手法です。本書の付録 A には、その他の一般的手法が紹介されています。

リーンは、すべてのアジャイル手法や技法における共通の起源です。これらはすべて共通の要素を共有しており、ここではアジャイルとスクラムに焦点を絞っているつもりですが、時折、スクラムに特有ではない方法や技法も含んでいます。

アジャイルスクラムを組織で使い始めたら、他の手法も検討するべきです。何故なら、使えそうな素晴らしものがあるかもしれないからです。

本書で定義されているプラクティスや原則、アーティファクト（作成物）の使用は、アジャイルを使用していると言えるための最低限のものです。

とはいえ、少なくとも最初は、1 つの手法にこだわるのがよいでしょう。スクラムは、そのシンプルさゆえに、スタートするのに非常に適しています。

スタートしてしばらくは 1 つの手法にこだわる理由は、アジャイルを文化に根付かせることに向けて、人や組織の変化の要素に注力する必要があるからです。

役に立つかもしれないアーティファクト、ツール、そして、技法などに注目しすぎると、本当に重要なことから焦点がずれてしまいます。アジャイルは、まずマインドセットを変え、次に組織を変え、人々の行動を変えるものです。

スクラムのすべては、こうしたアジャイルの旅を促進するために特別に設計されています。すぐに注意をそらしてはいけません。気をそらすと破壊的な状態になり、失敗の原因になります。他のフレームワークの方法と技法を導入し実験する前に、組織の一般的な考え方が変わり、新しい働き方への抵抗がなくなるまでは、基本に忠実であることに拘ることをお薦めしています。

アジャイルを組織で機能させるには、それに専念することと集中力が必要です。

本書の付録では、他の手法についても紹介します。今のところ、アジャイル手法としてスクラムに焦点を当てています。

## 2.1 アジャイルを採用する上での課題

リーンの場合と同様に、アジャイルでも組織全体で価値を生み出すことを理解することが重要です。それは、ひとつの製品、サービスのプロセス、あるいはチームに関することではありません。組織全体が機能横断的にプロセスを統合し、製品やサービスという形で顧客のために価値を創造しなければならないのです。

リーンでは、ビジネスの機能やプロセス全体に渡ってバリューストリームをマッピングします。プロダクトやプロダクトバックログだけに注目するのではなく、これらをビジネスのより広い文脈の中で捉えることが重要です。

同様に、アジャイルでは、ソリューションを生み出すチームが、多様な働き方、多様な専門性を持っていなければ、顧客が求めるような価値を生み出すことはできません。

しかし、クロスチームでのデリバリーを成功させるためには、組織の階層構造や既得権益などの問題に対処することになります。

スクラムの生みの親は次のように述べています。スクラムは、理解するのは簡単ですが、習得するのは難しいのです。

難しいのは、アジャイルやスクラムをうまく効果的に使うためには、組織の中で多くのことを変えなければならず、その変革は本当に難しいということです。アジャイルやスクラムの導入は、大規模なチェンジマネジメントの取り組みであり、過小評価してはならず、慎重に計画して実行しなければなりません。

チェンジマネジメントにはさまざまな方法や技法があり、どれを使うべきかを規定することはできませんが、ある方法がうまくいった場合は、その方法を使い続けてください。革命的な変革よりも進化的な方法の方がはるかに成功率が高いことは、これまでの経験で証明されています。

従って、アジャイルの考え方（マインドセット）の部分に関しては、忍耐強く進めることです。変化を急がないでください。ゆっくり、ゆっくりと近づきながら獲物を捕まえるごとくです。

ビジネスのチェンジマネジメントは、この本の焦点ではありませんので、主流の変更管理アプローチはここでは扱いません。しかし、ある環境に初めてアジャイルを導入する際に考えられる2つの代替アプローチをハイレベルで説明することは有益です。

## 2.2 変化へのマネジメント

変化は避けられないため、変化への抵抗は無駄であり、価値を段階的に提供するアジャイル手法を適応する際も同様です。

組織が「人」の側面の変化に対処しなければ、アジャイルの導入は失敗するでしょう。

Jeffrey M. Hiatt 氏と Tim J. Creasey 氏は、著書「*Change Management: The People Side of Change*」（2012年）では、チェンジマネジメントを、反復可能なサイクルに沿った総合的なツールセットを使用するプロセス、そして、変化を可能にし、組織の有効性を高めるための能力を生み出すことから、コンピテンシーと定義しています。

彼らはチェンジマネジメントの 5 つの原則を定義し、人々が変化に対処し、変化を受け入れるのを助けています。

**1.人が変わるには理由がある**

特定の望ましい成果を伴う未来の状態を達成するために変化しています。変化の理由は組織によって大きく異なり、コスト削減、顧客満足度の向上、アジャイルの場合は顧客にとっての価値の向上などが考えられます。

変化の必要性は、常に解決すべき機会や問題によってもたらされます。

**2.組織の変革には個人の変革が必要**

組織変革を実現するには、新しいツールやプロセスだけでは不十分で、組織内の個人が新しい価値観を持ち、新しい方法で仕事を始める必要があります。

**3.組織の成果は、個人の変化の集合的な結果である**

変化は個人的な出来事なので、考慮すべき人的要因があります。つまり、組織は従業員が変化を受け入れられるほど、望ましい成果の達成に近づきます。

**4.チェンジマネジメントは、変化する個人の側面を管理するためのフレームワーク**

組織において、特に変化の激しい環境では、変化に対する抵抗が常態化しています。人的側面から変化を管理することで、導入のスピードが上がり、習熟度が向上します。

**5.変化の恩恵と望ましい成果を実現するために、チェンジマネジメントを適用**
チェンジマネジメントの主な目的は、望ましい将来の状態の実現と期待される成果を推進し、支援することです。

## 2.3 ADKAR®と ADAPT

Hiatt 氏と Creasey 氏は、個人レベルでの変革を促進するためのモデルを定義し、「ADKAR モデル」と名付けました。ADKAR Model は、Awareness、Desire、Knowledge、Ability、Reinforcement の頭字語です。すべての従業員が 5 つのマイルストーンをすべて達成したとき、初めて変革は成功します。

本書で紹介されている手法の中では、ADKAR または ADAPT 手法が、伝統的なチェンジマネジメントの手法に最も近いと考えられます。

2つ目は、リーンマネジメントの原理と手法ですが、これについては後述します。

ADAPT[2]は、PROSCI®が開発したチェンジマネジメントの代表的な手法である ADKAR モデル[3]をベースに、オリジナルのアプローチから僅かな変更を加えたものです。

ADKAR は、その頭文字をとったものです。

- 認知（Awareness）
- 願望（Desire）
- 知識（Knowledge）
- 能力（Ability）
- 強化（Reinforcement）

スクラムの第一人者である Mike Cohn 氏は、ADKAR モデルを ADAPT に適応させました。仕事のほとんどが知識労働である環境では、知識（Knowledge）と能力（Ability）は切り離せないものだと主張したからです。そのため、彼はこれらを「能力（Ability）」という見出しでまとめました。

また、Cohn 氏は強化（Reinforcement）があまりにも曖昧な言葉であると主張し、結局、Promotion と Transfer という2つのステップに置き換えました。

したがって、ADAPT は、その意味を表しています。

- 認知（Awareness）
- 願望（Desire）
- 能力（Ability）
- 促進（Promotion）
- 移行（Transfer）

ここからは、ADAPT のハイレベルな説明になります。

**認知（気付き）**
現状維持が望ましくないと理解して初めて、真の変革が始まるのですが、これまで上手くいっていたことが上手くいかなくなっていることに、気づくのは非常に難しいことです。

人は一般的に、変化の必要性を認知するのに時間がかかります。これは多くの場合、全体像を把握していないことが原因です。

---

[2] Cohn, M.（2010）. Succeeding with Agile: Software Development Using Scrum. Addison-Wesley.第 2 章

[3] ADKAR モデルの詳細については、https://www.prosci.com/methodology/adkar

スクラムを採用したいという要望は、誰にも見えないてない要素が重なった結果かもしれません。しかし、変革の必要性がごく一部の人（例えば、新規顧客への売上減少に直面した人）だけにでも明らかになると、変化についての噂がそこから広まり始めるでしょう。

「変革」が組織にとってネガティブな行動や結果に繋がらないようにするためには、組織の全員が「変革」を促す事実を認知することが極めて重要です。

認知が憶測ではなく事実に基づいていることを確認するためには、問題を明確に伝え、客観的に測定された事実や測定基準で根拠を示すことが必要です。

変革によって問題がどのように解決されるかを周囲に説明し、実際にパイロットプロジェクトを実行して、変革によって実際に問題が解決されたことを証明することをお勧めします。

組織がアジャイルを導入したいと思う理由は様々あるかもしれませんが、メッセージは明確かつシンプルに、最も重要な理由に絞って、2つか 3 つ、それ以上は考えないようにしましょう。

**願望**

現在のプロセスが組織にとって機能していない、あるいは顧客にとって望ましい結果をもたらしていないという認知から行動に移すことは、人々にとって受け入れがたいことかもしれません。

プロジェクトマネジメントに関して言えば、これは間違いなく事実であり、私達の多くは、プロジェクトの伝統的なやり方を教えられたり、経験したりしてきました。

今まで効果があったのに、何故、今は変わったのか？

実際のところ、人々と少しの時間を過ごし、プロジェクトのネガティブな経験について話してみると、常に上手くいっている訳ではないことが、すぐに明らかになります。

経験したネガティブな結果を対比させ、アジャイルがそれを否定するのに役立つことを説明したり紹介したりすれば、よりポジティブな結果を求める気持ちが生まれてきます。

例えば、今必要なものと2年前に必要だったものが同じかと聞くと、「違う」と答えるでしょう。ウォーターフォールのプロジェクトでは、提供されたものが時代遅れであったり、不要になったり、現在のニーズに全く合っていないものであったりすることも認められます。

そして、もはや必要とされていないものを提供する古い方法と、アジャイルアプローチを対比させるのは比較的簡単です。アジャイルを使えば、プロジェクトチームは常に現在の要求に焦点を当て、短いサイクルで現在の要求に応えることができ、「はい、これは私たちが望んでいることであり、これは古いやり方よりも優れています」と言うのに大いに役立つでしょう。

変革への願望を生み出すには、より良い方法があり、組織がより良い結果を早急に必要としていることをステークホルダーに示すことです。アジャイルへの変革をきちんと説明すれば、ステークホルダーは同意するでしょう。

ただし、注意が必要です。過去がすべて間違っていたかのように振る舞うのではなく、本当に差し迫った、あるいは即時の変化をもたらす新しい利点だけに焦点を当ててください。さらに言えば、試験的に導入して、それがうまく機能することを示してください。必ず成功させてください。

物事が変わり始めたら、人々が新しい方法に変わることの難しさを乗り越えられるように手助けをします。忍耐強く、人々が前に進むのを助けます。ステークホルダーに、変化はそれほど恐ろしいものではなく、とても安全なものであることを示します。

可能な限り広くステークホルダーを巻き込み、時には変化へのインセンティブも検討します。しかし、変化の推進力となるインセンティブは、変化が起こる背景やカルチャーに左右されます。この点に敏感になってください。

Tom Peters 氏は、「世界で最も困難なことは、新しい考えを受け入れることではなく、古い考えを忘れさせることである」という有名な言葉を残しています。

古い考えを手放してもらうためには、それに取って代わる良いものを見せるしかありません。

認知と願望があっても、アジャイルの能力を身につけていなければ、チームは何もできません。

**能力**

先ほどの Tom Peters 氏の例のように、スクラムで成功するためには、チームメンバーは新しいスキルを学ぶだけでなく、古いスキルを捨てなければなりません。

スクラムチームが直面する大きな課題には、次のようなものがあります。

- 新しい技術スキルの習得
- チームとしての考え方や働き方の習得
- 自己管理の概念
- 1週間から 4 週間の短い一定の時間サイクル（タイムボックス）の中で、機能する製品やサービスを生み出す方法を習得

そのためには、適切なトレーニングを行うことが重要です。しかし、トレーニングだけでは十分ではなく、最初のうちはチームには付き添いとコーチングが必要です。そのためには、スクラムの導入に成功した経験と実績が豊富な社外の人に、チームのアジャイル・スクラム・コーチになってもらうのが望ましいです。

すぐに多くを期待してはいけません。また、新しい方法で正確に物事を行うことに責任を持たせたいと思っても、無理な目標を設定してはいけません。

検査、測定、フィードバックを行い、問題や課題の解決にステークホルダーを巻き込みます。

どれだけトレーニングを受けても、あなたとあなたのチームがそれを実行できるようになるわけではないことを覚えておいてください。現場で学び、失敗から学ぶことから始めなければなりません。それは学習の一部です。スクラムチームがスクラムに慣れていないときは、スクラムやアジャイル実践経験が増えるまで、ルールを厳格に守ったほうがいいでしょう。

**促進**

あなたは、自分の会社が商品の販売促進をしなければ売れないことを知っています。変化も同じです。変化を促進するために、変化の販売促進をします。

新しい手法（この場合はアジャイルスクラム）が適切に組織に組み込まれ、制度化されるように、新しい手法の利点や過去を手放すことの利点を絶えず売り込まなければなりません。

促進の段階では、3 つの目標があります。

1. おそらく小さく始めることになるでしょうが、パイロットプロジェクトは、アジャイルスクラム導入の次の部分の基礎を築くものでなければなりません。現在の成功を宣伝し、新しいアプローチに対する認識を高めることで、次の改善に向けて弾みができます。
2. 2つ目の目標は、既存のチームが達成した良いことのニュースを広めることで、既存チームのアジャイル行動を強化することです。
3. 3 つ目の目標は、スクラムの採用に直接関与していない人々の間で、認知度と関心を高めることです。

促進とは、成功例を公表し、他者が参加したいと思う雰囲気を作り出すことです。時には、観客参加型の展示、デモ、ゲームを行うことも、変化を促進するための素晴らしい方法です。

ステークホルダーにゲームやシミュレーションなどの活動に参加してもらうことで、実生活での変化に伴う痛みやマイナスの影響を受けることなく、実際に体験してもらうことができます。

### 移行

パイロットを実施して大成功を収めたら、アジャイルを組織全体に広めなければなりません。なぜなら、アジャイルはやり方が違うだけでなく、考え方も違うので、残りの組織が従わなければ、勢いを維持することはできません。組織の中で古い考え方が残っていると、ある組織が最高のスタートを切ったとしても、結局はそれが妨げになってしまいます。

スクラムを使うことの意味が他の部門に伝わらなければ、組織の免疫システムが常に変化を攻撃し、最終的には戦いに勝つことになります。

それは、残りの組織がスクラムを使い始めなければならないということではなく、その組織が少なくともスクラムに対応できるようにならなければならないということです。

ここでは、測定基準、人事やその他のポリシー、更にはガバナンスやコントロール構造の変化も見過ごすべきではありません。アジャイルチームの成功を、従来の階層型組織と同じ方法で測定することはできません。チームを管理し、測定し、構造化する方法を、古いものから新しいものに変えなければなりません。コントロールを変える必要がありますが、最も基本的なものは、プロジェクトとプログラムの管理、人事ポリシーと役割と責任、管理構造、パフォーマンス管理、製品管理、さらには組織内の資金調達、予算編成、財務慣行です。

アジャイルスクラムの導入は、組織全体に影響を与えます。その意味するところは、理想的に意外と早く、最終的には組織内の全てが変わるということです。

### ADAPT の最後の言葉

スクラムと同様、ADAPT も反復的なアプローチです。ADAPT は、変化が必要であり、現在の作業方法では許容できる結果が得られないという認識から始まります。認識が広まるにつれ、アーリーアダプターと呼ばれる一部の人々は、状況を改善するためにスクラムを試してみたいと思うようになります。

試行錯誤しながら、組織内のアーリーアダプターは、スクラムで成功する能力を身につけていきます。スクラムを採用していない広い組織の中で、少数のチームがスクラムの導入に成功するという新しい現状が生まれるかもしれない。

これらの初期スクラムチームがスクラムの使用を改善し続けると、彼らは自分たちの成功を宣伝し始めます。時には、他のチームの友人と昼食を共にするような非公式なものから、部門全体のプレゼンテーションや計画された変更プログラムのような正式なものまであります。

すると、他チームの人達が「認知」から「願望」、「能力」へと変化していき、やがて他のチームの人達も自分たちの成功をアピールするようになります。

## 2.4 プロダクトの意味するところ

アジャイルスクラムでは、“*プロダクト*[4]”という言葉は包括的なものです。すでにプロダクトという言葉をこのように使っている人もいて、プロダクトというのは商品でもサービスでもよく、実際のところ多くの場合、プロダクトは商品でもサービスでもあります。

また、製品とサービスを対比させて、製品とサービスについて語ることもありますが、この場合、製品は包括的な用語ではありません。

この言葉の定義は、スクラムのプラクティスによって定義されたものに基づいており、スクラムではプロダクトは包括的な用語であり、そのためサービスもプロダクトであるという立場をとっています。

[4] 注意）日本語訳では、“プロダクト=製品＋サービス”として表現します。

# 3 リーン・マネジメント

ADAPT は、チェンジマネジメントの手法として捉えることができます。変化を促進するもう一つの形は、より有機的で進化的なものです。

組織を管理・運営する方法としてリーンを採用することは、マネジメントスタイル、従業員の行動、組織構造、役割と責任の定義の仕方を根本的に変えることを意味します。

しかし、リーンは、革命的な組織改革を促進するのではなく、進化的なマネジメントアプローチです。つまり、時間がかかるので、組織のリーダーの揺るぎないコミットメントが必要になります。

すべてのアジャイル手法は、リーンが築いた基盤に基づいており、組織的なアジリティを構築してリーン組織になることは、その本質的に同じです。

本書の目的は、リーンを教えることではなく、アジャイル実践を成功させるための基礎を形成上で、中核部となるリーンの原則とプラクティスをいくつか共有します。

## 3.1 戦略・マネジメントアプローチとしてのリーン

ほとんどの場合、Lean はオペレーティング戦略であり、2つの主要なトピックに焦点を当ててビジネスを設計、変革、運営する方法であると解釈することができます。

- 顧客価値の最大化
- ムダの削減

第二次世界大戦後、日本は深刻な資源不足に直面していましたが、トヨタ自動車の指導者たちは、プロセスフローの継続性と製品の多様性の両方を提供するには、一連のイノベーションが唯一の方法であると考えました。

トヨタは、車を生産するために必要な原材料を最小限に抑え、顧客の注文から納車まで時間を短縮することに注力しました。

その考え方と手法は、トヨタ生産方式（TPS）として知られるようになりました。このシステムは、製造技術者の焦点を、個々の機械とその稼働率から、顧客の要求に基づいた全工程を通しての製品のフロー（流れ）に移すものでした。

トヨタはすぐに、工場労働者には筋力以上の多くの貢献があることを発見しました。機械を動かし、操作する人達は、革新的なアイデアを生み出し、製造プロセスを改善する可能性が最も高いのです。

リーンの基本となるのは、2つの基本原則です。

- **ジャストインタイム生産**：必要な時に必要なものだけを行う
- **自働化（ニンベンの付いた自働化）**：標準化と自動化

トヨタの発展とともに、TPS は、2001 年に出版された総合的な *「トヨタウェイ」*の一部となりました。現在、「自働化」と「ジャストインタイム」は残っていますが、それらを包含するトヨタウェイの理念には、2つの上位原理が含まれています。

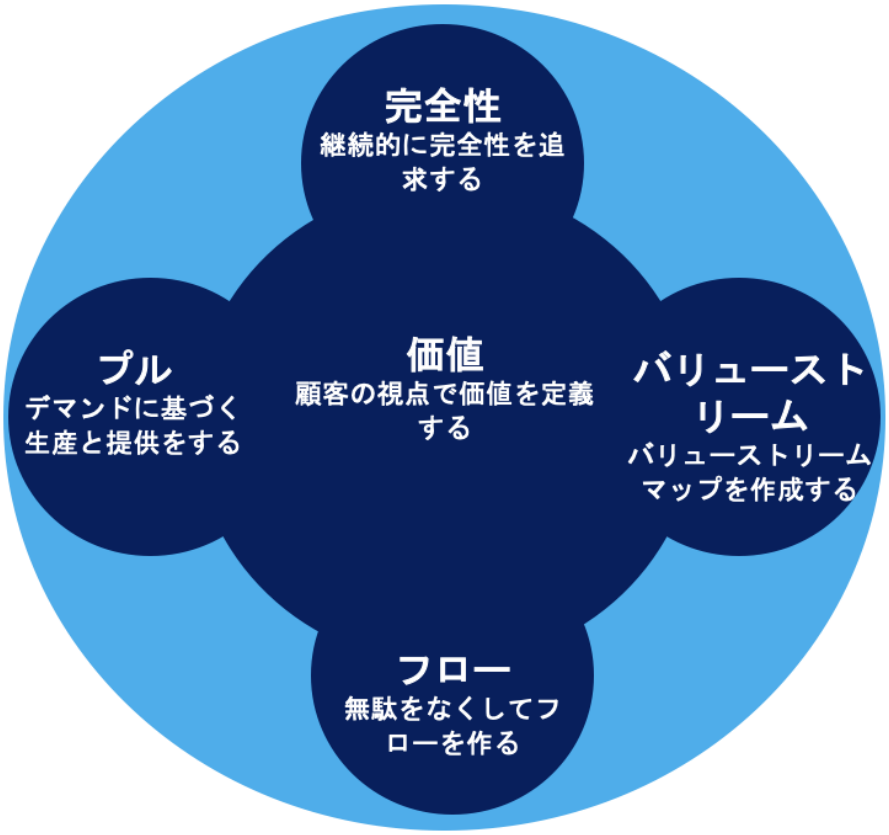
- 人間尊重
- 継続的改善

リーンは、設計や初期設定から運用と変革に至るまで、今日のビジネスに根本的な改善をもたらします。これらの概念は、顧客やステークホルダーのために長期的な成果を確立すること、全社的なプロセス能力の

構築と調整、持続的な成功を可能にする文化的要因、そして、毎日、どこでも、誰でも参加できる継続的な改善を中心に展開されます。

リーン企業は非リーン企業よりも成功しています。彼らはより良いビジネス価値を創造し、才能を保持することがより容易です。また、長期的に成功を維持することも容易です。

*図 2 リーンの原則*




画像は EXIN が作成。Botha, J.（2021）. *IT のためのリーン・ファンダメンタルズ [コースウェア]*. GetITright を参照.

第一に、リーンは顧客価値に焦点を当てるのに役立ちます。そうすることで、組織は製品やサービスにさらなる価値を付加すると同時に、ムダの原因を減らし、適応するために、敏捷性と能力を高めます。顧客やエンドユーザーとの結びつきを強化し、対話を改善することで、組織は顧客満足度を高め、顧客ロイヤルティを大幅に向上させることができます。

第二に、リーン組織はプロセスパフォーマンスを継続的に改善します。サービスの質が向上し、納期が短縮され、プロジェクトや運用の効率が向上し続けています。

リーン組織はまた、今日、ほとんどの従業員が知識労働者であり、組織の最も重要な資産はその人材であることを認識しています。リーン組織では、従業員の関与の度合いが高く、従業員のモチベーションや仕事への満足度も高くなっています。

第三に、**プロセスのムダを省き、付加価値を生み出す作業を最適化することで、さらなる付加価値を生み出すための時間を確保すること**で、財務上の価値が期待できます。しかし、組織全体で価値を生み出すために使われている「プロセス」を理解していなければ、このようなことは考えられません。**バリューストリームマッピング（VSM）**と呼ばれる手法は、改善するだけでなく、ムダをなくすために必要な洞察を与えてくれます。

ムダを省く方法の一つとして、注文を受けてから納品するまでの期間を短縮することが挙げられます。これは、システム内での仕事のフロー（流れ）を合理化し、すべてのプロセスとシステムを**プッシュ型**ではなく**プル型**に変更することで実現しています。

ワークフローの改善は、原材料の在庫の必要性を減らすことで、キャッシュフローの改善にもつながります。強調しておきたいのは、収益性の向上はリーンの第一目標ではないということです。期待はできますが、それは付加価値を生まない活動に費やす労力と時間を改善・削減することによる副次的な効果です。

品質の悪さは、リーン導入の原動力にもなります。品質低下は、組織の内外に影響を及ぼします。評判の低下や顧客の信頼の喪失などの側面から、組織に多額のコストや罰則が課せられることがあります。また、人材のムダ遣いや予期せぬ数の欠陥は、組織内にストレスをもたらします。この場合も、検査、やり直し、またはやる気をなくすことで、コストが増加します。

リーンはパラダイムをシフトさせます。例えば、伝統的なパラダイムは「知識は力」です。組織は、プロセスの仕組みを知っている少数の重要な人材に依存していることに気づくかもしれません。一部の人に頼るのではなく、すべての従業員が成長し、能力を発揮できるようにしたほうが良いのです。

アジャイルやスクラムの原則や仕事の進め方はすべて、自己管理型チームの概念、意思決定権の委譲、誰もが貢献し意見を述べることができる安全な環境の構築、罰則のない失敗からの学習を含むあらゆる手段による全員の学習の推進など、リーンの原則と実践に基づいています。

# 4 スクラムと継続的改善

アジャイルやスクラムは本来、継続的な改善を目的としたものです。アジャイルのアプローチが反復的であるのはそのためです。

デミングの PDCA サイクルのような継続的な改善手法の中心にあるのは、経験主義の考え方と、絶え間ない継続的な学習の概念、そして学習の結果としての改善です。これらは、リーンやアジャイルの中心的なテーマでもあります。

アジャイルは、プロジェクトを遂行するためだけの方法ではなく、継続的な改善を実現するための方法として捉えることができます。

ビジネスや顧客、ユーザーからの新たな要求は、継続的な改善と、ステークホルダーにとってより良い価値創造を行う終わりなきプログラムを推進します。

改善とは、今何が重要なのかによって行われ、緊急の懸案には柔軟に対処し、必要であれば方向転換でさえ行います。

しかし、これは計画的かつ予測可能な方法で行われるべきです。それは、変化と価値提供の一定のリズムを組織の中に作りだし維持することです。

スクラムを継続的な改善手法として用いて、好ましい変化のリズムを維持することは、スクラムが使用される環境に予測可能性と安定性をもたらすことを意味します。

# 5 スクラムの基本

## 5.1 スクラムの簡単な概要

スクラムは、組織が複雑なプロジェクトを段階的に進められるように作られたハイレベルなフレームワークです。提唱者がこのフレームワークを意図的にハイレベルなものにしたのは、組織が最も効果的な方法で“空白”を埋める方法を自分たちで開発することが必要だと考えたためです。

スクラムを他のアジャイル手法であるスケールド・アジャイル・フレームワーク®（SAFe®）、Disciplined Agile Delivery（DAD）、ABC/DSDM などと比較してみると、この軽量な構造は明らかで、有効性が悪くなるどころか、むしろその逆です。それは、自分たちのアジャイル手法を自由に開発できることで、スクラムをより有力なツールにできるからです。

スクラムはハイレベルなフレームワークであるため、規定的なものではなく、詳細なものでもありません。しかし、スクラムの基本的な概念は、長年にわたって慎重に設計され、テストされており、今日ではスクラム導入を成功させるための基盤を形成しています。スクラムを使用する最善の方法は、基本を維持しながら*採用し、適応する*ことです。

与えられたガイダンスに沿ってスタートし、数ヶ月間それに従ってから物事を修正し始めるのがベストです。スクラム手法がなぜ物事を行う特定の方法を提案しているのか、その理由は自明ではないことが多く、スクラム手法に基づいて何かをすることがなぜ賢いやり方なのか、その理由が自明になるまでには時間がかかることがあります。

スクラムはよく次のように定義されます。

> 人々が複雑な適応問題に対処し、生産的かつ創造的に可能な限り高い価値製品を提供することができるフレームワーク。
> スクラムは、軽量で理解しやすく、習得するのが難しいです。

文章の最後の部分が興味深いです。スクラムがとてもハイレベルで理解しやすいものであるならば、なぜ習得するのが難しいのでしょうか？

理由はいくつかありますが、次のようなことが挙げられます。

1. なぜなら、それは伝統的な組織の管理方法ではないからである
2. なぜなら、それは伝統的な組織の構造化とは異なるからである
3. なぜなら、伝統的な組織では見られなかった関係者の行動に依存しているからである
4. スクラムの進め方は1つだけではなく、それゆえ、個々の状況認識は不可欠となる

要するに、組織を大きく変える必要があるため習得が難しく、変化を受け入れるまでは使いこなすことが難しいのです。しかし、一旦、組織（および組織内の人々）が変化を受け入れ、組織、顧客、従業員にとってのメリットを確認すれば、以前のやり方に戻ることはできません。

最初の大きな組織的変革は、スクラムではすべての作業が自己管理のチームによって行われるという事実であり、これは従来の管理構造を持つ組織にとっては非常に異質な概念です。

スクラムでは、説明責任は 3 つしかなく、その中に従来の管理職の役割はありません。

最初に大きな疑念を抱かせる2つ目の要因は、プロジェクト開始前にプロジェクト全体の完全な詳細計画がないため、壁に貼れるような終了日が入ったガントチャートがないことです。

最初は、誰もコミットしようとしない真っ白なページのように見えて、マネジメントはこれに懐疑的になるかもしれません。

従来の方法の問題点は、たとえコミットメント（確約）を作って完璧な計画を立てたとしても、現実はその計画通りにはいかず、約束は守られないことが多いということです。それが現実であることは誰もが知っています。ただ、それを認識しなければなりません。

不完全な計画は、プロジェクト参加者の勤勉さの欠如の結果ではありません。むしろ、組織のような複雑適応系（complex adaptive systems: CAS）で、将来の変化の状態を完璧に計画できるという前提がそもそも間違っているのです。

スクラムは、前もって完璧な計画を立てることが不可能であることを認め、不可能なことをやろうとするのではなく、より良い解決策を提供します。ビジネスにとって最も重要なことに継続的に取り組み、より広範な（軽量な）計画の中でそれを実行するのです。それにより、作業スケジュールではなく、ビジネス価値の実現に集中することを可能にするのです。

## 5.2 異なる働き方

フレームワークと関連するアプローチの基本は、3 つのシンプルな説明責任（役割）、いくつかの計画されたイベントとツール（アーティファクト（作成物））、そして明快で単純なルールを身に着けた自己管理チームに依るものです。

これらの基本的な構成要素は、スクラムの必須要素であり、スクラムが機能するために必要なものです。スクラムを使っている組織は、進め方を調整したり変更したりする自由度が高いのですが、経験上、これらの基本的な構成要素に手を加えれば、すべてが壊れてしまいます。

フレームワークの基本的な要素と関連する「ルール」にこだわることをお勧めします。スクラムの各構成要素には特定の目的があり、フレームワークの成功には欠かせません。

スクラムはソフトウェアプロジェクトを管理するために開発されましたが、多くの組織では、他の製品やサービスのプロジェクトで同様に効果的に使用されています。

反復的な価値の提供と漸進的な知識の伝達が重要であるならば、スクラムが適しています。

スクラムの用語では、作業を行うチームを「開発者」と呼びます。スクラムの 3 つの説明責任の 1 つとして開発者という用語は残っていますが、これはチームがソフトウェア開発者だけで形成されていることを意味するものではありません。より広い意味では、特定のスプリントやイテレーションの成果物を開発、構築、提供するチームと解釈すべきです。スクラムチームには、スクラムマスター、プロダクトオーナー、そしてすべての開発者が一緒に参加します。

あなたの組織がソフトウェア開発会社であろうと、工業製品を製造していようと、サービスを提供していようと、あるいは学校であろうと重要ではありません。学習は反復的かつ漸進的であるため、先進的な学校ではスクラムが主要な教育方法として使用されています。スクラムは、小規模で、自律的で、機能横断的なチームが永続的に価値を創造するものであることを忘れてはならないです。広い意味では、これらのチームは大きなチームになるために自律性を失うのではなく、それぞれが自律的なチームでありながら、ネットワークを作り、協力し、相互運用を行います。従って、スクラムは、このフレームワークを採用する伝統的な階層的な組織に大きな影響を与えます。

## 5.3 スクラムの背後にある本質的な理論

注：本書は、さらにスクラムを学ぶためのガイドとして書かれたものであり、すでにスクラムの基本を知っていることを前提としています。そうでない場合は、*『アジャイル・スクラム・ハンドブック』*（Rad, 2021）をお読みください。

ここでの目的は、あなたがより良いスクラムマスターやプロダクトオーナーになるためのお手伝いをすることです。本書では、スクラムマスターとプロダクトオーナーの説明責任とプラクティスに焦点を当てます。

なお、スクラムマスターはプロダクトオーナーシップについて知っておくべきであり、その逆もまた然りです。

スクラムの背後にある考え方は新しいものではなく、経験的なプロセスコントロールやリーンで使われている原則とプラクティスにまで遡ることができます。

私たちが何を想定しているかではなく、私たちが知っていることに基づいて判断し、実際に行動して知識や経験を深めていくという考え方です。それはまた、私たちが何かが分からない状況で、答えを見つける唯一の方法は経験的なアプローチを採用することであることを意味します。

科学では、経験的アプローチの基本は、仮説を立て、その仮説を検証するためにテストを行うことです。仮説を検証した後、その仮説を証明するか反証するかのどちらになります。いずれにしても以前より多くのことを知ることになります。

これは、スクラムを使用することの大きな便益の1つであり、プロジェクトの計画のすべてが前もって行われ、そのほとんどが間違っていることが多いという仮定に基づいているウォーターフォール方式とは対照的です。

スクラムでは、わからないことを隠すことなく認め、経験的なアプローチでできるだけ早く答えを見つけようとします。

これは、実験に基づくプロセスコントロールの 3 つの柱の1つである**「透明性」です**。やるべきことを理解しようとするときに学ぶ実験や教訓は、**「検査」**と**「適応」の2つ**です。

## 5.4 スクラムの 3 つの柱

3 つの柱をもう一度見てみましょう。

- 透明性
- 検査
- 適応

**透明性**

透明性とは、仕事をするために使用するプロセスや行わなければならない作業について、チームメンバーに見えるようにすることです。

共通の標準、言語、手法、ツール、測定基準、共通の語彙が使われていれば、誰もが理解しやすくなります。また、作業も可視化する必要があります。これにより、誰もが実行する作業を確認して理解できるようになります。

**検査**

検査とは、やっていることがうまくいっているかどうか、仮定（仮説）が正しく妥当であるか、継続的にチェックしていくということです。作業が完了したら、作業方法や関連するスクラムのアーティファクト（事象物）を頻繁に検査して妥当性確認をしたり、3 つの柱の最後にある変更、修正、改善を行う必要があります。

**適応**

適応とは、プロセスまたは作業のいずれかの要素が設定された標準から逸脱した場合、実行されたプロセスまたは作業を調整して、設定された標準および提供されるべき品質に迅速に戻るようにする必要があることを意味します。

### 5.4.1 スクラムイベント

スクラムには 5 つのイベントや儀式しかなく、その一つ一つが検査と適応の機会を提供しています。

それらは次の通りです。

- スプリント
- 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

また、プロダクトバックログリファインメントはイベントとして定義されていませんが、重要な活動です。

多くの組織は、スクラムやその他のアジャイル手法やフレームワークを採用するために必要なカルチャー変化を過小評価しています。カルチャーに大きな影響があります。

組織カルチャーは一朝一夕には変えられないことは周知の事実ですが、スクラムやアジャイルを機能させるためには、組織はスクラムの根底にある一連の価値観を支持する必要があります。

アジャイル化を目指したにも関わらず、アジャイル手法を使うことを放棄した組織の多くが失敗するのは、おそらくこの一点に尽きるでしょう。

スクラムの核にあるのは経験主義とリーン思考であることを忘れてはいけません。これは、知識は経験から得られるものであり、意思決定は観察されたことと学んだことに基づくべきという信念に基づいています。リーン思考では、何が機能したのかに集中し、物事をシンプルに保つのに役立ちます。リーンのムダ削減への取り組みは、資源の消費だけでなく、新たな価値ある製品やサービスの構築にもつながっています。

スクラム、リーン、その他のアジャイル手法は、効果的に機能するために一連の価値観に依存しているため、次のように自問するのが良いでしょう。これらは、現在の組織の価値観やカルチャーに合っているのか?と。これらの価値観と合わなければ合わない程、組織にとって変化が難しいものになり、スクラムやアジャイルの採用は難しくなります。

しかし、その価値観とは何でしょうか?

## 5.5 アジャイルスクラムの価値観

スクラムチームは、自己管理型の自律型ユニットとして機能し、チームが行っていることに対して実行責任と説明責任を負います。チームが失敗したり、ミスをしたりした場合は、チームの学習と成長のために、顧客に価値を提供する限り、外部からの干渉を受けずにそこから学び、修正することができなければなりません。価値の提供とは、チームが顧客の要件を解釈・検証し、価値提供ための実行タスクに要件を分解することで、これらと関連する成果を確実に達成することを意味します。

ここでの重要な問題の 1 つは、チームが安全だと感じ、ミスから学びさえすれば、懲罰的な措置は取られないと知っていることです。スクラムやアジャイルが組織で失敗する最大の理由としては、おそらくこれが 1 つの課題になります。

スクラムの価値観は

- 確約 (Commitment)
- 勇気 (Courage)
- 集中 (Focus)
- 公開 (Openness)
- 尊敬 (Respect)

これらの価値観が活きず、実現しなければ、スクラムの三本柱は崩れます。

さらに、組織内や顧客など、自分が関わるすべての人がこの価値観を信じ、支持していることを信じ、信頼しなければなりません。

要するに、チームメンバーは価値観を信じて実践するだけでなく、組織内で、その価値観の整合性を確保する必要があります。

では、その価値観が実際に何を意味するのか見てみましょう。

**確約（Commitment）**
確約とは、チームメンバー全員がチームや作業に専念し、その結果に約束をすることです。

**勇気（Courage）**
チームメンバーは率直に発言する勇気を示さなければならず、他のチームメンバーはそれに耳を傾けなければなりません。チームは、約束を果たすために、できるだけ早く問題に取り組まなければなりません。

**集中（Focus）**
約束したものを決められた期限内に提供するために、チームメンバーは個人的にも集団的にも作業を成し遂げることに集中しなければなりません。

**公開（Openness）**
人は誰でもミスをし、全てのことができるわけではありません。チームメンバーは、ミスを犯したり、業務上の障害に直面したとき、あるいは割りあてられたタスクの実行方法がわからないときには、率直に認めなければなりません。ミスを隠すことは、チームにとって許されない連鎖的な影響を及ぼします。そのためには、「自分の意見を述べても罰を受けない」と信じなければなりません。残念ながら、多くの組織ではこのようになっていない状況が続いています。

**尊敬（Respect）**
したがって、チームメンバーはお互いに敬意を持って接し、可能なときはいつでもどこでも助け合わなければなりません。

## 5.6 スクラムの説明責任の概要

### 5.6.1 スクラムチーム

スクラムの基本単位は、スクラムチームと呼ばれる小さなチームです。スクラムチームは、1人のスクラムマスター、1人のプロダクトオーナー、そして開発者で構成されています。スクラムチームの中には、サブチームや階層はありません。それは、一度に1つの目的、すなわちプロダクトゴールに集中する専門家のまとまった単位だからです。

スクラムチームは自己管理型で機能横断的であるため、チーム外からの干渉を受けることなく、自分達が仕事を達成するための最善の方法を合意し、選択することができます。プロダクトオーナーは顧客とビジネスを代表しているので、ビジネスの優先順位はプロダクトオーナーを通じてスクラムチームに知らされています。

スクラムチームは、機能横断的な開発者で構成されるべきであり、チームの各個人は、スプリントと呼ばれるタイムボックス形式のイベントで、選択された作業の一部を完了するために必要なすべてのスキルを持っているべきです。チームは、チーム外のスキルへの依存度をある程度低くすべきです。つまり、外部の開発者の関与をできる限り避けるべきです。但し、特定の要求されるスキルが不足している場合には、外部の開発者をその機会に利用することができます。これはまた、小規模チームで働くことが、スクラムの価値

を実践するのを容易にすることを意味します。大規模なチームは、小規模なチームに比べてより多くの欠陥を生み出す傾向があります。

スクラムチームは、アジャイルを維持するのに十分に小さく、スプリント内で大量の作業を完了するのに十分な規模でなければなりません。スクラムチームのサイズは、10 人以下が望ましいです。

なぜこんなに小さいのか？小規模なチームの方がコミュニケーションが円滑で、有害な過度の専門化が起こりにくく、生産性が高いことが証明されています。

チームの規模が大きくなりすぎた場合、大規模なチームを再編成して、それぞれが同じプロダクトに焦点を当てた複数のまとまったスクラムチームにすることを検討すべきです。チームは同じプロダクトに焦点を当てているので、同じプロダクトゴール、プロダクトバックログ、プロダクトオーナーを共有する必要があります。

スクラムチームは、プロダクトに関するすべての活動に責任を持ちます。具体的には、ステークホルダーとのコラボレーション、要件の検証、チーム運営の維持、チームプラクティスの維持などが含まれ、これらのプラクティスを改善する方法を継続的に実験します。これを達成するために、チームは研究開発活動や、チームとその作業方法を改善するために必要なあらゆることに時間を費やさなければなりません。スクラムチームは、自分たちの作業を管理する権限を組織から与えられています。同時に、チームは、何をしなければならないのか、顧客やユーザーにとって何が価値であるのかを理解するために、広く協議すべきです。この合意は、スクラムチームのパフォーマンスの重要な指標である“完成の定義（DoD）”を定義するための基礎となります。

スクラムチーム全体は、全てのスプリントで価値ある有用なインクリメントを作成する責任があります。

スプリントとは、1 週間から 4 週間にわたるイベントで、スクラムチームが 1 つ以上の有用なインクリメントを開発します。

有用なインクリメントとは、ステークホルダーが展開して使用することができ、ステークホルダーにとって価値を生み出すものを意味します。

なお、1 つのスプリントには、複数の有用なインクリメントが含まれる場合があることに注意してください。

スクラムでは、スクラムチーム内の 3 つの具体的な説明責任を定義しています。

- 開発者
- プロダクトオーナー
- スクラムマスター

この本の後半では、効果的なチーム、チームの構成、チームの構成方法について、より詳細に述べます。スクラムチームが持続可能なペースで一貫して作業を行うための主な方法は、スプリントの使用です。従って、スプリントはスクラムの基本的な構成要素です。

ここでは、*良い*スクラムチームの基本的なポイントをご紹介します。

- **スクラムチームは自己管理型です。**誰もスクラムチームに対して、プロダクトバックログアイテムから価値（製品／サービス）を生み出すインクリメントに変える方法を指示しません。
- **スクラムチームは機能横断的です。**プロダクトのインクリメントを作るためにチームが必要とするあらゆるスキルを備えています。これは理想的なことです。常に可能とは限りません。
- **スクラムは、**その人が行っている作業に関係なく、**肩書きや地位を必要としません。**スクラムマスター、プロダクトオーナー、開発者は、従来の「役割」ほど厳密なものではなく、むしろ説明責任を果たすことに近いです。ある時期に特定の「役割」あるいは「説明責任」を持つということは、単に誰かが特定の物事を確実に遂行しなければならないことを意味します。
- テスト、アーキテクチャ、運用、ビジネスアナリシスなど、対処すべき領域に関わらず、**スクラムでは、サブチームを認めていません。**

- スクラムチームの個々のメンバーは、専門的なスキルや重点分野を持っているかもしれませんが、**説明責任はスクラムチーム全体に属しています。**
- スクラムマスターとプロダクトオーナーの責任は、一人の人物では両立できません。この場合、責任の衝突を避けるために職務分離が重要です。

### 5.6.2 開発者

開発者は、各スプリントで使用可能なインクリメントのあらゆる側面を作成することをコミットし、そのインクリメント開発するために必要な特定のスキルを持っています。

開発者のスキル、能力、さらにはスキルのレベルは多岐にわたり、主にスプリントに関わる作業領域によって決定されます。

開発者は常に次の責任を負っています。

- スプリントプランニングとスプリントバックログの作成
- 完成の定義（DoD）を守ることによる品質の向上
- スプリントゴールに向かって確実に前進するために、必要に応じて計画や作業の進め方を調整
- プロフェッショナルとしての説明責任をお互いに果たす

### 5.6.3 プロダクトオーナー

プロダクトオーナーは、委員会ではなく、開発者やスクラムマスターと協力してプロダクトの価値を最大化する責任者です。

プロダクトオーナーはフルタイムの仕事が望ましいです。マルチタスクやプロジェクトの潜在的なボトルネックを避けるために、この点に注意を払う必要があります。プロダクトオーナーは、プロダクトバックログにある全てのステークホルダーのニーズを代表する者であり、プロダクトオーナーシップの説明責任をどのように果たすのかというトレーニングを受けなければなりません。プロダクトバックログを変更したい場合は、変更のメリットに基づいてプロダクトオーナーを説得することで可能になります。

- プロダクトオーナーは、プロダクトバックログの作成、管理、維持、プロダクト予算の管理、プロダクトローンチの調整に責任を負う
- プロダクトオーナーは、プロダクトゴールを策定して明確に伝え、プロダクトゴールを確実に達成するための要件で構成されるプロダクトバックログを明確に定義する。要件はプロダクトバックログアイテムとして定義される
- プロダクトオーナーは、組織の目標に沿ってプロダクトバックログアイテムを順序付けする
    - 注：バックログの順序付けは、主にビジネス上の優先順位によって定義されますが、それが唯一の判断基準ではない
- **プロダクトオーナーは、スクラムミーティングに出席し、顧客の声（VoC）の役割を果す。**要件を説明し、開発者がプロダクトバックログアイテムとその順序をよりよく理解できるように支援する
- **プロダクトオーナーは、異なるスクラムチーム間の調整を行い**、作業が重複しないようにし、依存関係を可視化することで全員に明らかにし、作業と依存関係が整合するように調整しなければならない
- **プロダクトオーナーは、プロダクトバックログのリファインメントを頻繁に行わなければならない。**バックログのリファインには、順序の変更、要件の適切な詳細レベルへの細分化、そして開発者からのフィードバックと支援が含まれる

この説明責任の重要性から、権限がない、あるいは少ない、顧客から尊敬されていない人物にプロダクトオーナーシップを与えてはなりません。

プロダクトオーナーは説明責任を負いますが、チームの他の人に実行責任を委任することもできます。

明記されていませんが、ビジネスにおけるプロダクトオーナーシップ（戦略的、戦術的、予算）を 1 つの説明責任に統合することをお勧めします。

プロダクトオーナーは、そのビジネスを代表するものであるだけでなく、ビジネス担当者でもあるべきです。つまり、プロジェクト側や技術側ではなく、ビジネス/顧客側から来るべきだということです。スクラムガイド（Scrum.org, 2020）には、プロダクトオーナーが成功するためには、組織全体がその決定を尊重しなければならないと明記されています。

プロダクトオーナーの決定は、プロダクトバックログの内容と順序、そしてスプリントレビューで検査・見直しできるインクリメントを通して、目に見える形になります。

プロダクトオーナーが決定を下した場合、その決定はビジネスとしてです。このレベルの権限と影響力を持つプロダクトオーナーが最も効果的であることが証明されています。

### 5.6.3.1 顧客の声（VoC）としてのプロダクトオーナー

顧客の声（VoC）とは、リーンで使われる用語で、顧客の期待、好み、ニーズを表すものです。

顧客の代表として、プロダクトオーナーは、顧客とユーザーの要求やニーズを深く理解していなければなりません。したがって、プロダクトオーナーは顧客とユーザーの声を代弁する存在であると言ってもよいでしょう。

そうは言っても、開発者はイテレーションの間にユーザーや顧客と関わり、要件やニーズをより深く理解し、改善していく必要があります。プロダクトオーナーは、エンドツーエンドのコミュニケーションを妨げるゲートキーパーになってはいけません。

顧客の代表として、スクラムチームは以下の**アジャイルの原則**を実践しなければなりません。

1. 顧客満足を最優先し、価値のあるソフトウェアを早く継続的に提供します。
    a. それはどんな製品やサービスでもかまいません。あなたの環境に依ります。
2. 要求の変更はたとえ開発の後期であっても歓迎します。変化を味方につけることによって、お客様の競争力を引き上げます。
3. ビジネス側の人と開発者は、プロジェクトを通して日々一緒に働かなければなりません。
4. シンプルさ（ムダなく作れる量を最大限にすること）がアジャイルスキルの本質です。

### 5.6.3.2 優れたプロダクトオーナーの属性

優れたプロダクトオーナーは皆、同じ属性を持っています。ここでは、効果的なプロダクトオーナーの特性について、いくつかのガイダンスをご紹介します。

1. プロダクトオーナーは、全ての聞き手と上手くコミュニケーションがとれる、**強力なコミュニケーター**です。彼らは、聞き手に合わせて言葉やコミュニケーションの方法を変えていきます。そして何よりも、優れたプロダクトオーナーは優れた聞き手であり、耳を傾け、観察し、鋭い質問をし、学んだことを熟考します。
2. プロダクトオーナーは、ビジネスニーズを具体的活動として効果的に移すことができるよう、**ビジネスを深く理解している**必要があります。適切なレベルの洞察力を持ち、ステークホルダーのニーズが明確でない場合や、さらに悪いことには互いに対立している場合でも、複雑な意思決定と妥協を行うことができるようにするためには、プロジェクト側ではなくビジネス側の人間でなければなりません。
3. プロダクトオーナーは、ビジネス側を代表して意思決定を行う際に、**ビジネスを尊重することができる**シニアでなければなりません。
4. また、**「必要なもの」と「欲しいもの」を見分ける能力**も必要です。ステークホルダーは、機会があればプロダクトオーナーに多くの要求を投げかけます。プロダクトオーナーは、“必要”と“あったら

良い”を見分けることができなければなりません。ここで、MoSCoW（モスコー）の技法[5]は、プロダクトオーナーにとって最初の段階では有利に働きます。

5. **ソリューションの考え方：**ビジネスの要求やニーズの多くは、否定的な言葉（問題）で表現されています。プロダクトオーナーには、「何が悪いのか」ではなく、「どうすれば問題を解決できるのか」という視点で議論を進める能力が求められます。
6. **結果駆動型：**アウトプットは必須ではなく、達成すべきは*成果*（インクリメントが提供された後に、ユーザー、組織、顧客がそれを利用できること）です。

効果的なプロダクトオーナーのために、他にもいくつかのスキルや属性がありますが、上述の 6 つは間違いなく必要です。

### 5.6.4 スクラムマスター

スクラムマスターの説明責任は、従来のプロジェクトマネージャーとは比べ物にならないほど、かけ離れたものになります。

スクラムマスターは、*スクラム*という言葉の由来となったラグビーのスクラムハーフに近い存在です。彼の仕事はスクラムでボールを手に入れ、ボールをバックラインに渡すタスク（ビジネス価値をもたらす）を実行することです。バックラインはそのボールをトライ（ゴール）しなければなりません。もし、あなたがラグビーというゲームに馴染みがなくても、スクラムの知識を身につけるために最初にラグビーを学ぶ必要はありません。この例にストレスを感じる必要はありません。

スクラムマスターは、作業の調整、会議の手配、メンバーの困難な状況の打開を支援し、視覚的なツールを用いて進捗状況を把握することで、誰もが進捗状況や課題を正確に把握し、必要に応じてメンバーが他のメンバーを助けることができるようにします。

スクラムマスターは、手法とプロセスとしてのスクラムを推進し、サポートする責任があります。彼らは、全ての人がスクラムの理論とプラクティスを理解するのを手助けすることで、これを達成します。スクラムマスターには、チームに対して以下のような説明責任があると言えるでしょう。

- トレーナー
- ファシリテーター
- コーチ

コーチやファシリテーターとしての説明責任の一部として、スクラムの技術的な側面だけでなく、他にも注意を払う必要があります。特に初期の段階では、人の側面も重要です。そして、スクラムマスターは変化をもたらすチェンジエージェントであり、結束したチーム作りを行い、チームを破壊しようとする力に対処します。

スクラムマスターは、スクラムチームに仕えるリーダー（サーバントリーダー）です。その方法は次のとおりです。

- 目標、範囲、プロダクト/サービスが理解されていることの確認。
- プロダクトバックログの管理を改善する方法を見つけます。
    - プロダクトバックログを管理するのではなく、プロダクトバックログをより良くするための提案をします。

- スクラムマスターは、開発者が**プロダクトバックログアイテムを理解するのを助けます。**その順序、優先順位、価値などについて。
    - これはプロダクトオーナーの第一の説明責任ですが、スクラムマスターが支援します。

[5] MoSCoW に関する詳しい説明は、「MoSCoW」の項をご覧ください。

- スクラムマスターは、チームメンバーが**スクラムとその全ての部分を理解し**、スクラムをチームや組織の状況に合わせて最適に使用・適応できるように支援します。
- スクラムマスターは、スクラムイベントのファシリテーターです。
- 見落とされがちですが、スクラムマスターは、**外部からの割り込みや、集中力を妨げるものからチームを守り**、チームのパフォーマンスを向上/サポートするために**業務上の障害を取り除き**、コミュニケーションを促進しなければなりません。
    - スクラムマスターは、チームがすべきことの中で、チームとして働くのに最適な環境を整える必要があります。

スクラムマスターは、開発者にならないことが望ましいです。こうした慣例は、スクラムマスターの説明責任への注意の欠如に繋がり、他の障害を引き起こす可能性があります。

他のアジャイル手法では、スクラムマスターを、プロジェクトマネージャー（良い考えではありません）、イテレーションマネージャー、アジャイルコーチ、チームコーチと呼ぶことがあります。

コーチとして、スクラムマスターは、彼らのスクラム経験に基づき、アジャイルの原則とスクラムの方法・技法を、特定の背景や状況に適用するために、チームが最善の方法を見出すのを助けます。

スクラムマスターとして、適切なトレーニングを受けていない従来のプロジェクトマネージャーを使う際には十分注意してください。ウォーターフォール型プロジェクトでよく使われる典型的な命令と制御（コマンド&コントロール）の管理手法が適用されると、失敗するか、大失敗するかのいずれかになります。

また、元技術リーダー（リードエンジニア）をスクラムマスターに起用すると、悪影響を及ぼす可能性があります。チームリーダーとしての役割を果たしていた彼らは、自分の経験に基づいて何をすべきかを体系的に指示することに慣れていますが、スクラムマスターは開発者を代表して意思決定を行うわけではありません。誰をスクラムマスターに任命すべきかを検討する場合、スクラムマスターになるためのトレーニングを受け、かつ自ら志願する人を優先することが望ましいです。特に、アジャイルスクラムを始める際には、物事を進める高度な経験をもった有能なコーチを登用するのがよいでしょう。アジャイルスクラムは新しい仕事の進め方であるだけでなく、ほとんどの組織で行われてきた従来の働き方を根本的に変えるものです。

あなたは多くの業務上の障害に直面することでしょう。経験豊富なコーチは、何がうまくいくか、何がうまくいかないか、学んだ教訓、そして絶対に避けるべきことを教えてくれます。

経験豊富なコーチは、第一に組織的チェンジマネジメントの達人であり、第二にアジャイルスクラムの専門家です。しかし、彼らはアジャイルスクラムの専門家でもあるため、困難な移行期間を通じて、専門家として組織を強力に支援してくれます。

### 5.6.4.1 優れたスクラムマスターの属性

スクラムマスターの説明責任について、そして素晴らしいスクラムマスターになるために必要とされる暗黙的なスキル（形式知でなく暗黙知）については、すでに触れました。

ここでは、効果的なスクラムマスターの属性と要件を紹介します。これらの特性の中にはソフトスキルもありますが、まずはスクラムマスターに必要なハードスキルをご紹介します。

スクラムマスターはコーチや教師であるため、スクラムやアジャイルの理論についてしっかりとした知識を持っている必要があります。

- **スクラムマスターは永遠の学習者でなければならず**、常に技術的・理論的なレパートリーを増やしていか**なければなりません。**これには、スクラムやアジャイルの理論とプラクティスだけでなく、他のチームが素晴らしい効果を生み出すために使用している最新のツールや技法も含まれ、組織や自分のチームがいつ、どのようにこれらの恩恵を受けることができるかを実証します。
- **スクラムマスターは監督者ではなく、主催者です。**チームが反復しながら使用するシステムを構築・管理し、相談しながら包括的にそれを行う必要があります。つまり、スクラムマスターは組織化

する力がある人であり、周りの全てがうまくいかないときに冷静に対処できる人でなければなりません。
- **スクラムマスターは、チームが使用するツールを熟知していなければなりません。**通常、スクラムボードやバーンダウンチャートなどのツールやデータの維持・管理に責任をもちます。
- **スクラムマスターは優れたトレーナーであり、コーチでなければなりません。**偉大なスクラムマスターは、何をすべきかを知っているだけでなく、スクラムで特定の方法で物事が行われる方法と理由を、関係者全員に説明できなければなりません。
- **スクラムマスターはまず、チームがスクラムをマスターしていることを確認しなければなりません。**このスキルは絶対に必要です。スクラムマスターはチームの一員である一方で、彼らはチームメンバーの改善や協力をコーチし、奨励することでもあります。スクラムマスター以上にチームメンバー全員の長所と短所を知っている人はいないはずで、各人がチームへの貢献を最大限に高められるようにサポートできなければなりません。
- スクラムマスターは技術者である必要はありませんが、**中心的な技術のスキルを持っている**か、少なくともチームが機能する技術的背景をよく理解していると役に立ちます。これは、環境、主要な役割を担う人、何をすべきか、物事がうまくいかないときに誰に相談すべきかをよく理解している必要があることも意味します。

あるメンバーがスクラムマスターとして、または特定のチームのスクラムマスターとして機能するために必要なハードスキルを持っているかをチェックするのは比較的簡単です。

その他のスキルは、事前に確認するのは簡単ではありません。

- **優れたスクラムマスターは学習能力が高く**、特定の環境で機能するために必要なソフトスキルを速いペースで身につけます
- 人がチームとして働くときには必ず衝突が起きます。**スクラムマスターは衝突の解決を促進し**、チームが個人ではなく集団として機能するようにしなければなりません。
- すでに述べたように、**スクラムマスターは仕えるリーダー**です。リーダーは模範を示してリードし、貢献し、公平な仕事をします。サーバントリーダーシップとは、個人のニーズよりもチーム全体のニーズを優先し、他の人が能力を最大限に発揮できるように支援することです。

要約すると、優れたスクラムマスターは次のようになります。

- 責任感のある（Responsible）
- 謙虚である（Humble）
- 協調性がある（Collaborative）
- 熱心である（Committed）
- 影響力がある（Influential）
- 博識である（Knowledgeable）

## 5.7 スクラムイベントの概要

スクラムについて話すことは、常に鶏が先か卵が先かという話になります。物事がどのように機能するかについて話す場合、役割や説明責任についての議論になり、最初に説明責任について話す場合、物事がどのように機能するかについての議論に必ずなります。

このような場合には、説明責任について話をしてから、物事がどのように機能するかの議論をすることをお勧めします。どちらかと言えば、こちらの方がいいのではないでしょうか。

一般的に、スクラムの計画された活動は**イベント**と呼ばれ、使用されたか作成されるもの（リソース）は**アーティファクト**と呼ばれます。

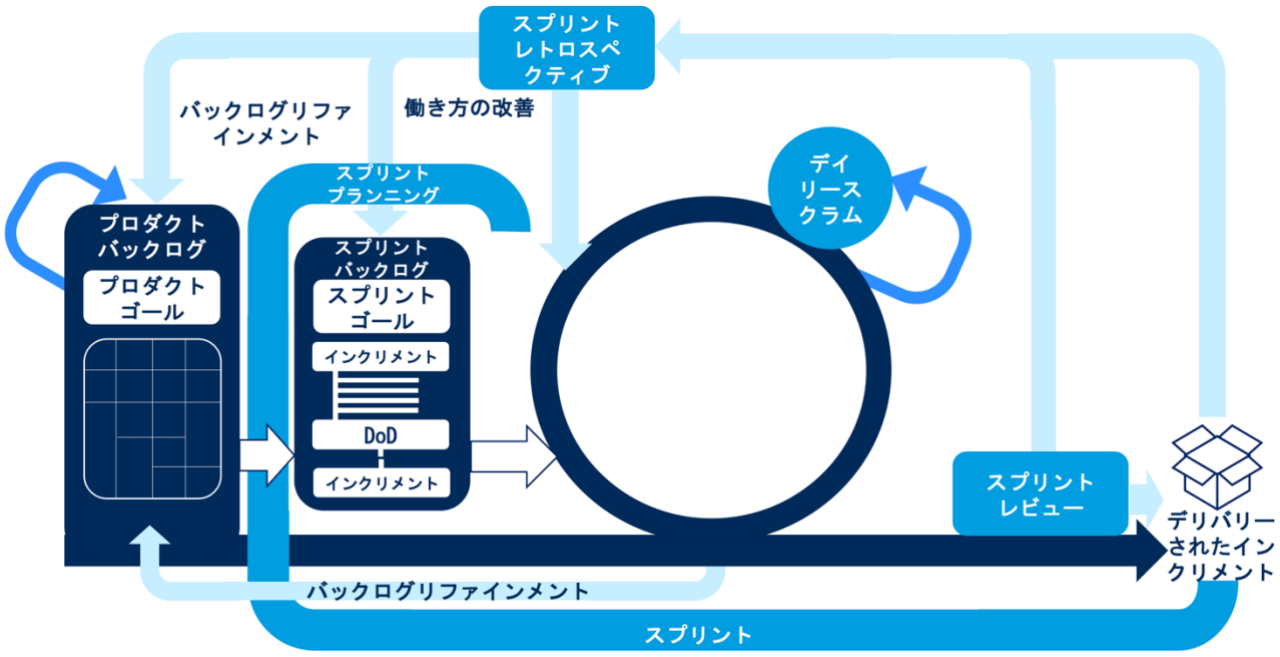
ステークホルダーの要求は、ユーザーストーリーとして収集・記録され、1つまたは複数のプロダクトバックログに反映されます。これらは後で詳しく説明します。

各プロダクトには、固有のプロダクトバックログとプロダクトオーナーがいます。彼は、スクラムチームの作業から得られるプロダクトの価値を最大化する説明責任を持ちます。プロダクトオーナーは、プロジェクトマネージャーではなく、対象となるプロダクトやプロダクトポートフォリオのビジネスオーナーであることが望ましいです。

プロダクトバックログのストーリーは、プロダクトオーナーが優先順位を決定し、スクラムチームの他のメンバーと協力して、優先順位に基づいて最短時間でプロダクトバックログのストーリーをデリバリーします。これは、収集したすべての要求がデリバリーされるまで継続して行われます。

要求に対するデリバリーは、**スプリント**と呼ばれる、短くて小さくて、プロジェクトのような、反復的で計画されたイベントによって行われます。

*図 3 スクラムのすべてを示すハイレベルビュー*




図は EXIN 制作：Botha、J.（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

スクラムチームは、プロダクトオーナーと相談して、決められた期限（スプリント）内に完了できる要件を選択します。設定された制限時間は、通常、すべてのスプリントで同じです。スプリントは**タイムボックス化された**、つまり一定の時間枠のイベントであり、通常は 1 週間から 4 週間の間に行われます。

組織によっては、すべてのスプリントに 4 週間などの固定期間を設定しているところもあれば、やるべきことに応じてスプリントの期間を変えることを認めているところもあります。一般的には、スプリントの長さは規定されていませんが、特に最初の頃は、固定の長さのスプリントを定義することに価値があります。

最初の段階では、4 週間の固定スプリントで作業することをお勧めします。スプリントの計画や作業を行う際にタイムボックスを使用することで、予測可能性が高まり、スプリントの成果物やプロダクト・フィーチャーのローンチ日をより明確に把握することができます。

繰り返し発生する問題により、タイムボックスが 4 週間未満でスプリントを終えることができない場合は、プロダクトオーナーに相談することでタイムボックスの長さを延長することができますが、4 週間を超えることはできません。

スクラムチームは自律的に機能します。自分たちで作業を選択・計画し、成果物に対して共同で責任を負います。スプリントの失敗は、全員の失敗です。

要件はプロダクトバックログの中でユーザーストーリーとして定義され、チームが達成すべき成果も定義されることはすでに述べました。ユーザーストーリーについての詳細は後述します。

スクラムチームは、要求をスプリント期間中に提供可能な小さな要求に分解します。スクラムチームは、**スプリントプランニング**の一環として、小さなストーリーをタスクに分解し、その中からチームメンバーが最も適切なものを選択して活動できるようにします。

スプリントプランニングの一環として、スプリントで提供される各インクリメントに対して、完成の定義（DoD）が定義されます。完成の定義（DoD）は、何が計画され、コミットされ、実行され、提供されたかを確認するための保証手段として機能します。なお、スプリントでは少なくとも 1 つのインクリメントを提供する必要があります。

スプリントプランニングもタイムボックス化されており、4 週間のスプリントのためのスプリントプランニング会議は 8 時間以内に終わらせる必要があります。

そして、チームは納品に向けた進捗状況を確認します。そのために使われる主なイベントが**デイリースクラム**です。15 分以内に行われるこのイベントでは、チームメンバーが進捗状況を報告し、直面している困難を話し合い、各人がどの作業に追われているかを他のメンバーに伝えます。時には、依存関係があるために、チームメンバー間の作業はデイリースクラムやスーパーストラクチャーの間に密接に調整されなければならないこともあります。そのような方法のひとつである「Nexus」については、後で詳しく説明します。

スプリントの終わりに、チームは、顧客やユーザーに使用可能で、必要であれば展開可能なプロダクトやフィーチャーをデモンストレーションします。機能的で展開可能なプロダクトやフィーチャーは、プロダクトが課題、統合、性能、ユーザービリティといったテストを完了した場合、*出荷可能なプロダクト*と呼ばれることもあります。このイベントは**スプリントレビュー**と呼ばれ、4 週間のスプリントの場合、4 時間以上かかることはありません。

出荷可能なプロダクトは、ユーザーが使用することができ、ユーザーとビジネスに価値をもたらすものです。要求されたすべてのフィーチャーを備えていないかもしれませんが、それが動作し、望ましい成果をもたらすことを検証する必要があります。出荷可能なプロダクトは、1 つのスプリントで提供される限り、1 つまたは複数のインクリメントであっても構いません。

スプリントレビューの後、チームはスプリント中の自分たちのパフォーマンスを評価し、今後どのようにすればよいかを考えます。この自己評価のためのイベントを「**スプリントレトロスペクティブ**」と呼びます。スプリントレトロスペクティブはタイムボックス型のイベントで、4 週間のスプリントで約 3 時間かかります。

スクラムイベントとして公式に認められているわけではありませんが、スクラムチームはしばしば、プロダクトオーナーがプロダクトバックログリファインメントを行うのを支援するために、通常はタイムボックスにおいて一定の時間を費やします。

注：プロダクトバックログリファインメントは、以前は「グルーミング」と呼ばれていましたが、グルーミングには否定的な意味合いが含まれるカルチャーもあるため、「リファインメント」に変更されました。

プロダクトバックログリファインメントは、次のスプリントが始まる前にプロダクトバックログアイテムが正しい順序であることを確認するのに役立ちます。

チームはプロダクトバックログリファインに特定の時間を割くことが多いですが、チームは継続的に学習しており、新たな依存関係を発見することもあるため、これは継続的な活動であることに注意してください。このようなことが起こるとすぐに、プロダクトバックログは最新の情報や洞察で更新されます。

ここでは、各スプリントイベントを完全に説明することが目的ではなく、スクラムのイベントからイベントへの流れの概要を説明することが目的であることに注意してください。次の章では、スクラムのイベントについて詳しく説明します。

## 5.8 スクラムイベント

前述の概要では、5 つのスクラムイベントが紹介されました。それらは次の通りです。

- スプリント
- 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

この章では、これらのイベントのそれぞれについて簡単に説明します。なお、スクラムのイベントは非常にシンプルでわかりやすいものです。

### 5.8.1 スプリント

スプリントは、アイデアを価値に変えるスクラムの心臓部です。

スプリントは、一貫性を持たせるために、1ヶ月以内の固定長のイベントです。新しいスプリントは、前回のスプリントが終了した直後に開始されます。

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など、プロダクトゴールを達成するために必要な作業は、いくつかのスプリントの中で行われます。

スプリント中：

- スプリント中にスプリントゴールを危うくするような変更はしてはいけません。
- 品質と品質要求は決して低下しない。
- プロダクトバックログは必要に応じて改良されます。
- スプリントに含まれるプロダクトバックログアイテム（PBI）についてチームが知るにつれ、作業範囲が明確になり、プロダクトオーナーと再交渉することもあります。

予測可能性は、プロダクトゴールやスプリントゴールに向かって進行しながら、アクティビティを検査し、適応させることで確保されます。スプリントが長すぎると、スプリントゴールが無効になり（したがって、プロダクトバックログの変更との整合性が取れなくなり）、複雑さが増し、リスクが増大する可能性があります。スプリントを短くすることで、学習サイクルを早くし、リスクを抑えることができます。各スプリントを短いプロジェクトとして捉えることもできます。

進捗状況の予測には、バーンダウンチャート、バーンアップチャート、累積フローなど、多くのツールを使用することができます。便利ではありますが、これらは経験主義の重要性に取って代わるものではありません。複雑な環境では、何が起こるかわかりません。将来を見越した意思決定には、すでに起こったことしか使えません。

*スプリントゴールが陳腐化した場合*、スプリントはキャンセルされる可能性があります。スプリントをキャンセルする権限があるのは、プロダクトオーナーだけです。

しかし、チームが何らかの理由でスプリントを完了できないことに気付いた場合（通常は、開発者が予想外の複雑さのために作業がタイムボックスに収まらないことに気付いた場合）は、プロダクトバックログの順番を修正して、どの作業を優先すべきかを確認した方が良いでしょう。

## 5.8.2 スプリントプランニング

スプリントプランニングは、スプリントで実行する作業を選択することで、スプリントを開始するものです。スプリントプランニングは、スクラムチーム全体の共同作業によって作成されます。

プロダクトオーナーは、最も重要なプロダクトバックログアイテムと、それらがプロダクトゴールにどのようにマッピングされているかを出席者同士で話し合う準備ができていることを確認します。スクラムチームは、アドバイスを提供するために他の人をスプリントプランニングに招待することもあります。

スプリントプランニングでは、次のようなテーマに取り組んでいます。

**なぜこのスプリントに価値があるのか?**
プロダクトオーナーは、現在のスプリントでプロダクトの価値や実用性をどのように高めることができるかを提案する。その後、スクラムチーム全体が協力して、そのスプリントがなぜステークホルダーにとって価値があるのかを伝えるスプリントゴールを定義します。スプリントゴールは、スプリントプランニングが終了する前に最終決定しなければなりません。

**このスプリントで何ができるのか?**
開発者は、プロダクトオーナーとの話し合いを通じて、プロダクトバックログから現在のスプリントに含めるべきアイテムを選択します。スクラムチームは、この話し合いの中で、これらのアイテムをリファインメントすることができ、理解と自信を深めることができます。

スプリント内でどのくらいの作業を完了できるかを選択することは、特にスクラムチームがアジャイルやスクラムに慣れていない場合には難しいかもしれません。このような状況では、ステークホルダーとベロシティの期待値を共有することに注意することをお勧めします。しかし、ステークホルダーの期待値をよく理解することは、スプリント内における作業量を管理できるようにするために有用です。

開発者は、チームの過去の実績、今後のキャパシティ、完成の定義（DoD）について知れば知るほど、スプリント予測に自信が持てるようになります。

**選ばれた仕事をどうやって実現するか?**
開発者は、インクリメントを完成させるために選んだ PBI ごとに、DoD を満たす作業を計画する必要があります。これは多くの場合、プロダクトバックログアイテムを 1 日以内の小さな作業項目に分解することで行われます。これは開発者の独自の判断で行われます。プロダクトバックログアイテムを価値あるインクリメントにするにはどうすればよいか、誰も教えてくれません。

スプリントゴール、スプリント用に選択されたプロダクトバックログアイテム、およびそれらを提供するための計画は、まとめてスプリントバックログと呼ばれます。

スプリントプランニングは、1ヶ月のスプリントの場合、最大 8 時間のタイムボックスとなっています。より短いスプリントの場合は、通常、イベントの時間は短くなります。

## 5.8.3 デイリースクラム

デイリースクラムの目的は、スプリントゴールに向けた進捗状況を確認し、必要に応じてスプリントバックログを調整し、今後予定されている作業を調整することにあります。

デイリースクラムは、スクラムチームの開発者を対象とした 15 分間のイベントです。複雑さを軽減するために、スプリントの就業日ごとに同じ時間と場所で行われます。プロダクトオーナーやスクラムマスターがスプリントバックログアイテムに積極的に取り組んでいる場合は、開発者として参加します。

開発者は、デイリースクラムでスプリントゴールに向けた進捗を確認し、次の日の作業のための実行可能な計画を作成する限り、好きな構造や技法を選択することができます。これにより、集中力が生まれ、自己管理能力が向上します。

スクラムを毎日行うことで、コミュニケーションが改善され、障害が特定され、迅速な意思決定が促進され、結果的に他の会議の必要性がなくなります。

開発者が計画を調整することが許されるのは、デイリースクラムだけではありません。開発者は、その日のうちにミーティングを行い、スプリントの残りの作業の適応や再計画について、より詳細な議論を行います。

以前のバージョンのスクラムガイドでは、各チームメンバーが答えるべき 3 つの具体的な質問が表記されていました。それらは

- 今日の私の仕事は?
- 昨日は何を完成させたのか?
- 何か障害があってできないことはありますか?

多くの人がこのリストを、尋ねて答えるべき唯一の質問であると解釈しました。そのため、2020 年のスクラムガイドから削除されました。

スクラムチームは、タスクが完了し、フローが維持されることを確実にするために、関連することは何でも質問し、答え、議論しなければなりません。

その日の仕事を最善の方法で計画・実行し、障害や課題に対処するために、15 分をどのように使うかは、すべてチーム次第です。

### 5.8.4 スプリントレビュー

スプリントレビューの目的は、スプリントの成果を点検し、今後の適応を決定することです。スクラムチームは作業の結果を主要なステークホルダーに提示し、プロダクトゴールに向けた進捗状況を議論します。

このイベントでは、スクラムチームとステークホルダーが、スプリントで達成されたことや環境の変化を確認します。これらの情報に基づいて、出席者は次に何をすべきかについて協力します。また、新しい機会に合わせてプロダクトバックログを調整することもあります。スプリントレビューはワーキングセッションであり、スクラムチームはそれをプレゼンテーションに限定することは避けるべきです。

スプリントレビューは、スプリントの最後から2番目のイベントで、1ヶ月のスプリントでは最大 4 時間のタイムボックスになっています。短いスプリントでは通常このイベントは短くなります。

### 5.8.5 スプリントレトロスペクティブ

スプリントレトロスペクティブの目的は、品質と効果を高める方法を計画することです。

スクラムチームは、個人との対話、プロセス、ツール、そして完成の定義（DoD）に関して、前回のスプリントがどうだったかを検査します。検査される要素は、作業の領域によって異なることが多いです。彼らを迷わせた仮定が特定され、その起源が探られます。スクラムチームは、スプリント中に何がうまくいったのか、どのような問題が発生したのか、それらの問題がどのように解決されたのか、あるいは解決されなかったのかを議論します。

従来、チームは 3 つの項目について具体的に話し合っていましたが、デイリースクラムミーティングと同様に、議論はこれらのトピックに限定されるものではありません。しかし、スプリントレトロスペクティブの際に、これらの質問をすることは役に立つかもしれません。

- 何をやめればいいのか?
- 私たちは何を続けていけばいいのでしょうか?
- 今後のスプリントで改善すべき点は?

なお、質問は会話を進めるのに役立つだけで、スクラムチームは効果を高めるために最も役立つ変更点を特定する必要があることに注意してください。最もインパクトのある改善点は、できるだけ早く対処されます。それらは次のスプリントのスプリントバックログに追加されることもあります。

スプリントレトロスペクティブは、スプリントの締めくくりです。通常の 1 ヶ月のスプリントでは最大 3 時間、それより短いスプリントでは通常もっと短い時間で行われます。

## 5.9 スクラムのアーティファクト

次の 3 つがスクラムのアーティファクト（作成物）とされています。

- プロダクトバックログ
- スプリントバックログ
- インクリメント

アーティファクトとしては定義されていませんが、これらの概念はスクラムのアーティファクトと密接に関連しています。

- プロダクトゴール
- スプリントゴール
- 完成の定義（DoD）
- ユーザーストーリー（および、エピックやフィーチャーなどのバリエーション）
- タスクの分解

3 つのアーティファクトと2つ目のリストの間のリンクは、アーティファクトを作成または維持するためのコミットメント（確約）と表現できます。

- プロダクトバックログのコミットメントはプロダクトゴール
- スプリントバックログのコミットメントは、スプリントゴール
- インクリメントの場合は、完成の定義（DoD）

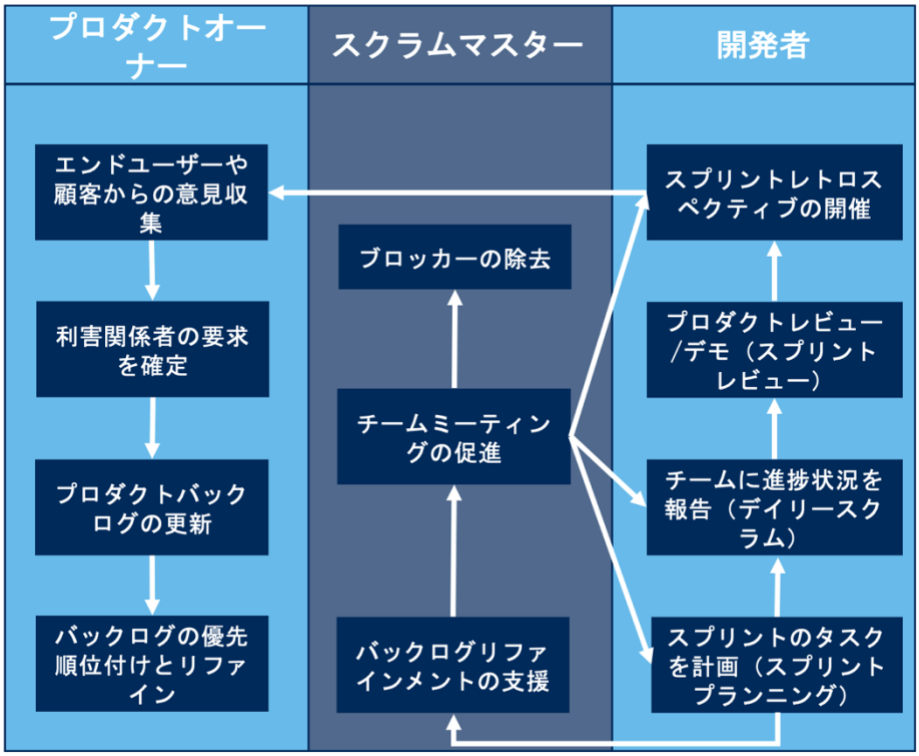
コミットメントではありませんが、ユーザーストーリーはプロダクトバックログアイテムを説明するために使用され、スプリントプランニングの際には関連するタスクに分解されます。

アジャイルスクラムで定義されている説明責任は、非常にシンプルであることに気づいたかもしれません。スクラムマスターやプロダクトオーナーが何をしなければならないかについて別の章を設け、イベントやアーティファクトの使用または作成における彼らの職務、あるいは実際にはその他の活動についてのみ説明するのは無意味でしょう。

その代わり、本書の残りの部分では、コンセプトやアーティファクトについて説明し、その際に、特定のイベントや、説明されたコンセプトやアーティファクトの一部における開発者、スクラムマスター、プロダクトオーナーの説明責任を説明するというアプローチをとっています。

要約すると、以下の図はスクラムの説明責任と関連する活動についての簡単なリマインダーになります。これはプロセスフローではなく、活動とアジャイルスクラムの説明責任の関係性マップと考えてください。

*図 4 スクラム活動の関係性を強調する（これはプロセス図ではありません。）*




画像は EXIN が作成したものを元にしています。Botha、J.（2021）. *[コースウェア]*. GetITright.

# 6 スクラムチームのその他の活動

## 6.1 ポートフォリオ、プロダクト、ロードマップ

厳密に言えば、スクラムはポートフォリオマネジメントについては、なんのガイドラインも示していません。スクラムのガイドラインはプロダクトのレベルから始まっています。実際のところ、プロダクトは何もないところから突然生まれてくるわけではありません。

重要な意思決定が必要な組織では、どのような場合でも、連続的な決定の連鎖（イベントホライズン）が発生します。このことは、それぞれの組織の中で適切な意思決定をするためのプランニングが含まれます。そうした決定は、十分に与えられた情報に基づく判断になります。

組織では、プランニング活動と関連した決定が上位から下位へ連続（カスケード）して下されます。プランニングは戦略レベルから始まり、プロダクトマネジメントレベルを経て、最終的にオペレーション（開発者）として作業を行うレベルに至ります。オペレーション環境は、*オペレーションの実行*（現状の運用）と*オペレーションの変更*（プロジェクト）に分けられます。スクラムは主に後者に焦点を当てています。

スクラムを機能させ、スケールさせるためには、イベントホライズンの基本、つまり戦略からオペレーション実行までのカスケードのレベルを理解する必要があります。スクラムの文脈では、次のようになるでしょう。

- 組織目標
- ポートフォリオの決定
- バックログ作成
- バックログリファインメント
- スケールドデリバリー計画
    - Nexus、Scrum@Scale、あるいはリリース計画
- スプリントプランニング

決定について話すのは有益ですが、アジャイル環境ではこれらの決定は固定のものではなく、決定や計画の変更は決して問題ではないことに注意してください（計画に従うことよりも変化に対応すること）。不必要な計画で時間を無駄にしないようにします。カスケードの各レベルで最小限の計画を行います。このようにすることで、状況や顧客のニーズについてより多くのことを学んだり（経験主義）、時間の経過とともに現実に対する理解が変化した場合に、計画を変更することが非常に容易になります。

Dwight D. Eisenhower（ドワイト D アイゼンハワー）元米国大統領は、かつてこう言いました。戦いに備えるとき、私はいつも、計画は役に立たないが、計画は欠かせないと考えている。

アイゼンハワーの言葉を思い出してください。計画を立てることは、学びと理解の旅の一部のようなものであるため、必要不可欠です。計画そのものではなく、計画し続けることに焦点を当て、変化と学習を奨励し、計画の変更をできるだけ簡単にできるようにしましょう。

## 6.2 ポートフォリオ計画

多くの大企業では、やらなければならない作業をより広範なポートフォリオで管理しています。そのポートフォリオとは、プロダクト、システム、バリューストリーム、サプライチェーン、投資、さらにはプログラムなどを指します。

ポートフォリオ管理とプランニングの目的は、どのプロダクトが組織の目標と目的をサポートするかを決定することです。

組織の優先順位に基づいて、どのような順番でプロダクトに取り組まなければならないのか、どのような条件で満たされるのかを理解することができます。

ポートフォリオ計画には、幅広いステークホルダーが参加し、プロダクトオーナーも参加する必要があります。ポートフォリオの計画期間は、より長期的なもので、6 ヶ月から12ヶ月先を想定しています。

質問はこうあるべきです。ポートフォリオのプロダクトは、組織の目標や目的にどのように近づけるのか?そして、アジャイル手法に則って、ポートフォリオの各プロダクトのロードマップからなるポートフォリオ・バックログを作成することで、組織のポートフォリオに対する考え方を表現することができます。

## 6.3 プロダクトを思い描く

潜在的なプロダクトの本質を思い描き、どのようにしてプロダクトを作ることができるのかという大まかなアイデアを作ることが出発点となります。プロダクトオーナーとステークホルダーが集まり、1 年以上の計画期間をもって、新しいプロダクトを高いレベルで構想します。しかし、世の中の変化はあまりにも早く、頻繁に起こるため、できれば短い計画期間に留めることをお勧めします。

参加者は、プロダクトの目的や目標を考慮し、議論が常にプロダクトを支える組織の目標や目的にリンクしていることを確認する必要があります。

プロダクトゴールを定義し、ハイレベルなプロダクトバックログを作成し、ハイレベルなプロダクトロードマップ（内容が更新され続ける文書）を定義することで、バックログをどのように価値に変えていくかについての見解を作成することが、構想*策定の主なアウトプット*となります。これらのアウトプットは、より高いレベルのポートフォリオ計画のインプットとなります。

最初のハイレベルなプロダクトバックログは、プロダクトオーナーが全体的で大規模なフィーチャーや、ステークホルダーがプロダクトに期待しているベネフィットを特定するための大まかな出発点となり、詳細な要件収集を開始することができます。

プロダクトロードマップは、プロダクトの目的や、プロダクトをどのように構築して提供していくかという段階的な性質を伝えるための効果的なツールであり、メカニズムでもありますが、当初の想像とは大きく異なることがよくあります。

## 6.4 プロダクトとプロダクトゴール

スクラムチームは、プロダクトゴールを計画の参考にする必要があります。プロダクトゴールは、プロダクトの将来の状態を記述し、プロダクトがサポートする上位の組織目標や目的が何であるかを示すからです。プロダクトゴールは、プロダクトバックログの一部です。通常、プロダクトバックログアイテムを定義する前に書き込まれ、プロダクトバックログに追加されたアイテムがプロダクトゴールの達成に組織を近づけるかどうかを確認するための第一の「チェック」となります。

プロダクトゴールとプロダクトバックログを定義するためには、まずスクラムの「プロダクトとは何か」という考え方を定義する必要があります。

スクラムでは、プロダクトを「誰かに価値を提供するための手法、目的達成手段」と定義しています。つまり、プロダクトからどのような「価値」が提供されるのかを理解するだけでなく、その「誰か」が誰なのかを知る必要があるということです。

明確な境界線を設定する必要があります。ステークホルダーが誰であるか、その役割は何か、その説明責任は何か、彼らにとっての価値は何かを知ることは、スタート地点として最適です。

スクラムでは、プロダクトとサービスの区別はなく、プロダクトは物理的なものから非常に抽象的なものまであります。

**プロダクトゴールとは、スクラムチームの長期的な目的です。**すべてのスプリントゴールは、プロダクトゴールを達成するためのステップでなければならず、チームは次の目標に取り組む前に、一つの目標を達成するか断念しなければなりません。

## 6.5 プロダクトゴールとビジネスバリュー

プロダクトゴールとは、創造するプロダクトと価値を描いたアウトプットであり、プロダクトとは何かをハイレベルに記述した声明文のことです。

しかし、ビジネスバリューとプロダクトゴールはどのように関係しているのでしょうか？

ビジネスは、その戦略を達成するために組織が実現しなければならない達成目標を定義します。組織のポートフォリオに含まれるプロダクトにより、時にはこれらの目標を達成できることもあります。私が「時には」という言葉を使ったのは意図的で、設定された目標のいくつかを達成するための手段が、他にも組織内に存在するからです。

したがって、組織目標とプロダクトゴールの違いは、前者がビジネス戦略に関するものであり、後者がプロダクトマネジメントに関するものであるということです。

スクラムにおける*プロダクトゴール*が、一般的なプロダクトマネジメントで使われる“プロダクトゴール”と異なるのは、スクラムではこの言葉がより特有な形で使われていることです。スクラムの古いバージョンでは、*プロダクトビジョン*への言及があり、プロダクトゴールという言葉の使用は、プロダクトマネジメント分野での一般的な使用よりもこの言葉に近いものです。

プロダクトゴールとは、そのプロダクトの長期的な目的や将来の状態を簡潔に表したものです。プロダクトゴールはプロダクトの内容（WHAT）のことで、理由（WHY）を補足するものであると言うこともできます。

また、プロダクトゴールは、プロダクトバックログに取り組むスクラムチームの全体的なコミットメントを反映しており、活動の焦点を提供するものとも言えます。

組織がその戦略的意図を達成できるようにすること自体が価値であり、それがプロダクトゴールの主眼です。

しかし、一般的にプロダクトは組織や顧客に可能性を与えるものであり、それゆえに様々な形でビジネスに価値をもたらすものであると考えることが重要です。つまり、プロダクトは、組織のプロセスや作業方法を改善するという内部的な価値と、エンドユーザーである顧客がこれまでできなかったことをできるようにしたり、これまでできていたことをよりよくできるようにするという外部的な価値の両方を生み出すことが多いのです。プロダクトはさまざまな方法でこれを実現し、同じプロダクト、あるいはフィーチャーであっても、そのプロダクトのユーザーによって異なる価値を生み出すことがあります。プロダクトの有用な/価値あるインクリメントとして価値を提供することは、プロダクトゴールで定義された範囲を超えます。

価値あるものを段階的に提供することは、プロダクト全体の目標に関連していますが、プロダクトゴールはプロダクトの価値のすべてを説明するものではありません。

気をつけなければならないのは、プロダクトが生み出すあらゆる価値を捉えてプロダクトゴールを複雑にしすぎないことです。複雑なプロダクトゴールは、焦点が合いにくく、使い勝手が悪くなります。

プロダクトゴールとは、そのプロダクトが実現する最も重要な戦略的目標や目的の説明だと考えてください。

この点に関して次に問うべきことは、「価値や価値が高いものとは何か」ということです。

価値は、プロダクトやフィーチャーが課題を解決する構成要素によって決定されます。価値は非常に主観的な用語であり、定義することは難しいですが、定義する試みはなされるべきです。

その主な理由は、価値を創造し提供するには資源を消費するため、目に見えるコストがかかるからです。したがって、価値についての質問は、次の質問に答えられなければなりません。"創造された価値は、その価値を創造するためのコストよりも金銭的に価値がありますか?"

アジャイル環境では、ここが面白いところです。

## 6.6 実質価値の測定

従来の財務比率では、アジャイルであることで得られた利益や、プロダクトの価値を実質的に測ることはできません。

その代わりに、改善と先行指標となる測定基準（メトリクス）に焦点を当てることをお勧めします。財務報告書が組織のパフォーマンスにプラスの影響を与えることはほとんどありませんが、日々のパフォーマンスに焦点を当てることは可能です。

従来の財務指標の構成と使用は、伝統的な工業化された世界観が大きく影響しており、アジャイル環境とは相容れないものとなっています。この課題は、アジャイル環境に限ったことではありませんが、1990 年代に組織がリーンを採用し始めたときにすぐに明らかになりました。

なぜ従来の財務比率は問題が多く、アジャイル環境では誤解を招くことが多いのでしょう?

一部の財務比率を使用した証拠は紀元前 300 年にさかのぼりますが、ビジネスとして企業への投資が盛んになったことで、財務比率の有用性が浮き彫りになりました。財務比率が解決した問題は単純でした。比率を使うことで、投資家は構成、産業、リスク、市場などが大きく異なる企業への投資を比較することができます。インドで木箱を製造している会社に投資すべきか、アメリカで自動車を製造している会社に投資すべきか、EU で金融機関に投資すべきか、投資家は比率を使って考えることができます。主要な比率を見ることで、その判断が容易になります。

ここで言う比率とは、伝統的な会計上の比率をすべて含みます。

- **流動性比率**は、企業の短期および長期の債務に対する返済能力を測定するものです。
- **レバレッジ比率**は、総資本に占める負債の割合を測定するものです。
- **効率性比率**（財務活動比率とも呼ばれる）は、企業が資産や資源をどのように活用しているかを測定するものです。
- **収益率**は、売上高、貸借対照表の資産、営業費用、および資本に対する企業の利益創出能力を測定するものです。
- **時価総額比率**は、企業の株式の株価を評価するものです。

ここでは、伝統的な比率のどれもアジャイル環境では役に立たないと述べることは意図していません。しかし、これらの比率の多くは、1ヶ月、6 ヶ月、1 年、5 年、さらには10 年と測定していくと、見方によっては全く異なることがわかります。

従来の会計実務とその結果としての財務比率の問題点は、会計士が財務報告期間を始まりと終わりのあるものと考えていることです。また、ウォーターフォール型のプロジェクトでは、従来の測定方法や指標を使用するのは簡単です。ウォーターフォール型プロジェクトの定義は、始まりと終わりがあるということです。この場合、コストと実現した利益を比較することで、プロジェクトの投資対効果（RoI）を簡単に計算できます。

アジャイル環境では、ビジネスのパフォーマンスを見る方法が大きく異なります。

ここでは、すべてのパフォーマンス指標を継続的に改善することに焦点が当てられており、そこには始まりも終わりもありません。そのため、従来の財務比率や会計プラクティスは、アジャイルやリーンを使用する際にはほとんど意味を持ちません。

実際、従来の比率を使用すると、短期的な改善につながり、長期的にはマイナスになることがあります。アジャイル環境では、短期的にはマイナスであっても、長期的な改善を行うことが推奨されます。

アジャイル組織における会計実務を考えるとき、リーンから学んだ教訓を非常に効果的に適用することができます。企業はすぐに、組織や「優れた財務実績」を構成するものについて、より広範なシステムビューを含む、根本的に異なる会計プラクティスが必要であることに気づきました。このことに気づいた瞬間、リーン会計が誕生したのです。

投資対効果（RoI）、内部収益率（IRR）、正味現在価値（NPV）をアジャイルの投資判断を行うための指標として使用すると、厄介なことになります。

ここでは、これまでの常識が通用しなくなる理由をご紹介します。

- アジャイルでは、**プロジェクトの開始日や終了日は**ありません。実際には、真の意味でプロジェクトは存在せず、価値を提供する短いインクリメントだけが存在すると言えるでしょう。
- 要求は時間の経過とともに進化することが認められ、または推奨されているため、時間の経過とともに**何が行われるのかについて、前もって限定されたものは**ありません。

RoI、IRR、NPV は、アジャイル環境で投資判断をしたい場合には、問題のある測定基準になります。そのため、別の測定方法が必要になります。

さらに複雑なのは、従来はオペレーションだけの活動であった継続的な改善が、すべてのスプリントの一部として期待されているため、「オペレーションのプロジェクト」と「オペレーションの実行環境」の境界がさらに曖昧になっていることです。

リーン環境（リーン会計より）と同様に、成功の最も適切な尺度は次のようになります。

- フローの改善
- 品質の向上または維持
- パフォーマンスの向上（納期、リードタイム、生産性）

そして、結果として：

- 収益性の向上

リーン（アジャイル）会計の焦点は、広い範囲でのシステムビューです。軌道修正のために特定の期間で改善を報告することがありますが、これらの報告は暫定的な測定であり、必ずしも長期的に会社の健全性を反映しているわけではないことを理解することが重要です。

このような考え方の変化は、部門やプロジェクトに資金を提供するという慣習がもはや意味をなさないことを意味します。組織の収入はプロダクトのパフォーマンスに直結しているため、プロジェクト/変革とオペレーションの両方を実行するプロダクトに資金を提供する方が賢明です。プロダクトを生み出し、販売し、サービスを提供し、メンテナンスを行うバリューストリーム全体を、そう、エンドツーエンドで考えなければならないのです。このようなシステムビューは、会計士が大学で学ぶ内容とは大きく異なります。

リーン／アジャイルの新しい RoI の見方は、よりホリスティックなものです。投資パフォーマンスを効果的に測定するためには、バリューストリーム全体でリソースを消費するすべてのものの合計と、プロダクトを顧客に販売することで得られる利益とを対比させる必要があります。報告は、企業のパフォーマンスを一連の関連性のない過程として反映する報告ではなく、時間的に連続したパフォーマンスと比較して、時間的にスライスされた過程にならなければなりません。

プロダクトを提供するバリューストリームの総コストと得られる収入を比較しても、本質的には RoI だと主張するかもしれませんし、それはそうです。しかし、それが将来的にどのように役立つのでしょうか？過去の RoI は、プロダクトの将来の RoI を測る上で非常に信頼性の低い指標です。

RoI のような比率の欠点は、遅行指標であるということです。つまり、望ましい投資収益率を達成できなかったことはわかるかもしれませんが、軌道修正するための指標としては役に立たないのです。リーン会計がパフォーマンスの先行指標として改善に焦点を当てているのは、まさにこの理由によるものです。

おもしろい余談ですが、トヨタの目標のひとつは、会社全体のすべてを毎年 2%ずつ改善することです。小さなことのように聞こえますが、彼らは1950 年からこれを達成しており、その複合効果は非常に大きいのです。

今月、先月、先々月と改善されていれば、RoI の改善に向けて順調に進んでいるはずです。もし今月改善されていないのであれば、すぐに何かをして軌道修正する必要があります。

アジャイル・アライアンスのメンバーは、2012年から 2016 年にかけて、アジャイル会計の標準化に取り組むプロジェクトを実施しましたが、このイニシアチブのアウトプットから判断すると、アジャイル環境における財務パフォーマンスを評価するための、バリューストリーム全体を統合した継続的なアプローチとして、アジャイルに目を向けることがほとんどできませんでした。

リーン会計は、今のところ、アジャイル環境での投資判断に最も適した方法ですが、会計とは何か、そして組織の中でどのように行われるべきか、という理解をかなり大きく変える必要があります。

では、これらを踏まえた上での結論は?

改善と先行指標となる測定基準（メトリクス）に焦点を当てます。財務報告書が組織のパフォーマンスにプラスの影響を与えることはほとんどありませんが、日々のパフォーマンスに焦点を当てることは可能です。

## 6.7 プロダクトバックログの管理について

まともなプロダクトバックログがなければ、スクラムはうまくいきません。プロダクトバックログは、すべてのスクラム計画活動の中心であり、スクラムチームの唯一の真実の源です。プロダクトバックログには、チームが行うべきすべての作業と、その作業の相対的な優先順位が記述されています。プロダクトバックログになければ、要件とそれに関連する活動は存在しません。

プロダクトバックログには、顧客のニーズだけでなく、顧客のニーズを確実に満たすために必要な技術的なタスクもすべて記載されています。したがって、プロダクトバックログは、プロジェクト中に提供する必要のある機能的および非機能的な要件の「優先順位付けされたリスト」（正確には「**順序付けされたリスト**」、以下で明らかになる）と表現することができます。要件が満たされると、関連するアイテムはプロダクトバックログから削除されます。

また、企業によっては、計画、リソース、その他の準備段階のように、要件に対して提供されるために必要なタスクも含まれています。

すべてのバックログは、そのプロダクトのバックログを管理する責任を負うプロダクトオーナーによって所有されています。彼らの仕事は、バックログが最新の状態であること、そして順序付けられていることを保証することです。ただし、これはスクラムチームの他のメンバーの助けを借りて行います。

一般的な慣習として、プロダクトバックログが 4 つの特定の基準を満たしていることを確認します。この概念は Mike Cohn 氏によって最初に定義され、現在ではスクラムコミュニティで広く使用されています。この基準は、頭文字をとって DEEP と呼ばれています。

- Detailed appropriately（適切な詳細さ）
- Estimated（見積もり）
- Emergent（創発的）
- Prioritized（優先付け）

スクラムが Prioritized という言葉を避けているのは、文脈や組織によって意味が異なるからであることはすでに述べたとおりです。スクラムは **Ordered** という用語の使用を好みます。スクラムの文脈での DEEP は、DEEO となります。

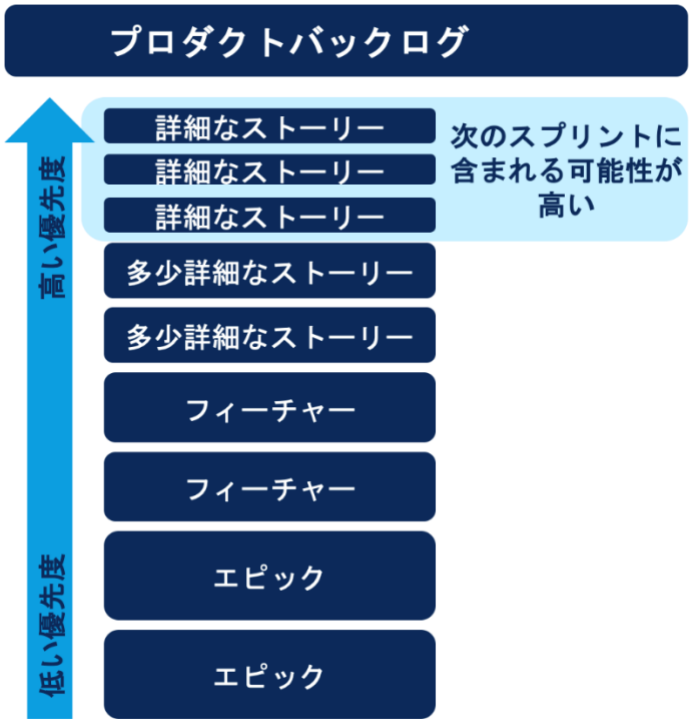
- Detailed appropriately（適切な詳細さ）
- Estimated（見積もり）
- Emergent（創発的）
- Ordered（順序付け）

### 6.7.1 適切な詳細さ

プロダクトバックログは、要求を詳細なアクティビティに分解する場所ではありませんが、要求に関連する作業を完了するためにどれくらいの時間がかかるかを把握できるように、十分に詳細なものでなければなりません。

要求は通常、ストーリーとして書かれますが、ストーリーはさまざまなレベルの詳細を持つことができます。

*図 5 プロダクトバックログの上位になるほど詳細度が増し、大きなストーリーは小さな（より実用的な）ストーリーに分解されます。*




図は EXIN 制作：Botha、J.（2018）. *スクラムマスターとプロダクトオーナー[コースウェア]*. GetITright.

優先度の高いアイテムは、その要件を満たすために必要な努力やスキルを判断できるように、十分に詳細なレベルまで分解する必要があります。

そのため、次のスプリントや2回目のスプリントで取り組む可能性のある重要なアイテムは、プロダクトバックログの詳細化やスプリントプランニングの際に、適切なレベルまで分解する必要があります。

つまり、プロジェクトの進捗に応じて、ストーリーや要件をより詳細なレベルに分解する作業が継続的に行われるということです。しかし、これは必要な場合にのみ行われるもので、すべてのストーリーや要件に対し

て前もって行われるものではありません。というのも、ストーリーが最初にプロダクトバックログに追加された時点では、このアイテムや関連するすべての要件が重要であるかどうか、あるいはバックログに残るかどうかさえも定かではないからです。優先度の低い要件がバックログから削除されたり、後から別の要件が出てきてそれらに取って代わることもよくあります。

アジャイルでは、使われないであろうものに労力を浪費することはありません。これが、成功するために「Just enough（ちょうどいい）」仕事をするという意味です。

エピックとフィーチャーは、ストーリーの大きさや複雑さを表現するための**補完的なプラクティスで**あり、スクラムガイドではストーリーの種類を区別していません。

### 6.7.2 見積もり

プロダクトバックログのすべてのアイテムには、そのストーリーを完成させるために必要な労力に関する何らかの見積もりが必要です。明らかに、ストーリーが大きければ大きいほど、見積もりは難しくなります。エピックの見積もりは精度が低くなりますが、それは問題ではありません。エピックとして残されたストーリーは、通常、当面の重要性が低いからです。

スプリントプランニングを行うためには、見積もりは必須の活動です。インクリメントとして機能を達成するための努力を見積もる際には、完成の定義（DoD）を考慮する必要があります。これは、1つのスプリントに複数のインクリメントが存在する可能性があるため、すべてのインクリメントには完成の定義（DoD）があることを意味します。

機能横断的なチームにより、活動の実行時に足りない情報をもたらすので、タスクを見積もる際に大きなメリットがあります。機能横断的チームは、外部リソースに依存せずに作業を達成するために必要なすべての能力を備えています。タスクを完了するための知識とスキルを持つチームメンバーが常に1人はいるので、タスクを見積もることができる人が常にいるのです。

### 6.7.3 創発的

創発的の意味は、プロダクトバックログとは、プロダクトのライフサイクルの期間に実現すべきことを確定的に記述したものではありません。顧客の要求は、プロダクトライフサイクルの期間中に変わり、バックログに新しい要求が追加されることがよくあります。しかし、バックログにアイテムが追加されるのは、その時だけではありません。

プロダクトを段階的に作成していく中で、それまで知られていなかった新しい要求が発見されることがよくあります。これらが機能するためには、他の要求に依存していることが多く、また、機能的ではなく、非機能的な要求が少なくありません。

つまり、プロダクトバックログが創発的であると述べられている場合、要はそれが進化するということです。

### 6.7.4 順序付け

プロダクトバックログのすべてのアイテムは、順序付けされたアイテムで表示され、最初に行うべきものはプロダクトバックログの最上部に表されます。順序とは、当該ストーリーの完成や実装のための重要性や実装順序、他の重要なストーリーへの依存度を示すものです。

## 6.8 プロダクトバックログのリファインメント

プロダクトバックログは常に変化するものであり、プロダクトバックログにアイテムが継続的に出入りしたり、変更されたりするため、注意深く維持する必要があります。

すでに語られている DEEO の活動は、プロダクトバックログのメンテナンスに関わるタスクです。この活動はプロダクトバックログリファインメントと呼ばれていいます。

プロダクトオーナーはリファインメントの説明責任を負いますが、これは一人で行う活動ではありません。プロダクトオーナーは、開発者の協力を得てこの作業を行います。開発者は、プロダクトバックログの既存のアイテムやプロダクトバックログに追加された新しいアイテムを再順序付けし、再見積もりするために、それぞれの技術的な洞察力で貢献します。

開発者が貢献できるのは、ストーリーの完成までにどれくらいの時間がかかるかを判断したり、重要なストーリーが他のストーリーに依存していて、その依存関係によって重要性が継承されているかどうかを認識できる技術的なノウハウを持っているからです。また、スクラムチームのメンバーは、大規模なストーリーをスプリントで実行しやすいように分割する方法を知っています。

プロダクトオーナーがプロダクトバックログのリファインメントに説明責任を持っているとはいえ、それは非常に協調的な活動です。

バックログリファインメントは定義されたスクラムイベントではありませんが、スクラムチームのメンバーはこの活動のための時間を計画しなければなりません。

なぜならば、それは計画された（タイムボックス）イベントではないためで、プロダクトバックログのリファインメントは、必要なときに行われ、多くの場合、他のスクラムイベント中に行われます。要求の変更、依存関係、発見された要求に関するコミュニケーションミスなどの問題が発生した場合は、その場で対処しなければなりません。

スケーリングやスプリントプランニングのイベントでは、必然的にある程度のリファインメントが行われます。また、スプリントレビューやスプリントレトロスペクティブでも同じことが起こるのは当然のことです。完成したストーリーは削除する必要があり、ストーリーは現在判明していることに照らし合わせて再編成する必要があります。また、終了したばかりのスプリントで依存関係や新しい要求が見つかった場合には、ストーリーを分解して新しいストーリーを追加することもあります。

毎日、時間をとってプロダクトバックログとスプリントバックログを確認することをお勧めします。例えばデイリースクラムの中で問題が発生した場合は、できるだけ早く話し合う必要があります。

推奨としては、チームはスクラムチームの時間の10%をリファインメントに費やすべきです。しかし、ほとんどのチームはプロダクトバックログのリファインメントに費やす時間はもっと少なくなっています。

プロダクトバックログのリファインメントには、通常、次のような活動が含まれます。

- **新たに発見されたアイテムは、バックログに追加しなければなりません。**これらは、顧客からの新しい要求であったり、前のスプリントで発見された不足している機能的、非機能的な要求や依存関係であったりします。
- **バックログアイテムは、**最も重要なアイテムを先頭にして**並べられます。**
- スプリントプランニングや見積もりをしやすくするために、**バックログアイテムは適切なサイズにする必要があります。**
- **大きな、あるいは重要性が曖昧なアイテムは、**次のスプリントで使用できる小さなユーザーストーリーに分解する必要があります。
- **重要なアイテムは、**スプリントプランニングを容易にするために、**リファインメントされています**（より良い記述）。
- **依存関係に注意を払い、**実行する必要のある順序を記録する必要があります。
- **バックログは、**すべての新しいアイテムが追加されているか、実行順序が決定されているか、サイジングや完成予想が適切であるか、不要になったアイテムがバックログから削除されているかなどを確認するために、**一度見直さなければならない。**

## 6.9 プロダクトバックログアイテムの作成

前述したように、要求収集はスクラムにおける一回限りのイベントではありません。プロジェクトの初期には多くの要求収集が行われますが、この活動は詳細な要求を得ることや、エピックを細かいユーザーストーリーに分解することに多くの時間を費やすことに重点を置くべきではありません。

したがって、大きなストーリーの分解は継続的な活動であり、作業が行われる直前に行うのがベストです。こうすることで、チームは作業する要求が最新の要求であり、2年前の要望ではないことを確認できます。要するに、これを行うことで、注力していることが最新であり、最大の価値を生み出すことができるようになるのです。

そのため、次のスプリントで取り組まれないストーリーの分解に多くの時間を費やすことは、実際には無駄であると考えられます。これは、アジャイルの原則である「Just enough（必要最低限）」を行うことと一致します。

一般的には、最初に要求を集め始めるときには、要求と成果を理解する必要があり、どのようにしてそれに到達するかという詳細な情報は必要ないと言ってよいでしょう。

要求を考える上での優れた方法は、まずロードマップを作成することで、プロジェクトで何を行うかというハイレベルな見解を定義し、次にロードマップの各ステージに対する要求を考え始めることです。

ロードマップを作成すると、すぐに提供できてすぐに付加価値がつくコアな要求を考え、その周辺部分に注力してプロダクトをより良いものにしていくことが必要になります。

このようなハイレベルな要求を集めるには、主要なステークホルダー全員が参加するブレーンストーミングセッションで行うことができます。セッションは、何をしようとしているのかを定義し、プロダクトのビジョンを作ることから始めます。どのような問題を解決しますか、あるいはどのような価値を提供しますか?

そして、プロダクトをハイレベルなコンポーネントに分解し、その開発をプロダクトロードマップの基礎とすることができます。

この時点では、細かいことにはこだわらず、大まかな話をします。これは、MVP（Minimum Viable Product）が何であるべきかを最初に定義するのと同じです。これが無いとプロダクトが機能しないといったコア機能に焦点を当てるべきです。

プロダクトバックログは時間をかけてリファインメントされていくものであり、より詳細で重要度の低い要求はいつでも後から追加できることを忘れてはなりません。

DSDM には MoSCoW という概念がありますが、ここで言及するのは適切なことかもしれません。

### MoSCoW

MoSCoW は、Dynamic Software Development Method[6]の開発に貢献した一人である Dai Clegg 氏によって開発された優先順位付けの手法であり、スプリントにおけるアイテムの順序付けに使用することを目的としています。要求発見プロセスで MoSCoW を使用することは非常に有益であり、スクラムチームに貴重な洞察を与えることができます。

要求を定義する際には、特定された要求が、次のいずれにかるかを問います。

- Must Have
- Should Have
- Could Have
- Won't Have

[6] DSDM は、現在では Agile Business Consortium の略で ABC とも呼ばれています。

このような場合、最初は Must Have だけに集中し、Should Have もいくつか入れておくことをお勧めします。

Must Have と Should Have の違いは、以下のように定義できます。

- **Must-Have** 要求は、必要な価値を提供するために、重要であり、すぐに実現すべきものです。これらに対して実現できなければ、そのプロジェクトは失敗となります。
- **Should Have** 要求は重要ですが、それほど緊急性の高いものではないかもしれません。一般的に、Should Have 要求は、プロダクトが完全に機能しているか、簡単に使用できることを保証するために、プロジェクトの終わりまでに提供されるべきものです。
- **Could Have** 要求は、あると便利な機能であり、Won't Have 要求は、プロジェクトの一部としてこれらのアイテムを含める正当な理由がないため、特に除外されます。

### 6.9.1 非機能要求を分解する

要求をすぐに分解してはいけないというルールには、1つだけ例外があります。それは、非機能要求です。

非機能要求とは、イニシアチブの残りの部分が依存しているにもかかわらず、顧客が求めていない部分のことです。非機能要求は、プロジェクトの残りの部分の基礎となるものなので、これらの要求に対して提供するために何が必要なのかをよく理解しておくとよいでしょう。

そのため、非機能要求は、それがわかった時点でチームが可能な限り分解します。

## 6.10 要求事項収集 – アウトプットと成果

ユーザーや顧客がどのような成果を求めているかを理解することは、スクラムにおいて非常に重要であると考えられています。これらの成果を理解することは、スクラムが最初から要求を収集する方法の基本的な部分です。

つまり、プロダクトバックログに入るための最低条件は、どのようなステークホルダーがその要求に依存するのかをよく理解し（これは文脈を提供し、依存関係を特定するのに役立ちます）、次に何を要求し、なぜそれが必要なのかを理解することです。

ユーザーストーリーがこのような構造になっているのは、このような理由によるものです。

```
As a <stakeholder ROLE>,
I want to <the WHAT of the requirement>,
so that <the WHY of the requirement>.
```

すでに収集された要求から作業を行う場合は、非常に注意が必要です。すでに収集した要求をユーザーストーリーに変換することは一切行わないことをお勧めします。

従来の要求収集技術は、上述したこととは全く逆のことを目指しています。これらの技術は、細かくて詳細な要求を定義しようとするものです。

これらをプロダクトバックログに翻訳したりコピーしたりするだけでは、非常に複雑なバックログになり、優先順位や創出された価値についての洞察はほとんど得られません。

従来の詳細な要求に基づいて作業をしなければならない場合は、要求をグループ化（アイテム分解の逆）することで、最終的にはより少ないアイテムで、より簡単に順序付け、並べ替え、見積もりができるようにして、これらからハイレベルの要求を作成することをお勧めします。

ここで使える技法は、すべての詳細な要求を付箋紙に書き、関連する要求のグループでアフィニティマップ（親和図）を作成することです。グループの説明は、粗い粒度の要求としてプロダクトバックログに追加することができます。

この技法を使うと、「Could Have」や「Won't Have」のカテゴリーに入るような要求を排除したり、優先順位を下げたりすることにもなります。

ユーザーストーリーやその他のストーリーをプロダクトバックログに表現する方法は、様々なものがあります。あらかじめ定義されたフォーマットを好む人もいれば、そうでない人もいます。あらかじめ定義されたフォーマットを使用することの欠点は、自動的に特定の方法で要求を考えることを強制されることです。

ユーザーストーリーには、上記のような文字通り一行だけのものもあれば、ステークホルダー、優先順位、依存関係、関連するアクティビティ、さらにはスプリント内で誰がそのストーリーに取り組むべきかなど、多くの詳細情報が含まれていることもあります。しかし、ほとんどの場合、これらすべてを記録するのは少し過剰です。

上記のユーザーストーリーのフォーマットは、バックログアイテムに絶対に必要な情報がすべて記録されていることを保証するものです。ストーリー形式では、「何を」「誰に」「何のために」達成しなければならないかを答えます。また、WHAT と WHY はどちらも性能基準または受け入れ基準であることに注意してください。

なお、スクラムでは要求をユーザーストーリー形式で定義することを規定していませんが、この方法で要求を記録することを強く推奨しています。

## 6.11 ユーザーストーリーについて

ユーザーストーリーがストーリーと呼ばれるのは、要求を持つステークホルダーが、感情のこもっていない言葉ではなく、物語の形で要求を伝えることになっているからです。彼らが何をしているのか、何をすべきなのか、なぜそれが重要なのかというストーリーです。

この最後の部分が実はとても重要なのです！反復的な提供はアウトプットだけではなく、ステークホルダーができるだけ頻繁に、できるだけ早く成果を達成できるように支援することが重要であると、以前に強調しました。その人のストーリーの WHY が成果なのです。

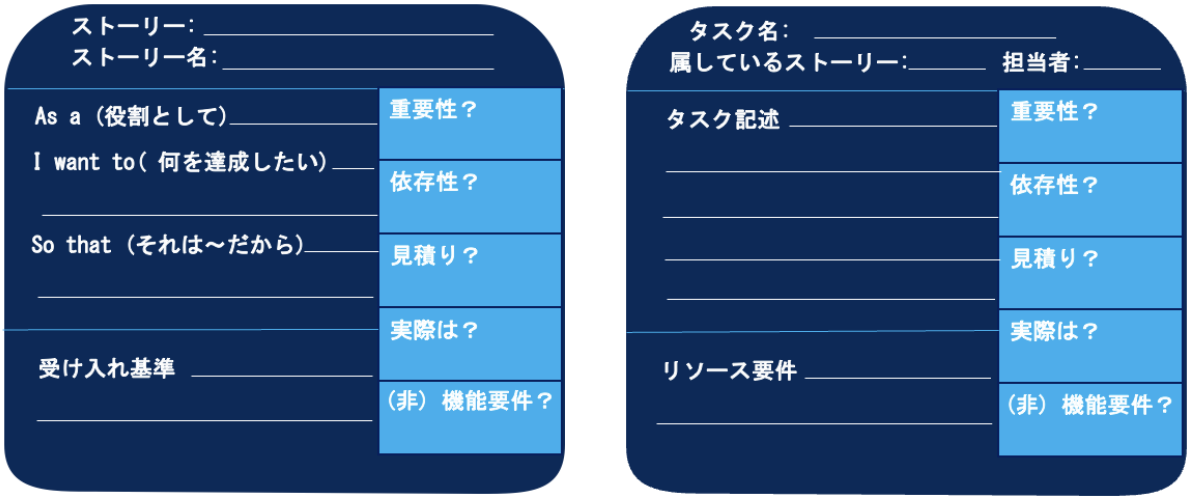
方法や理由を語ることで、それが理にかなっているかどうかストーリーを記録する人が評価するのに役立ちます。従来のように要求を書き出すだけでは、意味がありません。

意味がわからなければ、記録者は説明を求めることができます。また、理由や方法を尋ねることもできます。

具体的にはどのようにしているのですか?なぜそのようにしているのですか?なぜそんなことをしているのですか?同じ結果が得られるのであれば、他の方法でも良いのではないですか?

会話は理解につながり、理解は、ステークホルダー（多くの場合、ユーザーや顧客）のために価値を創造するためのより良い知識につながります。

*図 6 ストーリーとタスクチケットの例*




図は EXIN 制作：Botha、J.（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

ストーリー形式は、要求を記録し、理解するための非常に強力な方法です。

As a <stakeholder ROLE>,
I want to <the WHAT of the requirement>,
so that <the WHY of the requirement>.

要求をツールに保存することもできますが、物理的なカードや付箋に記録し、納品の進捗状況をカンバンボードにプロットすることをお勧めします。

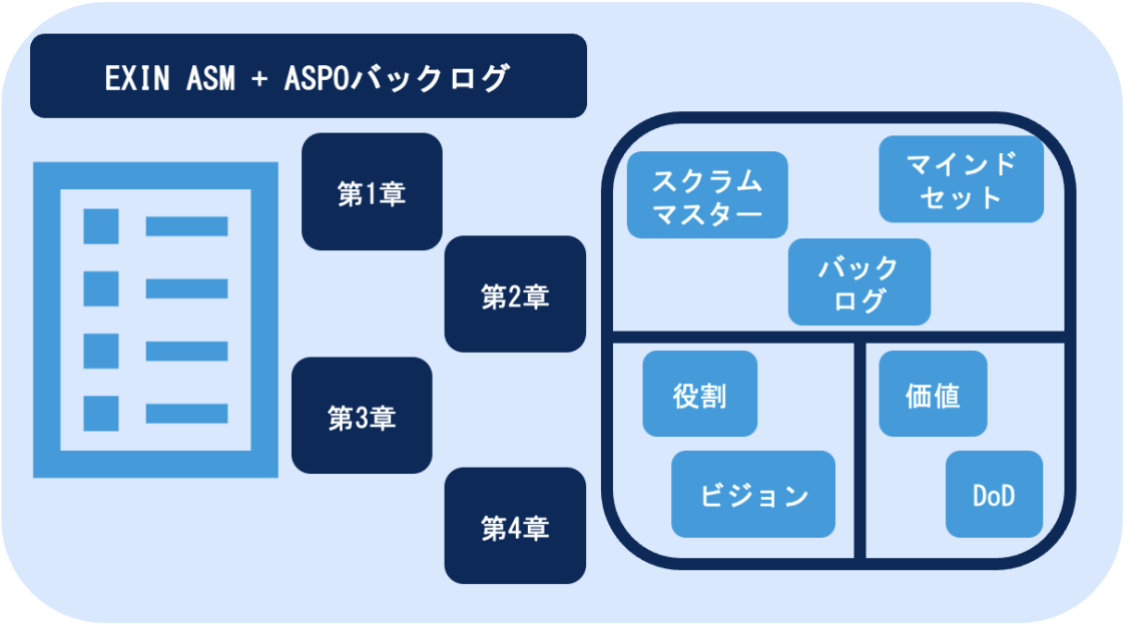
下の図はその一例です。これは、実際にこの本を書くためのカンバンで、左側にはプロダクトバックログがあります。この例では、ストーリーは非常にシンプルで、成果のみを記録していることがわかります。これは、プロダクト（この場合は本）を書くための十分な情報であり、言い換えれば、それだけで十分なのです。

このような個人的な例であっても、バックログを作成し、要求を定義し、途中で要求を改良することには価値があります。

この例では、水色の「付箋」は、上位のエピックに属するストーリーが分解されたもので、ここではより大きな濃い青色の付箋で表現されています。そして、それぞれの色には独自の意味があり、さらに情報が記録されていることもあります。これは当たり前のことではなく、高度にカスタマイズされた作業方法です。各スクラムチームは、時間をかけて独自の作業方法を適応させていきます。

要求の記録は旅の始まりに過ぎず、要求は完成して納品されるまで、どんどん洗練されていくことを忘れないでください。

*図 7 プロダクトバックログ*




画像は EXIN が作成したものを元にしています。Botha、J.（2021）. *[コースウェア]*. GetITright.

## 6.12 ユーザーストーリーとタスク分解

本章では、見積もりの技法とタスク分解の方法について説明します。ここで注意しなければならないのは、ストーリーを実現するために必要な作業がわかって初めて、ストーリーに取り組むことができるということです。

タスク分解とタスク見積もりを行うことは、スプリントプランニングの最後のステップです。スクラムマスターとプロダクトオーナーは、見積もりに関して開発者に影響を与えてはなりません。見積もりの際に発生する可能性のある問題を明確にするために、プロダクトオーナーはこの活動に参加しなければなりません。

タスク分解は単一のイベントではなく、スプリントプランニングやスケールアップした実装計画を開始する際にハイレベルで行う必要がありますが、スプリントで作業を開始する前に包括的に行う必要があります。また、必要に応じてスプリント中に再度行うこともあり、プロダクトバックログのリファインメントと同様に、効果が出て結果が出るまで繰り返し行うことを覚えておいてください。

理論的には、タスクはスプリントと同じくらいの時間がかかることもありますが、明らかにそれは悪い考えであり、現実的ではありません。そこで、2つの疑問が出てきました。

- タスクの所要時間は？
- 時間がかかる場合は、どうすればいいのでしょうか？

デイリースクラムの間、チームは通常、その日に取り組むことについて話すので、タスクが毎日のデイリースクラムのペースに合っているのが最善です必然的に 1 日以上かかるタスクが出てきますが、よく見るとサブタスクに分けて、それぞれの見積もりを出すことができるはずです。

あまりにも低レベルでタスクを分解するのも非生産的です。ルールとしては、タスクの分解は、チームメンバーのスキルセットに基づいてタスクを容易に受け入れられるレベルでなければならず、それ以下であってはなりません。

タスクを完了するための推奨最長時間は、8 時間以内とします。作業対象として選択されたユーザーストーリーに、さらにストーリーを追加したくなることがあります。次のスプリントに向けてストーリーをまとめることは、ストーリーが大きくなりすぎて管理しきれなくなる可能性があるため、お勧めできません。

8 時間レベルのタスク分解ができない場合は、デイリースクラムミーティングでチームメンバーが進捗状況を報告する時点で、追加のフィードバックを受けることを検討できます。これは非常に有益なプラクティスになります。1 日経ってもタスクに作業が残っている場合、チームメンバーは割り当てられた時間のうちどれだけそのタスクに費やしたかを報告するのではなく、タスクを完了するための推定時間がどれだけ残っているかを報告します。

このプラクティスには2つの大きなメリットがあります。

1. **必要な作業について考えさせられ**、将来、正確なタスク見積もり方法を学ぶことを助けます。。
2. **スクラムマスターを助けて、進捗を最新に更新します。**イテレーション完了までの残り時間を反映したバーンダウンチャートのように進捗を測定できます。ここで、純粋に言えば、バーンダウンは半分しか完了していないタスクを反映していません。厳密にはそうですが、未完了のタスクを反映することの利点は、チームがよりよい計画を立てるのに役立ち、また、タスクに追われているチームメンバーのモチベーションを高めることにもなります。

計画やタスク分解を行う際には、以下の「バケット」を使ってタスクの所要時間を見積もることができます。

*図 8 バケットシステムを適用したアクティビティへの見積もり工数の割り当て*




図は EXIN 制作：Botha、J.（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

## 6.13 プロダクトバックログとロードマップの作成と維持

前章では、要求収集について、まず粗い（ハイレベルな）要求から始める必要があることを説明しました。

スクラムガイドでは、プロダクトバックログを、すべての既知のプロダクト要求の順序付けされた、または大きさのあるリストと説明しています。プロダクトバックログは、最終的にこれらの要求を満たすためにプロダクトに加えられるあらゆる変更の唯一の情報源であり、プロダクトゴールと組織の目標をすべてサポートするものです。

「すべて」という言葉に注意してください。バックログには、非機能要求も含めるようにすることが重要です。また、スプリントレビューやレトロスペクティブの際に最適でないことが判明した、まだ修正が必要な重要な課題を含めるのも良いアイデアかもしれません。

プロダクトバックログは、4～5 個の粗い粒度のユーザーストーリー（エピック）から始め、イテレーションを行うごとに、これらを優先順位の高い順に分解していきます。目安としては、バックログのアイテム数は 100 を超えないようにし、最大でも 300 程度にしておきましょう。

これ以上になると、プロダクトバックログのリファインが事実上不可能になり、バックログの管理が非常に難しくなります。

100 アイテムのルールを守り、この数が増えてきたら、どのアイテムをバックログから削除できるかを確認することをお勧めします。前述のように、伝統的に収集された既存の要求を扱う場合と同様に、優先度の低い詳細なアイテムをエピックにまとめる作業を再度行います。

4 つか 5 つのハイレベルな要求から始めれば、プロダクトロードマップも簡単に作ることができます。つまり、それぞれの要求に実行予定の順番を割り当てればいいのです。

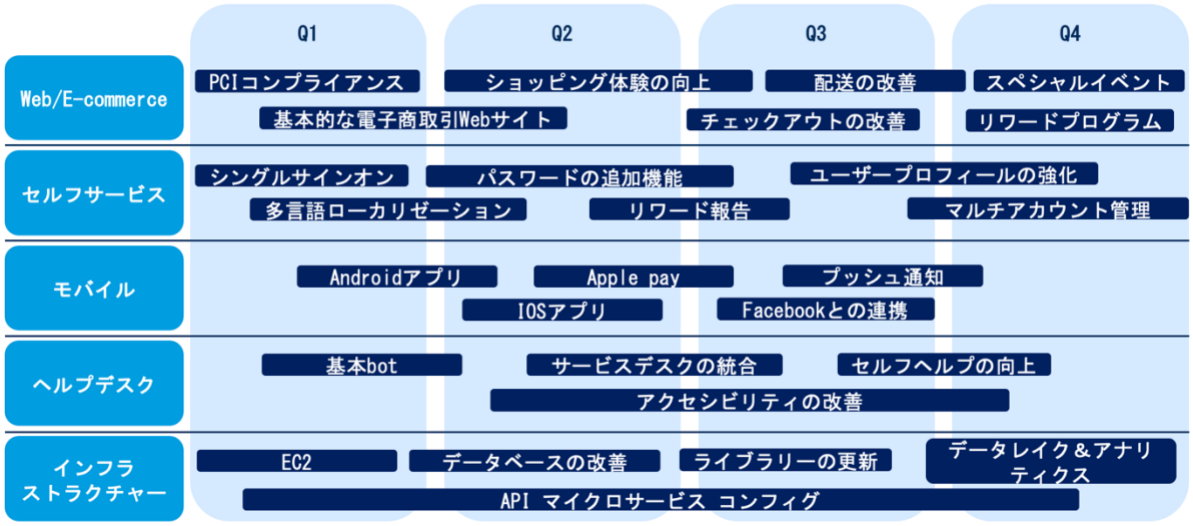
しかし、ほとんどの場合、そのような単純なものではありません。

ロードマップを作成するためには、ハイレベルな要求であってもテーマにまとめ、それをガイドラインとしてロードマップを作成することが必要になります。一般的にテーマとは、プロダクトやプロダクトカテゴリー、ビジネス用途（サポートされるプロセスを含む）、機能に関連する要求のグループと言えます。

しかし、実際には、複雑なプロジェクトのロードマップを作成するのに、特にプロダクト、プロダクトカテゴリー、ビジネス用途を使ってテーマを作成する場合には、テーマは完璧な方法ではありません。関連する機能性をグループにまとめ、それに基づいてテーマを作成する方が簡単で効果的です。

しかし、ロードマップの目的は、幅広いステークホルダーにどのようにして価値を生み出すかのアイデアを提供することであるため、これも必ずしも理想的ではありません。多くのステークホルダーは、テーマに基づいたロードマップがあまりにも技術的なものであれば理解できないだろうし、自分たちがやっていることややりたいことに関連するものでなければ意味をなしません。

*図 9 プロダクトロードマップの一例*




図は EXIN 制作：Botha、J.（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

テーマを設定する際には、なぜそのテーマを設定するのか、また、テーマを設定することでどのような効果が得られるのか、あるいは妨げになるのかを明確にする必要があります。常に自問自答してください。ユーザーや顧客は、テーマの意味を理解してくれるだろうか？ユーザーや顧客は、どのフィーチャーがどのテーマに属しているか理解できるだろうか？

プロダクトロードマップをどのように作成すればよいか、単純な答えはありません。その理由をいくつか探ってみましょう。

プロダクトロードマップは、プロダクトのビジョン、方向性、提供順序を共有するために使用されます。従来のプロジェクト計画や関連するガントチャートのように、約束されたタイムラインではありません。

その意図は、何が起きているかをステークホルダーに伝え、最新の情報を提供し、安心感を与えることにあります。

プロダクトロードマップは流動的であり、ビジネスに不可欠なものが進化したり、スクラムチームがアイテムに着手して依存関係を確認したりすることで変化していきます。

プロダクトロードマップは、プロジェクト内の有用な資料であり、ビジュアルコミュニケーションツールとしてもよく使われます。ここでは、前述の内容に加えて、チームはプロダクトの進捗を時間の経過とともに確認したいと考えます。このような使い方をすれば、アクションプランや進捗状況を報告する手段にもなります。

同じロードマップを見方の違う両方の人のために使わないでください。チームのためのロードマップを他のステークホルダー、特にマネジメントのロードマップとして使うと、顧客へのコミットメントに変えてしまうことになります。3 ヶ月後にはロードマップが全く違うものになっているかもしれないので、これは良くない考えです。

スクラムチームがパフォーマンス指標として使用するディメンションは、一般的なステークホルダー向けの進捗状況やパフォーマンス表示として使用してはならないことに注意してください。

## 6.14 バックログアイテムの順序付けの基準は?

Steven Covey（スティーブン・コヴィー）氏は、「The Seven Habits of Highly Effective People」の中で、優先順位は緊急性と重要性であると述べていますが、多くの人がこの2つの用語を扱う方法とは異なり、優先順位は以下のように割り当てられるべきです。

1. 重要かつ緊急なもの
2. 重要なもの
3. 緊急なもの
4. 重要ではなく、かつ緊急ではないもの

しかし、これらの言葉にはどのような意味があるのでしょうか?

- **重要と**は、それを迅速に行わなければビジネスの価値が大きく損なわれる、または重大な結果が生じることを意味します。
- **緊急**とは、それを成し遂げることに強い関心を持つ人がいることを意味します。たとえば、営業担当者は、機能が次のリリースに含まれることを約束したとします。

挙げられた2つの基準は、価値とリスクであると言うことができます。したがって、価値とリスクは、重要性とある程度の緊急性の2つの主要な決定要因です。

もちろん、顧客との約束通りにデリバリーすることは非常に重要です。しかし、先に何かをすることで組織が損失を被るのであれば、どちらを先にするかを決めるのは当然のことでしょう。短期的な約束よりも、価値やリスクの方が重要です。守るのが難しいかもしれない“取るに足らない” 目先の約束をするよりも、ステークホルダーと協力してやるべきことの価値を決める方が良いでしょう。

重要だと思われるアイテムが多すぎると、事実上、重要なものは何もないということになるため、重大なビジネスリスクとなります。この例えを使って、プロダクトバックログのアイテムの重要性と順序を決定する方法をステークホルダーに理解してもらいましょう。

もう一つの方法は、**パレートの法則**：プロダクトバックログの 20%だけのアイテムが重要であるとする法則、を使って考えることです。ステークホルダーに相談し、どのアイテムが20%になるかを決めます。

最終的にはプロダクトオーナーがバックログのアイテムの順序を決めますが、バックログのアイテムの順序を理解して決めるためには、幅広いステークホルダーを巻き込むのがベストです。

合意された基準に基づいて割り当てられたアイテムの相対的な重要性から、バックログ内のアイテムを順序付けます。次に行うべきことは、プロダクトバックログの一番上に置くことです。

また、上位レベルにあるアイテムが適切に分解され、アイテムに関連する正確な労力の見積もりを含む細かい要求になっていることを確認します。

これらは、次のスプリントのスプリントバックログに入る可能性の高いアイテムであることを忘れないでください。

ストーリーの分解作業中に、細かいストーリーの中に明らかに重要なものとそうでないものがあることに気づくかもしれません。次の数回のスプリントで行うべきアイテムだけがプロダクトバックログの一番上に残るように、プロダクトバックログを並び替えます。それ以外のアイテムは、細かいストーリーであってもバックログの下の方に配置してもよいし、ステークホルダーがバックログから完全に削除することを決めても良いです。

また、バックログの中に、殆ど、または完全に同じ成果を生むような類似した要求が他にあるかどうかの検討にも役立ちます。このような場合、バックログから重複したアイテムを削除する動機付けが容易になります。このようにしてアイテムを削除する場合、依存関係と要求が確実に対応付けされ理解されるようにして下さい。

重要度の低いアイテムの順序を決めるのに時間をかけすぎてはいけません。今時点で重要であることがはっきりするまでは、間違いなく重要ではないからです。

必須アイテムはよく理解され、文書化されるべきです。必要であれば、バックログアイテムをサポートするために別の文書を使用します。多くの場合、これはステークホルダーに対して質問を重ねたり、エピックの分解にステークホルダーを参加させたりすることで行われます。

非常に細かいアイテムの中に順序付けが難しいものがある場合、それらを理解しやすい一つのストーリーやアイテムにまとめ、バックログの中での位置を割り当てることにしてもよいでしょう。このプラクティスは、依存関係を理解するのが簡単なので役立ちます。スクラムチームは、スプリントプランニング時に再度分解することができます。

プロダクトバックログがリファインメントされるたびに、再優先順位付けが行われなければなりません。これは、バックログをクリーンで適切な状態に保つための最も簡単な方法です。

また、イテレーションの最初の結果が使用可能になった直後に、「新しい」要求が入ってくることに気づくでしょう。それらは新しいものだと思い込まないでください。それらの多くは、すでにエピックや上位のストーリーの一部として存在しています。

需要が高く、これらのアイテムがより重要になった場合、バックログアイテムを再順序付けする必要があります。

非機能要求は例外で、依存度が高く、プロダクトバックログの上位に位置する必要があります。識別されたら、すぐに分解し、最初から細かい要求としてリストアップする必要があります。また、リスクは順序の良い指標となります。

リスクや不確実性にはできるだけ早く対処したほうが、後で余計な作業をしなくてすみます。リスクや不確実性が高いアイテムは高いものとして分類し、できるだけ上の方に移動させます。

これは、仮定を妥当性確認するための初期の実験であり、本来は不確実でリスクを伴うものです。そうすることで、スクラムの原則のひとつである「経験的アプローチの使用」に準拠することになります。

未処理のリスクは、スクラムにおいてチームの首にかかる大きな石臼であり、後々まで足を引っ張ることになります。すでに行われたものが、最終的に持続不可能、真実ではない、使えないと証明されるかもしれないアイテムに依存していれば、関連するすべての努力が無駄になるということです。

バックログのアイテムをリストアップする際には、依存関係のあるアイテムや、ユーザーが使える「完成」を実現するために必要なアイテムをすでにグループ化しておくこともできます。このアプローチを具体的に使ったスクラムスケーリング手法（Nexus）を後ほど取り上げます。

ただ、関連するストーリーのすべてがスプリントのアウトプットに必要だと思い込まないように注意してください。多くの場合、スプリントはいくつかのコア要素を提供し、1つまたは2つのフィーチャーを除外しても、完全に使用可能であり、ビジネス価値を提供することができます。

スプリントプランニングの話をしているときに、Must、Should、Could のフィーチャーを選択するようにアドバイスされたことを覚えているかもしれません。Should と Could のフィーチャーは、完成の定義（DoD）の必要な部分ではないかもしれないので、スプリントの中で「バッファ」を作るために使われます。

スプリントが失敗することはもちろん望ましくないので、このアプローチはスクラムを導入する際によく使われます。これにより、組織内で失敗に対処するために使われてきた伝統的な手段（懲罰的行動）を公的に対処しなければならない状況に陥る可能性があります。

ステークホルダーのために早期に価値を引き出すことが目的であることを忘れないでください。できる限り早く真の価値を提供することを常に心がけてください。

## 6.15 ステークホルダーとのコミュニケーション

プロジェクトのような環境では、何が価値あるものと見なされ、何が価値を生み出すのかを知ることが最大の課題の一つです。

残念ながら、価値とは美しさと同じで、見る人の目に左右されるものです。ある人にとっては価値があっても、別の人にとっては価値があるとは限らないのです。

また、ステークホルダーが自分にとって価値のあるものを説明することはよくありますが、さらに調べてみると、すべての事柄に同等の価値があるわけではないことがわかります。ここでは、MoSCoW 技法が会話を誘導し、ステークホルダーが、絶対に譲れないもの（Must Have）、重要で効率や生産性を大幅に向上させるもの（Should Have）、物事をより良く、より簡単にするためにあると便利なもの（Could Have）、将来的に検討すべきだが現在は重要ではないもの（Wouldn't Have Now）を区別するのに役立ちます。

新しいプロダクトやサービスを設計・構築する際に、時間は非常に重要な議論の対象となります。プロダクトやサービスの中核を成すフィーチャーもあれば、それを可能にしたり強化したりするフィーチャーもあるからです。その中でも中核を最初に構築すべきなのは明白であり、スクラムチームは次に何が最も価値のあることなのかに集中すべきです。

アジャイルスクラムを使って新しいプロダクトやサービスを構築することは、エリック・リースが提唱したリーンスタートアップのアプローチと非常によく似ています。それは反復的なプロセスであり、提供されるプロダクトやサービスの中核についてステークホルダーがどのように考えているかを素早く知る必要があるからです。

スクラムチームは、**MVP（Minimum Viable Product）**を定義し、その MVP はステークホルダーが利用可能なものであり、かつ彼らの価値を引き出すものであるかどうかをステークホルダーと一緒に妥当性確認することができます。そうであれば、MVP を構成するすべてのストーリーを細かくして、プロダクトバックログの一番上に置くべきです。そうすると、バックログの一番上にあるものはすべて Must Have のアイテムであることに気づくかもしれません。しかし、もしそうだとしたら、MoSCoW はどのように役立つのでしょうか？

スクラムチームがプロダクトバックログをリファインメントする際には、Just-as-well の原則を使い、Must Have の作業が Should や Could であっても、簡単かつ効果的に行える作業を探します。

*警告:*非現実的な期待をしてしまうので、他のステークホルダーとフィーチャーについて話し合うときには、この原則を使わないことをお勧めします。スプリント(または Nexus)チームでのみ使用してください。また、MoSCoW 技法を使いたい場合にのみ使用してください。

ハイレベルなユーザーストーリー(エピックと呼ばれる)を分解していくと、自動的に Must、Should、Could のカテゴリーに分類されるフィーチャーの集合体があることに気づくでしょう。

多くの場合、MVP はテストして使用することができますが、これらを市場で販売することは非常に困難です。なぜなら、顧客が利用できるほどの十分な完成度がまだないからです。商業的に提供されるプロダクトやサービスの開発には、さらに多くのことが必要です。

その代わりに、**MMP(Minimum Marketable Products)**を開発することになりました。MMP のアイデアは、「Less is more」というリーンの原則に基づいています。MMP には、初期ユーザーのニーズを満たす最小の機能セットのみが含まれており、彼らはそのためにお金を払うことをいとわないでしょう。MMP は、市場投入までの時間を短縮するために、プロダクトやサービスのフィーチャーを削ぎ落としたバージョンを提供し、時間をかけてフィーチャーを追加していくことで、より完成度の高い、ユーザーにとってより価値のある(フィーチャーが豊富なことが価値提供とは限らない)プロダクトにしていくという方法です。

## 6.16 プロダクトゴールの設定

プロダクトゴールは、プロダクトがサポートするビジネス目標の要約であり、すべてのステークホルダーの要求を評価し、プロダクトバックログに並べたものです。プロダクトの全体像、顧客やその他の主要なステークホルダーの要求、そして組織の目標をどのようにサポートするかを把握して初めて、効果的なプロダクトゴールを定義することができます。

プロダクトバックログに対して実行しているすべてのスクラムチームにとって、プロダクトゴールは真北を表すものであることを忘れてはなりません。すべてのスプリントゴールは、プロダクトゴールを実現するための旅の足がかりとなるものでなければなりません。

プロダクトゴールは、要求の上位に位置するため、勝手に出てくるものではなく、具体的に策定する必要があります。

*図 10 プロダクトゴールとは?*




画像は EXIN が作成したものを元にしています。Botha、J.(2021). *[コースウェア]*. GetITright.

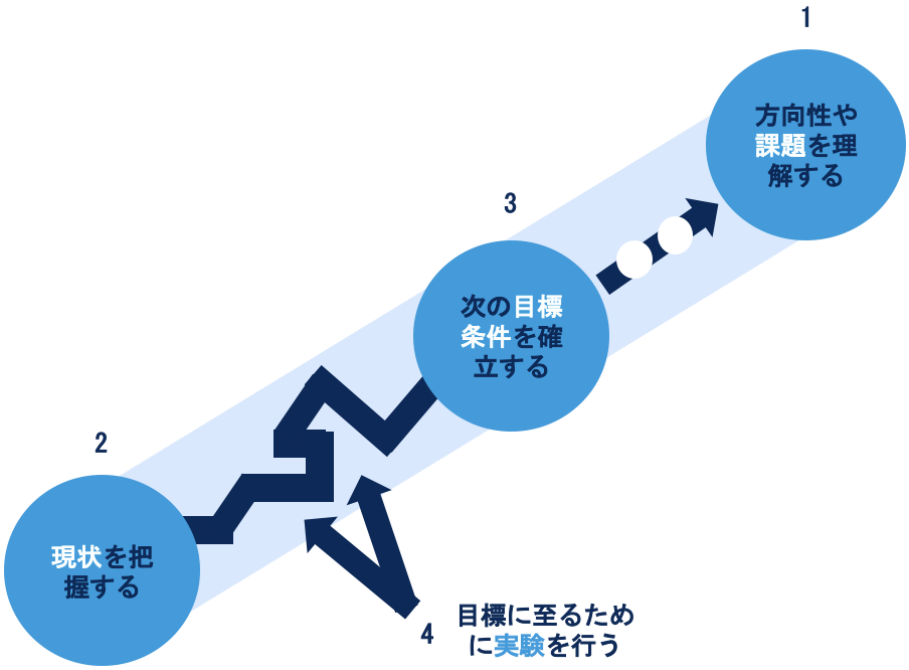
特に最初の段階で、プロダクトゴールだけでなく、プロダクトバックログを明確にし、洗練させるために、**トヨタ改善のカタ**のアプローチが有効な手法の一つです。

トヨタ改善のカタは、アジャイルと同様に、問題、ゴール、目標を理解するための科学的、経験的なアプローチです。伝統的には、ゴールや目標を明確にするためには使われません。しかし、ゴールや目標が正しく定義されていないことは、問題であるとも言えます。

トヨタの改善のカタは、一般的にこの 4 つのステップに沿って行われます。

1) **方向性や課題を掴む**：方向性を理解し、より大きな、そして長期的なビジョン、それが何であるかを明確にする
2) **現在の状況を把握する**：現在の状況を確認し、事実に基づいて明確に定義する
3) **次の目標を明確にする**：現在の知識と能力の限界を伸ばし、課題に向かって前進し、ある期日までに達成できる、次の目標を決定する
4) **実験を行う**：次の目標状態に到達するために、方法的、科学的に実験する

*図 11 トヨタ改善のカタのステップ*




図は EXIN 制作：Rother、M.（2018）. *トヨタ・カタの実践ガイド。1 日 20 分で優れた結果を出すための科学的思考スキルの実践。*マグロウヒル・エデュケーション。

プロダクトゴールを定義するのに、ストーリーなどのおなじみの手法を使うことができます。しかし、ステークホルダーからの十分なインプットがあれば、それに越したことはありません。まず、組織の目標を理解する必要があります。また、長期的かつ主要な組織の目標を推進するためには、組織の原則と目的を理解する必要があります。

こんな風に始めてみてはいかがでしょうか。

- **上位の組織目標を定義する**：<x 年後>に<あなたの組織>は<あなたが目指すもの>になる/達成する/持っている
- **目標を達成するための計画をする**：私たちは、<いつまでに><あなたがやるべきこと、または成し遂げるべきことを、測定可能な形で定義する>が必要
- そして、**現在の状況を評価し**、埋めなければならないギャップを特定する。ギャップを埋めるには、次のように段階を踏んでいくとよい
    - "私たちは<いつまでに><何を>する"
    - より高い目標が不要であれば、設定された組織の目標全てに同じアプローチを適用する
- **プロダクトバックログを能動的に管理することは、“カタ”で示されている実験をしていることになる**

多くの場合、物事を行う方法や順序は、ステークホルダーの要求や優先事項に影響されます。目標とする状態に到達するためのハイレベルなステップを定義する際には、ステークホルダーのニーズを考慮する必要があります。

スクラムの原理、原則に忠実な人は、トヨタの「改善のカタ」はこのような使い方をしてはいけないと言うでしょう。しかし、これは効果的に機能することが証明されています。トヨタの改善のカタのアプローチには、多くのリソースやガイダンスが用意されており、優れたプロダクトゴールを定義するのに適しています。

アジャイルと同じように、トヨタ改善のカタでは、コミュニケーション、チームワーク、コラボレーションを特に重視しています。

## 6.17 要求事項の収集

アジャイルで要求を収集する主な方法は、ユーザーストーリーを記録することです。

ユーザーストーリー形式は、要求の期待されるアウトプットと成果を 1 つのステートメントで定義することにより、粗い粒度の要求と細かい粒度の要求を集めることができます。

そのため、ストーリーは、プロダクトとプロダクトバックログが最初に定義されたときに集められることが多く、この段階では、ストーリーの粒度が粗く、規模が大きくなります。それらをエピックと呼びます。

要求を満たすための作業が進むにつれ、ユーザーストーリーはより細かいユーザーストーリーに分解されていきます。これらの細かなユーザーストーリーは、タスクに分解され、インクリメントとして簡単に作業できるようになります。

ユーザーストーリーを定義する際には、INVEST[7]メソッドを使用します。

- **Independent（独立している）：** 各ユーザーストーリーは、他のストーリーから独立していなければならない
- **Negotiable（交渉可能である）：** ストーリーは契約ではなく、交渉と変化の機会である
- **Valuable（価値がある）：** すべてのストーリーは、ユーザーやステークホルダーにとって価値のあるものでなければならない
- **Estimable（見積もり）：** すべてのストーリーの時間とコストは、ドメインや技術的な知識に基づいて計算可能でなければならない
- **Small（or simple）（小さい）：** ユーザーストーリーは、見積もりや実装が簡単にできる程度の小さなものでなければならない
- **Testable（テスト可能である）：** テストが可能なユーザーストーリーでなければならない

このような要求収集の方法や絶え間ないリファインメントは、人々には少々混沌とした印象を与えることがあります。このような場合には、ユーザーストーリーを他の情報や文書で補うことが必要になります。

追加情報は、技術的なものか、非機能要求に関連するものが多く、コンプライアンスやプロセスの標準が含まれることもあります。実際、すべての技術的および非機能要求は、可能な限りストーリー形式で定義することが推奨されています。

要求は常にリファインメントされていくものなので、定期的にステークホルダーと会話をしたり、リファインメント活動に参加したりすることが重要です。

---

[7] Bigelow, S. J.（2020）. *7 techniques for better Agile requirements gathering.* https://searchsoftwarequality.techtarget.com/tip/7-techniques-for-better-Agile-requirements-gathering

また、要求が明確でない場合には、他のステークホルダー、特にユーザーを巻き込む必要があります。ステークホルダーは要求のエキスパートであり、アイデアを共有したり、提案をしたり、さらには各イテレーションで要求を文書化するのを手伝うこともできます。

プロダクトバックログを常にリファインメントし、時間の経過とともに要求を更新していきます。要求は固定的なものではなく、時間の経過とともに変化していくものです。プロダクトバックログのリファインに顧客やユーザーを随時参加させることで、新しい要求や、変化するビジネスニーズや優先事項を特定することができます。

開発者がスプリントでインクリメントを構築する作業を始めると、これまで知られていなかった情報や依存関係を発見することがよくあります。このような情報は、プロダクトバックログとスプリントバックログの両方を改善するために使用する必要があります。

スプリントバックログのリファインは、現在のスプリントでの作業を再計画することを意味しますが、新たに発見された問題が現在のスプリントでは対処できないことが多く、そのような場合にはプロダクトバックログのリファインを行わなければなりません。プロダクトバックログのリファインメントは継続的な活動であり、イベントではないことを忘れないでください。

# 7 アジャイル計画と見積もり

この章では、見積もりと計画に対するいくつかのアプローチについて説明しますが、大まかに言えば、スクラムでは2種類の見積もりと計画技法が使われています。ベロシティベースとコミットメントベースです。どちらを選択するのかが非常に重要です。

ベロシティベースの見積りとスプリントプランニングを行う場合、前のスプリントで完了した作業（ストーリーポイントや T シャツのサイズなど）と同等の作業をスプリントに割り当てようとします。

ベロシティベースのスプリントプランニングを使用する際の典型的な手順は次の通りです。

- **チームのベロシティを計算します**。特にスクラムを始めたばかりの頃は、前回行った時から変わっているかもしれません。
- プロダクトゴールに基づいて、**スプリントゴールを定義します**。
- チームのベロシティに応じてプロダクトバックログの上位から**ストーリーを選択します**。
- **ストーリーをタスクに分割し**、それぞれにかかる時間を決めます。

コミットメントベースのスプリントプランニングを行う場合、チームはアイテムをスプリントバックログに追加する前にアイテムにコミットする必要があります。スプリントバックログ内のアイテムの合計がタイムボックスの時間と等しくなると、チームはバックログにそれ以上のストーリーを追加するのを止めます。

なぜなら、このチームは自然に作業を所要時間に変換するためで、チームは理想的な日数または時間の見積り手法を使用しています。

スプリントプランニングセッションは通常、このように実行されます。

- プロダクトゴールに基づいてス**プリントゴールを定義する**
- プロダクトバックログの先頭から**プロダクトバックログアイテムを選択します**
- **ストーリーをタスクに分割し**、完了するまでにかかる時間を決定します
- ストーリーを完成させることが可能であるという**チームのコミットメントを取得し**、ストーリーをスプリントバックログに追加します
- スプリントタイムボックスに残っている時間を確認する
- スプリントタイムボックスがいっぱいであるためにスプリントバックログにアイテムを追加できなくなるまで、このプロセスを**繰り返します**。

前述したように、様々なタイプの見積りを行うために、どのようなアプローチを取るべきか、どのツールが最適かは、非常に意見の分かれるところです。ここでは主な技法を説明しますが、どれが最適かはチームの判断に委ねられます。

アジャイルは前もって詳細な計画を立てるものではないと言っているのに、なぜ計画を立てるのでしょうか？

アジャイルサイクルの最初に計画を立てる目的はただ一つ、ポートフォリオに追加するプロダクトをきちんと理解し、ポートフォリオマネジメントやプロダクトマネジメントを効果的に行うためです。

計画に必要なレベルは、今後半年から 1 年の間に適切な決定を下すことができるレベルで、少なくとも、更にその先に起こりうることと、そのために必要なリソースについて見当をつけることです。企業ポートフォリオで示されたゴールと目標を達成するために、資金とリソースを割り当てることができるようにしたいものです。

つまり、事前計画の答えは、ビジネスの優先順位を決定し、ビジネスに最も大きなプラスの影響を与える取り組みに資金やその他のリソースを割り当てることができる「ちょうど良い」ものであるべきだということです。

初期計画には常に欠陥があります。ウォーターフォール型のプロジェクトではそのことがよく知られています。しかし、初期計画は洞察をもたらし、議論や発見、ビジネス全体の戦術的計画のきっかけとなります。

アジャイルアプローチ、特にスクラムが採用されない理由の1つは、組織が適切な戦略的・戦術的意思決定を行うためのハイレベルな視点を持っていないことです。アジャイルとスクラムを純粋に見た場合、この反論は部分的には正しいです。しかし、最近のすべてのアジャイル手法は、これらの初期の反論を克服し、修正する方法を持っています。

本書の後半でスケーリングを取り上げる際には、スクラムのアプローチを補完するものとして考えられている現在の手法のいくつかについて言及します。

しかし、すべてのアジャイル手法が採用しているアプローチは、どのレベルにおいても、計画や実行のそのレベルでの質問に答えられるように、十分な計画を行うことです。

ですから、最初の計画や見積りが変化する可能性があることを、前もって合意しておきましょう。これは、ウォーターフォール型のプロジェクトにも、アジャイル型のプロジェクトにも言えることです。最初の計画は、地に足をつけるためのものであり、ベースラインであり、顧客への真の価値に近づくほど、より多くのことを学び、より正確になるでしょう。

Bruce Martin 氏は、PMI の Project Management Quarterly に掲載された1980 年代の記事「*The goal: to improve credibility in the reporting of engineering progress*」の中で、今日でも有効なチャートを描いています。この記事では、アジャイルやスクラムについてではなく、ウォーターフォールのプロジェクトについて書かれていました。スクラムに対する反論のほとんどの根拠は欠陥があり、非常に誤った前提に基づいていることが改めて証明されました。

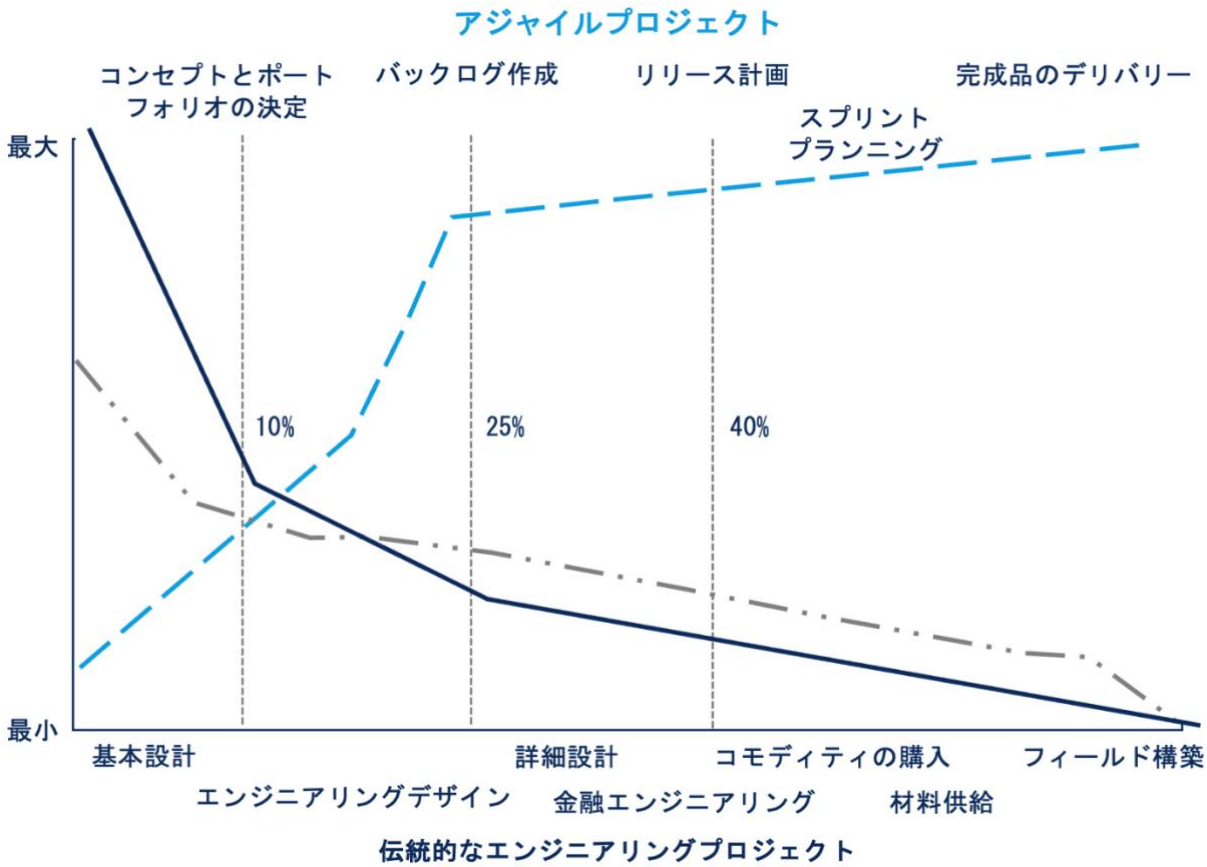
その前提として、プロジェクト開始時にはハイレベルなアジャイル計画よりも詳細なウォーターフォール計画の方が良いとされています（次の図を参照）。

では、なぜ悩むのでしょうか？ハイレベルな見積りを頻繁に行い、ポートフォリオを改善すればするほど、時間の経過とともにその能力が向上していきます。これは、アジャイルの見積り手法を使うメリットの1つです。新しいプロジェクトや成果物を、既知の古いプロジェクトと比較することが多いからです。やればやるほど、学ぶことが多くなり、上達していくのです。

詳細な計画の信頼性は、プロセスの初期段階では疑問視されますが、作業の実施に近づけば近づくほど、計画と見積りの信頼性は高まり、コストとスケジュールの管理の必要性は低くなります。

学習と改善のプロセスは、プロダクトバックログとロードマップの作成と維持が、時間の経過とともにより正確になることを意味しています。その結果、組織が提示する未来像への信頼度が高まります。

*図 12 詳細計画の信頼性*




画像は EXIN が作成：Martin、B. A.（1980）. *目標：エンジニアリングの進捗状況の報告における信頼性を向上させること。*Project Management Quarterly, 11（2）, 14-22.

このように、見積りを含めた全プロジェクトのハイレベルなプロダクトロードマップは、ポートフォリオ管理のインプットとなります。これにより組織にとって不可欠なものに注力でき、戦略的なイニシアチブとそれを実現するためのプロジェクトを関連付けることができるようになります。

それは、「何を、いつ、どのように作るべきか」「それは重要なのか」という疑問に答えるものです。

計画は、マーケティング、プロダクトブランディング、セールス、広告などの直接的なサブプロジェクトにも役立ちますし、プロジェクト デリバリーに備えてトレーニングやスキルアップの取り組みを入れておくことはよくあります。

良い計画は、その計画を使用したいレベルで、十分に詳細な計画をすることです。つまり、アジャイル計画の反復計画とは、その計画を実行するのが差し迫っている計画期間を示しています。

## 7.1 アジャイルプランニングは何が違うのか？

実務者は、伝統的な計画の立て方をしないように注意する必要があります。従来のプロジェクト計画は、直線的かつアクティビティベースのものでした。

アジャイルでは、価値の提供は直線的に行われるのではなく、どのフィーチャーが顧客にとって最も価値のあるものかを常に再優先して行います。

直線的な計画と実行の欠点に加えて、クリティカルパスに影響を与える遅延が、パフォーマンスの停止を引き起こします。そのため、ウォーターフォールアプローチは、リリースまでの完璧な計画があることを前提としていますが、これは有効ではありません。

つまり、ウォーターフォール型のプロジェクトでは、クリティカルパス上の何かが遅れることは、顧客に価値を提供するプロジェクトが遅れることを意味するのです。

もう1つの問題は、パーキンソンの法則によれば、プロジェクトチームは遅れによって生じる「空き時間」や「待ち時間」を埋めてしまうということです。このような現象が起こるのは、この種のプロジェクトでは、少なくとも何かをしているということが第一の評価基準になっているからです。下流のアイドルタイムは、依存しないタスクには利用されません。なぜなら、それらはリストの次にやるべきことではないからです。その結果、遅延の影響はほとんどの場合、下流に伝わってしまいます。

アジャイルチームが大切にしている 4 つのことを思い出してください。

| | | |
|---|---|---|
| 価値1. | **個人の対話** | *プロセスとツール*よりも |
| 価値2. | **動くソフトウェア** | *包括的なドキュメント*よりも |
| 価値3. | **顧客との協調** | *契約交渉*よりも |
| 価値4. | **変化への対応** | *計画に沿って*よりも |

これらの価値観は、計画の立て方を含め、すべての行動の一部であるべきです。

アジャイルでは、作業を行う直前に計画が行われることに注意してください。最初の計画とプロダクトバックログの作成から多くのことを学んだであろうことを考えると、この時期に計画を立てるのがベストです。アジャイルやスクラムの反復的な性質は、教訓を得続けることにより、更に良い計画を考えられることを可能にします。

前述のプランニング期間でも強調されていたように、さまざまなレベルのさまざまなチームで、チームとして作業します。プランニングのあらゆるレベルで意味があるように、幅広いステークホルダーを含めます。そして関係者の知恵と理解を最もよく表すものを、会話し、質問し、尋ねて考え出します。

わからないことは定義しようとせず、後回しにしましょう。計画の反復は、作業の実施に近づくにつれて、より詳細なものになっていきます。

計画は軽くすべきです。適切な判断を下すことができるように、「Just enough（本当に必要な最低限のことだけ）」というリーンの原則を用います。多すぎなく、完璧でもなく、包括的でもない、本当に必要な最低限のことだけです。

計画を立てるときには、参加型で視覚的な手法を使うことをお勧めします。もし、壁に付箋で覆われていなかったり、ボードに落書きがいっぱいされていたら、それは努力が足りないということかもしれません。

それに、そのプランニングが次のレベルの参加者にきちんと引き継がれるかどうかも確認してください。何かを壁の向こう側に放り投げて上手くいくことを期待してはいけません。次の活動をしなければならない当事者が部屋にいない場合、彼らを部屋に入れ、説明し、議論し、討論し、コンセンサスや合意を得ます。

アジャイル計画は、フィーチャーが生み出す価値に基づいて、優先順位を決めます。アジャイルでは詳細な計画は前もって行われず、アクティビティはハイレベルなアジャイル計画にはまったく存在しません。

このアプローチには 4 つの大きな便益があります。

1. **アクティビティベースのプランニング**は**行わ**ないので、このアプローチの落とし穴を避けることができます。
2. 作業内容自体は、価値の提供を実行する専門家の手に委ねられています。彼らは、その作業を行うための最善の方法を決められる最も優れた能力を持っています。
3. アクティビティを気にする必要がないので、顧客や組織の価値を引き出すための最適な順序を考えるために、**プランニングの視点を高める**ことができます。
4. すべての活動を詳細に把握しているわけではないので、**未来について不完全な考えを持っていることを受け入れ**、旅の各ステップで、継続的に、やらなければならないことをよりよく理解し、不確実性を減らすように努めることができます。

また、上記の最後のポイントは、プロジェクトのポートフォリオやロードマップの見方は、コミットメントではなく、計画された進捗状況を示すものであり、組織や顧客の状況や優先順位が変化した場合には、すべてが急遽変更される可能性があることを、ステークホルダーに伝える必要があるということです。

このアプローチでは、顧客に不安を与えてしまうのではないかと思われるかもしれません。

正直なところ、答えは *「イエス」です*。でもそれは、*「最初はイエス」*です。

なぜ?

なぜなら、顧客は価値を提供するサイクルを通じてプランニングに参加し続けなければならず、顧客を参加させ続けるのがプロダクトオーナーの仕事だからです。

スクラムチームがバックログのリファインメントやプロダクトバックログアイテムの再優先順位付けに関わることは前回述べました。また、プロダクトオーナーはビジネスの代表として、バックログリファインメントを行う際に、ビジネスの優先順位についてビジネスの相手方と会話をしなければなりません。

スクラムチームのタスクを議論する際には、イテレーションやスプリントの作業を計画する必要があると述べられています。また、プロダクトオーナーは、リリースやイテレーションを最適に計画・実行するために、プロダクトバックログ（プロダクトレベル）の作業を計画・順序付けしなければならないと述べられています。顧客は、優先順位に変更があった場合、プロダクトオーナーに報告する必要があり、プロダクトオーナーは、顧客の優先順位に変更がないことを確認する必要があります。

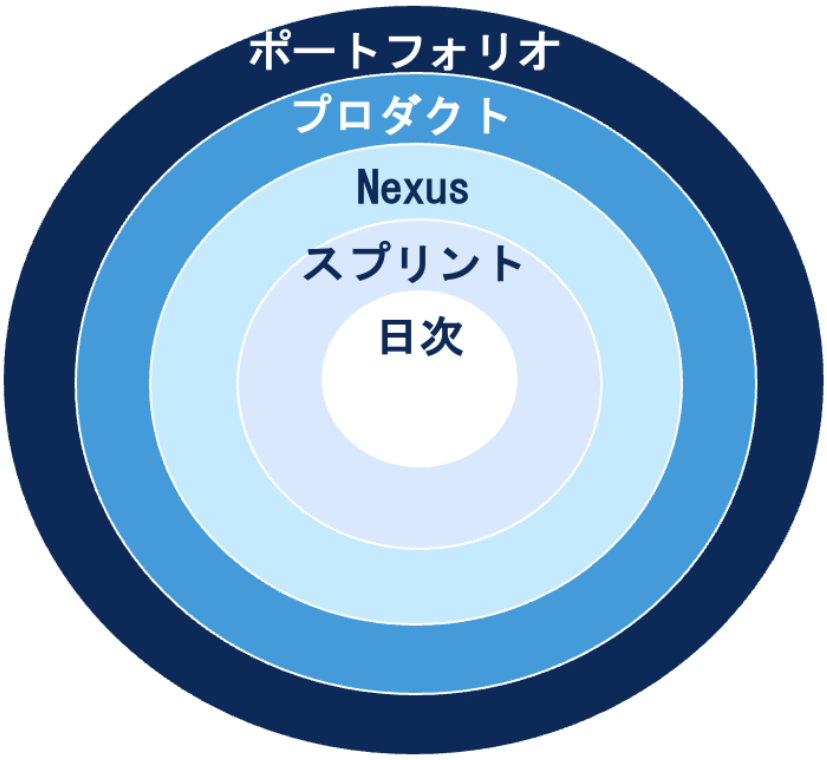
プロジェクトを組織のポートフォリオに組み込むには、そのプロジェクトを完了するために必要な労力、予算、リソースの量を見積もる必要があります。また、この情報は、組織内の他のイニシアチブとの関連で、そのイニシアチブに優先順位を割り当てるためにも使用されます。

これは、プロジェクトのライフサイクルを通じて、少なくとも 4 レベルの見積りと計画が必要であることを示しています。多くの組織や複雑なプロジェクトでは、さらに 4 レベル、5 レベル、あるいは 8 レベルの計画と見積りが必要になるかもしれません。重要なのは、質問に答えるために必要なレベルの計画と見積りを、各レベルで行うことです。

計画や見積りをリファインメントさせるレベルとは通常次の通りです。あるプロジェクトをポートフォリオに追加すべきかどうかを判断するため、バックログを作成するため、プロダクトバックログをリファインメントするため、スケールアップした実装やリリースを計画するため、スプリントを計画・実行するために、私たちはそれぞれのレベルで何を知る必要がありますか?

計画と見積りもスプリントの中で行われます。このスプリントでは何をするのか、今日は何をしますか?

*図 13 計画の層は、作業の実施に近づくにつれて、より詳細になります。アジャイルでは、すべての層で「ちょうどいい」計画を立てるという原則を採用しています。*




図は EXIN 制作：Botha、J.（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC – Kdp Print Us.

これまで、プロダクト、リリース、スプリントレベルでの計画と見積りについては説明してきましたが、ポートフォリオレベルでの計画と見積りについてはどうでしょうか？

この質問に答えるためには、組織のポートフォリオを作成・更新する際に必要な質問が何であるかを知る必要があります。

組織のポートフォリオの目的は、組織のゴールや目標の達成を管理することです。質問は本質的なものです。ゴールや目標を達成するためには、何（イニシアチブ）を実行する必要がありますか？そして、特定されたイニシアチブは評価され、順序付けされる必要があります。

- まず何をすべきか？
- 最も大きな影響を与えるのは何か？

残念ながら、これらの質問に答えるのはそれほど簡単ではありません。というのも、通常はリソースや予算などの制約があるからです。

そこで疑問となるのが

- どのような組み合わせであれば、最も大きなインパクトを与えることができるのか？

この質問に答え、このレベルの見積りと計画を行うためには、その取り組みによって生み出される価値、それを実現するためにかかる時間、コストと使用されるリソースについて、十分な理解を得る必要があります。次に、相互依存性についても考慮する必要があります。

この質問に対する答えは、せいぜい経験的な推測であることは理解していますが、過去のイニシアチブや既知の作業と比較すると、このレベルの見積りはすぐに 80%以上のレベルに達することができます。ポートフォリオプランニングでは、80%の見積りで十分なのです。

これらの質問に答えることは、行うべき作業の本質を扱うだけではないことに気付きましたか?関与するチームの速度も影響してきます。

最終的に誰が作業をするのか分からない場合は、チーム間の平均ベロシティに基づいて作業することをお勧めします。

## 7.2 誰がプランニングに参加するのか?

これは非常に良い質問で、どのレベルのプランニングを指すかによって答えが変わってきます。

ポートフォリオプランニングでは、プロダクトオーナー(マネージャー)、ビジネススポンサー(顧客)、シニアマネージャー、エグゼクティブマネージャーが議論に参加することになるでしょう。

プロダクトバックログの作成には、プロダクトオーナー、ビジネススポンサー(可能であれば)、そして最初は幅広いユーザーの代表者が参加する必要があります。関係ユーザーは、バックログに入るストーリーとその相対的な順序について十分に理解していなければなりません。

順序という言葉を選んだのは意図的なものです。プロダクトバックログでは、重要度や優先度だけで作業の順番を決めるわけではありません。プロダクトバックログでは、重要度や優先度だけではなく、リスクや依存性などの他の要素に対しても重要な役割を果たします。

プロダクトバックログのリファインにも同じ関係ユーザーを含めることができますが、実際には幅広いユーザーが必要とされることはほとんどありません。現在バックログに取り組んでいる開発者や、必要に応じてスクラムマスターも参加させることを忘れないでください。スクラムチームで得た教訓をインプットすることが不可欠です。

スケールドインプリメンテーションやスプリントプランニングでも、同様の関係ユーザーが適切です。特定のスプリントに向けて計画が進むにつれて、顧客やユーザーの関与は少なくなります。だからといって、まったく関与していないというわけではありません。スプリント全体を通してユーザーや顧客の関与が高いスクラムチームは、そうでないチームよりもはるかに優れています。ユーザーや顧客からのフィードバックを得るために、スプリントレビューを待つ必要はありません。

プロダクトオーナーは、複数の開発者チームやスクラムマスターを巻き込んで、プロダクトバックログの定義、リファインメント、プランニングの活動を行う必要があります。

プランニングは、実際のスプリントに近づけば近づくほど、徐々に技術的になっていきます。ここでは、異なるチームが同じバックログに対してどのように作業するか、依存関係とその結果、作業やアクティビティの順序付け、必要なコミュニケーション、そして最終的にはスプリントで誰が何をするのかを理解する必要があります。これが、大規模で複雑な環境では、まとまったスケールのあるアプローチが非常に重要になる理由です。

すべてのタイプのプランニングをタイムボックス化されたイベントとして捉えるのは良くありませんが、多くの場合タイムボックス化されたイベントとして実行されます。

スケールドインプリメンテーションとスプリントの両方とも、ゴール、目標、およびデリバリーが達成するための成功基準や受け入れ基準を持つべきですが、スケールドインプリメンテーションのレベルではもう少し流動的であるべきです。しかし、スプリントでは、明確に定義されたスプリントゴールと各インクリメントの完成の定義(DoD)があり、これにはさらに詳細な関連受け入れ基準が含まれる場合があります。

## 7.3 見積もり技法

プランニングを行う際には見積りが必要なのは明らかですが、この言葉の本質が重要なことを物語っています。

見積りは100%正確ではありません。どのレベルの見積りであれば十分なのかを知ることがコツです。

考慮すべき2つの要素は、**精度**と**労力**です。人生のほとんどのことがそうであるように、さまざまな試みをプロットすると、おそらく典型的なベルカーブ（正規分布）を示すでしょう。より良い見積りを得るための一つの方法は、より意味のあるデータを持つことであり、チームはやればやるほど自然にうまくなっていきます。

チームメンバーが積極的に参加して見積りの経験を積めば積むほど、チーム全体で良い見積りができるようになります。アジャイルの見積り技法はすべて、チームの共有されたインプットに依存していることに気づくでしょう。

あるストーリーの見積りが難しいとします。そのような場合、チームはそのストーリーを推定しやすい小さなストーリーに分割し、小さなストーリーの合計を大きなストーリーの推定値として割り当てることを検討します。この技法をディスアグリゲーションとアグリゲーション（分離と集約）といいます。

**しかし、チームがノーアイデアだったらどうでしょう？**
専門家や複数の専門家に依頼することもできますが、チーム外の専門家はチームのスキルや能力を把握していないため、これでは忠実度の高い見積りはできません。しかし、推測するよりは良いでしょう。

チームは、現在の作業を以前の作業と比較することもできます。理想的な日数やストーリーポイントに関する作業は、実際にはそれだけです。しかし、比較対象となるプロジェクトがない場合は、唯一実行可能な選択肢について外部の意見を聞くことになります。

### 7.3.1 再見積もり

ストーリーポイントと理想的な日数を使用する場合、一度作業が中断されると再見積もりが必要になることを意味しています。ストーリーポイントと理想的な日数を使った見積もりは、フィーチャーやストーリーの見積もりを簡単にするために使用され、それに関連するすべての複雑さを含みます。フィーチャーやストーリーのタスクのレベルまで詳細化を行う必要はありません。

つまり、スプリントプランニングにおいてタスクの分解が行われると、関連するタスクの合計をフィーチャーやストーリーに集約（アグリゲーション）することができるため、再見積もりが必要になることが明らかになります。

このレベルではストーリーやフィーチャーにはもはや関心がないので、これは不要だと主張する人もいるかもしれません。しかし、最初の見積もりとより詳細な見積もりの差分をチームが認識することができ、将来的に同じ性質の作業を見積もる際のチームの能力を大幅に向上させるため、アグリゲーションは有効です。

レトロスペクティブで話題になるのは、過去の見積もり、詳細な見積もり、実際の作業時間との間に大きな差があることが判明したケースです。

これは、スプリント終了直後にチームがプロダクトオーナーのプロダクトバックログリファインメントを手伝う可能性が高いことを考慮して、議題の1つとして特に推奨されています。

そのため、この項目を入れることで、今後の見積もりの向上にすぐに役立ちます。

また、部分的に完了したストーリーは、スプリント後すぐにプロダクトバックログに戻す必要があることを覚えておいてください。ストーリーの内容は変わりませんが、見積もりは変更される可能性があります。なぜな

ら、一部の作業はすでに完了しているからです。ストーリーは、完全に完了するまでプロダクトバックログから削除されません。

スプリントプランニングのセクションでは、プロダクトバックログのアイテムを見積もる際にも使用できる見積技法を紹介します。

## 7.4 スプリントプランニングでは、他に何ができるのか?

念のため、スプリントプランニングの際には以下の質問に答える必要があります。

- このスプリントはなぜ価値があるのか?
- このスプリントで何ができるのか?
- 選択した仕事をどのように成し遂げるのか?

プランニング活動の最後には、以下のことが明確に定義されていなければなりません。

- スプリントゴール
- スプリントで選択したプロダクトバックログ項目とアイテム
- スプリントバックログアイテムを提供するための計画

先にスプリントプランニングの概要を説明しましたが、このセクションでは、スプリントプランニングの際に使用できる技法について少し説明します。

### 7.4.1 スプリントプランニング

スプリントプランニングは、スプリントの長さ1週間毎に1～2時間程度のタイムボックス型のイベントです。

スプリントプランニングでは、スクラムマスターは開発者にフィーチャーの見積もりについて指導することはあっても、見積もりを決定することはありません。一方で、スクラムチームは、組織をプロダクトゴールの達成に近づける一連のプロダクトバックログアイテムを完成させることに合意します。この合意を、スプリントゴールを定義しスプリントバックログにアイテムを追加することで反映します。

**どれだけ引き受けるかをどうやって知るのですか?**

最初のうちは、開発者が作業できる速度に関するデータが得られるまで、バックログアイテムの見積もり時間は試行錯誤にならざるを得ません。これはチームのベロシティと呼ばれるもので、後ほど詳しく説明します。ここでは、スプリントプランニングというタイムボックス型のイベントに焦点を当ててみましょう。

スプリントの準備は、スプリントプランニングの前に始まります。プロダクトオーナーは、スプリントプランニングセッションで生産的な議論ができるように、プロダクトバックログが適切に精査され、分解され、必須アイテムの順序とサイズが可能な限り決められていることを確認しなければなりません。

スプリントプランニングには、スクラムチーム全員が参加します。その目的は、プロダクトオーナーとチームの他のメンバーとの間で、どの作業をスプリントに含めるかについて合意を得ることです。

スクラムを実践する人たちは、スプリントとインクリメントという言葉を同じものとして扱うことが多いですが、スクラムガイドによると、ひとつのスプリントには複数のインクリメントを含めることができます。インクリメントは、プロダクトゴールに向けた具体的な踏み石と定義されています。

それぞれの**インクリメント**は、それ以前のすべてのインクリメントに追加され、徹底的に検証され、すべてのインクリメントが一緒に機能するようになっています。価値を提供するためには、インクリメントが使用可能でなければなりません。

スプリントでは、複数のインクリメントを作成可能です。

インクリメントをまとめたものをスプリントレビューで提示することで、経験主義がサポートされます。ただし、スプリント終了前にインクリメントをステークホルダーにデリバリーする可能性もあります。スプリントレビューについて、価値をデリバリーするための関門と見なすべきではありません。

完成の定義（DoD）を満たさない限り、作業をインクリメントの一部と見なすことはできません。

この定義によると、各インクリメントはそのインクリメントの「完成」を定義しなければならないが、スプリントゴールはむしろ包括的な達成事項です。

スプリントプランニングは、スプリントの理想的かつ大まかな成果について話し合うことから始まります。プロダクトオーナーは、その準備として、チームに対してプロダクトバックログの上位にあるアイテムの概要を伝えたり、プロダクトゴールを思い出させます。

これは、スプリントの最初の大まかな目標を定義するための背景となります。大まかなという言葉は意図的なもので、もしそれが厳密すぎると、チームはスプリント中に何をしなければならないか選択できなくなります。これは、どの作業をどのように行うかを決定する自律性を持った自己管理型チームというアジャイルの原則に反するものです。

この会話の結果、このスプリントのスプリントゴールが定義され、そしてそのスプリントゴールはプロダクトゴールをサポートするものになるはずです。

スプリントゴールは、特定のスプリントに設定された目標であり、チームはプロダクトバックログに記載されたアイテムを実装することで目標を達成することができます。スプリントゴールは、プロダクトオーナーと開発者の間で交渉された結果であり、具体的で測定可能なものでなければなりません。開発者は、目標が現実的なものであることを確認する必要があります。これは、開発者が成し遂げることを確約するものだからです。

スプリントゴールを決定するために、Roman Pichler 氏は考慮すべき3つの質問[8]を提案しています。

- **なぜスプリントを実行するのか?**なぜスプリントを実行する価値があるのか?何を達成すべきか?
- **どのようにしてそのゴールに到達するのか?**どのようなアーティファクト、妥当性確認技法、テストグループを使うのか?
- **目標が達成されたことをどうやって知ることができるのか?**例えば、5 人のユーザーのうち、少なくとも 3 人が 1 分以内にユーザービリティテストを成功させたらなど。

また、以下の質問に答えることも重要です。

- **このスプリントはなぜ価値があるのか?**プロダクトオーナーは、プロダクトの価値と有用性を今回のスプリントでどのように高めることができるかを提案する。次に、スクラムチーム全体が協力して、そのスプリントになぜ価値があるかをステークホルダーに伝えるスプリントゴールを定義する。スプリントゴールは、スプリントプランニングの終了までに確定する必要がある。

- **このスプリントで何ができるのか?**開発者は、プロダクトオーナーとの話し合いを通じて、プロダクトバックログからアイテムを選択し、今回のスプリントに含める。スクラムチームは、このプロセスの中でプロダクトバックログアイテムのリファインメントをする場合がある。それによって、チームの理解と自信が高まる。スプリント内でどれくらい完了できるかを選択するのは難しいかもしれない。しかしながら、開発者が過去の自分たちのパフォーマンス、今回のキャパシティ、および完成の定義（DoD）の理解を深めていけば、スプリントの予測に自信が持てるようになる

[8] Overeem, B.（2016）.スプリント目標を使う 11 のメリット」https://www.scrum.org/resources/blog/11-advantages-using-sprint-goal

- **選ばれた作業はどのようにして行われるのか?**開発者は、選択したプロダクトバックログアイテムごとに、完成の定義（DoD）を満たすインクリメントを作成するために必要な作業を計画する。これは多くの場合、プロダクトバックログアイテムを 1 日以内の小さな作業アイテムに分解することによって行われる。これをどのように行うかは、開発者だけの裁量とする。プロダクトバックログアイテムを価値のインクリメントに変換する方法は誰も教えてくれない。

この2組の質問を通じて、よく練られたスプリントゴールを作ることができます。それ以降のスクラムチームの行動はすべて、スプリントゴールにどのように貢献しているかで評価されます。

次のステップは、チームがスプリントゴールを達成するために必要な作業を、そのスプリントに定義されたタイムボックスの中で現実的に完了できるかどうかを評価することです。多くの場合、このスプリントに関する議論では、今回のスプリント以降のスプリント群で成し得る将来の作業内容にも光が当たり始めるでしょう。

つまり、プロダクトオーナーは、スプリントバックログのアイテムをプロダクトバックログ一覧の最上位部分に反映します。そして、次のスプリントで扱う可能性があるアイテムをプロダクトバックログ一覧のその次の部分となるように、プロダクトバックログのアイテムを並べ替える必要があるかもしれません。また、開発者の意見がプロダクトバックログに反映されているため、次のスプリントのアイテムを選択するために費やす時間を最小限に抑えることができます。

スプリントに含まれるプロダクトバックログアイテムの量は、チームのキャパシティとベロシティに依存します。スクラムを始めたばかりの頃は、チームのベロシティを計算するためのデータがないため、推測にならざるを得ません。

スプリントに含めるために選ばれたプロダクトバックログアイテムは、小さく、見積もり可能で、テスト可能なアイテムに分かれていなければなりません。チームはプロダクトバックログアイテムをスプリントバックログに追加する前に分解すべきです。

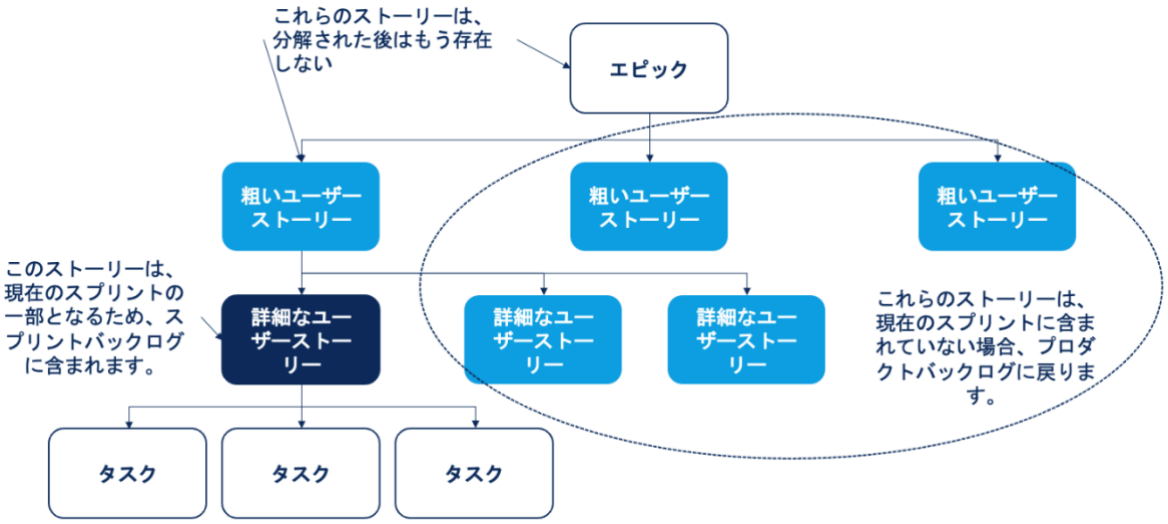
開発者のチームによって、スキル、能力、ベロシティのレベルが異なるため、プロダクトバックログアイテムはスプリントバックログに追加する前に再見積もりされます。

プロダクトバックログアイテムはできるだけ小さく、絶対に数日分以上にはならないようにしましょう。そうすることで、作業の内訳をシンプルで管理しやすいタスクにまとめることができます。また、仕掛（WIP）を制限し、バーンダウンやバーンアップチャートを使って進捗状況を把握しやすくなります。

ユーザーストーリーは複数回、複数のスプリントにわたって分解できることに注意してください。このスプリントで完成させる作業の一部でない限り、上位のストーリーのブランチすべてを、このスプリントで分解する必要はありません – このスプリントで作業されるものだけを分解します。

もしチームがこのスプリントで分解されたストーリーのブランチに取り組んでいない場合は、適切な優先順位と見積もりをつけてプロダクトバックログに戻すことができます。

*図 14 エピックからタスクへ – 要求の分解*




図は EXIN 制作：Botha、J.（2018）. *スクラムマスターとプロダクトオーナー[コースウェア]*. GetITright.

また、スプリント期間中のパフォーマンスに影響を与える可能性のある、プロジェクトの背景にある現在の課題を考慮します。依存関係に着目し、その影響を考慮します。

開発者以外の関係者に依存していることがよくあります。このような状況は絶対に避けたいものですが、実際には起こりえます。どうすれば依存関係を解消できるか、解消できない場合はどのように管理すればよいかを検討してください。

例えば、開発者が作業をするためにクラウド環境を必要としているような場合です。要求するのは簡単ですが、得られるかどうかは予測できません。スプリントのパフォーマンスに影響を与えないようにするために、何かできることはないでしょうか?もしそうなら、何をすべきで、どのように適切な手順を踏むことができるでしょうか?

チームがスプリントに含めるプロダクトバックログアイテムについて合意に達したら、チームは納品すべきプロダクトバックログアイテムをまとめたスプリントバックログを作成します。スプリントバックログは定期的に更新する必要があります。できるだけ最新の状態を保つために、1日1回の更新を行うことをお勧めします。

その後、チームはスプリントの一部を構成する各インクリメントの完成の定義（DoD）に合意しなければならず、そのインクリメントやスプリントで完了する必要のあるプロダクトバックログアイテムに取り組むことがチームのコミットメントとなります。

完成の定義（DoD）とは、インクリメントの成功に到達するために満たさなければならない要求のリストです。開発者のチームの中には、バックログアイテムをすべて完了させることを完成の定義（DoD）に含めているところもあれば、そうでないところもあります。

完成の定義（DoD）は、インクリメントの一部としてどのような作業を完了させるべきかという共通の理解を全員に与えるものです。完了したプロダクトバックログアイテムが完成の定義（DoD）を満たしていない場合、そのアイテムを使用可能にしたり、スプリントレビューでプレゼンテーションすることはできません。代わりに、将来のスプリントに含めるためにプロダクトバックログに戻さなければなりません。

インクリメントの完成の定義（DoD）に、組織の標準の一部が含まれている場合、すべてのスクラムチームは、標準的なセキュリティまたは品質要求として定義された非機能要求など、最低限それに従わなければなりません。要件が組織の標準ではない場合、スクラムチームはプロダクトとプロダクト要求に適した完成の定義（DoD）を作成しなければなりません。

開発者の成果物は、完成の定義（DoD）に準拠することが求められ、複数のスクラムチームが共同でインクリメントに取り組む場合は、相互に完成の定義（DoD）を定め、同じ定義に準拠しなければなりません。

プロダクトバックログアイテムが完成の定義（DoD）を満たした瞬間に、インクリメントが生まれます。

補足：MoSCoW 技法を使っているチームは、スプリントに含めるプロダクトバックログアイテムを Must、Should、Could Haves から選ぶことが多いです。通常、80%は Must Have（実行すべき重要性の高いアイテム）と Should Haves（便利で価値があるが必須ではないアイテム）、20%は Could Haves とします。このようなアプローチをとることで、何か不測の事態が起こったときに、スプリントの中でバッファを作ることができます。

しかし、チームがバッファアイテム（Should と Could Haves）を含むシナリオでは、通常、Must-Have ストーリーのみが完成の定義（DoD）に含まれます。

スクラムの標準的なプラクティスは、プロダクトバックログの上位にあるアイテムのみをスプリントバックログに含めるというものです。しかし、多くのスクラムチームは DSDM のプラクティスを採用しており、そのアプローチがあなたの組織にとって有効であれば、それはあなたの選択です。

スプリントバックログに含めるプロダクトバックログアイテムを選択する際には、依存関係と相互依存関係を考慮することが不可欠です。**依存**関係とは、多くの場合、技術的またはインフラ的な性質のものです。依存関係がある場合は、その依存関係にあるストーリーをスプリントに含めることも忘れてはいけません。

**ストーリーの完了はいつですか？**
インクリメントの完成の定義（DoD）に合致したときです。

ストーリーと関連するタスクには、受け入れ基準も必要です。これらの基準は、ストーリーがタスクに分割されるスプリントの開始時に設定されます。アクティビティとそれに関連する基準は、開発が進むにつれて、またフィーチャーや要求についてさらに発見があったときに変更されることがありますが、すべての作業は、使用可能になる前に受け入れ基準に照らしてテストされなければなりません。

上記の説明は、ソフトウェア開発で用いられる XP のテスト駆動開発アプローチによく似ています。この場合、「完成」の最終テストは、ソフトウェアがデプロイ可能で、使用可能で、安全であるかどうかを問うことになります。

*完成*とは、設定された個々の基準を満たすだけでなく、チームがインクリメントで完成させたものが、すぐに使用できる状態であることを意味します。

補足：ソフトウェアの開発であれば、*完成*という言葉は、デリバリーされたものが CI/DC（Continuous Integration/Continuous Delivery）パイプラインに入り、インクリメントが事実上すぐに利用可能になることを意味します。

CI/CD は、事前に定義されたツールと自動化されたワークフローを使用して、ソフトウェアコードをできるだけ早く本番環境に提供するためのコンセプトです。ソフトウェアをテストし、可能な限り自動化された方法で適切にデプロイするための包括的な規律は、しばしば DevOps と呼ばれます。DevOps と Scrum を統合することは、ユーザーや顧客のために早急に価値を創造する素晴らしい方法です。

チームがスプリントバックログに何を含めるべきかに合意したら、スプリントプランニングの残りの部分は、各ユーザーストーリーに属する作業の分解、タスクの見積もり、作業アイテムの受け入れを行うことに費やさなければなりません。

スプリントプランニングの後半でのプロダクトオーナーの職務は、質問を明確にし、受け入れ基準を推敲することです。プロダクトオーナーには、チームがどのように仕事をしているかを評価する役割はなく、バックログアイテムの相対的な重要性についてのみ意見を述べます。

スクラムマスターは、会議中に提起された新たな課題や懸念、または計画中に発見された仮定や依存関係が記録され、速やかに、よりよく理解されるようにする必要があります。

### 7.4.2 プロダクトバックログアイテムのサイジング

見積もりの技法については、プロダクトバックログの作成とリファインメントについてのセクションで述べました。同様の技法は、スプリントとプロダクトバックログの両方でストーリーを分解する際に使用されます。

バックログアイテムはプロダクトバックログリファインメントの際に見積もられ、サイズが決められますが、スプリントバックログに含まれるアイテムは、現在判明していることに照らし合わせて再評価する必要があります。バックログアイテムを再評価する際には、要求、環境、開発者の能力やキャパシティの変化を考慮します。

プロダクトバックログアイテムのサイズを適切かつ現実的に決めることは非常に重要です。そうしないと、チームはスプリント終了時に完成の定義（DoD）に沿った成果を出すことができないかもしれません。

スプリントバックログにプロダクトバックログアイテムが含まれていても、作業の分解を行っているうちに、見積もった時間や労力が不適切であることが判明することがよくあります。チームは、スプリントバックログの構成を再検討し、現実的にスプリント中に完了できないアイテムをプロダクトバックログに戻すようプロダクトオーナーと交渉することができます。それによって生まれるギャップは、他のより小さなプロダクトバックログアイテムをスプリントバックログに含めることで埋めることができます。

ただ、これを行うと、スプリントゴールや完成の定義（DoD）に影響が出て、修正が必要になる可能性があることを覚えておいてください。本来であれば、スプリントゴールに影響を与えない範囲での変更が望ましいのですが、基本的にはまだ計画に追われている状態なので、必然的にスプリントゴールを修正する必要が出てきます。

スプリントプランニングが完了したら、スプリントゴールに影響を与えるような変更はしないでください。プロダクトバックログアイテムのサイジング（大きさを調整）する方法はいくつかありますが、次節以降で簡単に説明します。

### 7.4.3 ストーリーポイント

ストーリーポイントに公式な定義はありませんが、一般的にストーリーポイントとは、過去に行われた同様の作業と、これから行われる作業を比較する、相対的な測定単位です。

したがって、ストーリーポイントは相対的なものであり、スクラムチームのベロシティと合わせて考慮する必要があります。この見積もり方法は、作業環境の変化に強い方法です。

これまでの作業と比較して、新しい作業アイテムは半分の時間で済むとか、3 倍の時間がかかるとか言うかもしれません。しかし、比較するための単位は同じでなければなりません。基準となる単位は、最小の数字ではなく、中規模または中規模よりやや小さめの数字を選択するのがベストです。例えば、1 から 10 までのスケールを使用する場合、基準となる単位は 3 または 4 が理想的です。通常のスケールが 1 から 10 であっても、ストーリーポイントが 20 もあるストーリーが時々あるかもしれませんが、これは、そのストーリーが重大な分解を必要とするエピックであることを示しています。

プロダクトバックログのレベルでは、時間見積もりのような技法を用いるための技術的な理解が不足しがちなプロダクトオーナーにとって、この技法はとても扱い易い技法です。

ストーリーポイントのバリエーションとして、T シャツのサイズを利用する方法があります。下の表は、1 から 10 までの代表的なストーリーポイントの参考にしてください。

*表 1 – T シャツのサイズとストーリー数の比較*

| T シャツサイズ | ストーリーポイント数 |
|---|---|
| XS | 1 ストーリーポイント |
| S | 2 ストーリーポイント |
| M | 3 ストーリーポイント |
| L | 5 ストーリーポイント |
| XL | 8 ストーリーポイント |
| XXL | 13 ストーリーポイント |
| XXXL | 21 ストーリーポイント |
| | |

なお、ここではフィボナッチ数列を相対的な基準として使用しています。

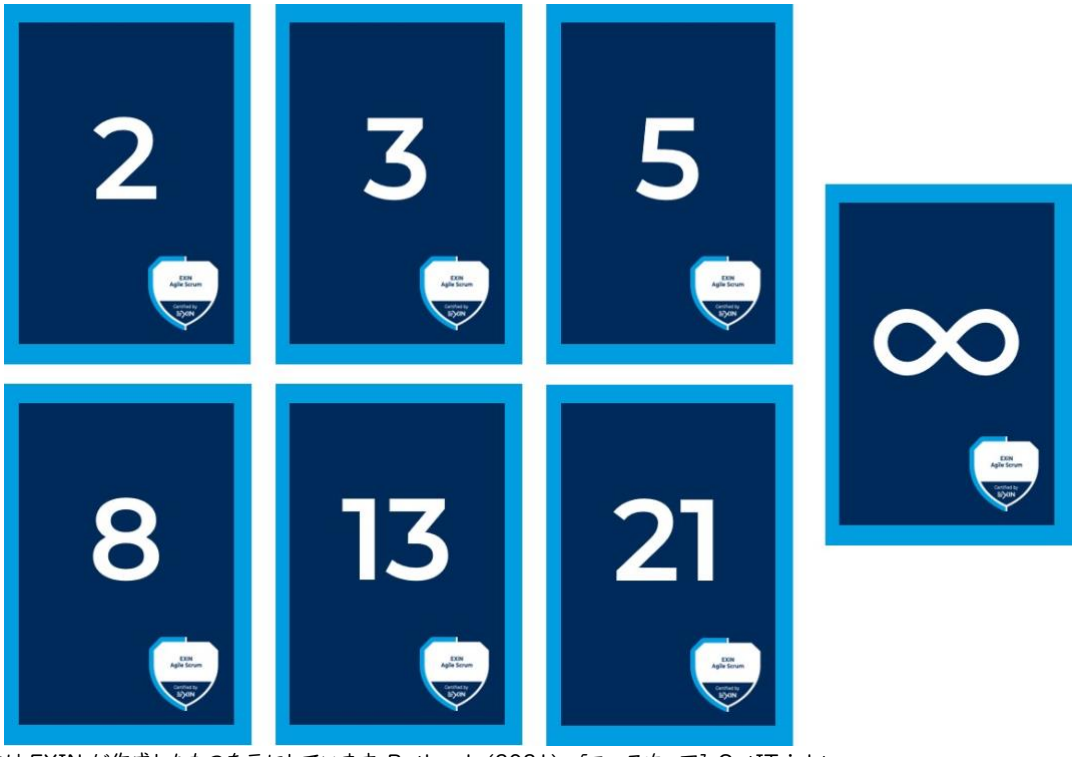
### 7.4.4 プランニングポーカー(スクラムポーカー)

プランニングポーカーは、実はポーカーをプレイするというより、UNO やスナップ(Snap)をプレイしているようなものです。

各開発者にはフィボナッチ数列が書かれたカードの組が与えられます。作業の大きさの推定値を聞かれた各メンバーは、自分の推定値と同じか、次に大きい数字のカードをテーブルの上に置きます。

スクラムマスターとプロダクトオーナーは、ここでの見積もりに関わるべきではありません。これは、開発者からのインプットを得る活動です。プロダクトオーナーは、プロダクトバックログにアイテムを追加する際に見積もりを行いますが、プランニングポーカーは実際の作業を行う人々からより良い洞察力と意見を得るための機会です。

*図 15 スクラムポーカーカード*




画像は EXIN が作成したものを元にしています。Botha、J. (2021). *[コースウェア]*. GetITright.

ここでの目的は、アクティビティやストーリーを完成させるためにどのような労力が必要か、チームメンバー間の意見の相違の度合いを測ることです。

場に出た数値の差が少ない場合は、慎重を期して高い方の数値を選択するといったやり方で、比較的容易に意見の一致をみることができます。

しかし、本当の価値は、外れ値にあります。例えば、チームのほとんどの人が 3 か 5 のストーリーポイントと言ったのに、グループ内の誰かが 20 と言った場合、2つの可能性があります。外れ値のカードを場に出した人が、やるべき作業の範囲や背景を理解していないか、チームの他のメンバーが知らないことを知っているかです。

外れ値を出した人には必ずと言っていいほど質問が投げかけられます。*なぜそのように考えたのですか?*

最初の問題に対処するのは簡単です。背景を説明することで、その人はおそらく次のラウンドのプレイで見積もりを修正することを決め、そうであればもう1ラウンドプレイするでしょう。

2つ目の可能性は、非常に興味深い議論や、プロダクトオーナーも他のチームメンバーも知らなかった、あるいは理解していなかった新しい洞察につながることがよくあります。この会話は通常、新たな依存関係やリスクの発見につながり、しばしばバックログに追加すべきストーリーの作成にもつながります。

場に出た値のグループに動きがなくなるまで、ラウンドを繰り返します。

前述したように、テーブルの周りに10 人の人がいて、8 人がタスクを 3 と見積もって、2人が 5 と見積もった場合、3 にします。半数が3で残り半数が5だった場合、5にします。全員が同じ数字に同意しようとして時間を無駄にする必要はありません。推定値が似通っていれば、それは許容できる結果です。

## 7.4.5 理想日数、理想時間

理想日数または理想時間での見積もりは、労力の指標として実時間を使用します。理想日数を考えるにあたっては、以下の条件を満たす必要があります。

- 作業開始時には、必要なものがすべて手元にあります。
- 見積もるストーリーは、あなたが唯一作業するものです。
- 中断することはありません。

もう一度言いますが、見積もりはその作業だけではなく、**仕事の一部というだけではなく、「(強調)チームの」仕事の一部です。**すべてのチームが同じ能力、キャパシティ、ベロシティを持っているわけではありません。

ちなみに、ポーカーカードの数字は日にちや時間を表すのにも都合が良いので、スクラムポーカーは理想日数や理想時間にも使えます。

### 7.4.5.1 なぜ理想日数、理想時間なのか?

この見積もりは、チームが中断なしで作業した場合に、ストーリーを完成させるのにかかる労力の日数または時間数に基づいています。つまり、気が散ったり、中断したり、タスクを切り替えたりしない理想的な状態を表しています。このような中断は通常、1日あたり約2時間を占めるため、理想的な1日は平均 6 時間ということになります。

現実には、ストーリーポイントを使ったとしても、どこかの段階でそれを時間に戻す必要があります。チームのベロシティがストーリーポイントで測られているのであれば、そんなことをする必要はないと主張する人もいるかもしれませんが、それは解釈の問題です。チームが1つのスプリントで 60 のストーリーポイントをこなすことができると言っても、スプリントはタイムボックス化されたイベントなので、事実上、どのような場合でも時間をさかのぼって作業したことになります。

ストーリーポイントを使用する真の価値は、少なくとも最初のうちは、組織のカルチャーを変えることです。多くの組織では、タイムラインが与えられると、デフォルトではコミットメントとみなされます。そこで、ストーリーポイントを使うことで、「でも 8 時間しかかからないと言っていたじゃないか」と言われる事態を避けることができます。

顧客と話すときに、理想日数や理想時間を使ってはいけません。お客さまはそれを約束したものと受け取ってしまいます。

### 7.4.6 付箋紙を使った高速見積もり

Roman Pichler（ローマン・ピヒラー）の技法は、チームが他の見積もり技法を使う時間がない場合に使用できます。

ボードにフィボナッチ数列を数字と数字の間にかなりのスペースを空けて書き、各チームメンバーに見積もるプロダクトバックログアイテムをいくつか与え、それを付箋に書いて適当と思われる数字の下に貼ってもらいます。

全員が作業を終えたら、チームメンバー全員がボードを見て、問題がありそうなプロダクトバックログアイテムや正しく見積もられていないと思われるプロダクトバックログアイテムを見つけます。そのメンバーがすぐに付箋を移動させるか、見積もりを修正するかのどちらかです。

別の方法としては、チームでそのことについて簡単な会話をしてから、ボード上の適切な場所に移動させることもできます。

チームメンバーには、見積もりを過度に批判しないように指示してください。ここではそのような意図はありません。

この手法は、精度は低いですが、チームが多くのことを素早く成し遂げることができることを理解してください。

おそらく大部分の人は、最初からこの技法を使うと、チームが成熟するにつれてより良い結果が得られるであろうことが理解できると思います。

### 7.4.7 その他、知っておくべきこと

ストーリーやタスクを見積もるためにどのような手法を使っても、最初のうちはおそらく間違ってしまうでしょう。しかし、心配しないでください。やればやるほど、あなたの見積もりは良くなっていきます。

タスクやストーリーを熟知している人だけが見積もりをするべきだという意見もありますが、チーム全体を巻き込むことで、新たな知見が得られることも少なくありません。第2に、チームの理解が深まり、最終的には学習と成長に貢献することになります。

もし、チームのメンバーが「自分にはその労力がわからない」と感じたら、見積もりを出すのを控えることができます。毎回、自分の手札を出す必要はありません。

#### 7.4.7.1 もしあなたが労力を見積もる方法を知らないとしたら?

それは見積もることと同じくらいシンプルです。もっと明らかにする必要があります。このような状況は、チームが新しいタイプの作業に対応しなければならないときによく起こります。

明らかにすることは、その特定のアイテムの作業を継続する前に、最初に完了しなければならない新しいストーリーになります。バックログに新しいストーリーを追加して、見つけることができるようにしてください。

そのためには、モックアップやプロトタイプを作成したり、調査をしたり、経験のある人に話を聞いたりすることが多いです。

#### 7.4.7.2 非機能要求についてはどうか?

非機能要求とは何でしょうか?一般的には、非ユーザー要件や技術要件、インフラ要件と捉えられがちです。しかし、スクラムではそのように定義されていません。

非機能要求とは、システムレベルまたはプロダクトレベルの制約のうち、機能に関係しないものを指します。その代わりに、性能、正確性、移植性、再利用性、保守性、相互運用性、容量などの属性を指します。プロダクトバックログアイテムがステークホルダーに完全に受け入れられるために必要なものです。非機能要求は、品質基準に関連することが多いですが、必ずしもそうではありません。

多くの場合、機能要求の受け入れは、非機能要求に依存します。機能要求を見積もる際には、それに依存するすべての非機能要求も考慮する必要があります。

スクラムやカンバンのボードに非機能要求を置くと、何度もボードに戻さなければならないような気がします。つまり、他の人が依存しているために発生し続けている要求がある場合、それは非機能要求である可能性が高いです。

これらの非機能要求を満たさずに*完了した*場合、プロダクトやサービスは使用できないか欠陥があるため、すべてのスプリントにこれらの非機能要求の一部を含める必要があります。依存性のある非機能要求が完成の定義(DoD)に含まれていることを確認することが重要です。

### 7.4.8 その他のストーリーは、他のステークホルダー

その他のストーリーは、他のステークホルダーのために書かれた単なるユーザーストーリーです。つまり、技術的要求とは、ユーザーや顧客が要求していないが、他の要求を実現するために必要なもので、異なるタイプのユーザーストーリーです。ここでのユーザーは、ユーザーや顧客の成果を促進する組織システムの一部である社内スタッフです。

リーンの観点では、これらの要求はおそらく付加価値のないものと考えられるでしょうが、必要なものです。なぜならば、顧客が求めていないので費用すら払われない可能性が高くなりますが、もしそれがなかった場合、顧客が求めているものを提供することができなくなるからです。

このような技術的なストーリーを、プロセスストーリーやインフラストラクチャストーリーなど、さまざまな名前で呼ぶ人がいます。スクラムでは、これらはすべてユーザーストーリーであり、ユーザーが変わるだけなのです。

## 7.5 スプリントの一環としての改善活動

### 7.5.1 障害物と制約理論(TOC)について

スクラムマスターの責任の一つは、スクラムチームが遭遇する障害物の解決を支援することです。これらの障害物は、チームメンバーが最大限の能力を発揮するのを妨げるものであれば何でも構いません。

スクラムマスターは、誰かが障害物や制約について知らせてくるまで待つ必要はありません。優れたスクラムマスターは、以下に説明する POOGI 技法を利用して、積極的に制約を特定します。

人々はしばしば、障害物を作業の妨げになるものと考えます。その通りですが、ほとんどの障害物は作業を完全に止めるものではなく、邪魔やボトルネックになることがあります。スクラムチーム、特にスクラムマスターは、Dr. Eliyahu M. Goldratt(エリヤフ・M・ゴールドラット博士)が 1984 年に発表した*『ザ・ゴール』*という本の中で定義した制約理論(TOC)から多くを学ぶことができます。

Dr. Goldratt(ゴールドラット博士)が提示した前提は、組織やチームは、スループット、運用コスト、在庫という 3 つの指標の変動によって測ることができるというものです。スループットは、システムが価値を生

み出す速度と見なすことができ、運用コストと在庫を合わせたものは、リーンにおける仕掛（WIP）として定義することができます。仕掛（WIP）については、後でもう少し詳しく説明します。

ゴールを達成するためには、安全性や品質、法的義務など、必要な条件を満たす必要があります。

ほとんどのビジネスでは、スループット、在庫、およびオペレーショナルエクスペンスを改善することで利益を上げることがゴールとなります。

では、これがスクラムにおける障害物や障害の管理とどのような関係があるのでしょうか？

### 7.5.2 5 つのステップに集中せよ

制約理論とは、ゴール指向型システムのゴール達成率は、少なくとも 1 つの制約によって制限されるという前提に基づいています。制約が存在しなければ、システムのアウトプットは無限になりますが、それは不可能です。

制約を通過するフロー（流量）を増やすことによってのみ、全体のスループットを向上させることができ、そのための最良の方法は、5 つの（簡単な）ステップに従うことです。

1. 制約条件を明らかにする。
2. システムの制約をどのように活用するかを決める。
3. 他のすべてを上記の決定に従属させる。
4. システムの制約を緩和する。
5. 前のステップで制約が解消された場合は、すぐにステップ 1 に戻ります。

この 5 つのステップは、制約条件に焦点を当てた継続的改善を目的としており、継続的改善のプロセス（POOGI）と呼ばれています。

**しかし、exploit（活用）、subordinate（従属）、alleviate（緩和）などの言葉は何を意味するのでしょうか？**

ステップ2では、「制約を**活用する**」という考え方で、新たな投資をせずに、制約が残っている場合に、どのようにすれば最良のアウトプットが得られるかを理解します。

**従属とは、**この制限の中で作業が行われるように設計する必要があるということです。たとえシステムの他の部分の速度を落とすことになっても、仕事の流れを確保することが不可欠です。あなたの労力の効果を評価するには、フローを見れば簡単にわかりますし、仕掛（WIP）を制限することでもできます。

これにより、症状ではなく真の根本原因を特定し、根本原因を恒久的に除去する方法を考える時間が得られます。これらの対策を実施すると、制約は**緩和さ**れますが、すぐに新たな制約が発生する可能性が高いため、このサイクルを再び開始する必要があります。

制約はもはや存在しないので、制約を利用するためにステップ2で課された制限を取り除くことを忘れないようにすることが重要です。

問題を解決するためには投資が必要であり、解決策をプロジェクトや改善活動として実施しなければならないほど複雑で大規模な場合もあるため、制約を利用するだけで緩和されない状態でチームが長期間運営される可能性が非常に高くなります。

### 7.5.3 継続的改善バックログ（CIB）について

スクラムでは改善バックログを特に認めていませんが、優れたアジャイル組織はすべて改善バックログを持っています。

スクラムでは、改善点はプロダクトバックログアイテムとして定義されるべきだとしていますが、特定のプロダクトバックログに属さない改善点はどうなるのでしょうか？

多くのスクラム専門家は、一般的な継続的改善のバックログがアジャイル組織に存在することを提案しています。改善が明らかにプロダクトに属するものであれば、それを記録するのに最適な場所はプロダクトバックログになるでしょう。

しかし、プロダクトに関する改善が特定のプロダクトバックログに記録されている場合、どのアイテムが一般的な継続的改善のバックログに属するのでしょうか？

プロダクトバックログアイテムを改善するためのソースは 3 つあります。

- プロダクトに依存しない**監査、CSI、またはカイゼン**（経験学習および改善）の取り組みの**結果**
- プロダクトやシステムを横断して機能させる**活用された制約**（例：多くの非機能要求）
- 組織、システム、プロセスにおける**技術的負債を特定し**、負債を返済するための取り組みを定義する（Lean と DevOps の両方で行われており、Scrum 環境でも適用されることができる）。

これらの取り組みを紹介することが本書の目的ではありませんが、リストの最後の項目の意味を明確にしておくことは有益です。

### 7.5.4 技術的負債

この言葉をグーグルで検索すると、ソフトウェア開発と密接に関連した定義が出てきます。

> 技術的負債とは、ソフトウェア開発における概念であり、時間がかかるより良いアプローチを使用する代わりに、簡単な（限定的な）解決策を選択することで生じる追加の手直しの暗黙のコストを反映したものである。[9]

しかし、すべての組織は技術的負債を抱えています。簡単に言えば、技術的負債とは、組織の中で間違っていると分かっていることの総体であり、通常、人々は「時間ができたら直します」と言います。

現実には、これではいつまでたっても直らないことが多いのです。

問題は、このような中途半端で性能の低い、制約の多い問題が、徐々に蓄積されていき、一つの巨大な制約となって、大きなマイナスの影響を与えることです。

後で直すと言わず、後で直すための時間を作った方がいい。

そこで登場するのが、CIB（Continuous Improvement Backlog）です。（後で直す）と言うならばそれを記録するのです。他のプロダクトバックログと同様に、CIB にもプロダクトオーナーが必要であり、スプリントサイクルごとに CIB の中にある課題群を解決するために何かを行わなければなりません。

### 7.5.5 改善インクリメント

前述したように、スクラムのルールでは、すべてのスプリントで少なくとも 1 つのインクリメントを提供しなければなりません。これは CIB にも当てはまります。

しかし、多くの企業では、すべてのスクラムチームが CIB からインクリメントを提供することを選択しています。また、専門の CIB のプロダクトバックログアイテムだけに取り組む専門のスクラムチームを持つ企業もあります。どのようなアプローチを選択したとしても、改善インクリメントを提供することは、持続可能なアジャイルビジネスを構築する方法の 1 つです。

---

[9] テクニカルデット。（2021）. https://en.wikipedia.org/wiki/Technical_debt から 2021 年 2 月 14 日に取得。

# 8 スプリントでは他に何が行われるのか？

前述の通り、スプリントはスプリントプランニングから始まります。そして、1つまたは複数のインクリメントを完成させるため、スプリントの大半の作業を通して実施されます。その作業の後に、スプリントレビューとスプリントレトロスペクティブが続きます。スプリント期間中、スクラムチームはデイリースクラムを開催し、お互いの情報交換、計画の調整を行います。

スプリントプランニングについては、すでに大事な注意点を説明しましたが、計画と見積もりの章では、さらなる洞察が得られるはずです。

次のセクションでは、まだ触れていない残りのスプリントの活動に関連した情報と、DevOps の分野でスクラムの使用法について、いくつかの考え方を述べます。

## 8.1 デイリースクラムの詳細

開発者は、毎日同じ時間に15 分間のタイムボックスミーティングを行い、次の日の作業を計画します。その目的としては、スプリント開始以降に行われた作業を検査することで、チームのコラボレーションとパフォーマンスを最適化することであり、次の作業を計画し残っている作業の予測をするためです。

前述したように、デイリースクラムでは、すべての開発者が 3 つの質問に答えます。

- 昨日、私は…
- 今日、私の予定は…
- 私ができなかった理由は…

これらの質問に答えることで、有益な洞察を得ることができますが、デイリースクラムで行われるのは、これだけであると多くの人が誤解していました。

これら 3 つの質問は、必ず求められるものではなく、チームにとってより適切な他の質問をしても構いません。これは完全にチームによります。質問は会話を始めるための手段に過ぎません。

検査と適応はスクラムの中心であり、デイリースクラムはまさにそのためのものです。

開発者は、デイリースクラムにてスプリントゴールに向けた進捗状況を確認し、スプリントバックログの完了に向けて、進捗状況を確認する必要があります。これは、スクラムチームが最新の現状に目を光らせ、実績と計画が一致しない場合には対応できるようにするためのものです。

スクラムチームは自己管理型であり、これは、スプリント期間中の変化、課題、問題に、チームが適切に対応し可能な限り迅速に対応することで、結果を出すことです。そのことを忘れてはなりません。

開発者は即座に反応することができ、適切な対応を定めることができます。

しかし、デイリースクラムは詳細な計画会議ではなく、潜在課題、依存関係、深刻で解決すべき問題の特定に焦点を絞ることに注意してください。多くの場合、適切な対応策を直ぐに定義できますが、それができない場合は、1人以上のチームメンバーで対応を策定することが、通常の業務の一つになり、スクラムチームのメンバーは、デイリースクラムの直後にミーティングを行い、詳細な議論を行ったり、スプリントの残りの作業を調整したり、再計画したりすることが多くなります。

デイリースクラムは、コミュニケーションを向上させ、スクラムにおける検査と適応をするための主要な手段の1つです。

デイリースクラムは、他のミーティングの開催を不要にするか削減を行い、スプリントゴールに向かって進むことを妨げる障害を特定します。また、全員に情報が行き渡り、迅速に意思決定が行われ、すぐに行動に移すことができます。

デイリースクラムは、開発者が主催者として議題と進め方を決めます。スクラムマスターはしばしば会議をファシリテートすることを任され、記録を残し、カンバンやバーンダウンチャートで進捗状況の更新をしたりしますが、スクラムマスターが主催者となる会議ではありません。

とはいえ、デイリースクラムが15 分を超えないようにするのは、スクラムマスターの責任です。開発者が問題を特定し、その明確な解決策がなくても、スクラムマスターは会議を前に進めるべきです。スクラムマスターは、解決策を見つけるために誰が関与しなければならないのか、また、何らかの結論を出すために特定のリソースがいつ可能となるかを尋ねます。これらの質問に答えられたら、デイリースクラムを前に進めます。開発者以外の人がミーティングを中断できるのはこの時だけです。

## 8.2 レビューとレトロスペクティブの実施

### 8.2.1 スプリントレビューの詳細

スプリントレビューは、スプリントの最後に行われ、インクリメントを検査し、必要に応じてプロダクトバックログに変更を加えます。

スプリントレビューは、リリースやデプロイメントのゲートステージではなく、または状況確認や承認をするための会議体でもないことに注意ください。この会議は、フィードバックを引き出しコラボレーションを促進することを目的としています。

スプリントに複数のインクリメントがある場合は、インクリメントが完成すると、速やかにデプロイされ利用可能にする必要があります。

スプリントレビューでは、検査と適応をするための能力について再度見直します。

スプリントレビューでは、スプリント中に何が行われたか、また、スプリント中に計画からの変更がなされたかを、スクラムチームとステークホルダーが議論します。

何が完成したのかをデモし、スプリントに含まれるすべてのインクリメントについて、完成の定義（DoD）をいかに満たしているかが示されます。

そして、ステークホルダーが協力して、価値を最適化するために次に何を完成するとよさそうかを特定します。

スクラムマスターは、イベントが開催されると、参加者がその目的を理解していることを確認します。また、会議がタイムボックス内に収まるようにする必要があります。

このイベントの参加者は、スクラムチームと、通常プロダクトオーナーが招待する関連するステークホルダー（例：ユーザー、顧客、管理者など）になります。

スプリントレビューを、どのような形式で実施するかは組織によりますが、彼らの固有ニーズに基づいて決定されます。ここではスプリントレビューで行われることについて、経験豊富なアジャイルコーチの提案をいくつか紹介します。

- プロダクトオーナーは、どのプロダクトバックログアイテムが実施され、どのアイテムが実施されていないかを説明する
- 開発者は、スプリントで何がうまくいったのか、どんな問題に直面したのか、その問題をどのように解決したのかを議論する
- 開発者は、完成した作業成果をデモし、インクリメントに関する質問に答える

- プロダクトオーナーは、現状のプロダクトバックログについて議論する。必要であれば、現在までの進捗状況を元に、期待できそうな目標と納期を予測する
- 次に何をすべきかについて、グループ全体でコラボレーションし、スプリントレビューの結果が、次のスプリントプランニングに貴重な情報を提供できるようにする
- プロダクトの市場や潜在的な用途がどのように変化したか、次に行うべき最も価値のあるものは何かを検討する
- 次に予定されるプロダクトの機能や性能のリリースについて、そのスケジュール、予算、潜在能力、市場について検討する

スプリントレビューの結果として、次のスプリントで正しいと思えるプロダクトバックログアイテムが定義され、その内容で改訂されたプロダクトバックログとなります。また、プロダクトバックログは、新たな機会やニーズに合わせて全体的に調整されることもあります。

### 8.2.2 スプリントレトロスペクティブの詳細

スプリントレトロスペクティブは、スプリントレビューの後、次のスプリントプランニングの前に行われます。何がうまくいったのか、何が改善できるか、何を将来避けるべきなのか、そして、次のスプリントの一部を成すことになりそうな分野について、スクラムチームが話し合うための会議です。4 週間のスプリントの場合、スプリントレトロスペクティブは通常 3 時間行われ、スプリント期間が短い場合はスプリントレトロスペクティブも短くなります。

スクラムマスターは、イベントが開催され、参加者がその目的を理解していることを確認します。これもまた、スクラムチームが、今後取り組むべき改善点を特定することによって、検査と適応をするための機会となります。

スクラムチームのメンバー全員が参加しなければなりません。

スクラムマスターは、次のスプリントに向けて、より効果的で素晴らしいものになるように、スクラムチームにプロセスやプラクティスの改善を促します。各スプリントレトロスペクティブにおいて、スクラムチームは、製品プロダクトまたは組織の基準と矛盾しないように、作業プロセスを改善するか、完成の定義（DoD）を改善することにより、プロダクトの品質を向上させるための計画をします。

スプリントレトロスペクティブが終わるまでに、スクラムチームは次のスプリントで実施する改善点を特定すべきです。次のスプリントでこれらの改善策を実施することは、スクラムチーム自身の検査への適応をすることです。

改善はいつでも実施できますが、スプリントレトロスペクティブは、検査と適応に焦点を当てる正式な機会となります。

多くのスクラムチームは、簡単に修正できない項目については継続的な改善バックログを作成し、改善項目をすべてのスプリントに盛り込みます。それによって、それぞれのスプリントでプロダクトと改善のためのインクリメントを作成します。

# 9 複雑で大規模なプロダクトバックログ

全てのプロダクトには1つのプロダクトバックログしかないことを覚えていますか?しかし、そのプロダクトが大規模で複雑であり、プロダクトのビジョンを実現するために多くのスクラムチームが必要となる場合はどうなるでしょうか?

複数のチームの作業を1つの調整されたバックログで進めるのは、控えめに言っても簡単ではなく、複雑なものになります。

## 9.1 スケールする方法

スクラムには様々なスケールさせる方法があり、多くのアジャイル手法はアジャイルをスケーリングすることにフォーカスしています。

複雑さや規模に対応するために、次のような多くの試みがなされています。

- SAFe
- LeSS
- Nexus
- Scrum@Scale

SAFe や LeSS は非常に複雑でオーバーヘッドが大きい手法であるため、ここでは説明をしません。Nexus と Scrum@Scale は比較的シンプルで軽量であり、全てのスクラムのプラクティスを事実上「そのまま」使用しています。

これらはどのような方法でしょうか?

**Scaled Agile Framework® (SAFe®)** は、企業規模で「アジャイル」プラクティスを実施するための組織とワークフローのパターンのセットです。このフレームワークは、役割と責任、作業の計画と管理方法、守るべき価値観について体系化されたガイダンスを含む知識体系です。

**大規模スクラム (LeSS)** は、プロダクト開発のためのスケーリングフレームワークと見ることができますが、組織のためのスケーリングフレームワークと見なすこともできます。LeSS はプロダクト開発において問題を種々のより明快で単純な方法で解決しようとすることで、組織の複雑さを解消するものです。ここで明快で単純なとは容易さのことを意味していません。特に短期的に見るとそうです。LeSS はシンプルであっても導入の初期段階では難しいです。

**Nexus** とは、大規模なプロダクトの開発と維持を行い、また、ソフトウェア開発のための新たな取り組み方で、そのためのフレームワークです。Nexus はスクラムをその構成要素として用いています。

同様に、**Scrum@Scale** はスクラムの基本的な構成要素に依存しています。しかし、ここで用いられているいくつかのプラクティスの中には、従来の考え方や長い事前計画へ大きく依存している側面もあるため、結果的に Nexus がスケールさせる上で最もシンプルな選択肢となります。

ソフトウェア開発は複雑であり、正常に動くソフトウェアにするためには、統合された作業として調整されなければならない多くのアーティファクトや活動が含まれ、それによって完成度の高い結果が生み出されます。作業は系統的に整理され、順序付けられ、依存関係が解決され、そして、段階的に成果が得られます。

多くのソフトウェア開発者は、スクラムフレームワークを使用して、チームとして共同で作業を行いながら、動作するソフトウェア・インクリメントを開発しています。しかし、もし複数のスクラムチームが、同じプロダクトバックログやプロダクトと同じコードベースで作業し、お互いが影響し合う環境で作業をしている時、彼

らはどのようにコミュニケーションをとるのでしょうか？異なるチームで彼らが働いている場合、どのように作業を統合し、どのように統合されたインクリメントをテストするのでしょうか？

こうした課題は、2つのチームで作業を統合するときに発生し、3 つ以上のチームでは非常に複雑になります。

これは、スクラムが使われている他の多くのタイプのプロジェクトにおいても同じことが言えます。したがって、ここで説明することは、最も複雑で相互依存性のあるプロジェクト環境においても同様に適用できます。

本書では、スケーリングの主要な手段として Nexus を紹介します。これは、最も複雑にならないアプローチとして考えられるからです。もちろんですが、あなたの組織に最適な方法を自由に選択しても構いません。

外骨格[10]のように、Nexus はスクラムフレームワークの周りを囲むようなフレームワークです。Nexus は、複数のスクラムチームの作業結果を 1 つの統合されたインクリメントにまとめます。Nexus はスクラムと一貫性があり、その構成要素はスクラムのプロジェクトに携わったことのある人にとっては馴染み深いものです。その違いは、スクラムチーム間の依存関係や相互運用に対してより多くの注意を払うことにより、少なくとも 1 つの完成した統合インクリメントを、全てのスプリントで提供することです。

## 9.2 アジャイルスクラムのスケーリングに対する考え方

同じプロダクトに対して複数のバックログを持たないことが基本ルールであるため、同じバックログの下で作業する複数のチームは、効果的なコミュニケーションとコラボレーションを確立する必要があります。

1 つの技法として、バックログの内容について機能性よりもハイレベルで考えることです。プロダクトバックログアイテムの分解は、エピック、テーマ、機能とすでに説明しました。テーマに基づいて異なるチームに作業を割り当てることも可能です。このアプローチにより、対立やチーム間の依存関係が発生する可能性を低くすることになるでしょう。

ここで注意することは、テーマをプロダクトの機能性を表したプレースホルダー（四角い枠）として記述することです。バックログの構造化と優先順位付けに役立ちます。これは、SAFe で説明されている戦略的テーマ、つまり、ポートフォリオを企業戦略に紐付けることでビジネス目標の差別化をする。といったことではないので注意をしてください。

アジャイルスクラムの中には、大規模で複雑なデリバリーやバックログにするためのスケーリング手法が定義されています。本書の後半では、Nexus についてより詳しく紹介します。Nexus はスケーリングに関して、伝統的なスクラムの考え方にある程度準拠していますが、そのアプローチにはいくつかの微妙な変更が加えられています。

Nexus にはいくつかの新しい専門用語が導入されています。これらの専門用語を矛盾なく正しく使用することで、価値を示すことができ誤解する可能性を減らすことができます。スケーリングおよび Nexus については、後ほど詳しく説明します。

[10] 外骨格。(2021)。https://en.wikipedia.org/wiki/Exoskeleton から 2021 年 9 月 23 日に取得。

# 10 ビジュアルマネジメント、スクラムとカンバンボード

カンバンとは、もともとリーンの概念です。アジャイルスクラムがリーンの原則に基づいて構築されている以上、スクラムを使用する際に、カンバンを手法として使用することは適切なことであり間違いではありません。このことは、David J. Anderson 氏が、カンバンとスクラムの技術を組み合わせた「スクラムバン（Scrum-ban）」を作成したことからも分かります。

ビジュアルマネジメントは、リーンおよびリーン関連のすべてのマネジメント手法において重要な概念となります。ビジュアルマネジメントツールは、文字情報や口頭によるコミュニケーションに比べて、より素早くメッセージを伝え人の目を引き付けることができます。

どうしても納得がいかないという方には、次のような例え話をしてみましょう。あなたが毎日車で通勤しているとして、あなたは停車した標識や信号の数を正確に答えることができますか？でも、あなたは停車しています。そうしないと、今頃多くの事故を起こしたはずです。それが、ビジュアルマネジメントと情報ラジエータの力です。

この種の情報が視覚的に示されていると、チームが問題を発見し行動を起こすことがより容易になります。また、情報ラジエータが使用されると、問題、欠陥、トラブルが発生したときにそれらが表示され、チームはより迅速かつ効果的な行動をとることができます。

進捗状況を可視化し、インクリメントの完成に向けたチームの進捗を追跡し、問題や欠陥を迅速に明らかにするための方法は、スクラムボードを使用することです。

チームはよくスクラムボードをカンバンボードと呼んでいますが、スクラムボードはカンバンボードから生まれた最もシンプルな簡易版なので、厳密に言えばそれは間違いではありません。しかし、手法としてのカンバンは、単に進捗状況を把握するだけのものではありません。したがって、スクラムボードとカンバンボードは違うもので、カンバンボードは一般的にスクラムボードよりも少し複雑になります。

ここでは、その相違点を簡単に紹介します。どちらの方法を使用するかは、チームで選択してください。

## 10.1 スクラムボード

スクラムボードを最も簡単に表現すると、スプリントバックログに存在するストーリーと関連するタスクを視覚的に表現したものです。これは、スプリントの全体、Nexus 計画、あるいはプロダクトゴールについて、スクラムチームから孤立や分断された感覚を軽減するための素晴らしい手法です。

ボードには通常、4 つ（またはそれ以上）の列があり、通常は以下の名前が付いています。

- ユーザーストーリー（またはスプリントバックログアイテム）
- 未着手（To do）
- 進行中（In progress）
- 完了（Done）

あるソフトウェア開発用のボードは、次のようになっているかもしれません。

- ユーザーストーリー
- タスク（To do）
- 進行中（In progress）
- テスト（Test）
- CD/CI（DevOps）パイプライン
- 完了（Done）

これはソフトウェア開発用ボードの1例であり、もしソフトウェア開発とは異なる分野でスクラムを使って価値を提供する場合、ボードは間違いなく違うものになるということを覚えておいてください。スプリントの開発者は、「To-do」から最終的に「Done」に至るまで、進捗状況を追跡したい適切なステップを選択すべきです。

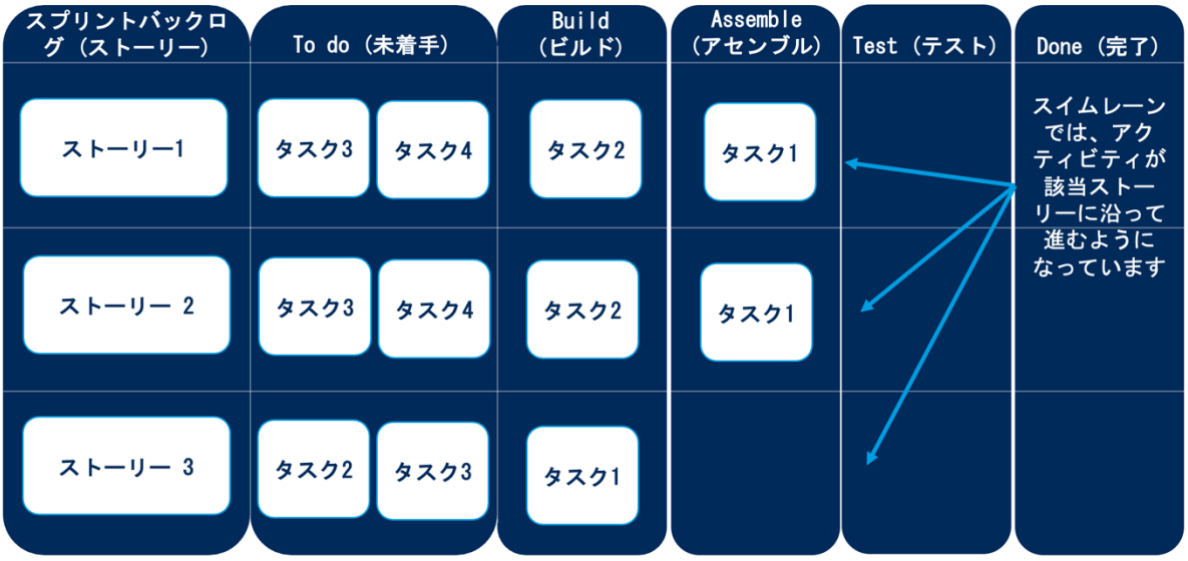
ストーリーとそれを構成するタスクは、通常、大小のメモカード（cue-cards）やポストイットに書き込まれます。

通常、タスクは必要とされる能力に応じて、実施する人が割り当てられ、ボードの左側から右側に移動する際に再割り当てされることがあります。

すべてのストーリーが顧客要求ではなく、依存関係や非機能要求を反映したもの、改善ストーリーとその関連タスクを反映したものがあることに注意してください。

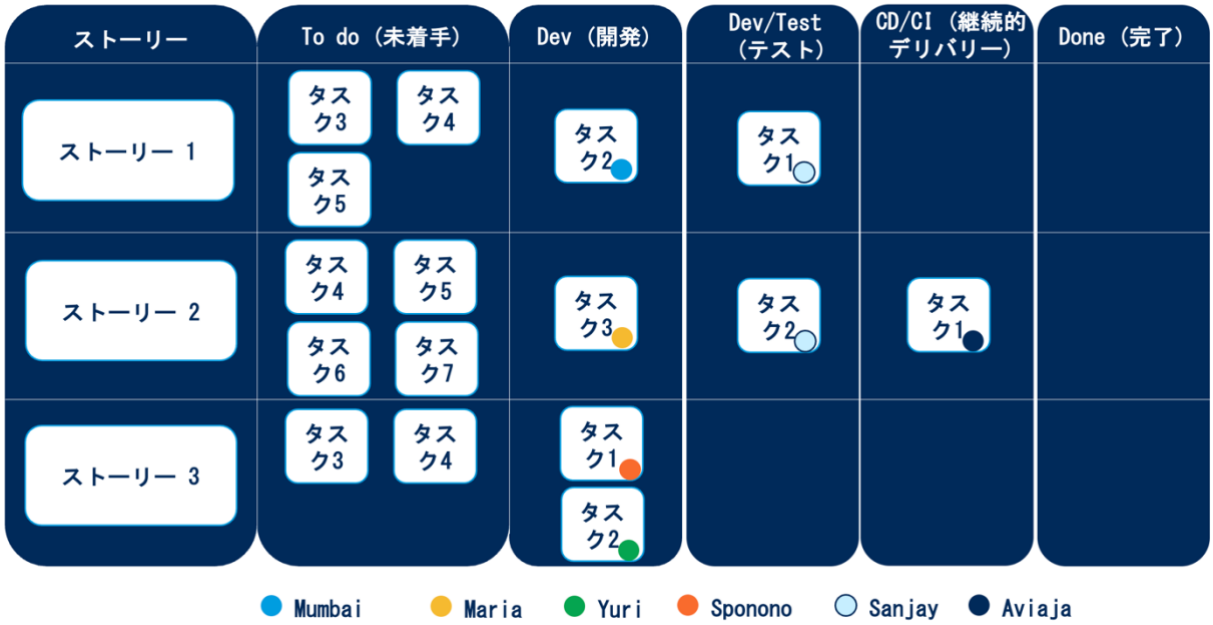
下の例は、スプリントの一部を形づくる様々なストーリーのスイムレーンが示された基本的なスクラムボードです。これにより、タスクがどのストーリーに属しているのかを容易に確認することができ、作業が完了したときチームメンバーの間を飛び回って再確認する必要性がなくなります。

*図 16 シンプルなスクラムボード（例：自動車のステアリングラックの製造）と、スイムレーンを使用してタスクを関連する PBI（ストーリー）にリンク*




図は EXIN 制作：出典 Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

*図 17 ソフトウェア開発プロジェクトにおける典型的なスクラムボード（タスクが誰に割り当てられているかを示すために色付きの付箋を使用）*




図は EXIN 制作：出典 Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

作業担当者を示すために、色のついたシールが使われています。

この例のようにシールを使う必要はありません。代わりにタスクカードや付箋に担当者が書かれていることが多いです。他の情報として、ストーリーやタスクを完成するための必要工数の目安、どのタスクが優先度が高いか、あるいは、ストーリーとタスク間、タスクとタスク間の依存関係がカードに書かれていることがあります。依存関係は図には示されていません。これについては、Nexus を使ったスケーリングの説明をするときに詳しく説明します。

## 10.2 スクラムボードを使う理由は何か？

スクラムボードは見える化を促進し、コミュニケーションを円滑にします。また、問題がどこにあるかをチームメンバーに示すことができ、お互いに助け合ったり（swarm on issues）、間違った前提に基づいた計画であると分かれば、それを変更することができます。このように、スクラムボードは、検査、適応、透明性を促進し、結果としてチームワークを促進するのに役立ちます。

また、やるべきことの全てを誰もが見ることができるので、チームがスプリントとそのスプリントに関連する作業にコミットすることを助けます。

このボードはスプリントプランニングの一環として作成されます。スプリントプランニングでは、まずチームがどのアイテムをスプリントバックログに含めるべきかを決定します。次に、ストーリーのタスク分解をしますが、そのストーリーを実現するために必要なタスクに分解します。

スクラムボードの素晴らしい点は、可視化されているのですぐに作業を開始できることです。スクラムはスクラムボードの使用を定めているわけではありませんが、使用することでチームにとって大きなメリットがあることを忘れないでください。

## 10.3 スクラムボードの使用

スクラムボードの作成は、スプリントプランニングの最後の活動として実施するのがベストです。スクラムマスターはスクラムボードの作成と維持に責任があるとよく言われますが、この記述は真実ではありません。

スクラムボードは開発者のものです。スクラムマスターがその維持管理を任されることもありますが、そうすることは必須要求でもなく規定要求でもありません。

スクラムボードは開発者のものであり、開発者が自分たちの作業を最適化するために役立つものです。

では、どうやってスクラムボードを作るのでしょうか？

開発者は彼らが行う作業のタイプを、最もよく表現したボードを作成する必要があります。どのようなボードにするかについては、すでに2つの例を挙げましたが、最低限でも、未着手「To-Do」、作業中「In Progress」、完了「Done」の列を設ける必要があります。

プロダクトバックログアイテム（PBI）は、通常ストーリーの形でスプリントバックログに追加されますが、同じようにスクラムボードにも追加されるべきです。PBI は、スプリントプランニング中に、通常は所要期間、依存関係、担当者、完了までの所要時間などの多くの属性を持ったタスクに分解されます。関連する全てのタスクを、それらが属する PBI の横のスイムレーンに配置します。また、同じ色の付箋やカードを使ってそれらを配置します。上述の例では、その両方が行われています。

こうすることで、最初のスクラムボードが作成され、スプリントプランニングが完了した時点でボードが使用可能になります。

デイリースクラムでは、スクラムボードに表現されている作業内容と、それが日々どのように変化しているかを中心に議論します。今でも 3 つの質問（これはスクラムガイドでは定義されなくなりましたが、いまだに役に立つでしょう）をすれば、どのカード/付箋がどの列にあるか、誰がタスクに割り当てられているかを直ぐに判断できます。ボードには、タスクの完了を妨げる問題や障害（一般にブロッカーと呼ばれる）も反映されます。開発者は自分で選択したタスクに取り組むことになりますが、スプリントプランニング中に前もってタスクの割り当てが行われることはない、ということを覚えておいてください。

ソフトウェア開発プロジェクトでは、スクラムボードはバグの追跡と報告にも使用できます。バグが発見されるとスクラムボードに追加され、それがプロダクトオーナーによって分析され、優先順位に沿って時間の枠内で解決されます。

カードの移動やタスクの割り当ては開発者の責任であり、スクラムマスターの責任ではありません。しかし、チームはしばしばスクラムマスターにスクラムボードを管理してもらい、現在のスプリントの状況やスプリントに関連したイテレーションの状況を常に反映するようにします。そのため、スクラムマスターがスクラムボードに関連した活動を行うことがよくあります。

## 10.4 カンバンとスクラムボードの違いは何か？

カンバンという言葉は、大まかに訳すと「見えるカード」という意味です。カンバンは、トヨタ生産方式やリーン生産方式の在庫計画システムとしてスタートしました。そうした経緯から、作業スケジューリングシステムとして使用される技法となりました。そのシンプルさと可視化できる能力は、スクラムやその他のアジャイル手法との相性が非常によく、先に述べたスクラムバンに至っています。

元々「カンバン」とは、ある生産工程で在庫が少なくなったときに、次の作業が（ジャストインタイムで）できるように供給を間に合わせるためのカードのことです。

同様に、カンバン方式のタスクスケジューリングボードは、次のタスクを何時しなければならないかを示すことで、仕事を「ジャストインタイム」で行えるようにします。

ここに、スクラムボードとカンバンボードの大きな違いがあります。カンバンボードは、進捗状況だけでなく、**フロー**（流れ、仕事がシステムの中でどのように進行していくか）を追跡します。

## 10.5 なぜフロー（流れ）が重要か？

仕事の正確なフロー（テンポ）を把握することは、無事に仕事を終わらせることと同じくらい重要です。事実プロセスにフローがなければ、最適なパフォーマンスを得ることはできません。例えば、あなたの街の信号機が一晩で全て撤去されたとしたら、交通のフローはどうなるでしょうか？

交通量は変わらないのにフローに大きな影響が出て、最終的にはすべての流れが徐々に止まってしまいます。

カンバンは、プロセス全体のワークフローを最適化することで、プロセスの全体的な予測可能性、効率性、有効性を向上させることを目的とした手法です。

スクラムは経験的プロセス制御理論に基づいていることを忘れてはいけません。経験的プロセス制御の中心となるのは、透明性のあるプロセスにおける検査と適応の頻繁なサイクルを回すことです。カンバン方式は作業結果の品質だけでなく、作業がシステムやプロセスをどのように流れるかについて、定期的な検査による改善サイクルに特に焦点を当てています。

カンバンボードは、作業内容や作業の流れを検査するために必要な透明性を提供します。チームは、品質の高い成果を達成するために、作業や作業方法を頻繁に適応させ変化させることができ、それを効果的かつ効率的に行うことができます。

さて、スクラムボードの話に戻ります。図17で、Sanjay が複数のタスクを抱えていることに気付きましたか?もし Sanjay が1つのタスクを午前中に、もう1つのタスクを午後に簡単に終わらせることができれば、大きな問題にはなりませんが、もしこれら両方のタスクが1日以上かかるとしたらどうしますか？

テスト作業が Sanjay の前で積み上がり、次の作業ができなくなるということです。Aviaja は、Sanjay がテストを完了するまで待たなければ、ソフトウェアをデプロイできません。

このシナリオでは、Sanjay がボトルネックとなり、他の人が Sanjay の前や後にどれだけ仕事をしても関係ありません。チームの最終成果物は、システムのボトルネック（この場合は Sanjay）によって決定されます。従って、彼が他のチームの仕事のテンポと合わせてテストを完了することは、スクラムボード全体の仕事の流れを最大化するために不可欠となります。

このような状況下では、Sanjay はボトルネックを作りたくないために、2つのタスクの間でタスクを切り替えながら対応しようとするのが普通です。タスク切り替えは一度に1つのことに集中できなく、作業を確実に行うことができません。2つのことを終わらせる代わりに、2つのことがうまくいかないか、全くできないかのどちらかになってしまいます。

注：リーンやアジャイルでは、タスクの切り替えを無くすことが基本原則です。

もし、Sanjay がこのチームで唯一のテスターだとしたら、Dev/Test の列に同時に1つ以上のタスクが入らないように仕事を計画しなければなりません。

そうすることで、Sanjay は1度に1つのことだけに集中し、それをうまくやり遂げることができます。彼の忙しいテストが完了し Aviaja に引き渡した後に、別のタスクを始めるようにします。そして、Aviaja がそれをデプロイしユーザーが利用できるようになります。

このような、仕掛（WIP）に制限をかける考え方を、仕掛制約（WIP-limit）といいます。1度に実行されるテストの量を制限することで、システムの全体的なデリバリー速度が速くなります。

もし、Sanjay が唯一のテスターであれば、DevTest の列の仕掛制約（WIP-limit）が 1 になり、それがスクラムボードの Test 列に適用され、システム全体がうまく働くことになります。さもなければ、もう 1 人のテスターがチームに追加されなければならず、そうすれば仕掛制約（WIP-limit）を 2 に増やせます。

実際のところ、仕掛制約（WIP-limit）が上限に達した場合、何人かの開発者が開発作業を中止することになります。

誰かが作業を止めることが良いことであると言うのは、直感に反しているように思えるかもしれませんが、実際、作業中のものを止めることが認められています。一旦その作業を止めると、新しい作業に着手してはなりません。それは、単にボトルネックを悪化させることになるからです。確かに、時にはチームメンバーが作業をしていないことになるかもしれません。しかし、彼らは他の人を助けて、その仕事を終わらせるための支援をいつでもできます。つまり、作業が再び流れを取り戻すようにすることがより重要です。（この場合では、Maria が自分自信の仕事をしていない限り、Maria は Sanjay のテスト作業を手伝うこともできます。）もしも、そのメンバーがボトルネックを解決するためのスキルを持っていない場合でも、その方法を学び始めることができ、新しいコアスキルを身につけることができます。

要するに、仕事が流れるように計画する際には、ボトルネックが発生することを避けなければなりません。スクラムのベースとなっているリーンの主要な目的の 1 つはムダをなくすことです。再び交通機関の例を使用してみましょう。信号機は交通を止めたり動かしたりします。本質的には信号機は交通を遅くするものですが、そうすることで交通の流れが良くなり、結果的に交通の流れが速くなります。

どのような仕組みになっているのでしょうか？

## 10.6　フローの理論を理解する

スクラムチームが、作業をより良く迅速に行うために測定すべきカンバン（リーン）の指標は 5 つあります。それらは[11]、

1. **仕掛（WIP）**：仕掛とは、開始されたが完了していない全ての作業のこと。なぜ作業がまだ完了していないのかは重要ではありません。重要なのは、仕掛（WIP）が完了する前に他のタスクを完了または開始してはならないということです。なぜならば、開発者が進行中の作業項目に依存し、または仕掛（WIP）に取り組んでいるからです。
2. **サイクルタイム（Cycle time）**：作業項目が開始されてから終了するまでの所要時間のこと
3. **作業経過時間（Work item age）**：作業項目が開始されてから現在までの時間のこと。まだ進行中の作業項目にのみ適用されます。
4. **スループット（Throughput）**：単位時間あたりに完了した作業項目の数のこと
5. **サービスレベル期待値（Service Level expectation：SLE）**：スクラムチームのワークフローの中で、ある項目が最初から最後まで流れるのにかかる予測時間のこと

バリューストリームマップピングを行ったことがあれば、これらの指標は馴染み深いものでしょう。スクラムの実践者としてリーンを学ぶことを強くお勧めします。これにより、スクラムを上手く使えるようになり、スクラムという特有の方法で非常に多くのことが行われる理由について、理解を深める事ができます。

では、なぜこれらの指標が重要なのでしょうか？

[11] Scrum.org., Vacanti, D., & Yeret, Y.（2021）. The Kanban Guide for Scrum Teams. Scrum.org. https://www.scrum.org/resources/kanban-guide-scrum-teams

## 10.6.1　リトルの法則

フローを理解するには、次のような「リトルの法則」を知らなければなりません。

$$\text{average cycle time} = \frac{average\ work\ in\ progress}{average\ throughput}$$

これは何を意味するのでしょうか？

**簡単に言えば、ある時点で多くのことに取り組めば取り組むほど、全体の作業を終えるのに時間がかかることを証明しています。**

論理的には、これは理にかなっています。常にタスクを切り替える場合、つまり進行中の作業項目が複数ある場合、1つのタスクを完了するのに時間がかかります（作業経過時間）。これに加えて、あなたは集中力を失い、その結果、（スループットが）低下するだけでなく、うまく機能しなくなります。

カンバンは、仕掛をよりよく管理することで、スクラムチームがサイクルタイムを短縮し、結果としてデリバリースピードを向上させるのに役立ちます。

サンプルのスクラムボードの例では、Sanjay だけがテストを行い、Aviaja だけがデプロを行えると推測され、これらの列は両方とも仕掛制約（WIP-limit）が1である必要があります。

開発「Dev」の列の仕掛制約（WIP-limit）は、Mubai、Sponono、Yuri、Maria の 4 である必要があります。ただし、4 人でコードを開発できる場合でも、開発されるコードの量は、テストやデプロイができる量を超えてはなりません。これがボトルネックに対処することであり、これによりシステム全体が早くなります。

開発作業は通常テストよりもはるかに時間がかかるため、過去の履歴データを利用して、開発「Dev」列に設定された理論上の仕掛制約（WIP-limit）を微調整することができます。実際には、「開発」の仕掛制約（WIP-limit）は 3（いつでも開発ができる人より1人少ない）である可能性がありますが、作業を行なうために利用できる（人的）リソースの数を超えることはありません。

Sanjay と Aviaja が実行可能な作業の速度を決定する場合、これはプルシステムと呼ばれます。プルシステムでは、前のワークステーションで作業が完了した時、次のワークステーションでそれを引き受けることができる場合にのみ、作業が実行されます。

プルシステムは、ワークフローの特定のステップの前に作業結果が蓄積するのを防ぎます。

チームが理想とするのは、リーンでいうところの「1個流し：シングルピースフロー」です。チームが目指すのは、スプリントのワークフローにおいて、どのステップの前でも停滞や待ち時間がなく滞りなく流れるようにすることです。このことは、スプリントのワークフロー作業計画を、スプリントプランニングの一部として行うことを意味します。

仕掛制約（WIP-limit）とフローの計画を加えることで、スクラムボードをカンバンに変えたことになります。

また、カンバンによって、スクラムチームはスキルと依存関係を明確に把握でき、また、自分たちに必要なスキルのバランスを図り、仕事を最適化させることができます。スクラムボードやカンバンボードで仕掛制約

（WIP-limit）を使うことで、システムの潜在的なボトルネックがすぐに明らかになり、チームはフローを最適化する方法[12]により作業を適応させることができます。

スクラムチームで通常起こることは何か、そして、どんな適応が求められるのか、それは、チームメンバーがお互いから学びながら機能横断的なスキルを身につけ始めることです。開発者は、機能横断的なチームで働くことにより、T 字型のプロフェッショナル[13]に変わっていきます。つまり、将来的に開発者がテスターになり、役割を変えることでシステムのボトルネックを解消できることを意味します。ただし、開発者が自分自身のコードや結果をテストし検証することはできません。その点は注意ください。

とはいえ、これは長期的な解決策です。即効性のある解決策としては、テスターの人数を増やせないか、テストの自動化ができないか、CI/CD パイプラインを構築できないかを検討し、Sanjay と Aviaja がタスクの切り替えなしに、1 日でできることを増やすことです。

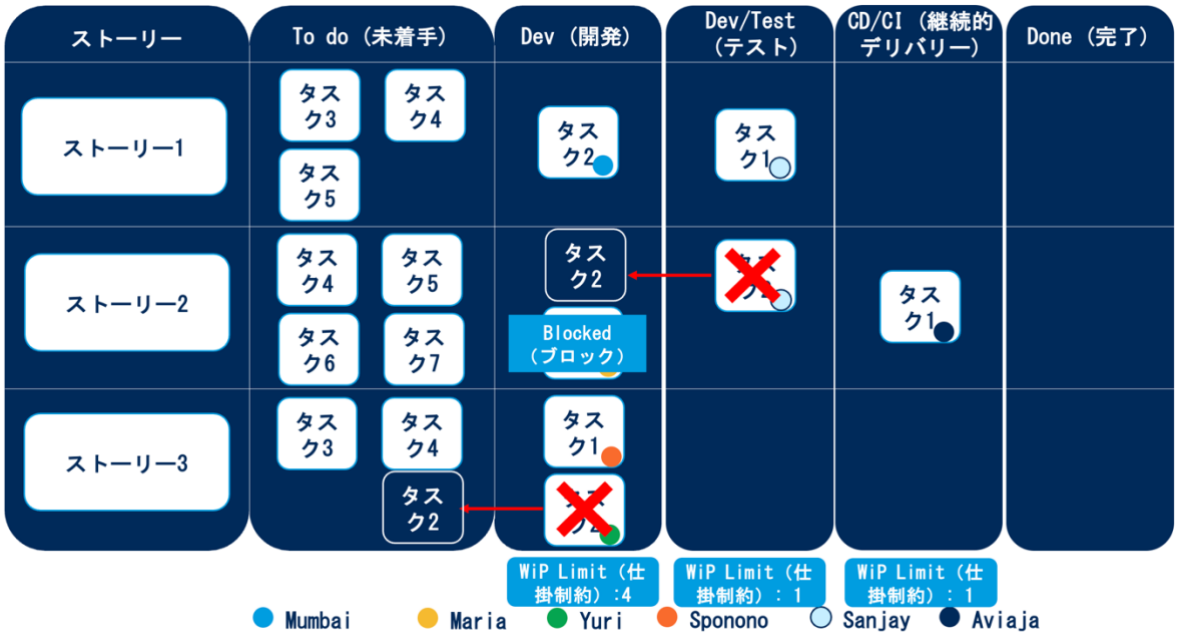
また、短期的な解決策としては、**スウォーミング**と呼ばれる行動があります。これは、1 人または複数のチームメンバーが自分の作業を中断して、フローを妨げるものを取り除くというものです。

## 10.7 カンバンの手法で、スクラムボードはどう変わるのか？

スクラムボードとカンバンボードの主な違いはその観点（興味・関心）です。スクラムボードは実行中の作業を視覚的に表現したものであり、カンバンボードはフローを最大化するように作業を管理するためのツールです。

最も基本的な形としては、スクラムボードに仕掛制約（WIP-limit）を導入するだけで、すでにカンバンの中心的な原則を実現したことになります。

*図 18 WIP-limit によるボトルネックの回避とプルシステムの構築*




図は EXIN 制作：Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

[12] 制約条件の理論については、エリヤフ・ゴールドラット博士の代表的な著書「*The Goal*」（1984 年）に詳しく書かれています。

[13] T 型プロについての詳細は、https://www.exin.com/about-career-path-certifications

前述の仕掛制約（WIP-limit）が正しい方法であるとすれば、ボードは次のようになります。（図 18 参照）

ストーリー2のタスク2は、テスト（Sanjay）に引き渡すべきではなく、その結果、Dev 列に戻されます。このタスクが Dev 列を占めると、開発（Dev 列）の WIP-limit が 4 であるため、他の1つの開発タスクは開始すべきでありません。

Yuri は、彼に割り当てられたコードの開発を、まだ始めるべきではありません。では、彼は何をすべきでしょうか、何もしませんか？

正解は、Yuri は、同じ品質を維持しながら、Sanjay がタスクをより速く実行できるようにどんな支援ができるかを考えるべきです。彼らは協力して、チームのコミットメントに集中する必要があります。そのスキルがないのであれば、実際には何もできないので、作業システムの中で仕事をしたほうがいいでしょう。

この例では、Yuri にできることは他にありませんが、ほとんどのスプリントでは、Yuri がボトルネックに依存しないタスクを見つけて、それに取り組むことが可能です。

スクラムでは、開発者はインクリメントを提供するための作業システムの改善にも取り組むべきだと推奨しています。要するに、スクラムでは、あらゆるスプリントにおいて継続的な改善が計画された作業の一部であるべきだと推奨しているのです。ここの例では、もし Yuri がテストを手伝うことができないのであれば、Sanjay のテストスキルに依存しない、改善ストーリーに属する活動から始めるべきです。

これにより、スクラムチームは、作業項目がワークフローを離れるのとほぼ同じ速度でのみ、作業項目がワークフローに取り込まれ、その結果として作業項目が不必要に長く時間を経過することがないようにします。

ボトルネックが発生した場合、チームメンバーは、ブロックされ、またはキューに入れられた作業項目について、チームが予想したサイクルタイムレベル（**サービスレベル期待値：**SLE）を超えている作業項目と同様、迅速に対応することができます。

スクラムチームは SLE を使って、常に変動するフローの問題を見つけ、その期待値を下回った場合には検査して適応します。

SLE は2つで構成されます：所要時間の予測期間、通常は日数、そして、予測期間に関連する見込み（例えば、70%のタスクがスプリント 7 日目までに完了するだろう）。

SLE は、スクラムチームの過去のサイクルタイムに基づいて算出する必要があり、算出したら、当該スクラムチームはそれを書き留め、誰もが見られるようにしておく必要があります。スクラムボードには、SLE のほか、プロダクトゴール、スプリントゴール、完成の定義（DoD）のための場所を残しておくことをお勧めします。

もし、新しいチームのため、これに基づく過去のデータがない場合、スクラムチームはベストの推測を行います。

## 10.8　ブロック項目とは？

上の図のストーリー2のタスク 3 が**ブロック**されていることに気付いたと思います。

通常のスクラムボードでは、作業項目の進行を妨げる問題を指摘することをお勧めしますが、スクラムとカンバを組み合わせる場合は、これらを使用することが特に重要になります。

上の例では、依存関係または問題により Maria はタスクを完了できません。タスクが開始されたため、To-Do 列に戻すことはできません。ここで、チームは Maria が依存関係に対処するのをどのように支援するかを考えなければなりません。依存関係が内部的なものであれば、チームは作業の順序を調整して

変更し、Maria のタスクを完了できるようにするか、あるいは、Maria が原因を理解し、問題の根本的な原因を修正できるように、チームメンバーが問題に集まり（スウォーミング）支援します。

依存関係が外部にある場合、スクラムマスターは通常、外部の関係者と問題を取り上げて解決を試みます。

ブロックされた状態の作業項目（ブロッカーチケット）は、前の列に戻れないと以前に述べましたが、実際のところ何もできない場合、チームはプロダクトオーナーと相談して次のことを決定することがあります。スプリントの目標を達成できる場合に限り、タスクをスプリントバックログからプロダクトバックログに移動します。または最後の手段として、スプリントの後半で取り組むことができることを期待し、To Do 列（他の列ではない）に戻すことができます。

もし、チームが依存関係や問題の原因について何もできない場合、つまりスプリントの目標と完成の定義（DoD）を達成できない場合、スプリントは失敗し、作業項目はプロダクトバックログに戻されることになります。

## 10.9 スクラムボードとカンバンボードの比較

下の表は、スクラムとカンバンを使った計画のアプローチがどのように異なるかを示す例です。

*表 2 従来のスクラムボードとカンバンの違い*

| スクラムボードの使用 | カンバンボードの使用 |
|---|---|
| 優先順位に基づいて作業を行う | バリューストリームの流れを最適化するために作業を行う |
| タスクは通常、スプリントの途中で追加されることは無いが、多くのタスクが進行中である可能性がある | 作業を完了する能力によって、タスクに取り組むか否かが決まる |
| プッシュ式があり得る | プルシステムのみ |
| | |

## 10.10 カンバン方式の活用

ボードだけでなく、かんばん方式を使用して製品やサービスの形で価値を提供することは、スクラムと同じではありません。2つの方法にはいくつかの基本的な違いがあります。

上記2つのボードの違いに加え、カンバン方式を使うことで次の違いが出てきます。

- アプローチ
- 役割
- 報告
- 働き方

次の表は、両方の方法について追加の違いを示します。

*表 3 スクラムとカンバンの違いの詳細*

| スクラムの利用 | カンバン（方式）の利用 |
|---|---|
| タイムボックス型イベント（スプリント、デイリースクラム） | タイムボックスがない：チームはタスクをその都度処理 |
| 非常に具体的な説明責任（プロダクトオーナー、スクラムマスター、開発者） | 決められた役割や説明責任無し：必要に応じて割り当てられる |
| スプリントでの作業は通常、スプリントバックログに限定 | 分割されたイテレーション・バックログはない |
| タスクは通常、スプリントの途中で追加されない | 新しい顧客要求から、新作業項目を継続的に作成可能のため、タスクはいつでも変更可能 |
| チームは、どれだけ作業が行われ、どれだけ残っているかでパフォーマンスを測定できる（ベロシティチャート、バーンダウンチャート） | 仕事は時間枠で設定されていないため、何が行われたかを測定することの方がより意味がある（累積フロー） |
| スクラムチームは、スプリントの終わりにスプリントレトロスペクティブを開催し、良かった事、悪かった事、今後の改善を話し合う | イテレーションの後に、議論するための個別イベントは無い。しかし、リーン環境のチームは、通常、同様のことを行う（カイゼン） |
| | |

## 10.11 スクラムとカンバンの併用で、スプリントのタスクはどのように変わるのか？

### 10.11.1 スプリント

カンバンをスクラムと一緒に使用する場合、スクラムのルールが優先されます。それは、スプリントの周期が以前と変わらないということです。2週間のスプリントで作業していた場合は2週間のスプリントを、4 週間のスプリントの場合はそのように続けることになります。

### 10.11.2 スプリントプランニング

フローをベースとしたスプリントプランニング会議では、スプリントバックログを作成するためにフローメトリクス（指標）を使用します。過去のスプリントでのスクラムチームのスループットが、次のスプリントの作業計画をするためのチーム能力として使用されます。

これは、ベロシティメトリクスによく似ているようですが、その通りです。しかし、今は代わりに SLE の方式を使っています。

### 10.11.3 デイリースクラム

デイリースクラムは、開発者のミーティングであることを忘れないでください。プロダクトオーナーとスクラムマスターは通常出席しますが、相談されない限り会議での議論に影響を与えることはしません。これはカンバンを使用している場合でも同様です。

デイリースクラムでは、作業自体を完了させることよりも、作業がシステムを流れるようにすることが焦点になります。ボトルネックが、デイリースクラムで対処しなければならない障害の主要なカテゴリーになります。

スクラムチームは、何に取り組んでいるのか、その日のうちに何が完了するのかを理解することに加えて、次のことについても話す必要があります。

- **仕掛制約（WIP-Limit）の制限のいずれかが破られましたか？** もしそうなら、状況はどのように修正されますか？具体的には、ブロックされた作業項目を解除するにはどうすればよいですか？
- **どのような作業が予想よりも遅れて流れていますか？** ここでチームは、進行中の各アイテムの作業項目経過時間、作業項目が SLE を超えたか、または超えそうか、また、その作業を完了させるためにスクラムチームは何ができるのかを確認しなければなりません。
- ボードに**表示されていない要素や作業項目**で、スクラムチームが今日の業務遂行能力に影響を与えるものはありますか？
- **今日学んだ教訓のいずれかが、**明日予定される作業に**影響を与えますか？**計画された作業の順序または優先順位を変更する必要がありますか？
- スクラムチームが次に取り組む計画を変えるような、**何か新しいことを学びましたか**？

### 10.11.4 スプリントレビュー

カンバン方式を使う場合でも、スプリントレビューは変更する必要はありません。

カンバンのフローメトリクス、特にスループットのレビューは、新たな会話の機会を生み出す可能性があります。スプリントレトロスペクティブで詳細に話し合うと良いでしょう。

### 10.11.5 スプリントレトロスペクティブ

フローメトリクスとその分析結果を検査することで、スクラムチームがプロセスをどのように改善できるかを判断することができます。また、スクラムチームは、次のスプリントでフローを最適化するために、「ワークフロー」の定義を検査し適応させる必要があります。

チームは累積されたフロー図を使用して、チームの仕掛（WIP）、平均サイクルタイム、平均スループットを可視化できます。これは、プロセスを改善するため、またはチームが次のスプリントで実行する作業量を計画するときに使用できます。

ただし、ここで説明した活動は、スプリント期間中の検査の一部であり、レトロスペクティブの時だけではないことに注意してください。

### 10.11.6 インクリメント

スクラムでは、チームはスプリントごとに少なくとも 1 つのデプロイ可能なインクリメントを作成することが求められます。スクラムの経験主義は、スプリント中に複数のインクリメントを作成し、スプリントの終了時だけでなく、チームによって作業が完了したらこれらのインクリメントをデプロイすることを推奨しています。

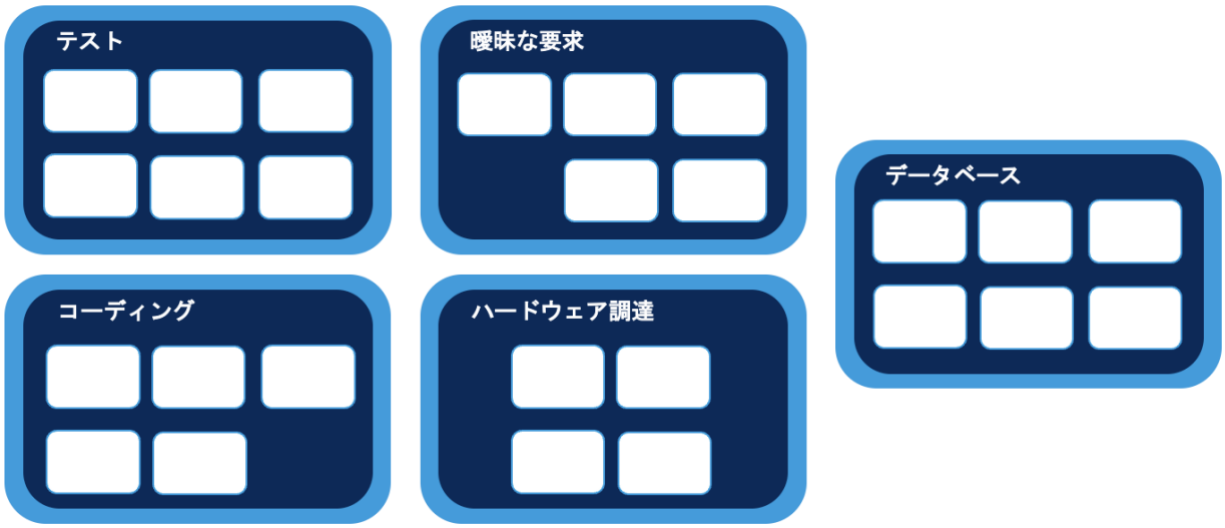
一定のフローを獲得することは、顧客やユーザーがスクラムチームの活動から継続的に利益を得ることを意味します。

## 10.12 良いスクラムボードには他に何が必要か？

スプリントボードやカンバンボードは、スクラムチームが集まる場所に設置された情報ラジエータの一つに過ぎません。スプリントボードやカンバンボードに加え他のツールを使用して、全てのアーティファクトや関連する作業を可視化し、そうすることで、良好なコミュニケーションと調整、およびスプリントの実行を促進することができます。これらのツールの中には、スプリントゴールやスプリント中に作業されるインクリメント、

および関連する作業と完成の定義（DoD）やブロッカーチケットの記録など、すでに説明したものもあります。しかし、見える化するとチームに役立つ情報は、それだけではないかもしれません。

*図 19 KJ 法を用いたブロッカーチケットのグループ化*




図は EXIN 制作：Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

ブロッカーチケットとその使用方法についてはこれまで述べてきましたが、障害が取り除かれた後にブロッカーチケットを廃棄しないようにすると、チームにとって非常に有益です。解決した課題や障害を記録しておくことは、スプリントレトロスペクティブを行う際に非常に有用です。スクラムチームの中には、解決されたブロッカーチケットを後で（レトロスペクティブ中に）参照できるように「留め書き」するエリアを確保したり、スプリント中に多くの問題が発生した場合には、アフィニティダイアグラム（親和図式）を使って論理的に関連するテーマに整理したりするチームもあります。

アフィニティ・ダイアグラム（親和図式）は、日本の文化人類学者である川喜田二郎氏の名前をとって「KJ 法」とも呼ばれており、構造化されていない曖昧なデータ（この場合はブロッカー・チケット）を整理するために考案されたグラフィック・ツールです。ブロッカーチケットを、「親和セット：似通ったもののグループ」と呼ばれる意味のあるカテゴリーに分類します。これらの親和セットは、ばらばらで異なるチケットを根本的なテーマに集めて結びつけることで、問題を明確にし、1つまたは複数の解決策を体系的に分析するための構造的な物を提供します。

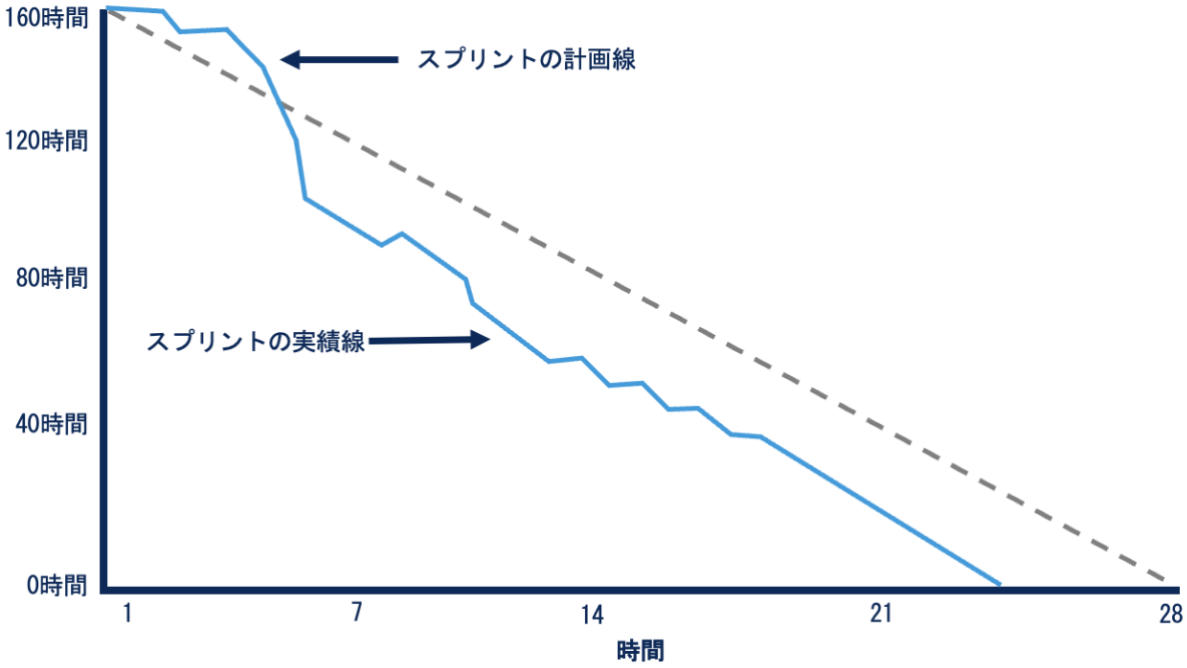
## 10.12.1 バーンダウン/バーンアップチャート

バーンダウンチャートやバーンアップチャートは、進捗状況を把握するためのビジュアルマネジメントツールですが、スプリント内での作業速度であるベロシティを測定することもできます。

スプリントは、通常2週間から 4 週間の一定のタイムボックスで完了できるように計画します[14]。アジャイルでは、スプリントの開発予算と割当時間（タイムボックス）を一定にして守らなければなりません。変化するものは、どれほどの価値を提供できるかです。完成の定義（DoD）とはこれを考慮しているので、あるスプリントでは少なくとも1つの完成の定義（DoD）から生まれる成果を提供しなければなりません。

[14] 実際には、もっと長いスプリントに遭遇することもあります。しかし、スクラムガイドによれば、スプリントの最長期間は 1 ヶ月です。

*図 20 典型的なバーンダウンチャート*




図は EXIN 制作：Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC – Kdp Print Us.

タスクが完了して見積もりと照合することで、残りの作業量が計算されてチャートに表示されます。その傾斜角度として、残りの作業量がゼロになるように下向きにする（バーンダウンチャート）場合と、2週間から 4 週間の間にいくつのタスクが完了したのかを上向きで示す（バーンアップチャート）場合のどちらかになります。

殆どのユーザーは、どれだけの作業が完了したかよりも、決められたタイムボックスに対してどれほどの作業が残っているかに関心があるため、バーンダウンチャートを選びます（カンバンボードは、いずれにしてもそれを示します）。

計画時には、タスク数を用いて 0 日目から28 日目までの直線でチャートテンプレートを作成します。ここでは、4 週間のスプリントで残りのタスクが 0 になるという例を示しています。この直線は、タスクが完了するまでのベロシティを表しています（点線はベロシティの見積もりを表しています）。

実際の進捗状況を、毎日追跡してグラフ化をします。線が点線の上の方にあれば、スプリントが計画通りに進行していないことを意味し、チームとして対処しなければなりません。日々のプロット線が点線の下の方にとどまっていれば、スプリントは予定通りに完了し、計画したものをすべて提供できます。

## 10.12.2 ベロシティとは?

ベロシティとは、スクラムチームが 1 回のスプリントで完了できる作業量を示す指標です。これには業界標準のようなものはありません。ベロシティは、スプリントの終わりに、完全に完了したすべてのユーザーストーリーのポイントを合計し、以前に記録された合計と比較することで計算されます。未完成の作業は、ベロシティを測定する際にカウントされたり、考慮されたりすることはありません。

チームのベロシティを知ることは、スプリントプランニングの貴重な情報となり、チームがスプリント内で完了できないほど多くの作業にコミットしないようにします。

いかなるスプリントにおいても、チームのベロシティとは、どのような作業をしているのか、また、その作業がチームのスキルや能力にどれほど適合しているかによっても異なることに注意しましょう。

チームのベロシティの傾向を評価することは、スプリントレトロスペクティブの際の貴重な情報となり、改善点を見出すのに利用できるかもしれません。

一般的に、チームのベロシティは時間の経過とともに増加するはずであり、ここで使用する適切な指標は、各スプリントで10%の改善です。経験の浅いスクラムチームでは、スプリント中にベロシティが予想よりも低いことが明らかになるのが一般的です。このような場合には、速度を上げる方法を特定するか、それが不可能な場合には、速度を予測し直すことが推奨されます。

プロダクトオーナーは、ベロシティの変化、予測可能性の向上、柔軟性などを考慮して、スプリント後にプロダクトバックログアイテムの順番を並べ替えることを決めることもあります。

### 10.12.3 ベロシティと SLE の違いは何か?

先に紹介したサービスレベル期待値（SLE）の概念は、バーンダウンチャートやバーンアップチャートに描かれた理想的なベロシティに似た予測指標です。

SLE がベロシティと異なる点は、予測が、1つのスプリント全体で行う作業量に基づくだけでなく、行うべき作業の複雑さのレベル、タスク完了までの時間を考慮している点です。バーンダウンチャートでは、完了した活動のみマークできることを忘れないでください。5 日かかるタスクが 80%完了していても、バーンダウンチャートには表示されませんが、これでは進捗状況やベロシティの見方が歪んでしまいます。

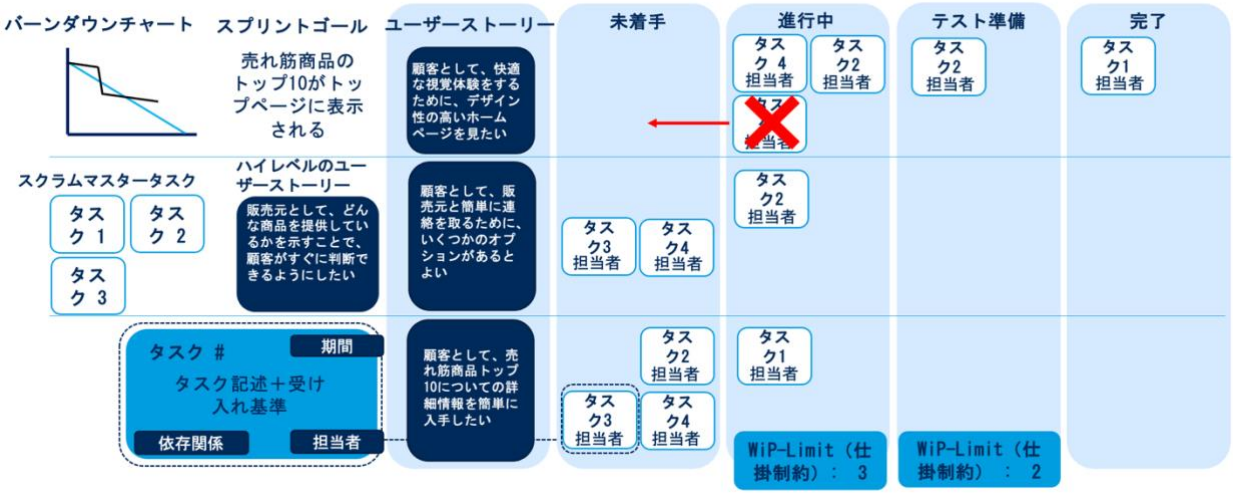
SLE を使ってバーンダウンチャートに理想的な進捗ラインを描くことで、進捗状況をより正確に反映させるチームもあります。

## 10.13 情報ラジエータと集合場所

情報ラジエータは常設され、チームメンバー全員が常に目にすることができるようにし、チームの活動はすべてこの周りで行われるべきです。

スクラムチームが集まる場所の背後の壁には、次のようなものをよく目にします。

*図 21 典型的なスクラムチームのスペース*




図は EXIN 制作：Botha、J.（2020）. *Scrum Lego-game workbook [コースウェア].* GetITright.

典型的なスクラムチームのスペースには、スクラムボードの他にも情報ラジエータがあります。また、ここでは、バーンダウンチャートがあり、集めた障害を JK 法を使って分析を始めたスプリントレトロスペクティブを見るかもしれません。

# 11 原則がパターンかアンチパターンを決める

本書の前バージョンでは、この章は従来のスケーリングアプローチ、特にスクラム以外のアジャイル手法でよく推奨されるアプローチに焦点を当てていました。

この章では、スクラムの「スケーリング」方法について説明します。

しかし、組織がアジャイルやスクラムをスケールすると言うとき、何を意味するのでしょうか？

一般的に、スケーリングという言葉は次の 2 つの意味で使われます。

一つ目は、組織が大規模で複雑なプロジェクトを、そのまま大規模で複雑なプロジェクトとして運営しようと考えている場合に使われます。それを行う唯一の方法は、従来のプロジェクト管理手法を採用することです。しかし、このアプローチには問題があります。そのような方法を採ることで、実際にはアジャイルでなくなってしまうからです！

もう一つは、組織におけるスケーリングとは、経営陣が組織全体をアジャイルにしたいと考えている場合に使われます。これは良い目標ではありますが、このアプローチは、現在のプロセスを「アジャイル」なプロセスに置き換え、現在の組織構造を「アジャイル」な組織構造に変更することで達成しようとします。しかし、このような大規模なチェンジマネジメントの取り組みによって組織をアジャイルにしようとする方法は、決してうまくいきません。なぜなら、アジャイルを行う（Doing Agile）のではなく、アジャイルである（Being Agile）必要があるからです。このアプローチは、アジャイルでない行動が強化され、組織の人々は、考え方や行動を変える必要はない。ただ単に、新しいやり方で今まで通りのことをやるだけ、と認識してしまいます！

上記の例はどちらも、私たちがアンチパターンと呼ぶものであり、スクラムのスケーリングを行う際に、アジャイルの 12 の原則を無視、または矛盾するような方法を取ることは、アジャイルの成功を妨げる要因になります！

本書のこのバージョンでは、この特定のアンチパターンへの言及は削除し、しかし、よくあるアンチパターンについて説明することで、どのようにすれば、組織にとって有益なアジャイルパターンを確立し、アジャイルな組織としての能力を高めることができるのか、この点を説明することが価値あると考えます。

アンチパターンについては多くの研究がなされてきましたが、Jon Smart 氏が『Sooner Safer Happier』を出版するまで、決定的な参考文献はありませんでした。この章は Smart 氏の研究に基づいており、この重要なテーマについては、本章だけでなく、さらに深く学んでいただくことをお勧めします！

## 11.1 アンチパターンとは何か？

簡単に言うと、アンチパターンとは、組織が問題を解決しようとして行うことが、実際には状況を悪化させたり、アジャイルマインドセットやそれに伴う行動を促進しにくくすることを指します。

ここで明確にしておきたいのは、アンチパターンは決してダメなアイデアではないということです。それらはしばしば伝統的な産業化時代の考え方や管理慣行に基づいており、産業化時代のコンテキストでは十分に機能することが証明されています。しかし、それらはアジャイル環境では通用しないため問題になるのです。むしろ、多くの場合、組織がアジャイルになる能力を損なったり、完全に破壊したりしてしまいます！

そのため、私たちがアンチパターンとして指摘するものを読んだとき、あなたは「一般的な常識では、それは賢明な判断だ」と思うかもしれません。しかし、アジャイルのコンテキストでは、それが非常に悪いアイデアであることがすでに証明されているのです！

アンチパターンを取り除くのが非常に難しいのは、それらが長年にわたる学習や習慣に反するからです。

もしあなたが、本当にアジャイルを、あなたと組織のために機能させたいのであれば、アンチパターンに対して勇気を持ち、徹底的に排除する姿勢が求められます。

## 11.2 「良いアイデア」…ではないもの!

この章は、アンチパターン（およびパターン、つまりあなたがむしろ行うべき良いこと）を決定するためのガイドではなく、その概念を紹介するものです。

ここでは 6 つの代表的な例について説明しますが、他にも数多くあります。

では、どのようにしてアンチパターンを見分ければよいのでしょうか？

それらを見つける最も簡単な方法は、まず、それが12のアジャイルの原則と矛盾しているかどうかです。Smart 氏の著書では、アジャイルを成功させた組織を分析し、追加の原則も定義しています。

### 11.2.1 アジャイルの導入または変革を行うこと

アジャイル導入プロジェクトを実施する場合、通常はチェンジマネジメントプロジェクトとして実施されます。つまり、取り除きたいものを、それを取り除くために使っているのです。これはナンセンスです！（アジャイルになろうとしながら、そのために非アジャイル的なアプローチを使っているという意味）

そうではなく、理解し、教育し、新しい働き方を試すボランティアを募り、組織の免疫システム（古いルール）から彼らを保護してください。なぜなら、新しい働き方の多くは古いルールを破るからです。ルールがないと言っているのではなく、ルールは必要ですが、少数のルールと、新しい考え方に基づいたルールが必要です！

重要なのは、望む結果に焦点を当て、現在の指標ではなく、将来的に望むものに基づいて新しい指標（成功基準）を定義することです。

成功の測定方法（そして結果として人々の測定方法）を変えないと、確実に失敗となります。

Peter Drucker 氏は「測定されるものが実行される」と言っています。もし新しい行動を期待しながら古い行動を測定していれば、何を得られるかわかると思います！

成功させたいのであれば、最初に意欲的な参加者を集めて、小さく始めることで成し遂げます。

### 11.2.2 アジャイルスケーリング

（前のセクションで述べた）スケーリングの二つの例のうち、どちらがアンチパターンでしょうか？

答えは両方です。

アジャイルを行う（Doing Agile）変革を行うことは、行動を変えずに組織全体をアジャイルにすることです。

ここで、なぜエンタープライズ規模のプロジェクトがアンチパターンなのかを見ていきます。

大規模で複雑なプロジェクトは、しばしば水晶玉を持つ人々（非現実的な予言者）によって考案された壮大な計画です。理想的な世界ではうまくいくかもしれませんが、私たちが生きる急速に変化する世界では、大きなアイデアでさえ一夜にして時代遅れになります。長期の詳細な計画は矛盾であり、組織が価値を提供するためにウォーターフォールからアジャイルなアプローチに移行したい理由の一つです。

アジャイルマインドセットを持っていれば、私たちにできることは方向性を定め、長期的に（今や 1 年でさえ長期です）達成したい高いレベルの成果について話すことだけだと気付くでしょう。

どのようにそこに到達するかを、一歩ずつ、従業員が考え出すことを許すのが最善です。このアプローチには、非常に頻繁に方向修正を行い、常に組織の現在の状況に合わせて提供できるという追加の利点があります。実際には、アジャイルであることで、常に追いつく必要はなく、起こった時に未来を創造することができます。

事前に決定された確実性を持って、大規模で複雑なプロジェクトを実行しようとするすべてのスケーリングメカニズムは、非アジャイルな方法を使用します。

これが私たちの組織の現実であり、これが私たちの組織と業界で運営できる唯一の方法であると、あなたは言うかもしれません。Google、Cisco、Microsoft がそうしていることに驚くかもしれません。そして、「しかし、私たちは IT 業界にいる」と言う前に、Barclays、Boeing、John Deere、Airbus、Amazon、US Airforce、Panera Bakeries、Sony、Lego、KPMG なども同様です!

次に進む前に、最後に注意しておきます。私たちは、アジャイルメソッドが万能薬だと言っているのではありません。なぜなら、いくつかのシナリオではウォーターフォール手法がうまくいくからです。私たちが言っているのは、アジャイルであること、アジャイルで考えること、アジャイルに行動することは、進化のプロセスであるということです。

実際、これは次のアンチパターンに完璧につながります。

### 11.2.3　すべてに当てはまる唯一の方法を想定することの間違い

(今日の世界のような)変化の激しい環境では、「XXXX アジャイル」を定義しようとすることは大きな間違いです。「スター・トレック:ディスカバリー」のロルカ船長は「普遍的な法則は従者のためのものであり、コンテキストこそが王のためのものである」と言いましたが、私たちも同意します。

これは、組織を一晩で変革したいという一部のリーダーの願望にもつながります。

あなたのコンテキストによって最適な方法は何でしょうか？ それは状況によります。そして、組織を細かく見ていくと、あらゆるレベルで状況は異なります。だからこそ、自己管理と真の草の根レベルでのオーナーシップがアジャイルの基本原則であり、それらが重要なのです!

アジャイルを管理するシステムを構築し、それを会社全体に展開したくなる気持ちは理解できますが、それ自体が産業化時代の考え方です。あなたがどれほど努力しようと、あるいはどれほど賢明であろうと、実際にはそれはできません。なぜなら、それはアジャイルのモデルに適合しないからです。あなたは組織の人々にアジャイルを押し付けることはできません。なぜなら、押し付けることは定義上アジャイルではないからです!

では、どのようにして全体をまとめていけばよいのでしょうか?

Smart 氏は VOICE を使うことを提案しています

- Values:価値観と原則に焦点を当てる。
- Outcomes:目的をもってマネジメントし、結果を測定する。
- Intent-based:リーダーシップは意図に基づいて、模範を示しながら、誰もが各自の方法で成果をだせるための余地を残す。
- Coaching:イニシアチブ、実験、学習をコーチングしサポートすることが、マネジメントの新たな標準になる。
- Experimentation:実験こそが、この方法が機能するかどうかを知る唯一の方法です！ つまり、測定方法を変える必要があります。もし、あなたの評価方法が失敗を罰するものであるなら、人々が実験することを期待することはできません!

### 11.2.4　指示と制御

アジャイル組織におけるマネジャーの新しい役割は、イネーブラー（支援者）であり、コーチです。もはや「私の言うとおりにしろ！」という時代は終わりました。

今日の仕事のほとんどは知識労働です。あなたに報告する人々は、おそらくあなたよりも仕事についてよく知っており、彼らに何をすべきかを指示するのは理にかなっていません。さらに、彼らに指示された通りにやらせた上で、彼らに責任を持てというのは無理があります。

アンチパターンは互いに影響し合っていることに気づきましたか？ そして、このリストが次に進むほど、前のアンチパターンすべてに対処しなければならない可能性が高まります。

職場の心理的安全性は極めて重要です。人々は間違いを犯しますし、新しいことに挑戦することで、むしろ多くの間違いにつながります。これは、人々が実践を通して安全に学び、その学びに対して報酬を与える（注意：失敗に対して報酬を与えるのではない）、「失敗しても安全な」環境を作ることがマネジメントの仕事であることを意味します。

アジャイルにおけるマネジメントとは、すなわちサーバントリーダーシップなのです！

### 11.2.5　制御できていないと感じることによる複雑な制御の導入

従来のマネジメントの考え方では、アジャイル組織で意思決定が委譲されるにつれて、マネジャーはますます不安を感じます。その本能的な反動として、コントロールを取り戻すために、より多くのコントロール（より強い統制・管理）を導入しようとします。

これがまさに間違った対応です！

マネジメントがやるべきことは、管理の仕組みをシンプルにすることです。そして同時に、その管理を 透明化することが、先述の不安への適切な対応となります。

コントロールが適切であるかどうかを確認してください。産業化時代に適していたコントロールのほとんどは、それ自体がアンチパターンになっています！

（局所的な）品質、プロセス遵守、活動ではなく、フロー、価値、そして成果に焦点を当ててください。

実際のところ、良い仕事をしているかどうかについての最も重要な指標は、顧客が「今までで最高の価値を提供されている」と言えるかどうかです！

### 11.2.6　デリバリ―スピードに焦点を当てる

多くの組織がアジャイルの導入を始めるとき、より速くデリバリーできるようになるからだと言うでしょう！

しかし、速さというのは誤った指標です。なぜなら、悪い行動を助長するからです。素晴らしい顧客価値を提供する代わりに、ベロシティ（速度）が目標になります。

最近「このインクリメントは完了したが、完全に完了したわけではない」という会話を耳にします。これは、納期を守るために、品質が犠牲になったか、技術的負債が蓄積されたか、どちらかを意味します。

これは非常に危険な状態です！

次のことを覚えておいてください。アジャイルの概念はリーン運動から派生したものであり、リーンでは欠陥を次の工程に渡しません！ たとえ納期に間に合わなくても、最初から正しく作るのです。なぜなら、そうしないと、いずれは多大な負債を抱え込み、納期に間に合わなくなり、顧客に何の価値も提供できなくなるからです。

先ほど述べた通り、マネジメントは成果に焦点を当てるべきです。これがまさにその好例です！ もし納品物が基準以下なら、最終的な成果はどうなるでしょうか？

あなたは、納品したものが完璧であることに満足していますか？

では、このような状況に陥った場合、どうすればよいのでしょうか？

正しいことをして、完璧に仕上げるか、さもなければ、納品、販売、展開をしないか、どちらかです。これこそが、まさにトヨタの工場でアンドンが行っていることです！

# 12 Nexus とスケーリング

## 12.1 Nexus とは?

Nexus は、持続可能な方法で大規模なプロダクト開発を実行するためのフレームワークで、スクラムを基本構成要素として使用しています。

スクラムの説明責任、イベント、アーティファクト、ルールの使用法など、どんな定義変更も行っていません。Nexus はスクラムを変えるのではなく、スクラムに追加してます。

この意味で、Nexus はスケーリングされた開発フレームワークであり、Nexus スプリントは、統合された「インクリメント」を提供する多くのイテレーションによって構成された開発ユニットです。それによって、顧客に価値を提供するものです。

イテレーションやスプリントでは、すべての労力がそのスプリントの成果に集中し、これは Nexus スプリントの目標と完成の定義（DoD）により定義されます。

全てのスプリントで出荷可能なプロダクトを提供しなければならないとは言っても、多くの開発環境は複雑であるため、実際はそれが不可能な場合もよくあることです。

実際の顧客を満足させるためには、相互依存性や様々なチームのスキルセットを考慮しながら作業を順序立てて実行し、デリバリーしなければなりません。そのため、スプリントの完成の定義（DoD）が使えなくなることがしばしば起こります。つまり、顧客のニーズや製品・サービスがもたらす結果を考慮すると、インクリメントが出荷可能なものではなくなり、また顧客さえ理解できないものになる、という可能性があります。

**これは現実にはどういう意味なのか?**

多くの場合、複数のスクラムチームが同じプロダクトバックログから価値を提供します。ソフトウェア開発やその他多くのプロダクトにおいて、このことは同じ構築要素:building blocks（codebase）を使用することを意味し、これらの異なるチームの作業間で結果的に生じる依存関係や相互依存関係の問題に直面します。また、異なるチームから提供されるインクリメントは、顧客が価値を享受するために、一緒にまとめてテストしデプロイされなければならないことが多くなります。

さらに複雑なのは、スクラムチームが同じビル、都市、国に居ないことが増えていることです。連携されていない取り組みは問題を引き起こします。

チームは、自分たちの仕事が他のチームにどのような影響を与えるのか、他のチームが自分たちの成果物にどのように依存しているのか、仕事を完了しなかった場合にどのような結果になるのか、注意を払わなくてはなりません。さらに、それぞれのチームが開発したものの要求事項、使われ方、適用業務が、他のチームによって共通の理解がされる必要があります。

チームが増えれば増えるほど、複雑になっていきます。

このような依存関係があるため、スクラムをスケールさせるためには、様々な取り組みを調整し、要求を理解し管理する方法が必要となります。

そのための方法として、具体的には次の対応が必要です。

- **要求：**関係するすべてのチームのメンバー間で全体的な理解を確実にし、ある要求の作業が別の要求の作業にどのように影響するかを確認すること
- **ドメイン知識：**行うべき作業が、彼らの技術およびビジネススキル、知識および能力に基づいて、チームに割り付け（マッピング）されること
- **プロダクト、テストと統合、およびアーティファクトの品質保証：**構成部品を統一された成果物としてテストするための統一されたアプローチを確保すること

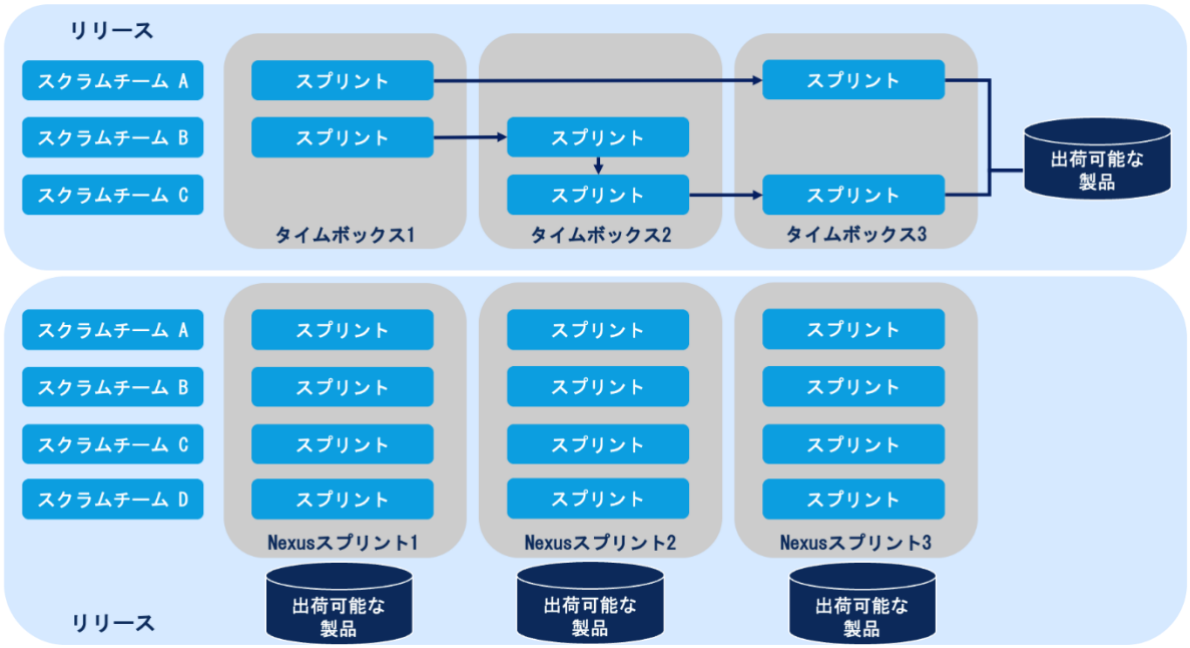
Nexus のようなアプローチを用いることで、割り当てられた作業がピンポン玉のようにチーム間を行き来することを防ぎ、また、作業の割り当ては、そのドメイン知識に依るだけではなく、関連し依存するアーティファクトの開発でチームが得る経験も考慮にいれて割り当てることになります。統合されたアプローチだけでなく、最適化されたアプローチが求められます。

Nexus は、基本的なフレームワークを提供し、詳細についてはチームの判断に任せることで、こうしたすべてを達成します。

## 12.2 Nexus とリリースアプローチの違い

Nexus は、これまでアジャイルリリースと呼ばれていたものと非常によく似ていますが、Nexus は一度に1つのスプリントまたはタイムボックスのみに焦点を当てているという点で根本的に異なります。これはリリースアプローチとの大きな相違点であり、そこから離れるためのポイントでもあります。プロジェクトの初期段階でリリースを計画することは、不完全な知識に基づいていることが明確に認識されています。そして、ウォーターフォールアプローチからの離脱を促したのもこの明らかな事実でした。

*図 22 Nexus vs リリースアプローチの比較*




図は EXIN 制作：Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

多くの人は次のように主張するでしょう。リリース計画は詳細なものではなくハイレベルなものであり、また、プロジェクト期間中に要求や理解の変化に合わせて変化するものだと。これは確かに真意であり、そうなると思われます。しかし、現実には、このアプローチは、殆どのプロジェクト参加者がウォーターフォールプ

ロジェクトで行ってきたこととよく似ているため、リリース計画の変更は推進者が想定または提唱するよりもはるかに少なくなります。

多くのリリースでは順番にイテレーションやスプリントが行われます。一方、Nexus では常に 1 つのイテレーションのみに焦点を当てます。人は両方のアプローチにメリットがあると主張します。

ただ、リリースアプローチはスクラムで必須なものではなく、他の手法から採用されたものです。一方、Nexus はスクラムのために設計されたものであることを覚えておいてください。

すぐ上の図を見ると、リリースはウォーターフォールプロジェクトで使われるガントチャートに、どことなく似ていることがわかります。

Nexus スプリントの一部であるスプリントは、1 つのイテレーションの一部でもあります。Nexus スプリントの一部であるイテレーションのアーティファクトを統合したものが、Nexus スプリント全体の完成の定義（DoD）の中で定義されます。

## 12.3 Nexus フレームワークによるスケールアップの新手法

Nexus は、既存のスクラム手法、説明責任、アーティファクトなどを使用するフレームワークと見なされますが、新たにスクラムに追加することで、ビジネス価値を提供する統合されたインクリメントの作成を行います。 従って、Nexus チームの焦点は、純粋なスクラムあるいは社内向けのリリースアプローチを併用したスクラムを使った場合よりも、依存関係、相互運用、ビジネス成果、そして創出される価値を重視することになります。

Nexus チームは、イテレーションごとに Nexus 全体として 1 つの成果を出します。

これを可能にするために、スクラムの説明責任を拡大する必要があり、新しい説明責任として Nexus 統合チームが作られます。Nexus 統合チームの役割は、Nexus とスクラムを適用するための調整、指導、指揮をすることで、顧客にとって価値ある成果を最大化することです。ただし、開発者の作業を監督するといったような自己管理の原則に反するようなことはしません。

Nexus 統合チームは、チームの新しい説明責任の定義であり、1 名のプロダクトオーナー、1 名のスクラムマスター、各スクラムチームから代表者 1 名が参加します。代表者とは、通常のスクラムチームの開発者を代表する統合チームのメンバーのことです。

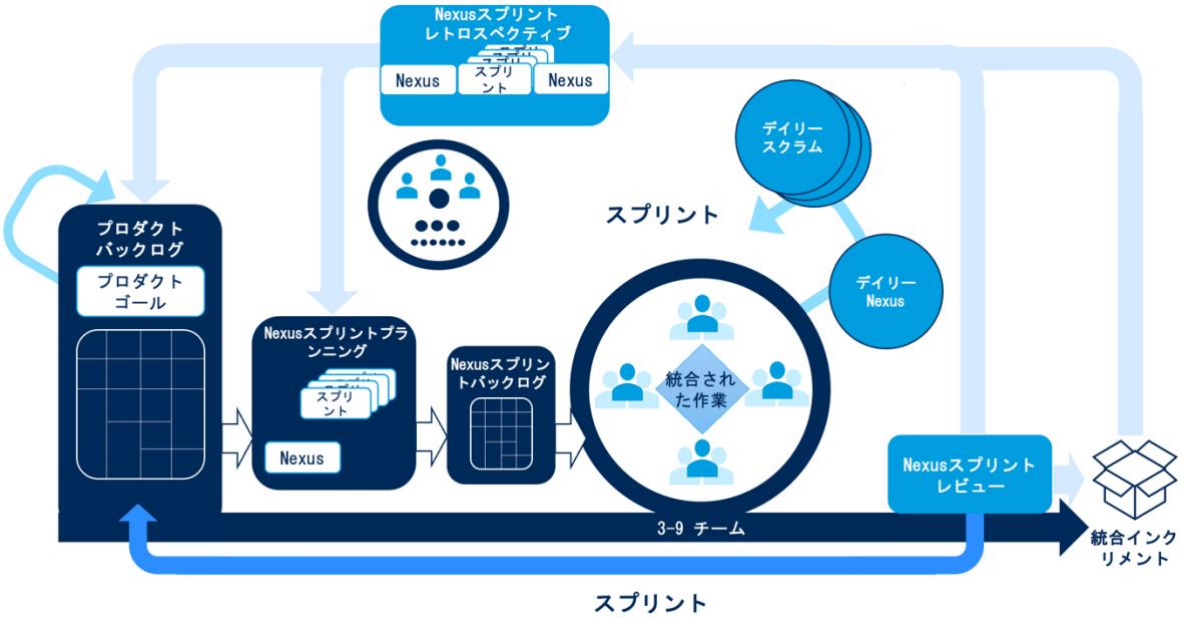
全てのスクラムチームは、同じプロダクトバックログに基づいて作業する必要があり、リファインメントの活動では、すべてのチームメンバーが次に作業すべき項目をよく見て理解している必要があります。また、アイテムの順序付けには、ビジネスニーズの即時性を理解するための視覚的な手段を使用することが最善です。

スクラムチームがアイテムを選択したら、通常の方法でスプリントを実行し、自分たちのスプリントバックログに取り組みます。

既存のスクラムイベントはどれも置き換えられません。その代わり、一部のイベントは、Nexus に参加している全てのスクラムチーム、あるいは少なくとも全てのチームの代表者が合同で行う活動になります。

次の図は、Nexus フレームワークで作られた外骨格の概要です。

*図 23 Nexus とスクラムの併用*




図は EXIN 制作：Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us. and Schwaber、K., & Scrum.org（2021）. *The Nexus™ Guide - The Definitive Guide to Scaling Scrum with Nexus*. Scrum.org. https://www.scrum.org/resources/nexus-guide

## 12.4 Nexus のプロセス

注：この部分は、「The Nexus Guide」からそのまま紹介しています。

Nexus スプリントにおける全ての作業は、その Nexus スプリントを構成するスプリントのメンバーであれば誰でも行うことができます。Nexus のスクラムチームの全てのチームメンバーは、より広範な機能横断的チームの一部として働きます。そのため、バックログアイテムに取り組むメンバーは、スキルセットと作業項目の相互依存性に基づいて選ばれ、不必要に作業分解することを最小限に抑えます。

Nexus の計画では、バックログアイテムをリファインし、それに取り組む可能性のあるチームをできるだけ早い段階で特定する必要があります。つまり、これはプロダクトバックログのリファインメントの際に追加される新しい活動です。

次に、Nexus スプリントプランニングと呼ばれる統合的プランニングセッションが開催されます。各スクラムチームの代表者が集まり、リファインされたプロダクトバックログを議論・検討し、各チームのためにバックログからアイテムを選択し、それらを各チームのスプリントバックログに並べます。また、結合された Nexus バックログを保持します。それにより、Nexus の成果をレビューし、統合されたレトロスペクティブを行うための統合的アプローチをファシリテートします。

結合された Nexus バックログを保持する具体的な理由は、依存関係や相互依存関係を明確に示すためです。

この記述から明らかなように、チームのスクラムマスターは代表者として理想的な候補ではありません。それは自己管理の原則やサーバントリーダーとしてのスクラムマスターの職務と矛盾するからです。

各チームは、必要に応じて他のチームと作業を統合しながら、それぞれのスプリントを計画します。各チームは、スプリントの目標が、Nexus スプリントプランニングで定義・合意された Nexus 全体の目標と一致するようにします。

開発とテストは通常のスプリントで行われますが、一部の活動はチーム間の調整が必要になる場合があり、依存関係がある場合には統合テストを行うことがあります。

従って、共有化されたテスト、統合プラットフォーム、アプローチが合意され維持されることが望ましいです。

チームはデイリースクラムを行いますが、全てのチームの代表者も毎日集まり、統合上の問題や障害を具体的に話し合い、その結果が各チームへ順番にフィードバックされます。なお、Nexus ガイドには、チームの代表者を常に同じ人にすると書かれていません。確かに、継続性の観点ではそのほうがメリットがあるかもしれません。それでも、Nexus で今発生している問題に対処できる最良の人が、その時点で代表者になるべきです。

したがって、Nexus のデイリースクラムは、個々のデイリースクラムの前に行われるべきであり、フィードバックとより良い気付きが全てのチームの日次計画の中に反映されるようにします。

スプリントレビューは Nexus として行われます。Nexus スプリントレビューでは、統合された Nexus の「完了」インクリメントをレビューします。また、プロダクトオーナーはこの情報をもとにプロダクトバックログを更新し、適切な調整を行うことができます。

Nexus のスプリントレビューは、各チームの代表者を集めて行われます。課題、問題、改善は各チームに持ち帰られ、各チームはそれぞれのスプリントを振り返ることになります。

その後、各チームの代表者がチームレトロスペクティブの後に再び集まり、学んだ教訓や全チームで取るべき行動について話し合います。

特記事項：最近のプロジェクト、変更、改善の取り組みは、複雑さが増しています。作業の自動化や複雑なワークフローの最適な管理方法についても検討することが賢明です。

## 12.5 Nexus の説明責任をより詳しく

Nexus 統合チームの説明責任は、イテレーションごとに統合されたインクリメントが生成されることを保証することです。スクラムチームを組み合わせることで Nexus ゴールを満たすインクリメントを開発します。

Nexus 統合チームは、以下のメンバーで構成されるスクラムチームです。

- 1 名のプロダクトオーナー
  - 1 つのプロダクトバックログと 1 人のプロダクトオーナーの存在
- 1 名のスクラムマスター
  - 通常、参加チームから 1 名
- Nexus 統合チームのメンバー
  - 参加チームを代表して各参加チームから 1 名以上

Nexus 統合チームのメンバーが個々のスプリントの一部として作業する場合は、Nexus 統合チームのための作業を優先しなければなりません。Nexus 統合チームの一員としてメンバーが行う作業は、常に個々のスクラムチームの作業より優先されます。これが重要な理由は、チームの連携方法に関する問題の解決、あるいは Nexus の成果に影響を与える可能性のある問題や多くのチームに影響を与える問題の対処、これらは関係者全員への影響やリスクのために優先されるからです。

Nexus チームのメンバー構成は Nexus ごとに必ずしも同じである必要はありませんが、Nexus から Nexus への作業の流れを理解するなど、Nexus 統合チームのメンバー構成を固定させるメリットはあります。しかし、チームメンバーは特定の Nexus に固有の要求も考慮しなければならないため、時間の経過とともにチームは変化する可能性が高くなります。

Nexus 統合チームは、統合に関するすべての問題に責任を持ち、彼らの仕事は、スクラムチームが自分たちの職務を、常に全体の中の一部として考えるようにすることです。Nexus を阻害する可能性のある制約や依存関係は、焦点を当てて注意を払うべき分野です。

また、Nexus 統合チームはかれら自身のスクラムチームメンバーに、Nexus での役割をコーチングするという役割も担っています。

## 12.6 プロダクトオーナーシップ

Nexus をベースにしたチームは、単一のプロダクトバックログを扱うため、プロダクトバックログに責任のある 1 人のプロダクトオーナーと一緒に仕事をします。

従来のプロダクトオーナーの説明責任は、以前には存在しなかった Nexus の活動に参加することを除いて変わりません。

特記事項：前述のようにプロダクトオーナーのチーム構成を採用することは、純粋なスクラム提唱者の間では一般的に嫌われています。しかし、この2つのアプローチを組み合わせることに価値がある場合もあります。

## 12.7 Nexus 統合チームのスクラムマスター

前述のとおり、Nexus 統合チームには、チームメンバーとして少なくとも 1 人のスクラムマスターが必要です。このスクラムマスターは、構成するスクラムチームの 1 つのスクラムマスターを兼ねることができますが、複雑な環境、特に組織が Nexus を使い始めたばかりの頃は、この方法はあまりお勧めできません。

Nexus のスクラムマスターは、スプリントの場合と同様に、コーチング、コミュニケーションのファシリテート、Nexus の成果を可能にする責任があります。

## 12.8 Nexus 統合チームのメンバー

開発の進め方と価値の提供に対するスケーリングアプローチは、スクラムチームが慣れ親しんできたものとはかなり異なる可能性があり、それを機能させるためには、おそらく追加のツールや技法を適用する必要があるでしょう。Nexus 統合チーム（NIT）は、新しいツールや技法の効果的な使用を確実にするだけでなく、成功に必要なツール、プロセス、プラクティスの導入にも責任があるため、これらの分野におけるメンバー候補の能力やスキルに基づいて、NIT のメンバーを決めます。

NIT のメンバーは、Nexus を組織の中で機能させる専門家として、Nexus や関連するツールや技法の使用に関するコーチやガイド役にもなります。

よくある質問に、スクラムでアーキテクトの役割がどう変わるかというものがあります。スクラムにおけるアーキテクトは、アーキテクチャの原則を定義するだけで、実装されるべき詳細なオペレーションモデルは定義しません。Nexus では、これらのオペレーションモデルの定義と実装は、NIT の責任となります。

NIT のメンバーは、Nexus のスプリントで起こっていることをより俯瞰して把握しているため、特にイテレーション中の Nexus の要素を統合することに関して、品質管理の職務も果たしています。

## 12.9 Nexus に導入された新しいイベント

Nexus を構成するスプリントのハイレベルな外骨格として、調整と統合を保証する新しいイベントが導入されています。

### 12.9.1 Nexus スプリントプランニング

この Nexus では何が起こるのか、誰がそれを行うのか、相互依存関係はどうなっているのか、優先順位はどうなっているのか？これらはすべて Nexus のスプリントプランニングの際に尋ねるべき有効な質問です。

プロダクトオーナーは、スクラムチームごとに選ばれるプロダクトバックログアイテムの選択について参加者をガイドしますが、誰が何を作業するかを指示することはありません。

しかし、それを行う前に、異なるスクラムチームの代表者たちがプロダクトオーナーと協力してプロダクトバックログのリファンインメントを行います。具体的には、彼らが持ち合わせているドメイン知識を利用しながら項目の順序を調整します。多くの場合、ビジネスの優先順位は何か、それを成し遂げるために最初に対処する必要のある依存関係は何か、ということが議論されます。プロダクトバックログのリファンインメントは、ビジネスの専門家と、ビジネス要求を達成するために満たすべき技術的または実用的な要件との間のバランスを取る作業です。

可能であれば、スクラムチームのメンバー全員がリファンインメントされたイベントに参加して、理解とコミュニケーションを最大化し、誤解や思い込みを最小限に抑えるべきです。

Nexus スプリントの目的と Nexus へ参加するチームが達成すべき成果は、Nexus スプリントプランニング時に Nexus スプリントゴールで定義されます。すべてのチームによって最終的に理解されると、各スクラムチームはそれぞれのスプリントプランニングを進めます。

スプリントプランニングは、チームが継続的に協働でき、新しい依存関係、問題、リスクが発見されたときに共有できるように、同じ場所で行うのが最適です。

チームは同じバックログからアイテムを選択するため、たとえ依存関係があったとしても、自分たちのチームが行うべきことと、代わりに別のスクラムチームが行うことに意味があることを、頻繁に交渉し、再評価することになります。

プロダクトバックログがきちんと定義されていれば、新たに発見される依存関係や要求の出現は最小限に抑えられますが、Nexus のスプリントプランニング中にどうしても発見されてしまうものもあります。

新たに発見された要求や依存関係はすぐに見えるようにする必要があり、共通の可視化プランニングツールを使って提示すると非常に便利です。バックログアイテムをチーム間で移動させたり、作業の順序を変えたりして依存関係を最小化することは、Nexus のスプリントプランニングでは当たり前のことです。

Nexus のスプリントプランニングでは、プロダクトバックログのリファンインメントが良くなります。スクラムチームがより大きなコンテキストに触れることができるようになるため、Nexus を使用するメリットとして、時間の経過とともにプロダクトバックログリファンインメントが著しく良くなります。

### 12.9.2 Nexus デイリースクラム

Nexus デイリースクラムの目的は、現在統合されているインクリメントの進捗状況を確認し、チーム間の依存関係や統合上の問題を発見することです。

各スクラムチームは適切に代表を務めるべきです。ここで意図的な言葉にしたのは、問題を最もよく理解している人が Nexus デイリースクラムに参加すべきであることを示すためです。Nexus デイリースクラムへの参加と Nexus 統合チームのメンバーを混同しないでください。Nexus 統合チームのメンバーは少なくとも Nexus スプリントの間は一定ですが、Nexus デイリースクラムの代表は Nexus スプリント中に変わる可能性があります。

### 12.9.3 Nexus スプリントレビュー

Nexus スプリントレビューは、Nexus がスプリントで構築した統合インクリメントのフィードバックを提供するために、スプリントの終わりに開催されます。すべてのスクラムチームがステークホルダーと個々に会い、統合されたインクリメントをレビューします。プロダクトバックログに調整が加えられることもあります。

## 12.10 Nexus イベント

Nexus イベントの期間は、スクラムの該当するイベントの長さによって導かれます。該当するスクラムイベントに加えて、それらはタイムボックスとなります。

### 12.10.1 リファインメント

プロダクトバックログを大規模にリファインメントすることには、2つの目的があります。それは、スクラムチームのどのチームがどのプロダクトバックログアイテムを提供するかを予測するのに役立つと同時に、それらのチーム間の依存関係を明らかにすることです。この透明性により、チームは依存関係をモニターでき、最小化することができます。

Nexus によるプロダクトバックログアイテムのリファインメントは、プロダクトバックログアイテムが十分に独立した状態になり、過度な衝突がないように単一のスクラムチームによって作業できるところまで続けられます。

リファインメントの回数、頻度、期間、発生率は、プロダクトバックログに内在する依存性と不確実性に基づいています。プロダクトバックログのアイテムは、非常に大規模で漠然としたリクエストから、単一のスクラムチームがスプリント内で提供できる実用的な作業まで、さまざまなレベルのものがあります。

リファインメントは、必要かつ妥当な場合、スプリントを通して継続的に行われます。プロダクトバックログのリファインメントは、Nexus スプリントプランニングのイベントでプロダクトバックログアイテムを選択できるようにするために、各スクラムチーム内で継続されます。

### 12.10.2 Nexus スプリントプランニング

Nexus スプリントプランニングの目的は、1つのスプリントに対するすべてのスクラムチームの活動を調整することです。プロダクトオーナーは、ドメイン知識を提供し、選択と優先順位の決定をガイドします。プロダクトバックログは、Nexus スプリントプランニングの前に、依存関係を特定して削除または最小化したうえで、十分にリファインされている必要があります。

効果的な Nexus スプリントプランニングは、3 つの明確なステップで構成されます。

1. スクラムと同様に、**Nexus のチームは自分たちが行うスプリントの作業を選択します。**チーム間での作業の重複を避けるため、他のスクラムチームの代表者と共同で行うのがベストです。
2. **各チームは、通常のスプリントプランニングプロセスを実行します。**これはスクラムチームごとに行われ、並行して行うこともできます。
3. **その後、依存関係が発見されると、チームは作業をリファインし、**不必要な作業の中断やチーム間の依存関係を最小限に抑えるために、チーム間で作業を再編成することもあります。

Nexus スプリントプランニングでは、各スクラムチームの適切な代表者が、リファインメントイベントで作成された作業の順序を検証し調整します。リファインメントイベントでは、コミュニケーションの問題を最小限にするために、Nexus スクラムチームの全メンバーの参加が求められます。

プロダクトオーナーは、Nexus スプリントプランニングの際に Nexus スプリントゴールについて議論します。Nexus スプリントゴールは、スプリント期間中にスクラムチームが達成する目的を示しています。Nexus の全体的な作業が理解された後、各スクラムチームがそれぞれ個別にスプリントプランニングを行うことで Nexus スプリントプランニングが継続されます。スクラムチームは、新たに見つかった依存関係を Nexus の他のスクラムチームと共有し続けなければなりません。各スクラムチームが個別のスプリントプランニングイベントを終了すると、Nexus スプリントプランニングは完了します。

Nexus スプリントプランニング中に、新たな依存関係が発生することがあります。それらを透明化し、最小限に抑える必要があります。また、チーム間の作業順序も調整する必要があります。十分にリファインされ

たプロダクトバックログがあれば、Nexus スプリントプランニング中に新たな依存関係が発生することは少なくなります。スプリントのために選択されたすべてのプロダクトバックログアイテムとその依存関係は、Nexus スプリントバックログ上で可視化されるべきです。

## 12.10.3 Nexus スプリントゴール

Nexus スプリントゴールは、スプリントのために設定された目標セットです。これは、Nexus 内のスクラムチームのすべての作業とスプリントゴールを全て合わせたものです。Nexus では、Nexus スプリントゴールを達成するために行われた（開発された）機能を、ステークホルダーからのフィードバックを受けるための Nexus スプリントレビューでデモする必要があります。

## 12.10.4 Nexus デイリースクラム

Nexus デイリースクラムは、適切な代表者（個々の開発者）が統合されたインクリメントの現状を検査するためのイベントであり、統合上の問題や新たに発見されたチーム間の依存関係、チーム間の影響を特定するためのイベントです。

Nexus デイリースクラムでは、参加者は各チームが統合されたインクリメントに与える影響に焦点を当て、議論する必要があります。

- 前日の作業はうまく結合できたか?できなかった場合、その理由は?
- どのような新しい依存関係や影響が確認されたか?
- Nexus では、どのような情報をチーム間で共有しなければならないか?

チームは Nexus デイリースクラムを利用して、Nexus スプリントゴールに向けた進捗状況を検査します。少なくとも、Nexus デイリースクラムのたびに、Nexus スプリントバックログは、Nexus 内のスクラムチームの作業に関する現在の理解を反映して調整される必要があります。

その後、個々のスクラムチームは、Nexus デイリースクラムで明らかになった課題や作業を持ち帰り、個々のデイリースクラムイベント内で計画を立てます。

## 12.10.5 Nexus スプリントレビュー

Nexus スプリントレビューは、スプリントの終わりに行われ、Nexus がスプリントで構築した統合インクリメントのフィードバックを行い、必要に応じてプロダクトバックログを適応させます。

Nexus スプリントレビューは、統合されたインクリメント全体がステークホルダーからのフィードバックを得ることが焦点となるため、個々のスクラムチームのスプリントレビューに代わるものです。完成した作業をすべて詳細に示すことはできないかもしれません。ステークホルダーのフィードバックを最大化するための技法が必要な場合もあります。Nexus スプリントレビューの結果は、更新されたプロダクトバックログとなります。

## 12.10.6 Nexus スプリントレトロスペクティブ

Nexus スプリントレトロスペクティブは、Nexus が自らを検査・適応させ、次のスプリントで実施する改善計画を作成し、継続的な改善を図るための正式な機会です。

Nexus スプリントレトロスペクティブは、Nexus スプリントレビューの後、次の Nexus スプリントプランニングの前に行われます。

Nexus スプリントレトロスペクティブは、3 つのパートで構成されています。

1. 第 1 部は、Nexus 全体から適切な代表者が集まり、**より多くのチームに影響を与えている問題を確認**する機会です。その目的は、共有された問題をすべてのスクラムチームに透明化することです。
2. 第2部は、スクラムフレームワークに記載されているように、**各スクラムチームがそれぞれのスプリントレトロスペクティブを行う**ことです。彼らは、Nexus レトロスペクティブの第 1 部から提起された問題を、チームが議論するためのインプットとして使用することができます。個々のスクラムチームは、彼らのスプリントレトロスペクティブの際に、これらの問題に対処するためのアクションを作成する必要があります。
3. 最後の第 3 部では、スクラムチームの適切な代表者が再び集まり、**特定された**アクション**をどのように可視化し、追跡するかについて合意する**機会です。これにより、Nexus 全体としての適応が可能になります。

これらは一般的にスケーリングのための機能不全を起こすために、すべてのレトロスペクティブでは以下のテーマを扱う必要があります。

- やり残した作業はないか?技術的負債は発生したか?
- すべてのアーティファクト、特にコードは、頻繁に（毎日のように）統合されていたか?
- ソフトウェアの構築、テスト、デプロイが十分な頻度で行われ、未解決の依存関係が過度に蓄積されることはなかったか?

上記の質問に対して、必要に応じて対処する。

- なぜ起きたのか?
- 技術的負債はどのようにして元に戻せるのか?
- 再発を防ぐにはどうしたら良いか?

## 12.11 Nexus アーティファクト

スクラムガイドに記載されているように、アーティファクトは成果や価値を表し、透明性を提供し、検査と適応の機会を与えます。

### 12.11.1 プロダクトバックログ

Nexus 全体と Nexus で活動するすべてのスクラムチームのために、単一のプロダクトバックログがあります。プロダクトオーナーは、プロダクトバックログの内容、有効性、順位付けを含めて、プロダクトバックログに対して説明責任を負います。

大規模な場合、プロダクトバックログは、依存性が検出され最小限に抑えられるレベルまで理解されなければなりません。作業の解決をサポートするために、PBI に記述されている機能は、しばしば粒度が細かく*薄くスライスされ*ています。プロダクトバックログアイテムが、他のスクラムチームへの依存性がないか最小限に抑えられた状態となり、スクラムチームが実行するアイテムを選択できるようになった時点で、Nexus のスプリントプランニング会議の*準備が整った*と判断されます。

### 12.11.2 Nexus スプリントバックログ

Nexus スプリントバックログは、個々のスクラムチームのスプリントバックログのプロダクトバックログアイテムを 1 つにまとめたものです。このバックログは、スプリント期間中の依存関係や作業の流れを強調するために使用されます。最低でも毎日更新され、多くは Nexus デイリースクラムの一部として行われます。

### 12.11.3 統合されたインクリメント

統合されたインクリメントは、現時点までに Nexus によって完成された統合作業の総和を表しています。統合されたインクリメントは、使用可能であり、潜在的にリリース可能でなければならず、これは完成の定義（DoD）を満たすことを意味します。統合されたインクリメントは、Nexus スプリントレビューで検査されます。

### 12.11.4 アーティファクトの透明性

その構成要素であるスクラムと同様、Nexus は透明性に基づいています。Nexus 統合チームは、Nexus 内のスクラムチームや組織と協力して、すべてのアーティファクトの透明性をあげ、統合されたインクリメントの状態がチームメンバー全員に広く確実に理解されるようにします。

Nexus のアーティファクトの状態は、Nexus 期間中に行われる多くの意思決定を左右するものであり、アーティファクトの透明性のレベルに応じて効果を発揮することになります。部分的な情報や不完全な情報は、不正確で不完全な判断につながり、それが Nexus の規模になると拡大してしまいます。

インクリメントは、技術的負債が Nexus や進行中のオペレーションにとって受け入れられなくなる前に、依存関係が検出され解決されるように開発する必要があります。完全な透明性がないと、リスクを最小限に抑え、価値を最大化するために Nexus を効果的に導くことができなくなります。

### 12.11.5 完成の定義（DoD）

Nexus 統合チームは、各スプリントで開発された統合インクリメントに適用される完成の定義（DoD）を設定する責任があります。Nexus のすべてのスクラムチームは、この完成の定義（DoD）を遵守します。インクリメントは、統合され、使用可能で、プロダクトオーナーによりリリース可能と見なされた場合にのみ、完成となります。

個々のスクラムチームは、自分のチーム内でより厳しい完成の定義（DoD）を適用することを選択できても、合意されたものよりあいまいな基準をインクリメントに適応することはできません。

## 12.12 Nexus の核となる Nexus 統合チーム

Nexus は、3～9 のスクラムチーム、チームを跨いだ 1 名のプロダクトオーナー、そして 1 つのプロダクトバックログから構成されます。Nexus 統合チーム（NIT）の構成は以下の通りです。

- プロダクトオーナー（唯一の 1 人）
- スクラムマスター（選ばれた 1 人）
- Nexus を構成するチームから選ばれた Nexus 統合チームメンバー

**プロダクトオーナー**

プロダクトオーナーは NIT に所属していますが、Nexus に所属するすべてのスクラムチームでもプロダクトオーナーの説明責任を果たします。プロダクトオーナーは、ステークホルダーにとっての価値が最大になるように、プロダクトバックログの順位付けを行い、リファインメントを確実に行います。（説明責任）

**スクラムマスター**

Nexus 統合チームのスクラムマスターは、手法としての Nexus が確実に理解され、機能させることに責任があります。また、彼らは通常、Nexus デイリースクラム、Nexus スプリントプランニング、クロスチームリファインメント、Nexus スプリントレビュー、Nexus レトロスペクティブなどの Nexus レベルのイベントをファシリテートします。少なくとも、それらをファシリテートする人をサポートします。

### Nexus 統合チームのメンバー

Nexus 統合チームのメンバーは、Nexus 内のスクラムチームの開発者で構成されています。NIT は、Nexus 内のすべての活動を調整します。チームメンバーは、スクラムマスターがスクラムで行っているように、自分のチームのコーチとなり、Nexus の実装、適用、使用を共同で調整し、指揮することで、最適な成果を実現します。

NIT のメンバーは、少なくともスプリントサイクルごとに、Nexus が完成させた成果をまとめた統合インクリメントを確実に作成する責任があります。

Nexus 統合チームの構成は、Nexus の現在のニーズに応じて、時間の経過とともに変わる可能性があります。

NIT のメンバーは一般的に Nexus 内のスクラムチームから選ばれ、一般的に NIT のメンバーはパートタイムですが、特に関連業務の調整が不可欠で複雑な環境においては、NIT のメンバーの中にはフルタイムの方がよいメンバーもいます。

ただし、NIT メンバーは、スクラムチームのメンバーとしてよりも、NIT メンバーとしての役割を常に優先しなければなりません。全体は部分よりも重要なのです。

#### アカウンタビリティ

Nexus 統合チームは、少なくともスプリントごとに統合されたインクリメントを確実に作成する説明責任があります。これにより、統合されたプロダクトが、個々のチームの努力だけでなく、チーム全体の目標に重点を置くようになります。

スケールの大きいソフトウェア開発では、個々のスクラムチームが頻繁に使用しないようなツールやプラクティスが必要になります。したがって、Nexus 統合チームは、技術的に熟達した専門家で構成されるべきです。しかし、統合とはコードの作成より幅広い意味になります。チームがどのように協力し協働するかは、統合の重要な部分です。例えば、ビルドを分割したり、集団的コード所有権を実践したりするチームの対応は、すべて統合を成功させる大きな要素です。Nexus 統合チームには、「ハード」（技術）と「ソフト」（人材）の両方のスキルを持つメンバーが必要です。

#### アクティビティ

Nexus 統合チームのメンバーは、メンバーのスクラムチームのコーチとして機能します。そのため、Nexus 統合チームのメンバーは、Nexus 内のスクラムチームが統合インクリメントの状態を継続的に改善できるように、必要なスキルや特性を持っている必要があります。

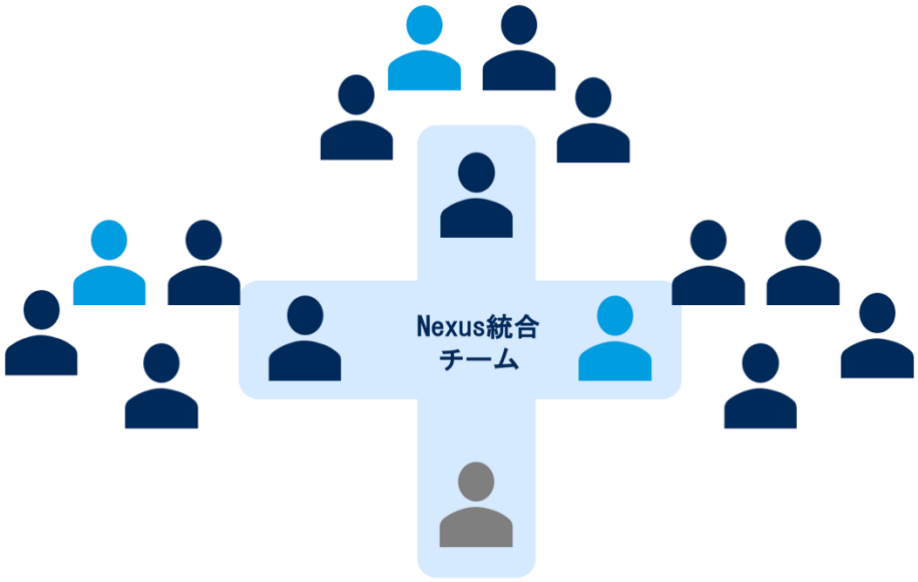
Nexus 統合チームが日々行う活動は以下の通りです。

- チーム間の作業調整を支援
- 可能な限り早い段階での依存関係の認知度向上
- 統合ツールとプラクティスの周知と使用の徹底
- コンサルタント、コーチ、コミュニケーションリンクとして仕える役割
- ときには、必要に応じて作業をサポート
- アーキテクチャ/インフラの共有化をファシリテーション
- 統合するための透明性を常に提供

Nexus 統合チームのメンバーは、*サーバントリーダー*であり、改善のためのコーチング手法を持っていることが重要です。これは「オールスター」のチームではありません。

図 24 に示すように、Nexus 統合チームに Nexus 内のスクラムチームのメンバーを配置することで、「私たちと彼ら」の分裂を避けることができます。

*図 24 Nexus 統合チームの作成*




図は EXIN 制作：Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

Nexus 統合チームは、スクラムチームのメンバーで構成されています。Nexus 統合チームは通常、プロダクトバックログから離れてスクラムチームに戻り一緒に働くことはありません。それよりも、Nexus 内のスクラムチームにサービスを提供するコーチとガイドのコミュニティなのです。彼らはまたスクラムチームのメンバーでもあるので、Nexus は本質的にそれ自体がリーダーシップを発揮します。

**スクラムチームとしての Nexus 統合チーム**

統合するための深刻な懸念や問題がある場合、最後の手段として Nexus 統合チームは、バックログの作業を中断してスクラムチームとして作業する必要があると判断することがあります。この場合、彼らは問題が解決するまで、既存のスクラムチームから Nexus 統合チームにフルタイムで移ります。

次のような理由で、これを選択する場合があります。

- Nexus 統合チームのメンバーだけが持っているスキルを必要とする時
- プロダクトの状態が分らなくなった時
- Nexus 内のスクラムチームが一時的にうまく統合できなくなった時

これは、持続可能な作業モデルではありません。それは、Nexus がスクラムをうまくスケーリングできず、作業が Nexus 統合チームという 1 つのチームのペースにまで減速していることを意味し、結果的に Nexus 統合チームしか作業できなくなっていのであれば、スケーリングは失敗に終ったと言えます。

ただし、この方法で作業を行うかどうかの判断は、プロダクトオーナーを含む Nexus 統合チームが、その説明責任を果たすための一時的な最後の手段であると考えることが必要です。

Nexus 統合チームがスクラムチームとして働くことは、"失敗モード"とみなされます。これは一時的で戦術的な選択肢であるべきです。通常、Nexus 統合チームのメンバーが Nexus 内のスクラムチームと協力して、ペアリングやコーチングを通じて不足しているスキルや知識を移転するのが良いでしょう。

**メンバーシップ**

プロダクトオーナーの説明責任以外では、Nexus 統合チームのメンバーは恒久的である必要はありません。実際のところそれは状況によります。必要とされる能力の種類は時間とともに変化し、チーム構成も変化することになります。

Nexus は、Nexus 以外の人を Nexus 統合チームに招き入れることがあるかもしれません。その中には、Nexus の成功に欠かせない分野の代表が含まれるかもしれません。彼らが統合に関して同じ説明責任を共有していれば、Nexus 統合チームへの参加は良い結果につながります。

**Nexus 統合チームに関する最後の一言**

Nexus 統合チームは、管理チームでもなければ、狭い範囲の専門や経験を持った人のチームであってはなりません。通常は Nexus 内の個々のスクラムチームから選ばれた専門家チームであり、プラクティスやツールが実装、理解され、依存関係が検出されることを確実にします。彼らは、少なくともスプリントごとに統合されたインクリメントを確実に作成する説明責任があります。彼らは統合することに注力し、Nexus 内の技術的／非技術的な両方の課題に対処する必要があります。

Nexus 統合チームは、階層構造の管理者としての「管理担当」ではなく、サーバントリーダーシップとコーチングを通じて、まさにスクラムチームに対するスクラムマスターのごとく行動します。彼らは、チームから得られるボトムアップの知見を使い、問題の解決と改善をします。大きな成功はすべてのチームが継続的にプロダクトの開発と向上に貢献することから生れます。

## 12.13 Nexus スプリントバックログの可視化とクロスチームリファインメント

Nexus スプリントプランニングの目的は、1 つの Nexus スプリントにむけて Nexus に参加するすべてのスクラムチームの活動を調整し連係させることです。Nexus スプリントバックログは Nexus スプリントプランニング中に作成され、Nexus 全体の作業を可視化したものです。特に依存関係が明確に強調されています。

合同の Nexus スプリントプランニングは、構造的な方法で行われるべきで、それによって作業が適切に調整され、すべての依存関係が考慮されることになります。

新しい依存関係は Nexus でしか発見されないことが多いため、複雑な環境では継続的なリファインメントが必要です。

チームを横断するクロスチーム Nexus リファインメントボードは、Nexus 内のスクラムチームが行った作業を検証し変更するための有用なツールであることが実証されています。

## 12.14 Nexus のクロスチームリファインメント

プロダクトバックログリファインメントは、スクラムでは継続的に行われている活動の 1 つですが、必須のイベントではありません。Nexus では多くのチームが 1 つのプロダクト（プロダクトバックログ）に共同で取り組むという複雑さのため、リファインメントは公式の必須イベントになります。

共有されたバックログのリファインメントでは、チームが自分たちの提供できる作業と、今後のスプリントでの作業の提供順序を理解できるように、プロダクトバックログアイテム（PBI）を十分に分解することに重点を置きます。また、このイベントでは、チーム間の依存関係を最小限に抑え、取り除くことにも焦点が当てられます。

特記事項：Nexus のリリース計画に代わるものは何かと思っている人にとって答えは*ありません*。しかし、バックログリファインメントを共有することで、今後の複数のインクリメントやスプリントで提供される PBI の重要性、依存性、順序についての相互理解が生まれます。以下の質問をご覧ください。

Nexus では、複数のスクラムチームが 1 つのプロダクトバックログから作業を行っています。それ故、バックログをリファインすることに対する新しい質問に対応しなければなりません。

- どのチームがどんな仕事をしているのか?
- スプリントやチームを横断して、リスクや複雑さと価値の早期提供のバランスをとるために、どのように作業を進めるのがベストなのか?

スクラムをスケーリングする際、これらの質問には開発者自身がクロスチームリファインメントを通じて対応するのが最善であると推奨されています。

各チームの代表者は、クロスチームリファインメントの場に参加します。代表者は、チーム内での作業内容ではなく、リファインメントの対象となる作業に基づいて選ばれ、ワークショップに参加します。これらのワークショップには、必要なスキルに応じて様々な人が参加するのが一般的です。

しかし、どのチームがどんな仕事をしているかは、どうやって特定するのでしょうか?

プロダクトバックログアイテムの詳細化と順序付け。

プロダクトオーナーは、リファンインメントされるべき PBI について説明する必要があります。これらはまだスプリント内に収まるサイズに分解、またはスライスされていない大きなアイテムなのが殆どです。これはプロダクトバックログの最上位にある PBI は、スプリントプランニングが始まる前には必ずリファインメントされているという従来の常識とは異なります。この分解を遅らせたことで、リファインメント中に明らかになるスキルや依存関係に基づいてアイテムを分解できる機会を得ることになります。

最初の会話は非常に流動的で、どのチームがその仕事を引き受けるために必要なスキルがあるか、に焦点を当てなければなりません。各チームに無い専門スキルについては、既存の制約を慎重に考慮する必要があります。

スクラムチームの代表者が PBI を理解し始めたら、それをより小さな項目に分解し、そのアイテムを自分のチームに持ち帰って通常のプロダクトバックログリファンインメントやスプリントプランニングを行うことができます。

*図 25 Nexus ボードの作成*




図は EXIN 制作:Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us.

最初のリファインメントでは依存関係も重視されることに注意し、従来のリリース計画と同様に、このスプリントで何ができるか、次以降のスプリントで何を待つべきかを明確にします。数ヶ月前に顧客と約束した事よりも、依存関係に沿って話し合いを進めます。そして、3 つのスプリントの先にある依存関係を探そうとするのは良いやり方ではありません。

それゆえ、次のスプリントの作業の流れを理解し、チームごとにそれを可視化することは、最初のリファインメントの活動の一部になりますが、これが終わると NIT のメンバーは、「価値の早期提供とリスクや複雑さとのバランスをとるために、スプリントやチーム間でどのように作業を順序付けるのが最適か」を問い掛ける必要があります。

これは、リファインメントボード上で依存関係を可視化し、その依存関係をどのように管理するのがベストかを検討することで行われます。

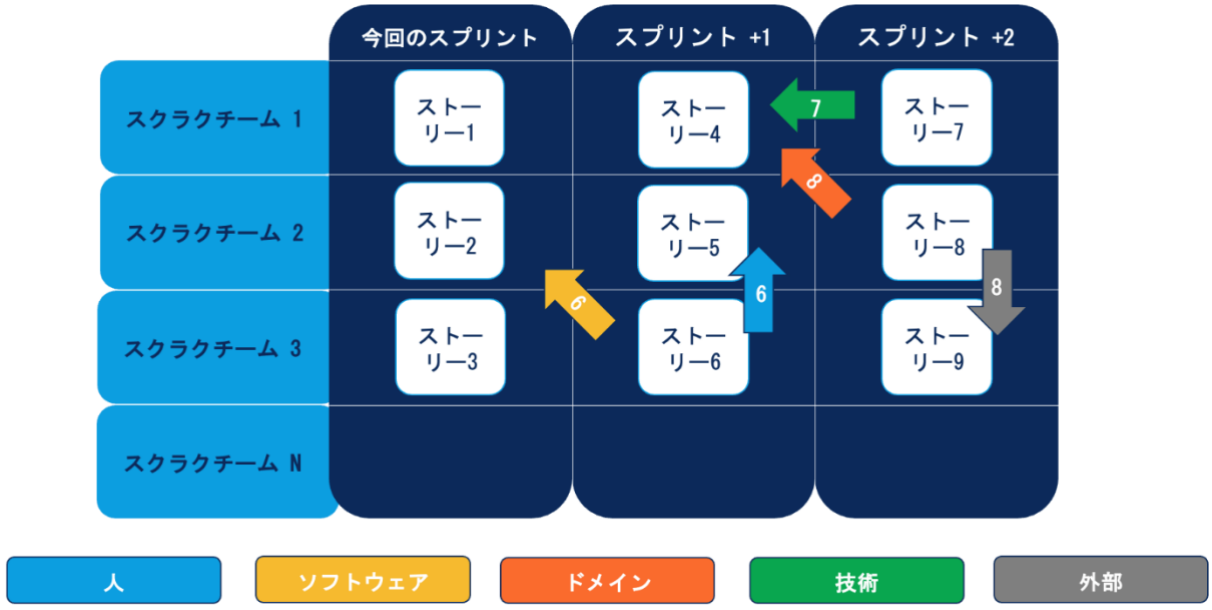
すべての依存関係が類似の性質を持っているわけではないので、チームは依存関係を定義する際に、依存関係もまた分類する必要があります。

一般的には、以下のカテゴリーとサブカテゴリーを出発点として使用することができます。

- **Build Sequence（開発順序）** – あるアイテムはその親が完了するまで完了しない。（技術、ドメイン、ソフトウェアを含む）
- **People/Skills（人／スキル）** – 特定の人やチームだけがアイテムを完成させることができる。
- **External（外部）** – 親アイテムが Nexus の外で提供されている。

依存関係は矢印で示され、矢印の色で依存関係の種類を区別することができます。また、矢印の方向は親子関係を表し、わかりやすくするために親ストーリーの番号を矢印に記載することもあります。

*図 26 Nexus スクラムチームに分散したタスクの依存関係への対応*




図は EXIN 制作：Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC – Kdp Print Us. and Schwaber、K., & Scrum.org（2021）. *The Nexus™ Guide – The Definitive Guide to Scaling Scrum with Nexus*. Scrum.org. https://www.scrum.org/resources/nexus-guide

少なくとも、外部との依存関係を特定することを検討ください。状況に応じて、色を追加することもできます。

矢印が多ければ多いほど、よりリスクが高くなり、明確な計画と調整が行われることが重要です。依存関係はこれからの作業やリファインメントのクリティカルパスを浮き彫りにします。次に、リファインメントによって依存関係が最小限に抑えられ、少なくとも依存関係による影響を最小限に抑えます。

以下のような一般的な決まりがあります。

🡐 水平方向の矢印は、単一のスクラムチームにおける依存関係を示しています。
（一般的に低リスク）
🡔 上向きの斜めの矢印は、時間やチームを超えた依存関係を示しています。
（一般的に中リスク）
🡑 縦の矢印は、チーム間での依存関係を示していますが、同じスプリントサイクル内であることを示しています。
（これは、1つのチームの遅れが Nexus 全体を危険にさらすことになるため、高リスクと考えられます。）
🡕 下向きの斜めの矢印は、時間を超えた外部依存性を示しています。
（中リスクと考えられます。）
🡓 下向きの縦矢印は、Nexus チームの外部にある別のスプリントサイクルの依存関係を表しています。
（これは高リスクの依存関係と考えられます。）

依存関係を可視化することで、Nexus チームは複雑な状況を理解し、NIT はチーム間で作業項目の再配分を試み、依存性のある作業を1つのチームが行うことで、チーム間の依存関係やリスクを無くすか、少なくとも最小限に抑えることができます。

このプラクティスの例外は、NIT がこのスプリント中に特定のアイテムのデリバリーを進めるために、チーム間で作業を分散することを決定した場合です。それは、1つのチームで完成するには大きすぎるからです。

ここで、依存関係とリスクを最小限に抑えるための方法を紹介します。

- ハンドオーバーやチーム間の依存関係を最小限にするために、チーム間で仕事を移動させる。
- チーム間の依存関係を少なくするために、チーム間で人を移動します。チーム内のスキル依存性を最小限にするために、1、2回のスプリントにおいてチーム間で特定のスキルを再配分すると、デリバリ―リスクが大幅に減る可能性があります。
- アイテムを別の方法で分割したり、やるべきことを再構成したりすることで、依存関係をなくすことができるかもしれません。
- ボード上に作業をプロットする際には、活動計画のできるだけ早い段階ですべてのリスクを強調し、チーム間のスプリント内の依存関係をできるだけ早く可視化するようにします。

リファインメントと依存関係の追跡は一度だけのイベントではなく、Nexus 全体とスプリントを通じて再検討する必要があります。

## 12.15　Nexus スプリントプランニングと Nexus デイリースクラムの詳細

クロスチームリファインメントの情報は、Nexus スプリントプランニングへのインプットとなります。Nexus スプリントプランニングでは、現在のスプリントの依存関係を考慮する必要があります。

また、依存関係に関する情報は、Nexus デイリースクラム中にチーム間の同期やリスク管理、進捗管理を日々行う際に、焦点となる情報として活用されます。

## 12.15.1 Nexus スプリントバックログ

スプリントバックログを可視化する唯一の方法が、スクラムボードを作成することであることを忘れないでください。Nexus チーム間のスプリントの依存関係は、以下の例のように Nexus スクラムボードを作成することで管理することができます。

ここでは PBI 1がスクラムチーム2によって提供されていますが、それはスクラムチーム1によって提供されている PBI 4 に依存しており、スクラムチーム 3 によって提供されている PBI 2は、スクラムチーム2によって提供されている PBI 8 に依存しています。

*図 27 チーム間の依存関係を利用する*




図は EXIN 制作：Botha、J（2019）. *Agile: A Manager's Guide to Unlocking Business Value*. Amazon Digital Services LLC - Kdp Print Us. and Schwaber、K., & Scrum.org（2021）. *The Nexus™ Guide - The Definitive Guide to Scaling Scrum with Nexus*. Scrum.org. https://www.scrum.org/resources/nexus-guide

Nexus スクラムボードの Blocked 列を使用すると、チーム間、スプリント内の依存関係を示すことができます。

この可視化は、スプリント内の依存関係と関連するリスクを明らかにするために重要です。また、チーム間の調整と正しい作業順序に焦点を当てることを促します。

Nexus デイリースクラム中、通常、Nexus スプリントバックログ（スクラムボード）が焦点となり、デイリースクラム中の会話を促進します。

もし、Nexus にスプリント内の依存関係がない場合、チームは代わりにクロスチームリファインメントボード内の現在のスプリント列を使って、Nexus スプリントバックログを表すことを選択できます。この場合、リファインメントボードがデイリースクラムの中心となります。

# 13 アジャイル・スクラムの導入と成功のために

調査によると、ほとんどすべての組織でアジャイルの取り組みが行われていますが、大多数の組織が自分たちのアジャイル成熟度は並以下であると考えています。

しかし、アジャイルの利点は明らかで、ソフトウェアの迅速な提供、優先順位の変更に対する管理能力の向上、ビジネスと IT の間の生産性と連携の向上などが挙げられます。

スクラムは、組織で使用されている中では圧倒的に人気のあるアジャイル手法であり、あなたは良い仲間達と共にいます。

しかし、今回の調査結果には、より暗い陰の側面も示されています。

アジャイルを採用して規模を拡大していく上で、最も上位にランクインした課題は、引き続き組織文化に関するものでした。これには、変化に対する全体的な組織の抵抗、経営陣のサポートやスポンサーシップの不足、リーダーシップの参加不足、アジャイルの価値観と対立する組織文化の存在などが含まれます。

しかし、これらの課題にもかかわらず、将来的にはさらにアジャイルになることは明らかであり、ビジネスアジリティを受け入れられない者は取り残されることになるでしょう。

## 13.1 すべては組織の変革のために

マッキンゼーは、*The journey to an Agile organization*（2019 年）という記事の中で、アジャイルな運用モデルへの移行は、特に既存企業にとっては厳しいものだと述べています。アジャイルをはじめとする多くの先進的なプラクティスの敵は、古いサイロ型の運用モデルとそれに付随する時代遅れの管理手法です。

アジャイルへのシフトは、チームが正しいことをすると信頼し任せることであり、マネジメントにとっては最も難しいことです。意思決定を可能な限り低いレベルに委ねることが、アジャイルの成功の核心であり、工業化時代の企業がこれを正しく行うことは、組織が直面する最も困難な試みです。

専門家の見解は全員が一致しています。企業全体のアジャイル変革が必要であり、アジャイルがもたらすメリットを実現するためには、それは包括的かつ反復的なものでなければなりません。

マッキンゼーはその記事の中で、変化の鍵となる分野を次のように強調しています。

- 人々
- 組織構造
- プロセス
- システムとツール

**人々**

人々のカテゴリーにおける成功の鍵は、組織のリーダーたちに自分たちの役割／職務の変化を理解してもらうことです。すべてのマネジメントスタッフは、仕事や人を監督することから、チームが複数の専門性を持った自己管理チームとして、物事を成し遂げるために必要なビジョンを提供することに焦点を移すよう、訓練されなければなりません。また、マネジメントの仕事は、指示することよりもインスピレーションを与えコーチングすることです。マネジャーは、仕事のモデル（WHAT）を支援するべきで、仕事のやり方（HOW）を指示するべきではありません。

これは、チームメンバー全員が物事を成し遂げるために貢献し、物事をどのように成し遂げるかを集団で決定する必要があることを意味します。スプリントで何が起こるかを考えてみると、これはまさにスクラムが提案していることです。

コーチングの一環として、マネジャーは社員が成長し、新しいスキルやコンピテンシーを身につけられるように、支援することにもっと力を入れる必要があります。事実、アジャイルとはマネジャーの仕事が活動、プロセス、手順の管理から、人々のエンパワーメントとチームの構築に変わることを意味します。今の仕事を見る良い方法は、むしろ価値の流れを認識し、人とプロセスがどのように顧客価値の創造に貢献するかを認識することです。

これは簡単なことのように見えますが、マネジャーがマネジャーとして成長する過程で形成されたテイラー主義の管理的行動を振り払って、組織におけるサーバントリーダー、イネーブラー、コーチ、メンターという新しい役割を受け入れるのは限りなく難しいことです。

また、組織の中の地位や身分よりも影響力の方がはるかに重要になってきていることを意味しており、その自然な流れは次のセクションで説明します。

**組織構造**

よりフラットな組織とクロススキルを持った従業員は、このようなビジネスのやり方の違いから自然に生まれるものです。

現在では、よりミッションを重視し、あるいは価値を重視したアプローチにより、チームのあり方、ひいては組織のあり方が決定されます。その結果、従業員の規模や配置に対するアプローチにも影響を与えることになります。

チームは複数の専門性を持ったユニットとして働き、多くは一時的な組織構造であることから、組織はレポーティング構造を大幅に簡素化し、階層を減らす必要があります。アジャイル組織は最初から分権化されており、大規模な企業の本社は助けになるというよりも、むしろ邪魔になるのです。

これはガバナンスを複雑にするので、結果としてガバナンスモデルを単純化、効率化する必要があります。他の組織がそれを意味があると分かっても、私たちはもはや自分たちを守るために何かをすることはできません。私たちはリスクを深く理解し、ガバナンス構造、方法（メソッド）、および関連する統制、特に組織のリスクに基づく統制を構築し、継続的に改善する必要があります。

**プロセス**

厳格なプロセスと適合主義体質は、今日では組織の助けになるどころか、むしろ組織の妨げになることが多くなります。プロセスは、ハイレベルで定義される必要があり、チームがハイレベルのガイダンスに適合する独自の方法を定義するための自由度を持つ必要があります。これは、各チームが同じこと（プロセス）を行うための独自の方法（手順）をもつことを意味しており、アウトプット、アウトカム、遵守すべき必須のコントロールが明確になっていれば、それで十分なのです。

組織は、パフォーマンスを測定する手段として、プロセスや手順をよく使用しますが、プロセスや手順が同じではない場合、パフォーマンスの測定基準は、今やその殆どを成果（アウトカム）ベースにすべきです。

**システムとツール**

マネジメントは組織が適切なツールやシステムを備えていること、そして、それらがアジャイルな働き方をサポートするよう適切に導入されていることを確実に行なってください。

それはまた、企業レベルのアーキテクチャは原則として一元的に定義し、時間の経過に伴って求められるアーキテクチャを設計しながら進化させることを意味します。

価値あるインクリメントを提供しても、すぐに使用できなければ意味がありません。特に製品やサービスの実現と提供のための自動化は、高い優先度を持つべきです。

ソフトウェアの環境設定において、これは、テストとインテグレーションプロセスを自動化し、高速な継続的デリバリーを可能にする自動化されたデリバリーパイプラインを意味します。例として、DevOps の3つの道を使います。

IT インフラの観点からは、柔軟性と拡張性の構築を意味します。クラウドの手法やアーキテクチャの採用は、ますます避けて通れないものとなっています。

## 13.2 変化をファシリテートする

変化をファシリテートするための手引書となる ADAPT モデルと ADKAR モデルについては、2章で説明しました。

リーンマネジメントの原則を採用することは、包括的で反発を受けにくい進化的な組織変革をファシリテートする方法でもあります。

リーンマネジメントの利点は、それがマネジメントから始まり、サーバントリーダーとしての新しい役割を担うマネジメントによって推進されることです。

これが非常に重要である理由として、*the Prosci Best Practices in Change Management benchmarking report*（2021）によると、管理者が変化に抵抗する主な理由は、コントロールと権限を失うことへの恐れです。管理者が物事をより良くするための新しい方法を自分のものにし、新しい働き方を受け入れることで、どんな組織においても変化をはるかに容易にファシリテーションできます。その1つの条件として、管理者が自分の行動を進んで変えることで、従業員に自分の行動を安心して変えることができることを示す必要があります。

報告書では引き続き、従業員と管理者の両方にとって主な理由を以下のように挙げています。

| 従業員 | マネジャー |
|---|---|
| 自覚意識の無さ | 管理権限を失う恐れ |
| 未知への恐れ | 時間がない |
| 雇用の不安 | 現状維持が快適 |
| 孤立する不安 | 個人的な損得勘定 |
| 全体デザインへの不関与 | デザインへの不関与 |
| | |

最も失うものが大きい人たち（実際の損失ではなく、彼らの心の中で認識された損失）からの最大の抵抗が予想され、また、変化のイニシアチブに反対する何らかの集団的な連係も予想されます。

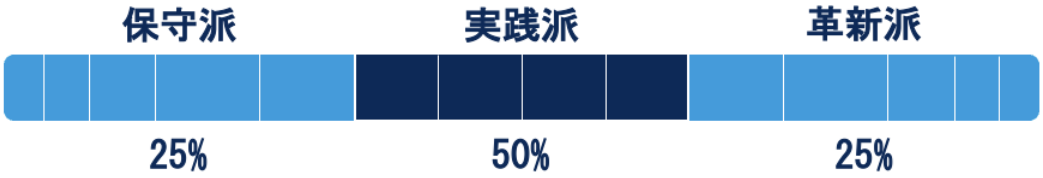
しかし、組織の中には、現状の不満から変化を心から受け入れる人もいます。そうした変化することを好み、刺激的と感じる小さな集団から支持を受けると思われます。

Chris Musselwhite 氏と Robyn Ingram 氏は、このような変化に対する個人の反応の範囲を「変化の連続体」と呼び、個人の変化に対する認識を示す 3 つの主要なカテゴリーにより、該当する個人を特徴づけるチェンジスタイルインジケータを作成しました。

連続した 3 つのカテゴリーがあります。

- コンサーバー（保守派）
- プラグマティスト（実践派）
- オリジネーター（革新派）

*図 28 変化のイニシアティブにおける 3 種類のユーザーを表すスケール*




図は EXIN が作成したものを使用しています。Underwood、J.（2013）. *Musselwhite and Ingram's Change Style Indicator*. https://innovategov.org/2013/08/15/musslewhite-and-ingrams-change-style-indicator/

- **コンサーバー**は、未知のものやそれがもたらす不確実性への恐れから、変化に抵抗します。（保守派）
- **プラグマティスト**は、変化の連続体の中央に位置し、決定的に必要な時には変化する。（実践派）
- **オリジネーター**は、変化を導入することに全く抵抗がなく、「試してみてどうなるか」という考え方を持っています。（革新派）

認識には大きな違いがありますが、それぞれが変化のイニシアチブに役立つ貴重な洞察力と特性を持っています。

コンサーバーの慎重な性格は、計画作りの役割に活かされるべきです。彼らは漸進的（進化的）な変化を好み、ビジネスに大きな混乱を招くことなく組織を移行させる方法で変化を実行します。

プラグマティストはファシリテートし、協力し、仲裁をします。一方でオリジネーターはビジョン、エネルギー、新規性を提供します。

チームを中心に考えるプラグマティストは、双方の立場を理解し、受け入れ可能な共通点を見出すための仲裁者を務めることもあります。適切なトレーニングやパイロットプロジェクトを行うことで、プラグマティストは変化を受け入れることができるでしょう。

オリジネーターは、変革のイニシアチブにおけるビジョンを持った人です。しかし、大局的な視点を持つ彼らにとって、実施の現実性や実用性は後回しになってしまうという欠点があります。

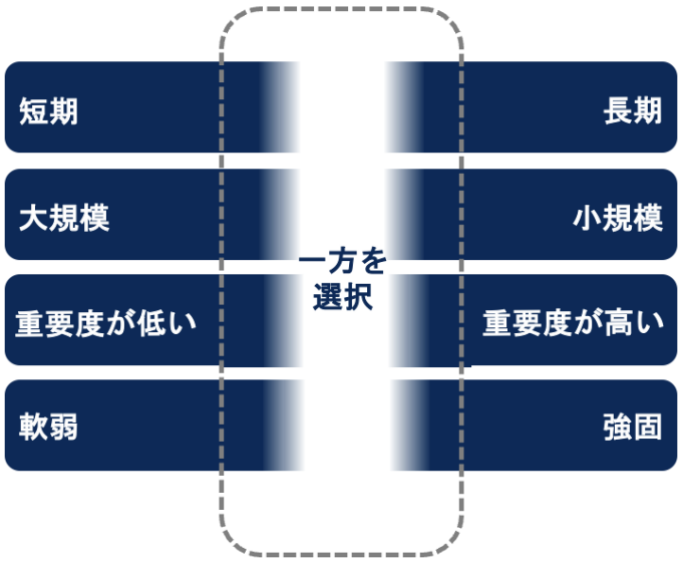
新しいアイデアを適用しようとするオリジネーターの意欲を組織が活用すれば、組織は新しいビジネスのやり方や、効率と効果の流れを生み出し、その結果、より収益性の高い組織となり、顧客にとっての価値も高まります。

## 13.3 パイロットを使った実装方法

組織がアジャイルスクラムを採用していることが知られると、すべての人がプロジェクトに注目することになります。成功を望む人も居れば、失敗を望む人も居ますが、大半の人はその判断を後回しにするでしょう。

最初のプロジェクトを成功させることは非常に重要であり、そのためにはパイロットプロジェクトを慎重に選択する必要があります。最初のパイロットプロジェクトを選ぶ際には、4 つの主要な指標に関するゴルディロックス（Goldilocks）の法則に従うことをお勧めします。

*図 29 良いパイロットプログラムの選択*




図は EXIN 制作：Cohn、M.（2009）. *4 Attributes of the Ideal Pilot Project.*
https://www.mountaingoatsoftware.com/blog/four-attributes-of-the-ideal-pilot-project

**期間**

非常に短いプロジェクトを選択すると、スクラムに反対する人たちが「スクラムは短いプロジェクトでしか機能しない」と主張することになります。

しかし、あまりにも長いプロジェクトを選択してしまうと、プロジェクトが終わるまで成功したとは言えなくなるリスクがあります。

従来の管理型プロジェクトの多くは、順調に進んでいると主張していながら、結局は予算超過や遅延が発生しています。そのため、スクラムプロジェクトが同じことを主張したところで、あまり説得力がありません。

プロジェクトの長さは、組織にとって通常の長さの中間に近いものを選ぶのがよいでしょう。6 カ月以上の場合は、成果を明確にしたサブプロジェクトに分割し、最初の 3 カ月間をショーケースまたはパイロットとして実施してください。

これは、チームがスプリントでの作業に慣れ始め、スプリントを楽しみ、プロダクトとチームの両方にとっての利益を確認するのに十分な時間だからです。また、3～4 ヶ月のプロジェクトは、スクラムがさらに長期の大規模プロジェクトの成功につながることを示すのに十分な期間です。

**サイズ**

できる限りメンバー全員が一か所に集められる 1 つのチームから始められるプロジェクトを選びましょう。

パイロットプロジェクトが最終的に多くのチームを含むように成長する場合でも、必ず 1 つのチームから開始してください。また、そのような成長するプロジェクトが組織内では当たり前であっても、5 つのチームを超えないようにパイロットを選択してください。

失敗をしないように、最初は多くのスクラムチーム間で作業を調整する必要のあるようなプロジェクトを選ばないでください。おそらく、1 つのチームから 5 つ以上のチームに成長する時間はないでしょう。

**重要性**

重要度が低く、リスクも低いプロジェクトをパイロットとして選びたくなるかもしれません。うまくいかなくてもリスクは低く、失敗しても気づかれないかもしれません。

しかし、批評家たちは、あなたが何をしているのか目を光らせ、「どうせ簡単なことだから」と、あなた達の努力を無視するでしょう。

代わりに、比較的リスクの低い重要なプロジェクトを選びましょう。スクラムに移行するために、チームとして正しく行うべきことの幾つかは難しいことであることを忘れないでください。重要でないプロジェクトでは、そうした求められる全てのことを行うことはないかもしれません。

**ビジネススポンサーとの関わり**

スクラムを採用するとは、技術的なプロジェクトリソースだけでなく、ビジネス全体に変化が求められます。アジャイルに関心を持ち、パイロットチームと一緒に働く時間と意欲を持つビジネススポンサーの存在は非常に重要です。

献身的で熱心なビジネススポンサーは、チームが対立を解決するのを助け、部門や個人によって厳格に実施されている凝り固まったプロセス、プラクティス、コントロールに対して、チームが挑戦するのを助けます。プロダクトオーナーは、特にパイロット期間中、ビジネスとのコミュニケーション、およびビジネスとスクラムチームとのコミュニケーションを推進する上で重要な役割を果たします。このような時期には、プロダクトオーナーが活動する場面が増えてくることは珍しくありません。

同様に、適切なスポンサーシップは、マネジメントのコミットメントを示しており、プロジェクトの成功を促進する有効な手段となります。特に、予想以上にうまくいったと彼らが言う時はそうした時です。

## 13.4 アジャイルスクラムを組織内に広める

パイロットを成功させた後はどうなるのでしょうか？アジャイルスクラムの使用を組織内でどのように拡大していくのでしょうか？

アジャイルを採用するためには、組織の他のメンバーにもアジャイルを導入してもらう必要があります。その方法としては、一気に導入するビックバンアプローチと段階的に導入する方法があります。多くの組織は、ビッグバンがうまくいくと言いますが、段階的なアプローチを用いることが推奨されます。

また、ある分野で成功した後に、別の分野でもうまくいくことを示すために、複数のパイロットを行うこともあります。よく耳にする言葉で、「XYZ 部門では問題ないが、我々の分野は同じではないのでうまくいかないだろう」という意見をよく耳にします。

垂直方向と水平方向へのスケーリングアプローチがあります。

- **垂直方向へのスケーリング** ビジネスユニット内へのスケーリング
- **水平方向へのスケーリング** 組織を横断したスケーリング

垂直方向に拡大することは、水平方向に拡大することよりも簡単です。同じビジネスユニット内の他のチームへアジャイルの使い方を教えるのは、環境が似ていて、学んだ教訓の多くをビジネス全体に容易に適用できるため、はるかに簡単です。

パイロットは、スクラムへの様々なチームの貢献とスクラムを使用することの価値を確立し、スクラムチームは機能横断型であるため、ビジネスユニット内の複数のチームメンバーがパイロットに参加することになります。

**Split-and-seed approach：分割と萌芽アプローチ**

パイロットスクラムチームの 1 人または複数のチームメンバーが、新しいスクラムチームのメンバーになると、スクラムの実務を経験した人が、問題や疑問を抱えて新しい作業方法について何かを知りたい学びたいと思っている、他のスクラムチームのメンバーの中に加わり一緒になることを意味します。このアプローチは、しばしば分割と萌芽アプローチと呼ばれます。

**Grow-and-split strategy：成長と分割戦略**
暫定的な戦略として、「成長と分裂」を適用することもできます。このシナリオでは、パイロットに参加するチームメンバー数を増やし、古いメンバーが2～3 回のスプリントで新しいメンバーを助け、その後、大きくなった新しいグループを分割して新しいチームを生み出します。このシナリオでは、新しいチームを生むときに、経験豊富なメンバーと経験の浅いメンバーの両方で編成されます。

**コーチング**
コーチングの活用も、アジャイルプログラムを拡大するための有効な仕組みであり、社内外のコーチを利用することができます。

ここでは、コーチは担当するチームと週に数時間を共に過ごし、スクラムの最適な使い方や、組織のコンテキストの中で得られた教訓などを共有します。要するに、社内コーチはコーチを務めるスクラムチームの常任メンバーではありません。

社内のコーチはコンテキストをよく理解していますが、社外のコーチはスクラムをよく知っていて、より多くの経験を持っているかもしれません。

スクラムを組織全体に普及させるためには、ここで紹介した技法を組み合わせて使うことをお勧めします。

## 13.5　人への対応

スクラムへの移行に抵抗する人にはさまざまな理由があります。ある人は理路整然とした論理で猛烈に主張するかもしれませんが、別の人は変更の試みを静かに妨害する事で抵抗するかもしれません。

Mike Cohn 氏は、著書『*Succeeding with Agile*』（2009 年）の中で、この点についていくつかの素晴らしい洞察を述べており、ここではそのいくつかを紹介します。

誤解された理解に基づいて、間違った議論をしたり行動したりするのをよく耳にしますが、それは、抵抗感によって正しいメッセージを受け入れることができないからです。

例えば、誰かが次のようなことを言ったり、考えたりするかもしれません。*「あなたは、ドキュメントなしが良いアイデアだと思いますか?ドキュメントがないところを見せてやるよ。」*アジャイルの原則が定義されたとき、ドキュメントが重要ではないということを意味する意図はありませんでした。しかし、変化に抵抗する人たちは、このような言葉につられて、たとえチームが何かをすべきだと合意していたとしても、何も書かずに進めてしまうかもしれません。

また、変化を静かに無視して、できるだけ古いやり方で作業し、次の変化がやってきてスクラムを一掃するのを待っている人もいるでしょう。

**では、人は行動を変えることに対して、どのように抵抗し、どのように対処するのか？**
まず始めに、人々はどのように抵抗するのか?積極的あるいは消極的かを特定することです。

- **積極的な抵抗**とは、アジャイル・スクラムの導入を妨げたり、失敗させたりするような行動をとることです。
- **消極的な抵抗**とは、誰かが行動を起こさない、行動を怠る場合は明らかです。彼らは、長い間無視していれば、消えてなくなると思っています。

第二に、人々が変化に抵抗している理由を明らかにする必要があります。ここでは、たいてい2つの主な理由が明らかになります。

- **現状維持が好き**：未知への恐れ
- **スクラムが嫌い**：アジャイルやスクラムに対する純粋な嫌悪感

彼らは、参考になるような嫌な経験をしているかもしれません。(それが自分自身の経験であるとは限りません)、あるいは、アジャイルで広められた考え方が、彼らにとってあまりに異質であり、それゆえ異議を唱えているのかもしれません。

つまり、人がどうやって抵抗するかは2通りあり、抵抗する理由も2通りあるということです。それをもとに、2x2のマトリックスを作ると、以下のようになります。

*図 30 変化の抵抗に対処する方法を理解する*




図は EXIN 制作：Cohn、M.（2010）. *Succeeding with Agile*. Addison-Wesley.

各 4 象限は、縦横軸のラベルが示す方法で抵抗する人の区分を表す名前が付けられています。

## 13.5.1 スケプティック（懐疑者）

スケプティック（懐疑者）とは、スクラムの原則や実践に賛同せず、移行にただ消極的に抵抗する人たちのことです。受動的抵抗者はスクラムに対して丁寧に反論し、デイリースクラムに参加することを忘れることが多いです。彼らは、移行を止めようとしている人たちであり、少し違和感を感じつつ試してみたいと思っている人たちとは対照的です。

スケプティック（懐疑者）の抵抗を克服するための有効な方法としては、次のようなものがあります。

- 時の経過に任せる
- トレーニングを提供する
- 仲間たちの話を聞く
- スケプティック主義者（懐疑主義者）に任命する
- 論争を進める
- 相互理解を築く

### 13.5.2　サボタージュ（妨害者）

スケプティックと同様に、サボタージュ（妨害者）はスクラムに対する嫌悪感から移行に抵抗します。しかし、サボタージュは、移行の努力を台無しにしようとすることで積極的な抵抗を行います。

ここでは、サボタージュを扱うときに役立つ方法を紹介します。

- 成功を示す
- 繰り返し粘り強く関わり合う
- 異動させる
- 彼らを解雇する（彼らが「井戸に毒を入れる」ので、時にはこれが唯一の解決策となる）
- 適切な人と話す

### 13.5.3　ダイハード（頑固者）

左側の人は、現状に満足している人たちです。彼らは現在の方法で物事を行うことに慣れており、彼らの仕事は過去幾度も同僚と一緒に高い評価の価値をもたらしています。原則として、彼らはスクラムに反対することはないが、現在の地位を危険にさらすような変化には反対します。

ダイハードは現状維持を好み、変化することに積極的に抵抗します。他の人を味方につけて、変化を防ごうとすることもあります。

ここでは、ダイハードに対処する方法は次のとおりです。

- インセンティブを並べる
- 現状に不満を抱かせる
- 脅迫感と向き合い、理解する

### 13.5.4　フォロワー（追従者）

フォロワーは現状維持を好み、変化には消極的である。フォロワーは、変化の可能性に怒ることはないが、変化が一過性の流行であることを期待して、できる限り何もしない。彼らを味方につけるには、スクラムが新しい現状になったことを示すことです。

最後に、フォロワーに対応する際に採用するとよい方法を紹介します。

- チームの構成を変える
- 正しい行動を褒める
- 彼らを巻き込む
- 自分自身が正しい行動のモデルとなる
- 真の障壁を見極める

### 13.5.5　抵抗の種類に応じた対処法

Mike Cohn のアドバイスは、実用的でシンプルなものです。しかし、変化に抵抗する人々にどう対処するかを決める前に、あなたが関わっている組織の文化、社会性、法的背景などを考慮しなければなりません。

## 13.6　適切な環境の構築

アジャイル・スクラムへの移行を開始する際に最初に直面する課題の1つは、自己管理型の機能横断的チームとして働くことです。アジャイルのバリューステートメントを考慮すると、アジャイルチームは本来以下のような働き方をしていると推測できます。

メンバーは...

- 一つのスクラムチームとして活動する
- 一度に一つの仕事をこなす
- 短い反復で仕事をする
- イテレーションごとに価値あるものを提供する
- ビジネス上の優先順位と成果に焦点を当てる
- すべてを検査し適応する

上記を見ると、従来の組織構造、設備、説明責任、職務、または役割にはいくつかの課題があることがわかります。

スクラムチームが機能横断的であっても、緊密に連携し、定期的にコラボレーションを行う必要がある場合は、物理的なスペースを設けるのがベストだと思われます。

**ビジュアルマネジメント**が重要であり、チーム全員がスクラムボード、カンバンボード、Nexus ボードなどの重要な情報ラジエータを見ることができなければなりません。そのためには、オープンスペースで再構成可能な作業スペースが理想的です。とはいえ、時には誰かが集中したり、誰かと会ったりする必要があるかもしれません。その場合、共有スペースの退屈な雰囲気は逆効果になります。

この目的のために、ブレイクアウト施設を用意することが必要で、全員がこれらのエリアに座って恒常的に仕事ができない程度にする必要があります。日常的な活動は共有スペースで行われなければなりません。これは「ケイブ&コモン」と呼ばれることもあります。

特に、デイリースクラムを行うことができるスペースがあることが重要です。毎日のミーティングが終わっても、チームのアーティファクトが見えるようになっていなければなりません。

スクラム／カンバンボードにソフトウェアを使用する場合は、毎日のミーティングが行われる共有エリアの壁に恒久的に投影されるようにしてください。

## 13.7 バーチャルチーム、リモートチームとしての活動

2020 年のパンデミックから学んだことが一つあるとすれば、それはバーチャルチームはこれからも存在するということです。2020 年以前にも、多くのスクラムチームは同居しておらず、世界各地に散らばっていることが多く、バーチャルチームに所属している人なら誰でも知っているように、これはいくつかの課題をもたらします。

**しかし、バーチャルチームとは何か?**
バーチャルチームとは、物理的に同じ場所にいなくても、同じ目的のために働く人たちで構成されるチームのことです。チームメンバーは、自宅で仕事をすることもあれば、異なる都市や国のオフィスで仕事をすることもあります。バーチャルチームを機能させるためには、バーチャルチーム成功の方程式を知っておくことが重要です。

現在、バーチャルチームの問題とその働き方については様々な見解がありますが、良いニュースとしては、成功するバーチャルチームとして働くための一部が、アジャイルスクラムの DNA にすでに組み込まれているということです。複数の分野にまたがるチームがまとまりのあるスクラムチームとして協働するための原則と実践は、バーチャルチームがまとまりのあるスクラムチームとして協働するためにも役立ちます。

では、バーチャルスクラムチームについて、どのように機能するのかを見てみましょう。

- **すべてのスクラムの活動のための明確なプロセスを作成します。**これは詳細である必要はありませんが、何がいつ、どのように起こるのかを全員が正確に知っている必要があります。以下を参照ください。

- 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
- プロダクトバックログリファインメント
- アドホックな問題解決のための会議

- **共通のコラボレーションツールと管理ツールを使用する。**どのツールを使うにしても、全員がタスク、進捗状況、課題を同じように見られるようにし、いつでも見えるようにします。
  - また、コラボレーションツールがチームのために確実に機能するようにし、**フェイス・ツー・フェイスのコミュニケーションが非常に重要である**ことを認識してください。コミュニケーションの 70%は非言語で行われるため、お互いの顔を見ることは非常に重要です。

  最も優れたコミュニケーションの形は、人々が一つの部屋にいることですが、現実にはそれが実現できないことが多いのです。したがって、すべてのバーチャルミーティングは、音声だけでなく、**ビデオベースのチームミーティング**でなければなりません。また、ビデオであるということは、マルチタスクに対応できないということでもあります。
  - 参加者は会議に参加する時、有益な貢献ができるよう、例えばベッドや電車内からではなく、**いつも決められた場所**から参加するべきです。
  - 会議中に質問に答えたり、問題について投票したりする必要がある場合には、**インタラクティブな質問や投票ソフトウェアを使用し**て、交流を深めます。
  - また、チームの WhatsApp グループを作成するなど、**日常のカジュアルな交流**のためのチームスペースを作成することもできます。

- たとえ何度も同じことが繰り返されたとしても、**すべての活動に対する期待値が前もって設定されていること**を確認してください。
  - この活動で、チームは参加者に何を期待していますか?
  - 参加者はチームに何を期待していますか?
  - イベントから期待されるアウトプットや成果は何ですか?
  - そのイベントの規範の一部を形成できないと予想される場合は、全員にそれを強調する機会を与えます。
  - 誰も会議のルールを守らなければ、その会議は終了し全員のスケジュールを変更しなければなりません。一度でもルールは破らないで下さい。

- **良いファシリテーターが重要です。**ここでは、スクラムマスターが大きな貢献をすることができます。しかし、誰の会議であるかを忘れてはいけません。デイリースクラムは開発者のミーティングであり、プロダクトバックログのリファインメントはプロダクトオーナーのミーティング etc.のように、彼らがあなたをファシリテーターに任命しなければ、あなたはファシリテーターになれません。
  - パワーポイントのようなツールは、一方通行のコミュニケーション媒体に過ぎないので避けましょう。バーチャル会議では、全員の会話と参加がより重要になります。
  - Mike Dwyer 氏からの素晴らしいヒント – 彼はこれを「NOSTUESO ルール」と呼んでいます。「**みんなが一度話すまで、誰も二度話さない（No One Speaks Twice Until Everyone Speaks Once）**」ということです。
  - これは、人々が発言するためのスペースを作ることでもあります。調査によると、最初の 5 分間に発言する参加者は、再度発言する確率が 3 倍だそうです。
  - また、女性が最初に発言すると、他の女性の参加率が格段に高くなることがわかっています。
  - スクラムチーム内で良好な関係を築き、積極的かつ率直なコミュニケーションを図ります。

# 14 説明責任 – スクラムと Nexus のイベントとプラクティス

この本の中では、イベントやプラクティスが参照され、誰がそれを行うべきかが示されています（例えば、スクラムマスター、開発者、プロダクトオーナーなど）。しかし、それぞれが何に対して説明責任と実行責任を負うのか、明確なリストはどこにもありません。

しかし、スクラムにおける主要な説明責任それぞれに何が求められているかを理解することは、もちろん重要です。本章は、各説明責任ごとに 3 つのセクションに分かれています。それぞれのセクションでは、各役割の説明責任と実行責任をスクラムのイベントとプラクティスに照らして説明します。

## 14.1 プロダクトオーナー

### 14.1.1 スプリントプランニング

| | 説明責任と実行責任 |
|---|---|
| 出席者 | スクラムチームだが、開発者が所有している。 |
| スプリントプランニングの準備 | プロダクトバックログが洗練され、現在のビジネスニーズに沿っていることを確認する。 |
| スプリントゴールの設定 | プロダクトゴールを再確認し、全員が現在のビジネスニーズを知っていることを確認する。スプリントゴールに同意する。 |
| スプリントバックログの定義 | 必要に応じて意見を述べ、ビジネスニーズがバックログアイテムに反映されるようにする。不一致があればそれを強調し、可能であれば開発者と交渉して変更する。 |
| 見積もりとタスク分解の実施 | 前提を検証し、要求やビジネスニーズをより明確にする。 |
| スプリントバックログの作成 | 相談を受けた場合、意見を述べる。 |
| プロダクトバックログリファインメント | チームメンバーからの意見を収集し、プロダクトバックログを更新し、新たな洞察に基づいてアイテムを並べ替えることができる。 |
| | |

### 14.1.2 デイリースクラム

| | 説明責任と実行責任 |
|---|---|
| 出席者 | 開発者とスクラムマスター – 時には、プロダクトオーナーを含むその他のステークホルダー。 |
| 検査と適応 | 必要に応じて意見を述べたり、質問をしたりする。 |
| プロダクトバックログリファインメント | チームメンバーからの意見を集め、プロダクトバックログを更新し、新たな洞察に基づいてアイテムを並べ替える。能力、必要性、時間、予算を考慮する。 |
| | |

### 14.1.3 スプリントレビュー

| | 説明責任と実行責任 |
|---|---|
| 出席者 | スクラムチームと、プロダクトオーナーが特定した重要なステークホルダー。プロダクトオーナーが議長を務めることもあるが、チームのミーティングである。 |
| キックオフ | 出席者がなぜその場にいるのかを理解するようにする。このイベントはスクラムチーム以外のステークホルダーを扱うため、スプリントレビューではプロダクトオーナーが主導することになる。 |
| デモンストレーションと検査 | デリバリーされたインクリメントのデモンストレーションを支援する。 |
| 自分の行動に対するフィードバック | 積極的に質問してください。 |
| 環境の変化をフィードバックする | 環境の変化を記録する。 |
| 課題・問題点 | 具体的には、納品の順番に影響を与える可能性のある依存関係に注意してください。 |
| プロダクトバックログリファインメント | チームメンバーからの意見を集め、プロダクトバックログを更新し、新たな洞察に基づいてアイテムを並べ替える。能力、必要性、時間、予算を考慮する。 |
| | |

### 14.1.4 プロダクトバックログの作成と管理

| | 説明責任と実行責任 |
|---|---|
| 要求の収集 | 集めた要求を受け入れる。プロダクトオーナーが自ら要求を収集することもあるが、顧客やユーザー、アナリスト、その他のチームメンバーが収集することもある。プロダクトオーナーは要求を吟味する必要があり、バックログに追加する前に、元に戻って要求をリファインすることもよくある。 |
| 初期のバックログリファインメント | 最初の要求を検討し、チームメンバーから意見を集め、ビジネス要求やその他の洞察に基づいてプロダクトバックログを作成する。能力、必要性、時間、予算を考慮する。 |
| 継続的なバックログリファインメント | チームメンバーからの意見を収集し、プロダクトバックログを更新し、新しい洞察力や変化するビジネス要求に基づいてアイテムを並べ替える。能力、必要性、時間、予算を考慮する。 |
| | |

### 14.1.5 スプリントレトロスペクティブ

| | 説明責任と実行責任 |
|---|---|
| 出席者 | スクラムチーム。このイベントは、他の説明責任、または役割の人からの意見をインプットに、開発者が進める。 |
| 検査 | 貢献はしてもリードはしない。 |
| 適応 | 貢献はしてもリードはしない。 |
| 学びの共有 | 学習は主にチームに焦点を当てていますが、プロダクトオーナーは、同じバックログに対して作業している他のスクラムチームと学習内容を共有することもできる。 |
| | |

### 14.1.6 Nexus スプリントプランニング

| | 説明責任と実行責任 |
|---|---|

| | |
|---|---|
| 出席者 | 参加している Nexus チームのメンバーとプロダクトオーナー 1 名。 |
| スプリントプランニングの準備 | Nexus は決して横断的なバックログで実行されるものではない。プロダクトオーナーは、より支配的な役割を果たすが、Nexus の計画に関わるチームを支援する役割でもある。実際には、プロダクトオーナーは Nexus の招集者であり、Nexus を構成し、チームを Nexus に参加させるという重要な役割を担っている。それ以外の役割は、通常のスプリントプランニングと同様。 |
| キックオフ | プロダクトオーナーは、Nexus のオーナーでもあり、招集者でもある。したがって、キックオフミーティングの計画と実行は、プロダクトオーナーの説明責任でもある。 |
| | |

### 14.1.7 Nexus スプリントレビュー

| | 説明責任と実行責任 |
|---|---|
| 出席者 | 参加している Nexus チームのメンバーとプロダクトオーナー 1 名。 |
| 実行 | Nexus のスプリントレビューの内容や形式はスプリントレビューと似ていて、調整と計画が必要となる。独立したスプリントレビューはなく、Nexus のスプリントレビューだけがある。このミーティングは、プロダクトオーナーが所有し、運営する。 |
| | |

### 14.1.8 Nexus スプリントレロスペクティブ

スプリントレトロスペクティブと同じです。

| | 説明責任と実行責任 |
|---|---|
| 出席者 | スクラムチーム。このイベントは、他の説明責任、または役割の人からの意見をインプットに、開発者が進める。 |
| 検査 | 貢献はしてもリードはしない。 |
| 適応 | 貢献はしてもリードはしない。 |
| 学びの共有 | 学習は主にチームに焦点を当てているが、プロダクトオーナーは、同じバックログに対して作業している他のスクラムチームと学習内容を共有することもできる。 |
| | |

### 14.1.9 Nexus プロダクトバックログリファインメント

Nexus プロダクトバックログリファインメントは、継続的な活動であるだけでなく、Nexus イベントとして定義されています。このイベントは、プロダクトオーナーによって運営・調整されます。

## 14.2 スクラムマスター

重要な注意事項：スクラムマスターはスクラムチームのメンバーを支援するものであり、イベントやアクションを推進するもので*はありません*。彼らはファシリテーション、コーチ、アシストを行います。

### 14.2.1 スプリントプランニング

| | 説明責任と実行責任 |
|---|---|
| 出席者 | スクラムチームだが、開発者が所有している。 |
| スプリントプランニングの準備 | ミーティングのコーディネートやセッティングを手伝う。 |
| スプリントゴールの設定 | ファシリテーター、仲裁者として行動し、衝突の解決を助ける。 |
| スプリントバックログの定義 | ファシリテーター、仲裁者として行動し、衝突の解決を助ける。 |
| 見積もりとタスク分解実施 | ファシリテーター、仲裁者として行動し、衝突の解決を助ける。 |
| スプリントバックログの作成 | スプリントバックログは開発者のものあるが、開発者はしばしばスクラムマスターに、スプリントバックログの作成、文書化、維持の支援を求める。 |
| プロダクトバックログの精査 | ファシリテーター、仲裁者として行動し、衝突の解決を助ける。 |
| | |

### 14.2.2 デイリースクラム

| | 説明責任と実行責任 |
|---|---|
| 出席者 | 開発者とスクラムマスター – 時には、プロダクトオーナーを含むその他のステークホルダー。 |
| 検査と適応 | 会話を促進し、必要に応じてコーチングを提供し、必要に応じて仲裁を行う。 |
| 阻害要因への対応 | スクラムマスターの重要な役割は、開発者が障害に対処するのを支援することです。通常、これには、会議の進行、支援、リソース、スキルの獲得、影響を受けるステークホルダーとの頻繁なコミュニケーションとフォローアップが含まれる。 |
| 進捗状況の確認 | 情報ラジエータを更新し続けることは、通常スクラムマスターが行うが、それは彼らの責任ではない。このタスクは開発者からスクラムマスターに委ねられることが多い。 |
| チームのコーチング | スクラムマスターの重要な役割は、チームがスクラムの実践に習熟し、スクラムの最新動向を把握できるようにコーチングや指導を行うこと。 |
| プロダクトバックログの精査 | 会話を促進し、必要に応じてコーチングを提供し、必要に応じて仲裁を行う。 |
| | |

### 14.2.3 スプリントレビューとレトロスペクティブ

| | 説明責任と実行責任 |
|---|---|
| ファシリテーション | 会話を促進し、必要に応じてコーチングを行い、必要に応じて仲裁を行う。スクラムマスターは、スプリントレトロスペクティブを幅広い範囲で準備する。 |
| | |

### 14.2.4 プロダクトバックログの作成と管理

| | 説明責任と実行責任 |
|---|---|
| 要求収集 | この活動では、スクラムマスターがプロダクトオーナーをサポートする。 |
| 初期のバックログリファインメント | 会話をファシリテートし、必要に応じてコーチングを提供し、必要に応じて仲裁を行う。 |
| 継続的なバックログリファインメント | 会話をファシリテートし、必要に応じてコーチングを提供し、必要に応じて仲裁を行う。 |
| | |

### 14.2.5 Nexus

参加するスクラムチームのスクラムマスター全員が、Nexus スプリントプランニング、Nexus スプリントレビュー、Nexus レトロスペクティブなどの Nexus 固有のイベントに参加するわけではありません。しかし、少なくとも 1 人のスクラムマスターは、スクラムマスターとして Nexus 統合チームに参加しなければなりません。この役割では、多くの時間がチーム間の活動の調整と改善に費やされます。これはフルタイムの仕事ではないかもしれませんが、複雑な Nexus の中で永続的で中心的な役割になればよいでしょう。Nexus の仕事は、常にスクラムチームの仕事よりも優先されます。

## 14.3 開発者

### 14.3.1 スプリントプランニング

| | 説明責任と実行責任 |
|---|---|
| 出席者 | スクラムチームだが、開発者が所有している。 |
| スプリントプランニングの準備 | スプリントプランニングは開発者の仕事ですが、開発者はミーティングの準備をスクラムマスターに任せることが多い。 |
| スプリントゴールの設定 | スプリントバックログは、開発者がプロダクトオーナーと相談して決定する。 |
| スプリントバックログの作成 | スプリントバックログの項目は、開発者がプロダクトオーナーの意見を参考にして選択し、時にはスクラムマスターの助けを借りる。 |
| 見積もりとタスク分解の実施 | 開発者は、プロダクトオーナーやスクラムマスターの支援を受けて、見積もりやタスク分解を行う。 |
| スプリントバックログの作成 | 開発者はスプリントバックログを作成しますが、しばしばスクラムマスターにスプリントバックログの管理人を依頼する。 |
| プロダクトバックログのリファインメント | 開発者はプロダクトオーナーがプロダクトバックログをリファインメントするのを手伝うべきである。これは継続的な活動であり、スプリントプランニング中に、その中で計画中に学んだことに基づいて簡単なリファインメントセッションを行うのが一般的。 |
| | |

### 14.3.2 デイリースクラム

| | 説明責任と実行責任 |
|---|---|
| 出席者 | 開発者とスクラムマスター – 時には、プロダクトオーナーを含むその他のステークホルダー。 |
| 検査 | 開発者は進捗状況を報告し、依存関係や障害物を発見する。 |
| 適応 | 開発者は、障害や依存関係をどのように処理するかを決定し、外部からの支援が必要な場合は、より良い結果を導くためにスクラムマスターやプロダクトオーナーに依頼することもある。 |
| 進捗状況の確認 | 進捗状況の把握は開発者の責任ですが、これはしばしばスクラムマスターに委ねられる。 |
| プロダクトバックログリファインメント | プロダクトバックログリファインメントが必要となるような新しい情報があれば、開発者が問題を明らかにし、プロダクトオーナーがリファンインメントの責任を果たす。 |
| | |

### 14.3.3 スプリントレビュー

| | 説明責任と実行責任 |
|---|---|
| 出席者 | スクラムチームと、プロダクトオーナーが重要と見なしたステークホルダー。プロダクトオーナーが議長を務めることもある。但し、チームのミーティング。 |
| キックオフ | このイベントはスクラムチーム以外のステークホルダーを相手にするため、プロダクトオーナーがスプリントレビューを主導する。 |
| デモンストレーションと検査 | 開発者は他のステークホルダーに自分たちの成果を見せ、質問に答える。 |
| 自分の行動や環境に対するフィードバックを得る | スプリントレビューはサインオフミーティングではなく、ステークホルダー、具体的には顧客やユーザーからのフィードバックを得る機会であることに注意すること。ステークホルダーは、スクラムチームが気づいていなかった新しい要求や環境の変化を共有することがよくある。また、フィードバックセッションでは、時折、問題点や課題が強調されることもあるが、すぐに制御できなくなる可能性があるため、これは推奨できない。 |
| プロダクトバックログリファインメント | 新しい情報、洞察力、フィードバックなど、これらの情報がチームの頭の中でまだ新鮮な間が、プロダクトバックログリファインを行う絶好のタイミング。 |
| | |

### 14.3.4 プロダクトバックログの作成と管理

| | 説明責任と実行責任 |
|---|---|
| 出席者 | プロダクトオーナーが開発者の協力を得て行う。 |
| 要求収集とプロダクトバックログリファインメント | 開発者は最初の要求収集に関わることはあまりないが、スプリント中のユーザーとのやり取りや、スプリントの計画・実行時に新しい要求を発見することがよくある。要求は、通常、プロダクトバックログリファインメントの中で、プロダクトオーナーに伝えられる必要がある。 |
| | |

### 14.3.5 スプリントレトロスペクティブ

| | 説明責任と実行責任 |
|---|---|
| 出席者 | スクラムチーム。 |
| 検査 | 様々な技法があるが、その一つが「何がうまくいったのか」「何が間違っていたのか」「何を改善すればいいのか」を問うことです。ここでは、スプリント中に集められた課題やブロッカーチケットの分析が役立つ。 |
| 適応 | 同様に、チームは何を変えるべきか、何を改善すべきか、何を避けるべきかを自問し、それを実行可能な計画に変えることで、改善の手段を生み出すことができる。 |
| 学びの共有 | 特に大規模な環境で実施すると、学んだ教訓を他の開発者と共有することが有効。 |
| | |

### 14.3.6 Nexus スプリントプランニング

| | 説明責任と実行責任 |
|---|---|
| 出席者 | 参加している Nexus チームのメンバーとプロダクトオーナー 1 名。 |
| コーディネート | Nexus スプリントプランニングは、Nexus 統合チームと呼ばれ、参加しているすべてのスクラムチームの代表者によって行われる。このチームには、参加しているスクラムチームから出向している経験豊富な開発者が参加することが多い。 |
| | |

### 14.3.7 Nexus スプリントレビュー

Nexus の期間は、統合されたスプリントレビューが実施され、そこでは Nexus の一部を構成するすべてのインクリメントを含めて、統合された Nexus の作業がレビューされます。最低でも、Nexus のすべてのスクラムチームの代表者が自分たちの作業を示し、フィードバックを得ます。Nexus スプリントレビューのそれ以外についてはスプリントレビューと同じになります。

| | 説明責任と実行責任 |
|---|---|
| 出席者 | Nexus スクラムレビューには、スクラムチームの代表者、プロダクトオーナー、そして選ばれたステークホルダー（顧客や主要ユーザー）が参加。 |
| | |

### 14.3.8 Nexus スプリントレトロスペクティブ

Nexus に関わるすべてのスクラムチームは、まず通常のスクラムレトロスペクティブを行い、チームの全メンバーが Nexus 全体のためにこの改善活動を繰り返す必要があります。ここで共有される教訓をすべてのチームに適用すれば、大きな利益につながります。

| | 説明責任と実行責任 |
|---|---|
| 出席者 | 構成するスクラムチームの全メンバー。 |
| | |

### 14.3.9 Nexus プロダクトバックログリファインメント

共有されたバックログリファインメントでは、プロダクトバックログアイテム（PBI）を十分に分解して、チームが自分たちの提供できる作業と、その作業を（今後のスプリントで）提供する順序を理解することに焦点を当てます。また、このイベントでは、チーム間の依存関係を最小限にし、取り除くことにも焦点が当てられます。

| | 説明責任と実行責任 |
|---|---|
| 出席者 | この活動には、最低でも Nexus 統合チームのメンバーが参加する必要がありますが、構成するスクラムチームから広く参加することを推奨。 |
| | |

# 付録 A – その他のアジャイル手法

これは、公開された情報ソースから派生したコンテンツに示されているいくつかのものから編集されたものです。このドキュメントの唯一の目的は、EXIN アジャイルスクラムプロダクトオーナーと、EXIN アジャイルスクラムマスターの認定資格の準備している読者のためのものです。

2001 年にアジャイル・マニフェストが発表されて以来、多くの企業が、チームが目標や成果を達成できるように、さまざまなアジャイル手法を採用してきました。最も有名な方法論はスクラムで、あらゆる企業で広く利用されています。初期の頃は、これらのアジャイル手法は IT 部門のみで適用されていました。それに対して、アジャイル手法は現在、IT 以外のさまざまなビジネス分野でも幅広く利用されています。

この付録では、スクラム以外で最も実践されているアジャイルメソドロジーの概要と簡単な説明を行います。

- クリスタル
- エクストリーム・プログラミング（XP）
- DSDM（現在は ABC と呼ばれています）
- LeSS
- SAFe
- カンバン

これらの手法はすべて異なりますが、共通しているのは「オープンなコミュニケーション」「協調や協働」「柔軟性や適応性」の 3 つであることを念頭に置いていることが重要です。これらの要素は、アジャイル・マニフェストの中核をなすものであり、すべての手法に共通しています。

## クリスタルの方法論[15]

クリスタルファミリーは、軽量なアジャイル方法論のグループです。これらの方法論が軽量であると考えられるのは、これらの方法論がプロセスを二次的に重要視し、代わりに他の要素に重点を置いているからです。

これらの要素は

- 人々
- 相互作用
- コミュニティ
- スキル
- 才能
- コミュニケーション

クリスタルメソドロジーは、1990 年代に Alistair Cockburn（アリスター・コーバーン）氏によって開発された。彼が行った調査の結果、開発者が正式な方法論を意図したとおりに使用していないにもかかわらず、プロジェクトを成功させていることがわかったことから作られた。Cockburn 氏は、いくつかの重要な側面の違いを示しています。

- **方法論**：一連の要素（例：プラクティス、ツール）のこと
- **技法**：スキル分野（ユースケースの開発など）
- **ポリシー**：組織がどのように行動すべきかを定義したもの

Cockburn 氏は、研究の結果、チームにおける人々の行動を次のように定義した。

- 人はコミュニケーションをする生き物であり、直接会ってリアルタイムで質疑応答をするのに最も適しています
- 人は長い間、一貫して行動することが苦手です
- 人は非常に変わり易く、日ごとに、場所ごとに変化します
- 一般的に人々は、良き社会人でありたいと考えており、周りを見渡し、率先して行動し、プロジェクトを成功させるために「必要なことは何でも」することが得意です

クリスタルの方法論は、8 つの色に分けられています。この色は、メソドロジーの「重さ」を示すのに使われます。6 人以下のプロジェクトに適した「クリスタル・クリア」から始まり、人命に関わるようなミッションクリティカルなプロジェクトに適した「クリスタル・ダイヤモンド」「クリスタル・サファイア」へと続きます。

また、重要な点として、クリスタル・ファミリーには 7 つの共通する性質があることにも注目します。これらの特性は

- 頻繁なデリバリー
- 自発的な改善
- 緊密で浸透するコミュニケーション
- 個人の安全性
- フォーカス
- エキスパートユーザーへの容易なアクセス

**頻繁なデリバリー**
開発中のソフトウェア/プロダクトのイテレーションを定期的にリリースする。

[15] Free after: Crystal Methods.（2021）. Retrieved on September 23, 2021 from https://en.wikiversity.org/wiki/Crystal_Methods

**自発的な改善**

プロジェクトチームは、時折、自分たちのプロセスをどのように改善するかを考えます。これは、開発作業から解放されるだけでなく、効率の向上にもつながります。

**緊密で浸透するコミュニケーション**

プロジェクトチームの中には、コミュニケーションの流れが必要です。これを実現するためには、プロジェクトチームが同じ部屋に集まり、プロジェクトの進行をサポートするコミュニケーションが促進されるようにする必要があります。

**個人の安全性**

チームの全員が、自由に発言できるオープンで安全な環境にいると感じなければなりません。質問をしたり、アイデアを提案したりしたときに嘲笑されるなどの否定的な反応は、完全に阻止しなければなりません。なぜなら、人々はお互いに信頼できなければならず、否定的な態度を受ける人は、もう積極的に参加しなくなる可能性が高いからです。

**フォーカス**

成功するためには、フォーカスを保つことが重要です。クリスタル・メソドロジーで「フォーカス」とは、「進捗」と「方向性」の2つのことを指します。進捗とは、プロジェクトのタスクに十分な時間を割いて確実に進捗させること。一方、「方向性」とは、プロジェクトがどのような方向に向かっているのかということです。

**エキスパートユーザーへの容易なアクセス**

開発中の新機能を実際に使用しているユーザーが、プロジェクトチームと連絡を取り合い、質問に答えたり、問題解決の手助けをしてくれる必要があります。このエキスパートは、実際の経験をもとにプロジェクトをサポートしてくれます。理想的には、ミーティング中や電話でも頻繁に連絡が取れるようにしておくべきです。

自動化されたテスト、構成管理、頻繁な統合など、技術的な環境が整っている必要があります。

問題（エラー、バグなど）を早期に発見するためには、頻繁な統合とテストが不可欠です。統合が継続的に行われていれば、早い段階で問題が解決されるため、問題の拡大を防ぐことができます。

## エクストリーム・プログラミング（XP）[16]

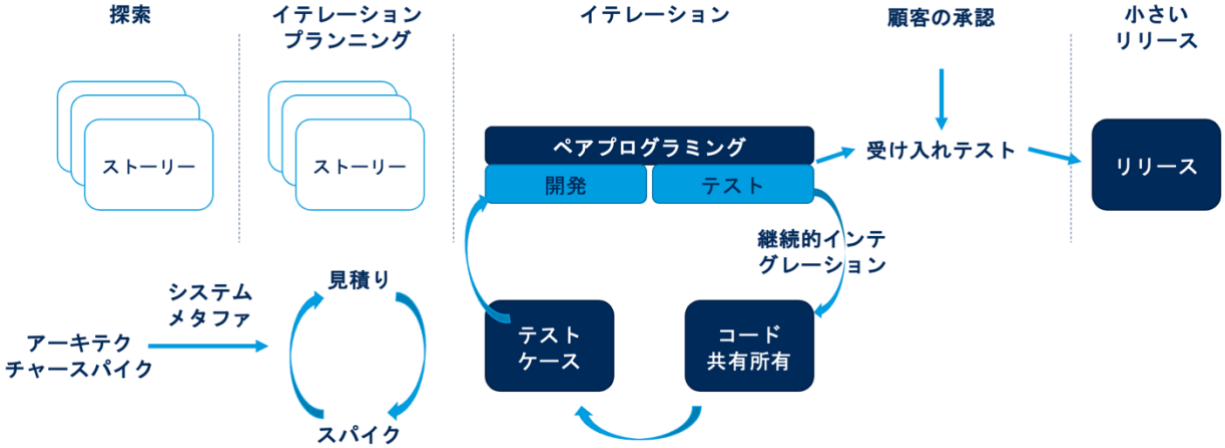
XP は、顧客満足に重点を置いたソフトウェア開発手法です。これにより、提供されているものが顧客のニーズに基づいていることを保証します。その方法の一つが、ユーザーストーリーを価値あるビジネス資産に変えることです。

XP 手法の特徴の一つは、継続的でオープンなコミュニケーションです。この重要な側面がなければ、期待通りの成果は得られません。このようなコミュニケーションスタイルの重要な利点は、顧客とプロジェクトチームの両者の間の信頼が高まることです。

XP は、サービスプロバイダーが新製品や新サービスを開発する際に直面する継続的な変化に対応するために、また変化する市場のニーズに応えるべく誕生しました。その成功の要因は、プロジェクトに関わるすべての関係者間の主なコミュニケーションとコラボレーションの側面にあると言えます。つまり、顧客、マネジャー、プロジェクトチーム、その他の関係者が、スコープが設定され、ストーリーが定義された時点から、テストや最終的なリリースに至るまで、協力して作業を進めるということです。

この方法論は小規模なプロジェクトチームのために設計されたものですが、大規模なチームも XP を使用しており、成功を収めています。

*図 31 一目でわかるエクストリーム・プログラミング（XP）の概要*




図は EXIN 制作：Meier、J. D.（2019）. *一目でわかるエクストリーム・プログラミング』*. https://jdmeier.com/extreme-programming-at-a-glance/.

## エクストリーム・プログラミング（XP）とは？

XP の最初のステップは、ユーザーストーリーの収集と、リスクとなりうるものに対するスパイクソリューション（小さな先行技術調査）の実施です。

最初のステップの後、リリース計画会議を開催する必要があります。この段階では、必要な人をすべて招待することが重要です。これには、顧客、プロジェクトチーム、マネジャーなどが含まれます。この段階での主な目的は、誰もが納得できる共通の目標を定めることです。この会議とゴールの設定に続いて、次のリリースに合意するためのイテレーション計画会議が行われます。

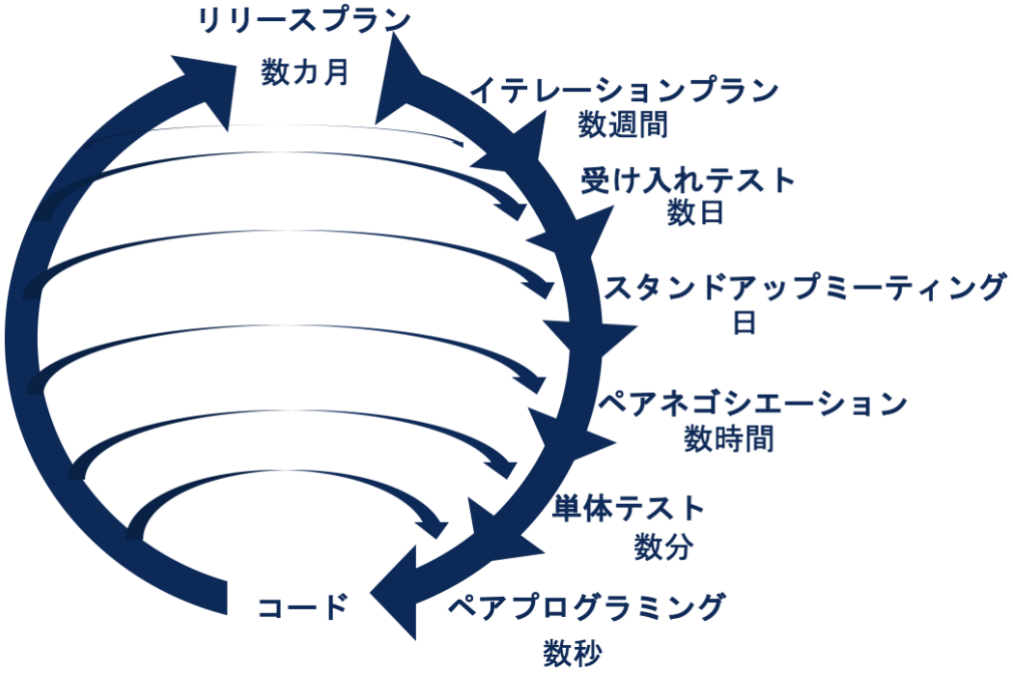
[16] Free after: Wells, D.（2013）. Extreme Programming: A gentle introduction. http://www.extremeprogramming.org/ and associated pages

この数ステップだけで、プロジェクトチームはすでに必要な機能を開発する方向に進んでいます。

## エクストリーム・プログラミングのルール

XP は、下図のようなフィードバックループが頻繁に発生します。

*図 32 XP でのループの計画*




図は EXIN 制作：Wells、D.（2000）. *Introducing Extreme Programming.*
http://www.extremeprogramming.org/introduction.html

これらのフィードバックループは、現在の視点では非常に長く感じられるかもしれませんが、1990 年代末には非常に画期的なものでした。

XP の核となるのは、ルールという考え方です。要するに、XP のルールとは

- プランニング
- マネジメント
- デザイン
- コーディング
- テスト

**プランニング**

- ユーザーストーリーの作成
- リリース計画でリリーススケジュールを作成
- 小さく頻繁なリリース
- イテレーションで区切られたプロジェクト
- イテレーションごとに始まるスプリントプランニング

**マネジメント**

- チーム専用のオープンなワークスペースを提供
- 持続可能な一定のペース設定
- スタンドアップミーティングを毎日開催
- プロジェクトのベロシティを測定
- 人々を動かす
- XP が壊れたら直す

**デザイン**

- シンプルさ
- システムのメタファーを用いる
- デザインセッションには CRC（クラス・レスポンシビリティ・コラボレーター）カードを使用する
- リスクを減らすためのスパイクソリューション（小さな先行調査）を作る
- 早期に機能が追加されることはない
- いつでも、どこでも、リファクタリングが可能

**コーディング**

- 顧客はいつでも利用できる
- コードは合意された基準に沿って書く
- ユニットテストから書く
- すべてのプロダクションコードはペアプログラミングされる
- 一度にコードを統合できるのは 1 ペアだけ
- 頻繁に統合する
- 統合のための専用環境を作る
- 共同所有する

**テスト**

- すべてのコードはユニットテストが必要
- すべてのコードはリリースまえにすべてのユニットテストをパスさせる
- バグが見つかればテストを実施する
- 受入テストを頻繁に行いそのスコアを公開する

## ペアプログラミング

ペアプログラミングとは、エクストリーム・プログラミング（XP）に由来するアジャイル手法のひとつで、2人の開発者が組んで 1 台のコンピュータ上で作業を行う。2人は共同でユーザーストーリーの設計、コーディング、テストを行います。理想的には、2人のスキルが同じくらいで、それぞれが同じようにキーボードを操作することが望ましいが、シニアプログラマーからジュニアプログラマーにスキルを伝授する効果的な方法でもある。

## DSDM[17]

DSDM（Dynamic Systems Development Method）は、プロジェクトの全ライフサイクルに焦点を当てたフレームワークです。今でも多くの人が DSDM と呼んでいますが、新しい組織名は「アジャイルビジネス・コンソーシアム」と呼ばれています。

このアプローチでは、次の項目に進むために、すべてのステップで最低限の作業のみを行うことを規定します。これには、プロジェクトには継続的な変化がつきものだという考え方が背後にあります。ビジネス要求は急に変更される可能性があることは周知の事実であり、必要なステップを完了したとみなすために必要な作業のみを行うことで、プロジェクトチームの労力、リソース、時間を節約することができます。

DSDM フレームワークは、新たな製品やサービスの開発から財務部門まで、組織内のさまざまな用途に使用することができます。ビジネスのどの部分でも恩恵を受けることができます。

## DSDM の原則

1. ユーザーの積極的な関与が不可欠
2. DSDM チームは意思決定の権限を持たなければならない
3. 頻繁にプロダクトをデリバリーすることに重点を置く
4. ビジネス目的への適合性が、成果物を受け入れる必須基準
5. 正確なビジネスソリューションに収束させるために、反復的かつ漸進的な開発が必要
6. 開発中のすべての変更は元に戻すことができる
7. 要求事項はハイレベルでベースライン化される
8. ライフサイクル全体で統合されたテストを実施
9. すべてのステークホルダーの協力と協調が不可欠

出典：アジャイルビジネスコンソーシアム

## DSDM フレームワークのフェーズ

DSDM フレームワークは 6 つのフェーズで構成されます。プレ・プロジェクト、ポスト・プロジェクトのフェーズがあり、そして、プロジェクトフェーズとして 4 つの主要なフェーズ：Feasibility（実現可能性）、Foundations（基礎）、Evolutionary Development（進化的開発）、Deployment（展開）がある。

フェーズ1．プレ・プロジェクト
フェーズ2．実現可能性
フェーズ3．基礎
フェーズ4．進化的開発
フェーズ5．デプロイメント
フェーズ6．ポスト・プロジェクト

### フェーズ1 – プレ・プロジェクト

プロジェクトの準備の一環として、明確な目的を定義します。このフエーズでは、プロジェクトが正しく設定され、適切なプロジェクトのみが開始されるようにします。

[17] Free after: Agile Business Consortium（2014）. The DSDM Agile Project Framework（2014 Onwards）. https://www.agilebusiness.org/page/TheDSDMAgileProjectFramework

## フェーズ 2 – 実現可能性

このフェーズの名前が示すように、このパートの焦点は、論点となっているプロジェクトが技術的および費用対効果の両方の観点から実現可能かどうかです。このフエーズでは、プロジェクトを検討する価値があるのか、それとも実行できそうにないのかを判断するだけなので多くの時間を割くべきではありません。

## フェーズ 3 – 基礎

このフェーズでは、仕事の範囲を理解することが目標となります。これには、実際の作業をどのように、誰が、いつ、どこで行うのかが含まれます。また、DSDM プロセスをどのように適用するかを検討し、プロジェクトのライフサイクルを決定します。

## フェーズ 4 – 進化的開発

ソリューション開発チームは、ソリューションを進化させるために、反復的開発、MoSCoW、またはタイムボクシングなどのプラクティスを適用します。その目的は、ソリューションが正確で、技術的な観点から「正しい方法」で組み込まれており、ビジネスニーズを満たすようにさらに進化させることです。チームは、プロジェクトの進行に合わせて、反復的に開発とテストを継続的に行います。

## フェーズ 5 – デプロイメント

このフェーズでは、3 つの主要な活動：アセンブル、レビュー、デプロイ、に焦点を当てています。このフェーズでは、最終的なソリューションまたはそのサブセットのリリースを目指します。最後のリリースをもって、プロジェクトは正式に終了します。

## フェーズ 6 – ポスト・プロジェクト

ポスト・プロジェクトのフェーズでは、ビジネス上の利益の期待がどれだけ満たされたかを確認するために、ベネフィット・アセスメントという形でチェックが行われます。

# LeSS

LeSS とは、Large-Scale Scrum（大規模スクラム）の略です。1 つの製品やサービスを開発する際に、複数のチームが共同で作業する場合に適用できる手法です。

LeSS は次のような場合に適用できます。

- ある特定の開発において複数のチームが必要
- 複数のチームが一つの共通の目標に向かって協力する
- 複数のチームが同時に製品やサービスに取り組む

もしあなたのプロジェクトがこれらの要件に当てはまらないのであれば、基本的なスクラムを使用する方が良いでしょう。

# LeSS 10 の原則

LeSS の 10 の原則は、次の図のようにまとめられています。

*図 33 LeSS で使われているコンセプト*




画像は EXIN が作成したものです。The LeSS Company B.V.（2014）. *大規模スクラムはスクラムである【ビジュアル】.* https://less.works/img/principles/principles.pdf。

## Large-scale Scrum（LeSS）はスクラム

LeSS は大規模な製品やサービスの開発に向けてスケーリングされているが、スクラムそのものです。複数チームという文脈の中で、スクラムを適用するためのガイド類をまとめた一連のルールを提供します。これは新しいスクラムのフレームワークではありません。

## 透明性

主な成果は、すべての関係者に見えるようにしなければなりません。短いフィードバックループを作ることは透明性を高めます。透明性は、変化への適応と改善を可能とするために不可欠です。

## より少ないリソースでより多く

この原則は LeSS の中心的なもので、複雑な組織によるソリューションは却って不利となり、解決するよりも多くの問題を引き起こす可能性があるという認識です。この原則は、問題に対してよりシンプルで異なるソリューションを適用することで、複雑さを取り除くことを目的としています。

## プロダクト全体にフォーカス

LeSS では、完成したプロダクトであることを重視します。なぜなら、プロダクトを構成する個々の部品は、それらが組み合わされて最終的なプロダクトが作られるまで、実際の価値を持たないからです。結局のところ、顧客が購入するのは部品ではなくプロダクト全体なのです。「全体」へフォーカスし続けることは、大規模な開発グループにおける最大の課題の一つです。

## 顧客中心

プロダクトフォーカスと同様に、LeSS の大規模化は顧客満足度を低下させるリスクがあります。LeSS では、プロダクトオーナーは、顧客/ユーザーとプロジェクトチームのコネクターとして機能します。顧客を重視するために、チームはエンドツーエンドの機能とコンポーネントに基づいて組織されています。

## 完璧を目指す継続的な改善

完璧を目指す継続的な改善はリーン思考の2つの柱の1つであり、LeSS の一部でもあります。LeSS では、変化は期待され、継続的であり、受け入れられます。

## リーン思考

「人間尊重」と「継続的改善」は、LeSS が採用した Lean の中心的なコンセプトです。

## システム思考

これは、即効性のある解決策ではなく、長期的なシステムの改善が望ましいという原則です。

## 経験的プロセス制御

LeSS ではこのスクラムのコンセプトを適用し、チームが「ちょうどよいプロセス」を持てるようにしています。このようにして、チームは透明性、検査、適応のサイクルの中で仕事をします。

## 待ち行列理論

システムのキューの中をアイテムがどのように移動するかというこの理論は、大きなバッチを処理するための能力を向上させるために使用できる思考ツールです。これは特に LeSS に関係しています。

# スケールド・アジャイル・フレームワーク®（SAFe®）[18]

Scaled Agile Framework は、アジャイルを 1 チームの枠を超えて拡張する際に遭遇する問題に対処するために作成された、自由に利用できる知識の体系です。この企業規模の開発手法は、リーンとアジャイルの原則を組み合わせたものです。フレームワークの組織とワークフローのパターンは、組織がリーンとアジャイルのプラクティスをスケーリングすることをサポートするように設計されています。

## SAFe のリーン・アジャイルの 9 つの原則

1. 経済的な視点を取る
2. システム思考を適用する
3. 可変性を仮定する。つまり、オプションを持ち続ける
4. 速く統合された学習サイクルとともにインクリメンタルに構築する
5. 動作するシステムの客観的な評価に基づくマイルストーンを決める
6. 仕掛（WIP）を可視化して制限し,バッチサイズを小さくし,待ち行列の長さを管理する
7. リズムを適用し,分野横断の計画策定で同期する
8. ナレッジワーカーの内発的なモチベーションを引き出す
9. 意思決定を分散する

## SAFe のフレームワーク

Scaled Agile Framework のウェブサイトでは、SAFe の概要が紹介されている。SAFe では、4 つの構成がある。

- エッセンシャル
- ポートフォリオ
- 大型ソリューション
- フル

**エッセンシャル**

Essential SAFe は、最も基本的な構成です。必要とされる最も重要な要素が説明されており、フレームワークの利点の大半を提供することを目的としている。チームやプログラムレベルのものも含まれており、これをアジャイルリリーストレイン（ART）と呼んでいる。

**ポートフォリオ**

Portfolio SAFe には、戦略的方向性、投資資金、リーンガバナンスへの懸案事項が含まれています。

**大型ソリューション**

Large Solution SAFe では、複数のプログラムにまたがって調整や同期を行うことができますが、ポートフォリオを考慮する必要はありません。SAFe の初期のバージョンでは、このレベルはバリューストリームと呼ばれていた。

**フル**

Full SAFe は、他の 3 つのレベルを組み合わせたものです。

[18] Free after: Scaled Agile（2021）. *Scaled Agile Framework.* https://www.scaledagileframework.com

# ディシプリンド・アジャイル・デリバリー（DAD）

ディシプリンド・アジャイル・デリバリー（DAD）は、高品質なプロダクトをより早く生産するために、企業のニーズに合った状況に応じたガイダンスを提供するフレームワークです。これは、スクラム、カンバン、XP、アジャイルモデリング、ユニファイドプロセスなど、世界的に実績のあるリーン・アジャイル手法を集めて形成されたハイブリッドモデルです。

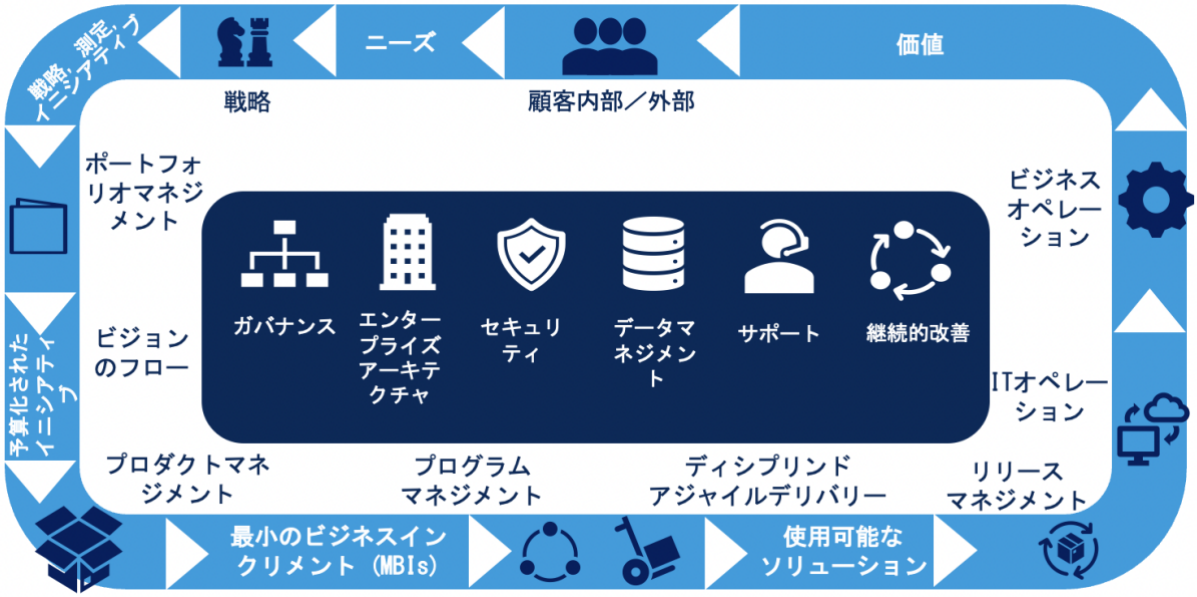
このフレームワークは、もともと 2009 年初頭から 2012年 6 月まで IBM Rational で開発されたものです。IBM チームは、Mark Lines（マーク・ラインズ）氏をはじめとするビジネスパートナーと緊密に連携し、Scott Ambler（スコット・アンブラー）氏が指揮を執りました。

DAD の知的財産権の所有権は、2012年10 月にディシプリンド・アジャイル・コンソーシアムに事実上譲渡されました。これは、2014 年 6 月に IBM によって法的に認められた事実です。

DAD はその後、包括的なツールキットに発展し、現在はディシプリンド・アジャイル 5.x と呼ばれるものが 2020 年 5 月にリリースされ、*『Choose Your WoW! A Disciplined Agile Delivery Handbook for Optimizing Your Way of Working (WoW)』*（Ambler & Lines, 2020）という本が出版されています。

DAD は現在、より広い範囲をカバーするディシプリンド・アジャイルのサブセットに過ぎません。

*図 34 DA Flex Workflow でのロールプレイヤーの連携方法*




画像は EXIN が作成したものです。Project Management Institute, Inc.（2021）. *Introduction to Disciplined Agile®（DA™）*. https://www.pmi.org/disciplined-Agile/introduction-to-disciplined-Agile

## カンバン

カンバンは、製造業でよく使われているリーン生産方式の一つで、最近では IT プロジェクトやオペレーションでも使用されるようになってきています。IT 環境における「カンバン」とは、通常、スクラムの様々な要素とカンバンの技法を組み合わせた開発手法のことを指します。

スクラムでのカンバンの使用については、本文の「カンバン方式の活用」ですでに説明しています。

# 付録 B – アジャイル環境におけるリリースとよく使われるリリース管理ツールの詳細

## リリースするかしないか、それが問題だ

スクラムでリリースの概念が使われることへの批判が実践者の間で高まっているが、それには理由があります。リリースという概念では、先を見越した計画を立てないと嘲笑されます。同時に、スクラムでリリースを使うことを擁護する人は、それは詳細な計画ではないと言うでしょう。そうであり、そうではないとも言えます。

2011年から公式にスクラムガイドからリリースの概念がなくなりました。実際のところ、なぜリリース計画がスクラムガイドから削除されたのでしょうか?その答えは簡潔でシンプルです。リリース計画を使わなくても、スクラムをうまく使うことは可能だからです。

リリース計画を必要としないことは、プロダクトの健全性を示すポジティブな指標かもしれません。スクラムは、短い反復で継続的に顧客に価値を提供することを目的としています。リリースを使用すると、価値の提供、顧客からのフィードバックが遅れ、そして投資収益率（RoI）が悪化する可能性があります。成熟した開発手法と優れたロールアウト戦略があれば、価値の提供をスプリントごとに行うことができます。”[19]

リリースマネジメントの支持者が提唱するであろう神話を払拭することで、スクラムにおいて従来のリリースマネジメントのアプローチが適切でない理由を探ってみましょう。すなわち：

### 神話 1

*複雑で大規模な環境では、事前の計画が必要です。*

**リアリティ1**

その通りです。しかし、必要なすべての事前計画は、プロダクトバックログを適切なリファインメントを確実にすることで実行できます。

### 神話 2

*プロダクトバックログのリファンインメントは、調整、依存関係、順序を考慮するには不十分な計画です。*

**リアリティ2**

その通りです。しかし、Nexus のようなスケーリングされたメカニズムを使用すれば、たとえスプリント間を跨っても、Nexus の一部であるスクラムチーム間の適切な連携を確保し、依存関係を適切に処理することができます。

[19] Scrum.org（2021 年）。*Gone Are Release Planning and the Release Turndown.* https://www.scrum.org/resources/gone-are-Release-planning-and-Release-turndown

## 神話 3

*Nexus のような手法は、連続したスプリントに対応していないため、時間の経過に伴う依存関係を適切に扱うことができません。*

**リアリティ 3A**

従来のリリース計画で行われるように、連続するスプリントで何が起こるかをかなり詳細に計画することは、スクラムをウォーターフォールに戻すことになります。順番、依存関係、次のスプリントで何が起こるかを理解することは、プロダクトバックログを適切に管理するだけで十分に対処できます。これに加えて、Nexus のようなスケーリング・アプローチでは、スプリント全体の依存関係の順序を特に注意して管理し、残りはプロダクトバックログを管理する時に発生します。

**リアリティ 3B**

リリースアプローチの最も不都合な結果の一つは、チームが「*スプリント2では、すでに決まっていたXXをやります*」という罠に陥ることです。*これ*により、アジャイル原則2に違反し、その結果、1、4、5、10、11にも違反する可能性があります。

## 神話 4

*リリース計画は、予測可能性を生み出し、ビジネス目標にコミットするのに役立ちます.*
*(ビジネスは完了予定日を求めます)。*

**リアリティ 4**

何?それは WaterScrumFall です。あなたはアジャイルではありません。

# リリースガイダンス

しかし、もしあなたが、リリース管理を使用することが、あなたの組織が継続して行うべきことであると考えるならば、ここにいくつかのガイダンスがあります。

- リリースサイクルを 3 スプリント以上に伸ばさないでください。
- ビジネスにおいて何も変化がなければ話は別ですが、イテレーションのたびにリリースプランが変更される可能性があり、また変更されるべきであることを全員が知っておいてください。
- リリースバックログ (スプリントバックログのようなものですが、リリースの一部を構成するすべてのスプリントのためのものです) の作成と使用は、ハイレベルで非常に柔軟でなければなりません。つまり、新しいアイテムが継続的にリリースバックログに入ってくるので、結果として、より多くのチームがリリースのために参加しない限り、アイテムを削除しなければなりません。
- リリース計画は、プロダクトゴールに集中するためのガイドであり、手段であると考えてください。新たな発見をし、それに素早く対応していかなければならないでしょうから。
- 柔軟性を持ち、計画を守るだけではなく、顧客にとって何が正しいのかを重視してください。
- プロダクトバックログの依存関係や連続性に注意ください。
- Nexus のようなスケーリング手法を使ってください。そうすれば、リリース計画と管理がよりシンプルでハイレベルなものになります。注:Scrum@Scale のような手法には、リリース管理のコンセプトが含まれているので、そこのガイダンスを調べてみるのも有用かもしれません。
- リリース計画ではタスク分解を行わず、スプリントのサイクルごとに行ってください。要するに、Nexus のようなスケーリング手法を使わないのであれば、スプリントサイクルごとに少しずつリリース計画を立てていくということです。
- WaterScrumFall をしてしまうことのないように。

## バーンダウンと見積もり

バーンダウン・チャートは、リリースを追跡するのに特に役立つツールです。考え方は、通常のバーンダウンチャートと変わりませんが、いくつかの工夫が必要です。

スクラムプロジェクトの進捗は、リリースバーンダウンチャートによって追跡することができます。リリースを管理するスクラムマスターは、各スプリントの終わりにリリースバーンダウンを更新する必要があります。

スプリント・バーンダウン・チャートのように、縦軸は各スプリントの開始時点で残っている作業量を示しており、これを表現するために、ストーリー・ポイント、理想日数、チーム日数など、組織で選択したものを使うことができます。横軸は時間を表していますが、この例では、リリース（プロジェクト）が終了するまでのスプリントの数を表しています。1つのリリースが 3 つのスプリントより長くならないはずなのに、なぜ悩むのでしょうか？

ここで、盲点を明らかにします。

あなたがリリース管理を使用する場合、あなたは神話 4 を信じてそうしている可能性があります。

作業が常に追加され、顧客がリリースバックログのプロダクトバックログアイテムの優先順位を下げたくない場合、実際には項目を削除してプロダクトバックログに戻すことができるようにすることで、誰もが到達できる唯一の論理的な帰結は「時間がかかる」ということになるのです。

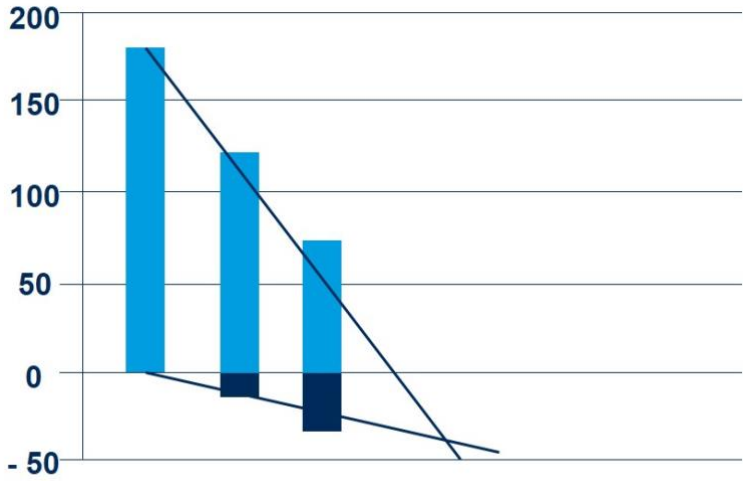
でも、あとどれくらい？

リリースバーンダウンバーチャートを見ると、その答えがわかります。

リリースバーンダウンバーチャートは、ゼロになるだけでなく、縦棒でマイナスにすることもできます。リリース中に作業が追加されると、スプリントサイクルごとにゼロラインの下に追加されていきます。

では、なぜこれが役に立つのでしょうか？このようにすることで、2つのトレンドラインを使って、リリースバックログの作業を完了するのに何回のスプリントが必要かを予測することができるからです。

*図 35 リリースバーンダウン・バー・チャート*




図は EXIN 制作：Botha、J.（2018）. *スクラムマスターとプロダクトオーナー[コースウェア]*. GetITright.

リリースバーンダウンには、2つのトレンドラインが表示されます。1つはゼロラインより上のデータ、もう1つはゼロラインより下のデータです。この交点は、リリースを指導しているスクラムマスターに、スプリント2と 3

で新たに追加された作業でリリースを完成させるには、追加のスプリントサイクルが必要であることを伝えています。

実際には、10 回以上のスプリントを計画することも珍しくなく、もっと悪い結果になることが多いのです。これがウォーターフォール型のプロジェクトです。

## ガントチャートを使ったリリース計画

一般的な方法として、ガントチャートによってリリースバックログの中の作業がどのスプリントで起こるか、チームに視覚的に思い出させます。また、プロダクトバックログアイテムとその関連タスクの相互依存関係を示すためにも使われます。

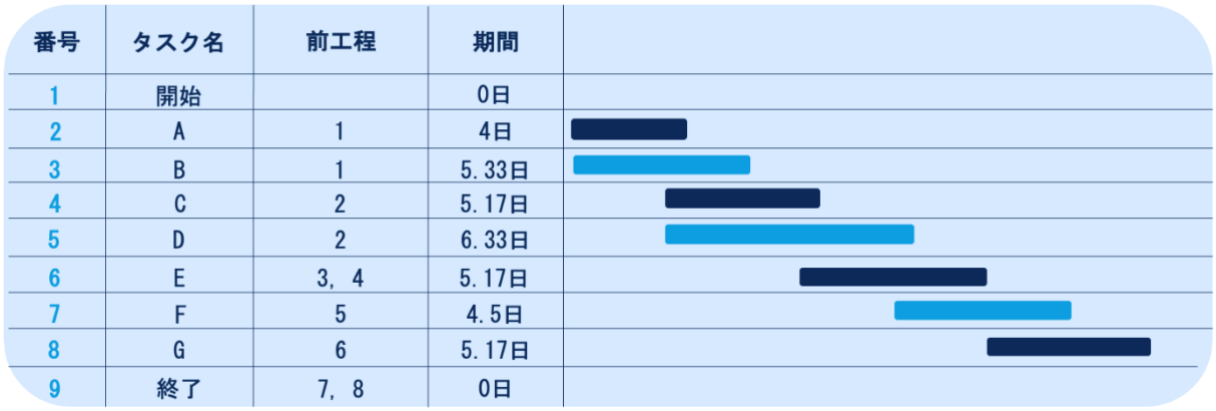
それでは、ガントチャートって何？

ガントチャートは、時間で予定されているタスクとその関係、依存関係を視覚的に表したものです。

ガントチャートは、1910 年頃 Frederick Taylor（フレデリック・テイラー）氏と一緒に働いていた Henry L. Gantt（ヘンリー・L・ガント）氏によって、工場や作業場の生産計画やリソースの負荷を表す手段として開発された手法です。ガントチャートの最初の主要な適用分野の一つは、第一次世界大戦中の米国で、戦争のための物流プロジェクトを計画するために使用されました。

皆さんもプロジェクト会議で見たことがあると思いますが、こんな感じのチャートです。

*図 36 ガントチャートによるリリース計画*



| 番号 | タスク名 | 前工程 | 期間 |
|---|---|---|---|
| 1 | 開始 | | 0日 |
| 2 | A | 1 | 4日 |
| 3 | B | 1 | 5.33日 |
| 4 | C | 2 | 5.17日 |
| 5 | D | 2 | 6.33日 |
| 6 | E | 3，4 | 5.17日 |
| 7 | F | 5 | 4.5日 |
| 8 | G | 6 | 5.17日 |
| 9 | 終了 | 7，8 | 0日 |

EXIN が作成した絵をベースにしたガントチャート。(2021)．https://en.wikipedia.org/wiki/Gantt_chart から 2021 年 2 月 14 日に取得しました。

基本的には、データを視覚化する手段としてガントチャートを使用することに問題はありませんが、重要なのは、それがどのような文脈で使われ、何を意味するかということです。詳細な事前計画に使われると、アジャイルやスクラムの考えに相反することになります。

# 書誌・参考文献


Agile Business Consortium（2014）. The DSDM Agile Project Framework（2014 Onwards）. https://www.agilebusiness.org/page/TheDSDMAgileProjectFramework

Ambler, S. & Lines, M.（2020）. Choose your WoW!: A Disciplined Agile Delivery Handbook for Optimizing Your Way of Working（WoW）. Project Management Institute.

Bigelow, S. J.（2020）. 7 techniques for better Agile requirements gathering. https://searchsoftwarequality.techtarget.com/tip/7-techniques-for-better-Agile-requirements-gathering

Botha, J.（2018）. Scrum Masters and Product Owners [Courseware]. GetITright.

Botha, J.（2019）. Agile: A Manager's Guide to Unlocking Business Value. Amazon Digital Services LLC - Kdp Print Us.

Botha, J.（2020）. Scrum Lego-game workbook [Courseware]. GetITright.

Botha, J.（2021）. [Courseware]. GetITright. Retrieved in personal communication from the author of the book.

Botha, J.（2021）. Lean fundamentals for IT [Courseware]. GetITright.

Cohn, M.（2005）. Agile Estimating and Planning. Pearson Education.

Cohn, M.（2009）. Four Attributes of the Ideal Pilot Project. https://www.mountaingoatsoftware.com/blog/four-attributes-of-the-ideal-pilot-project

Cohn, M.（2010）. Succeeding with Agile. Addison-Wesley.

Crystal Methods.（2021）. Retrieved on September 23, 2021 from

EXIN Holding B.V.（2019）. Agile Methodologies. EXIN Holding B.V. https://dam.exin.com/api/&request=asset.permadownload&id=3144&type=this&token=8659eabedff466d86ac2eba5ed72b33d

Exoskeleton.（2021）. Retrieved on September 23, 2021 from https://en.wikipedia.org/wiki/Exoskeleton

Gantt Chart.（2021）. Retrieved on September 23, 2021 from https://en.wikipedia.org/wiki/Gantt_chart

Goldratt, E. M. & Cox, J.（2012）. The Goal.（3rd Edition）North River Press.

Hiatt, J. M. & Creasey, T. J.（2012）. Change Management: The People Side of Change. Prosci Learning Center Publications. https://en.wikiversity.org/wiki/Crystal_Methods

Kano+（2020）. The Kano model – Assessing Product Features based on Customer Satisfaction. Kano+ https://kano.plus/about-kano#analyze-a-study

Maher, R., & Kong, P.（2016）. Cross-Team Refinement in Nexus™. Scrum.org. https://www.scrum.org/resources/cross-team-refinement-nexus

Maher, R., & Kong, P.（2016）. The Nexus Integration Team. Scrum.org. https://www.scrum.org/resources/nexus-integration-team

Maher, R., & Kong, P.（2016）. Visualizing the Nexus Sprint Backlog. Scrum.org. https://www.scrum.org/resources/visualizing-nexus-sprint-backlog

Martin, B. A.（1980）. The goal: to improve credibility in the reporting of engineering progress. Project Management Quarterly, 11（2）, 14–22.

McKinsey & Company（2019）. The journey to an Agile organization. https://www.mckinsey.com/business-functions/organization/our-insights/the-journey-to-an-agile-organization

Meier, J. D.（2019）. Extreme Programming at a Glance. https://jdmeier.com/extreme-programming-at-a-glance/
Overeem, B.（2016）. The 11 Advantages of Using a Sprint Goal. https://www.scrum.org/resources/blog/11-advantages-using-sprint-goal
Pichler, R.（2010）. Agile Product Management with Scrum. Addison-Wesley.
Project Management Institute, Inc.（2021）. Introduction to Disciplined Agile®（DA™）. https://www.pmi.org/disciplined-Agile/introduction-to-disciplined-Agile
Prosci（2021）. Best Practices in change management benchmarking report. https://www.prosci.com/resources/articles/change-management-best-practices
Rad, N. K.（2021）. Agile Scrum Handbook（3de ed.）. Van Haren Publishing.
Rother, M.（2018）. The Toyota Kata Practice Guide: Practicing Scientific Thinking Skills for Superior Results in 20 Minutes a Day. Mcgraw-Hill Education.
Scaled Agile（2021）. Scaled Agile Framework. https://www.scaledagileframework.com/
Schwaber, K., & Scrum.org（2021）. The Nexus™ Guide – The Definitive Guide to Scaling Scrum with Nexus. Scrum.org. https://www.scrum.org/resources/nexus-guide
Schwaber, K., & Sutherland, J.（2020）. The Scrum Guide – The Definitive Guide to Scrum: The Rules of the Game. Scrumguides.org. https://scrumguides.org/scrum-guide.html
Scrum.org（2021）. Gone Are Release Planning and the Release Turndown. https://www.scrum.org/resources/gone-are-Release-planning-and-Release-turndown
Scrum.org., Vacanti, D., & Yeret, Y.（2021）. The Kanban Guide for Scrum Teams. Scrum.org. https://www.scrum.org/resources/kanban-guide-scrum-teams
Senge, P.（2006）. The Fifth Discipline. Bantam Doubleday Dell Publishing Group Inc.
Smart, J. (2020). Sooner Safer Happier. Antipatterns and Patterns for Business Agility. IT Revolution Press.
Sutherland, J., & Scrum Inc.（2021）. The Scrum@Scale® Guide – The Definitive Guide to Scrum@Scale: Scaling that Works. Scrum.org. https://www.scrumatscale.com/wp-content/uploads/2020/12/official-scrum-at-scale-guide.pdf
Technical Debt.（2021）. Retrieved on September 23, 2021 from https://en.wikipedia.org/wiki/Technical_debt
The LeSS Company B.V.（2014）. Large Scale Scrum is Scrum [Visual]. https://less.works/img/principles/principles.pdf
Underwood, J.（2013）. Musselwhite and Ingram's Change Style Indicator. https://innovategov.org/2013/08/15/musslewhite-and-ingrams-change-style-indicator/
Wells, D.（2000）. Introducing Extreme Programming. http://www.extremeprogramming.org/introduction.html
Wells, D.（2013）. Extreme Programming: A gentle introduction. http://www.extremeprogramming.org/

Driving Professional Growth

# EXIN の連絡先

www.exin.com