アジャイルにおける自己組織化チームの構成

平井 直樹[1]

Structure of Self-Organizing Team in Agile Software Development

HIRAI Naoki

1. はじめに

現代において、将来は予測しづらく、顧客のニーズは多様化し、ビジネスのスピードは劇的に速くなっており、VUCA と呼ばれるような不確実性が前提となりつつある。こうした環境下では、これまでのような定型的で標準化された手法は通用しにくく、すなわち多くの企業を構成している中央集権的な統制を行うヒエラルキー型の組織自体が通用しにくくなっているともいえる。

不確実な環境の中では、絶えず変化する問題や状況に対して互いに協力し、知恵を出し合いながら解決していく知識集約型の組織が求められる。こうした中、これまでのリーダーや管理者を中心としてきた中央集権的な権限を、個人やチームそのものに権限移譲し、分散させ、さらに自律的に変化に対応して行動することができる組織である自律型組織や自律分散型組織がとりあげられることが多い。

ソフトウェア開発において、中央集権的で統制を行うヒエラルキー型の組織や構造の元で開発が行われるのがウォーターフォールであり、自律分散型の元で開発が行われるものがアジャイルといえる。アジャイルは、スクラム（Scrum）や XP（extreme programming）と呼ばれる、幾つかの軽量なソフトウェア開発プロセスの総称であり、ソフトウェアの仕様について厳密な決定をせず、開発プロセスを進める中で仕様を擦り合わせ、ある程度決まった部分だけを先に開発していくスタイルである（平井, 2019）。アジャイルの特徴は、機能横断型で自己組織化したチームのもと、反復型で漸進的な開発を行い、顧客との対話やフィードバックを重視して継続的な学習のもと、ふりかえりを行うことである。

このアジャイルの自己組織化と類似したものとして、フラット型の組織やティール組織、ホラクラシー組織などの分散型の組織構造が挙げられる。たとえば、ティール組織の特徴の一つである Self-management では、自律的なチームであり、上司や階層が存在せず、分散された意思決定などが行われている（Laloux, 2014）。

これまでビジネス書や技術書を中心にアジャイルの原則について解説がなされており、

[1] 昭和女子大学現代ビジネス研究所 研究員 / 立教大学大学院ビジネスデザイン研究科 助教

自己組織化については事例と共に説明されることもあるが、やや抽象的で 2〜3 行程度の簡易な説明に留まっていることも多い。この理由としてアジャイルは、2001 年に発表された価値観や原則をまとめた「アジャイルソフトウェア開発宣言」から広く知られるようになったが、新しい開発手法であることから、実務の現場にもあまり広まっておらず、そのためビジネス書や技術者が先行していると考えられる。

さらに、アジャイルのようなソフトウェアを対象とした研究の多くは、その技術的な特性からアルゴリズムの解析やプログラム作成の効率性の向上、ツールや手法の開発といったソフトウェア工学やそれに類するものであり、人的資源やそれを構成する組織についてなどの社会学や経営学に関する研究は非常に少ない（DeMarco and Lister, 2013）。そのため、こうしたアジャイルの自己組織化の特徴はいかなるものか、また、そうした自己組織化を促すもしくは形成するにはどのような手段が考えられるのか、自己組織化したチームはどのような成果を生み出せているのか、学術的にも研究がなされているとはいい難い。

本研究では、アジャイルの自己組織化に関する研究の前提として、アジャイルの代表的な開発の一手法であるスクラムにおける自己組織化の特徴について検討する。

2. アジャイルの特徴

アジャイルは、製品開発当初の計画に従うよりもユーザーの要望やビジネスの変化への対応を価値としており、ビジネスや市場に合わせて臨機応変に製品の仕様を変更していく。開発プロジェクトを進めていくなかで、顧客からの要求に基づくソフトウェアを作成し、その検査とフィードバックをもとに新たな要求の設計や解析といった作業をチーム内で PDCA サイクルのように反復活動を繰り返していくのである。

ある程度動くソフトウェアを成長させながら作成する、反復、且つ漸進型のプロセスが大きな特徴であり、顧客の要望から優先度の高いものをできるだけ早く作成し、少しでもできあがったソフトウェアを顧客に確認してもらうことで、早期にフィードバックを得ることを目的としている（平井, 2020）。

アジャイルの特徴をまとめたものが表 1 である。

アジャイルの特徴は、こうした反復型の取り組みを可能とする組織・チームであり、文化でもあり、たとえば、「組織的な学びの仕組み・改善のためのふりかえり（Retrospective）」、「顧客との共創（Co-Creation）」、「自己組織化（Self-organizing Teams）」「機能横断的（Cross Functional Teams）」、「使用価値の最大化（Maximize Value in Use）」といった要素があげられる。

表 1　アジャイルの要素

特徴	概要
Self-organizing Teams（自己組織化）	開発チームが状況に応じてミッションを実現するための選択を自分たちが決定し、行動できる。
Cross Functional Teams（機能横断的）	外部に頼らず開発をやり遂げるために必要なあらゆるスキルを持つチーム、メンバーが複数の役割を果たすことができる。
Co-Creation（顧客との共創）	フィードバックといった顧客の参画の度合いが強く、ゴールを共有しており、そのため、人と人とのコミュニケーションやコラボレーション、「共創」を重視する。
Retrospective（組織的な学びの仕組み・改善のための「ふりかえり」）	単なる試行錯誤ではなく、失敗からいかに学び、成長できるか。「反復活動」を小さく短く繰り返すことで、小さな失敗を繰り返し、多くを学んでいく。
Maximize Value in Use（使用価値の最大化）	顧客にとっての使用価値（使用と経験）が常に最大化されることを目的とする。

出典：平井（2022）をもとに筆者作成

3.　自己組織化

3.1. 自己組織化の定義

本研究が対象とする「自己組織化（self-organizing）」という言葉は、生命科学や環境科学、物理科学などの分野で広く使われてきた。自己組織化は、物事が一定の秩序をもち、有機的な働きをするように統一化したり、組織的にすること（小学館デジタル大辞泉）であり、複数の要素からなるシステムが時間と共に何らかの意味で自発的に秩序化する過程（山口, 2011）や、外からの働きかけなしに何らかの秩序やパターンが形成する現象（今井, 2001）であり、雪の結晶が水の分子が集まって自然にできあがる過程や、蟻や蜂が群れとして列を成す行動などが例に挙げられる。

こうした自己組織化による秩序立った構造の形成については、その後、社会学を中心に、経営学でも広く検討されてきた。組織の分野において、自己組織化は個々人の自律的な行動によってチーム全体としての成果に繋がる行動である。また、リーダーシップの分野では、自己組織化はチームに関わるものであり、意思決定の機会や生産的に前進する際にリーダーから独立して機能するもの（Osherove, 2016）である。

3.2. アジャイルにおける自己組織化

ソフトウェア開発の分野において、自己組織化は「アジャイルソフトウェア開発宣言」の 12 の原則に記載されている。「アジャイルソフトウェアの 12 の原則（Twelve Principles of Agile Software）」Cunningham（2001）によると、「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」[2]とされている。

ここでの自己組織的なチームとは、「メンバー一人ひとりが自らの行動、作業に責任を持つとともに、お互いの連携により、チームとしての成果を最大限に発揮することができる」（IPA 独立行政法人情報処理推進機構, 2020a, p.17）自律的なチームを指す。メンバーそれぞれが改善の意識を持ち続け、共通の目標や目的のもと、能力を発揮するのである。

この自己組織化について、アジャイルの代表的な一手法であるスクラムと、その公式のガイドラインである「スクラムガイド」[3]では「自己組織化（self-organized）」や「自己管理型（self-managed）」といった表記がなされている（表 2）。

表 2　スクラムガイド

スクラムガイド 2017 年版	スクラムガイド 2020 年版
スクラムチームは自己組織化されており、機能横断的である。自己組織化チームは、作業を成し遂げるための最善の策を、チーム外からの指示ではなく、自分たちで選択する。機能横断的チームは、チーム以外に頼らずに作業を成し遂げる能力を持っている。 スクラムにおけるチームのモデルは、柔軟性・創造性・生産性を最適化するように設計されている。	スクラムチームは機能横断型で、各スプリントで価値を生み出すために必要なすべてのスキルを備えている。また、自己管理型であり、誰が何を、いつ、どのように行うかをスクラムチーム内で決定する。

出典：「スクラムガイド 2017」p.5、「スクラムガイド 2020」p.6 をもとに筆者作成

スクラムのチームでは、開発者[4]とスクラムオーナー[5]とプロダクトオーナー[6]の 3 つの役割が存在する。スクラムガイドの 2020 年版では、「開発チーム」を「開発者」という表記に変更しており、スクラムチーム内に、スクラムオーナーと開発者による開発者チームが存在するような疑念を払拭している。このスクラムチームの自己管理とは、それまでの開発者チームの自己組織化とは異なり、よりチームとしての裁量が増しており、チーム内の分断をなくし「ワンチーム」になることを強調している（平鍋・野中・及部, 2021）[7]。スクラムで

[2] 英語原文では　“The best architectures, requirements, and designs emerge from self-organizing teams.”とされている。
[3] スクラムガイドは、スクラムを開発した Jeff Sutherland と Ken Schwaber 自身が、その定義やルール、知識体系をまとめ、文書化したものである。
[4] 開発者は、プログラミングを行うなど、ソフトウェアの開発を実際に行う者である（スクラムガイド, 2020）。
[5] スクラムオーナーは、スクラムチームが、スクラムガイドで定義されたスクラムに準拠して開発が行われるよう、コーチングを行うリーダーとしての役割をもつ（スクラムガイド, 2020）。
[6] プロダクトオーナーは、開発しているソフトウェア製品（プロダクト）の方向性を決め、その開発結果の責任者としての役割をもつ（スクラムガイド, 2020）。
[7] スクラムガイドにおいて、自己組織化という表記が自己管理に変更されたが、スクラムにおいて自己管理は最低限の基準であり、状況が許せば、自己組織化し、自己設計し、自己統治するとしており、スクラムは自己組織化を否定しているわけではない（Visotckym, 2020）。

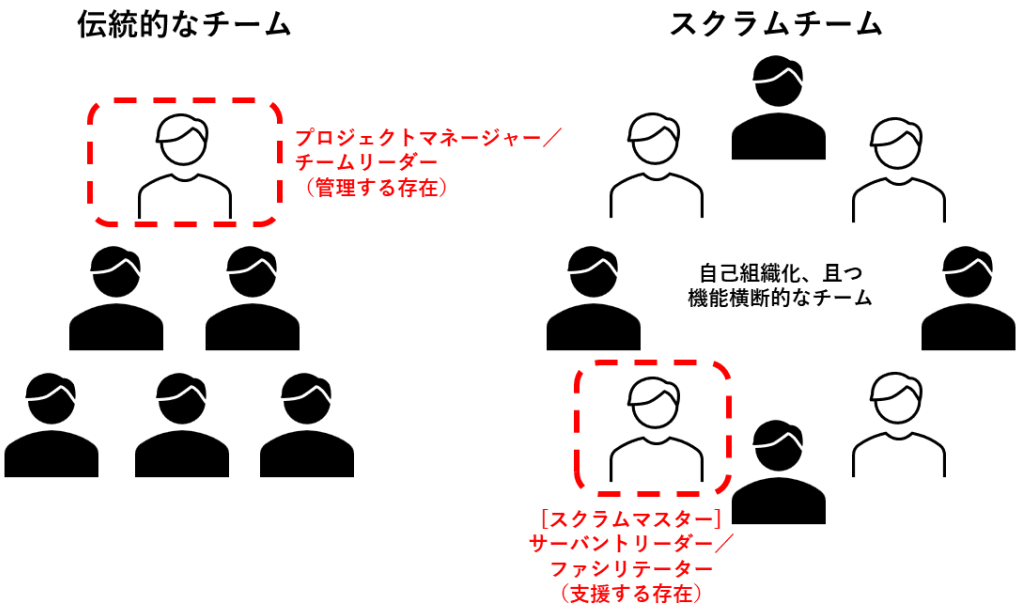
は役割を超えて協力していくことが重要となる。

こうした変更は、開発チームの自己組織化から、スクラムチームとしての自己管理に重点が置かれており、「誰が」「どのように」「何の」作業をするかを選択できることを説明している。それまでの開発者（開発チーム）内の自己組織化ではなく、開発者、スクラムオーナーも含めたスクラムのチームとしての自己組織化であり、スクラムチームに何が期待されているかをより明確にしたものとなっている（Visotcky, 2020）。

3.3. 自己組織化チームの構成

スクラムガイドでは、チーム外の影響を受けず自らのチーム内だけで意思決定ができるような権限移譲について、また、機能横断型チームのもと自分たちでやりきることができるスキルや能力について説明されている。

自己組織化したチームでは、従来の伝統的なチームにおけるプロジェクトマネージャーや管理するような存在は必要とされない。図 1 のように、スクラムにおいては、スクラムマスターは管理者やリーダーというよりも、スクラムチームを支援する存在である。



出典：「スクラムガイド 2020」をもとに筆者作成

図 1　伝統的なチームとスクラムチーム

こうしたスクラムチームは、異なる部門の個人が一つのチームとして働く機能横断的なチーム（cross functional teams）であり、スクラムマスターはチームメンバー各々がそれぞれ主体的にプロジェクトの成功に貢献できるような環境を構築する存在としてサーバント・リーダーシップやファシリテーターとしての役目が求められる。

自己組織化したチームには、個人とチームの両方に高いレベルの自律性が必要となる。さ

らに自律的に動くためには、チームとしてバランスの取れたスキルが必要となる。ここで関わってくるものがアジャイルの特徴の一つである機能横断的なチームであり、個々のチームメンバーが仕事をこなすための深いスキルを持ち、さらにチームとしてパフォーマンスを発揮できるスキルも持つ必要がある。

こうしたスキルに関して、スクラムチームの開発者を対象とした自己組織化チームの導入における調査によると、最も重要な障壁となったものは、開発者の高度に専門化されたスキルと、それに対応する作業分担であり、また、チームをサポートするシステムの欠如や、外部からの自主性の低下だとしている（Moe, Dingsøyr, and Dybå, 2008）。

スクラムにおいてチームは独立し、十分な自己管理の下で作業を行う。そのためには、機能横断的チームとして意思決定ができるだけ能力、つまり質の高い意思決定が下せる能力（Forsgren, Humble, and Kim, 2018）と権限を持つ必要がある。この質の高い意思決定を行うには、必要な専門知識がすべて揃っている必要がある。チームはできるだけ多くの問題を自分たちだけで解決できるような構成にする必要がある。

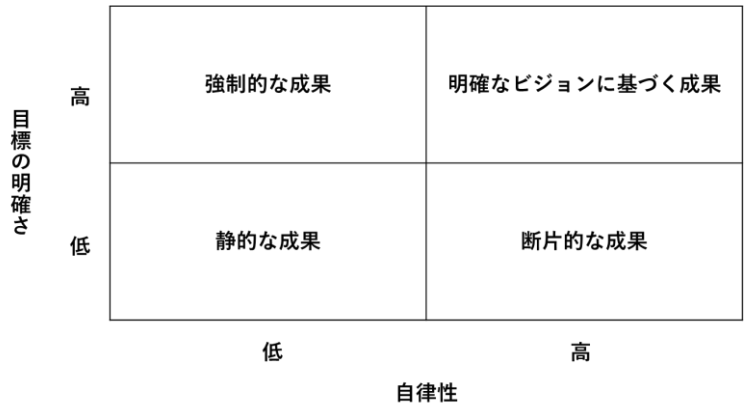
調査や把握のたびに、チーム外部の誰に調査すればよいのかの把握や外部の人間への説明や対応待ち、外部の調査結果の理解や把握など、チーム外の人間に頼ってしまうような場合、毎回の反復活動であるスプリントごとに対処していくことが難しくなる。

一方で、スクラムのチームは、原則として 5～10 名程度で構成され、専門知識や能力には偏りや限りがあるため、必ずしもすべてに対応できるわけでは無く、すべてを備えたチームを編成することは現実的ではない。メンバーは複数の役割をこなさなければならないが、不足するものについては、メンバーに新たなスキルを学習、習得させていかなければならない。当然ながら、開発に必要なすべての能力をもったような人材は存在せず、1 人の能力には限界やスキル領域ごとのレベルに差があるが、1 つのチームとしてその差をメンバー間で補っていくことになる（IPA 独立行政法人情報処理推進機構, 2020b）[8]。

また、自己組織化したチームは、作業の目的を理解している必要がある。作業の目的を明確に理解していないと、チームの意思決定は誤った方向に導かれてしまう可能性がある。そのため、チームの責務と説明責任を強調することで、仲間意識が強くなり、チームの目的が養われる（常松・川口・松元, 2023）。さらに、顧客との直接の交流を重視することは、チームの責務と説明責任に繋がり、チームの目的意識を養うのに貢献する（McConnell, 2019）。

チームの意思決定によって起こり得る結果は、チームの自律性とその目標、目的の明確さに応じて図 2 のようになる。

[8] 一般的な対応策は、2～3 回のスプリントごとにユーザーエクスペリエンスやアーキテクチャといった分野の専門家に臨時メンバーとしてチームに加わってもらう（McConnell, 2019）。



出典：McConnell（2019）p.195 をもとに筆者作成

図 2　自律性と目的の明確さ

3.4. 自己組織化チームの権限と支援

ソフトウェア開発の一般的なプロジェクトチームにおいて、プロジェクトにアサインするような体制では、目的を達成するとチーム構成を見直すことが多い。しかし、チームが機能するようになるには、メンバーが成長していかなければならず、長い時間がかかる。こうした状況下で、組織や管理者がメンバーを短期間で入れ替えてしまうと、せっかくメンバーが成長し、機能しかけているチームが破壊されてしまう。

こうした、チームメンバーの変更によりチームが破壊されてしまうことを防ぐには、プロジェクトとチームの関係性を見直す必要がある。開発で生じる課題は、チームの課題であって個人に拠るものではない。課題を共有することでその解決をチームで行うのであり、プロジェクトに合わせてチームを組成や解散するのではなく、チームにプロジェクトを与えるように発想を転換する必要がある（平鍋・野中・及部, 2021）（図 3）。

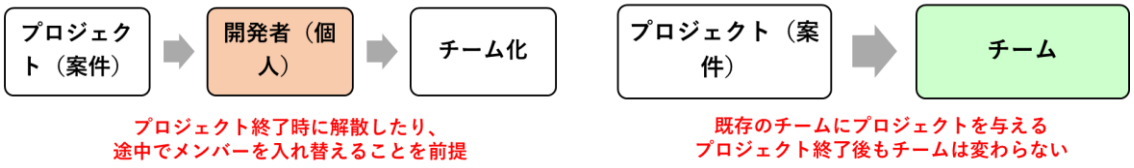


図 3　開発体制の基本単位（個人からチームへ）

チームには、意思決定を行えるだけの十分な権限が必要となる。重要なステークホルダー全員がチームに参加したり、組織の他の誰かが覆すことのできない拘束力のある意思決定の権限が必要となる（McConnell, 2019）。

組織のリーダーや管理者といった外部の人間は、スプリントが不可侵であることをスクラムチームに確約し、目的・目標といったチームの方向性が明確であることを確認する。ス

クラムチームによる開発作業は、外部である組織のリーダーや管理者からはスプリント期間中は指示されないように、つまりブラックボックス化される。これにより、外部から内容について確認はできず、外部から作業を追加したり変更することもできない。

スプリントの期間は 1〜3 週間程度のため、リーダーや管理者は、成果が出るのを待つことになる。これによりマイクロマネジメントが避けられ、組織のリーダーはよりリーダーらしい姿勢で臨むようになることが期待される（McConnell, 2019）。

一方で、チームが初めから自己組織化している状態にあるとは限らず、チームの成熟度を理解し、統率力、管理能力、指導力を提供しながら、チームの自己管理能力（自己組織化）を養うことも組織のリーダーの役割の 1 つとなる（McConnell, 2019）。

また、自己組織化が、チームメンバーのモチベーションと革新性にプラスの影響を与えるとの指摘もある（Kakar, 2017）。自己組織化を進めていくには、組織の上層部の支援が必要であり、組織がメンバーへのサポートを怠らず、責任を負う姿勢を貫くこと、そしてチームのメンバーが改善に対する努力を惜しまないことが必要となる。アジャイルの実践の成否は、意思決定のほとんどを自分たちだけで下すのに必要な人材をチームに積極的に配属するかどうかにかかっている。メンバーを集めただけでは自己組織化したチームはできず、成熟度に応じた組織の支援が必要となるのである。

開発者にとって成長の機会は、どの要因よりも強力なモチベーションになる。アジャイルは経験からの学習が重視されており、学習により開発スピードが向上することや、高度な開発が可能となることで、チームとして成長しているという熟達感を養うのに貢献する。

スクラムチームは必ずしも成熟したチームとは限らないこと、むしろ、チームは反復活動を通じて学習を続け、成長していくものであり、成熟に至るまでそうしたチームをサポートするシステムが必要である。また、外部からの中央集権的な権力の影響を受けずにチーム内ですべてを自律的に行うには、チームに権限が委譲される必要があるが、従来の組織のような管理者が存在していたり、チームとして自分たちで管理（self-managed）できるだけの十分な権限が委譲されていないような場合、チームの自律性の低下が引き起こされると考えられる。

失敗やそこからの学習を支援、奨励する作業環境を育むことで、自分の仕事を左右する意思決定を自ら下せるようになる。チームは失敗から学び、改善していくが、組織がスクラムチームを十分に信頼し、失敗が許されると理解していることが、チームにとって大きな励みになる。チームが下した意思決定を組織が信頼しないようでは、チームは自己組織化しているとはいえず、自律しているともいえない。失敗を許容し、お互いに敬意をもったチーム文化を育む責任は最終的には管理者（組織）にある（Forsgren, Humble, and Kim, 2018）。チームが成熟していくためには、失敗からの学びができる信頼が重要となるのである。

4. 結論と今後の研究にむけて

本研究では、アジャイルの自己組織化はどのような特徴を有するのかについて、特にアジ

ャイルのだい表的な開発手法の一つであるスクラムを対象に検討した。自己組織的なチームは、外部からの影響を受けず、自律的に作業を行っていくことになるが、個々の技術者のスキルレベルの違いやチームの権限移譲の問題など多くの課題が存在する。

自己組織的なチームは、チーム外の影響を受けずにチーム内で決定できるだけの質の高い意思決定能力や権限移譲を持つ必要があり、そうしたチームは機能横断型であり、自分たちでやりきるスキルや能力が重要となる。こうしたチームの開発体制を考えるときの基本単位は、個人ではなくチームであり、そうしたチームでは、複数の役割をこなす必要があるが、開発に必要なすべての能力をもったような人材は存在せず、メンバーが学習し、自己組織化、つまり成熟していく必要がある。十分な意思決定ができる能力は、初めからあるものではなく、学習を通じて成長していくことでできあがる。チームが成熟していくためには、失敗からの学びが必要であり、そうした失敗が可能な信頼が重要であり、成熟度に応じた組織の支援が必要となる。単にメンバーを構成し、権限移譲するだけでは自己組織化は成り立たないのである。

本研究では、先行研究より、意思決定のほとんどを自分たちだけで下すことができるかといったチームの自己管理能力を養うためにも、チームの自己組織化の程度（成熟度）に応じてメンバーへのサポートといった組織の支援が必要であることを述べたが、McConnell（2019）をはじめ、多くの先行研究は実務における実践的な手法をまとめたものであり、具体的な支援内容については十分に説明されておらず、定量的な調査も行われていない。意思決定ができるだけの権限とその移譲がどのようなものかについても、明らかとなっておらず、成熟度に応じて変わることも考えられる。組織構造・組織モデルや権限委譲等の紐づけ、特に権限移譲が何をもたらすのかについて、他産業や科学的な実証データをもとに分析する必要があろう。今後の研究方針として、各特長の詳しい要因の分析や、自己組織化チームで生じる課題やその成果に関する調査、そして自己組織化を促す要因について、引き続き検討していきたい。

＜謝辞＞
本研究は JSPS 科研費（若手研究） JP22K13483 の助成を受けたものです。

＜参考文献＞
今井宏明（2001）「自己組織化する結晶」『Journal of the Society of Inorganic Materials, Japan』8 (292), pp.234-240, 無機マテリアル学会。
常松祐一・川口恭伸・松元健（2023）『アジャイルプラクティスガイドブック チームで成果を出すための開発技術の実践知』翔泳社。
平井直樹（2019）「アジャイルの導入と本質 ―開発プロセスから学習する組織活動へ―」『立教 DBA ジャーナル』(10), pp.35-44。
平井直樹（2020）「顧客志向の反復型プロセス―リーン・スタートアップとアジャイルの組

織的仕組み―」『立教 DBA ジャーナル』(11), pp.45-58。
平井直樹（2022）「心理的安全性とアジャイル・アプローチ」『立教 DBA ジャーナル』(13), pp.9-23。
平鍋健児・野中郁次郎・及部敬雄（2021）『アジャイル開発とスクラム 第 2 版』翔泳社。
山口智彦（2011）「さまざまな自己組織化とその工学的応用」『表面技術』62(2), pp.74-79, 一般社団法人表面技術協会。
“Cross-Functional Teams in Product Development: Definition, Principles, and Examples”（https://www.altexsoft.com/blog/crossfunctional-teams/）, March, 3, 2023. Retrieved August, 1, 2023 from AltexSoft, 2023.12.03.
Cunningham, Ward.（2001）“Manifesto for Agile Software Development”（https://agilemanifesto.org/iso/en/manifesto.html）, Retrieved August, 1, 2023 from Ward Cunningham, 2023.11.7.
DeMarco, Tom. and Lister, Timothy R.（2013）*Peopleware: productive projects and teams (3rd Edition)*, Addison Wesley Professional.
Forsgren, Nicole., Humble, Jez., and Kim, Gene.（2018）*Accelerate: The Science Behind Devops: Building and Scaling High Performing Technology Organizations*, It Revolution Pr.
IPA 独立行政法人情報処理推進機構（2020a）「アジャイル領域へのスキル変革の指針 アジャイルソフトウェア開発宣言の読みとき方」2020 年 2 月。
IPA 独立行政法人情報処理推進機構（2020b）「アジャイル領域へのスキル変革の指針 アジャイル開発の進め方」2020 年 2 月。
Kakar, Adarsh Kumar.（2017）“Assessing Self-Organization in Agile Software Development Teams”, *Journal of Computer Information Systems*, 57(3), pp.208-217.
Laloux, F.（2014）*Reinventing Organizations*, Lightning Source Inc.
McConnell, S.（2019）*More Effective Agile*, Construx Press.（長沢智治監訳・クイープ訳（2020）『More Effective Agile －“ソフトウェアリーダー”になるための 28 の道標－』日経 BP.）
Moe, N. B., Dingsøyr, Torgeir., and Dybå, T.（2008）“Understanding Self-Organizing Teams in Agile Software Development”, Conference Paper, 19th Australian Conference on Software Engineering (aswec 2008).
Osherove, R.（2016）*Elastic Leadership: Growing self-organizing teams*, Manning Publications.
Visotcky, Rich.（2020）“The Scrum Team is Self-Managing”, November 18, 2020, Scrum.org, https://www.scrum.org/resources/blog/scrum-team-self-managing