アジャイル方式のゲーム開発プロジェクトにおける管理会計の役割について

アジャイル方式のゲーム開発プロジェクトにおける管理会計の役割について

坂　手　啓　介

1．ゲーム産業の現状とゲーム開発プロジェクトの高騰

日本における2021年のテレビゲームおよびスマホゲームにおける市場は，KADOKAWA Game Linkage 社の「ファミ通」の2022年の調査によれば，2兆円を超えており，10年前の2012年と比較し2倍の市場規模となっている．一方，世界市場は，約21兆8,927億円という規模であり，年々増加している．新型コロナウイルスは世界中で経済に大打撃を与えているが，GameIndustry.biz の調査や Newzoo の調査によれば，ウィルスの流行によるロックダウンでビデオゲームの市場やモバイルゲームの市場が急増したということである（Dring, 2020; Tianyi, 2020）．成長を続けるゲーム市場ではあるが，一方で，ゲームの開発コストも飛躍的な増加を見せている．開発費の高騰に関するエビデンスの一つとして，世界初の大ヒット MMO であるウルティマ・オンラインのリード・デザイナーである Koster が2017年に行った調査では，1985年以降，ゲームの開発コストが増加傾向にあることが示されている．同氏は，さらに，最近の30年で，ゲームサイズが122倍になり，コストが22倍になっている点を指摘している．このような開発コストの増加傾向に関しては，今後も続くと予測され，ゲーム開発プロジェクト・コストを予算内におさめ，予定通りの発売日にリリースするためには，プロジェクト・マネジメント（以降 PM）手法の適用が必要であろう．

一般的にプロジェクトはその進行が遅延すれば，コストが予算を超過するリスクを持つが，遅延が生じる理由の一つにプロジェクトの不確実性がある．ゲーム開発プロジェクトには，Schreier（2017）や Keith（2010）が指摘するような不確実性があり，その不確実性がプロジェクトの実行を困難なものにしていると考えられる．本稿では，そのような不確実性の高いゲーム開発プロジェクトではどのような PM 手法を適用すべきか，また，その PM において，管理会計がどのように貢献できるかについて考察したい．

2．ゲーム開発プロジェクトにおける不確実性とアジャイル方式

ゲーム開発プロジェクトの管理がその他の分野の開発プロジェクトと比較して困難である理由として，①ゲームのインタラクティブ性，②テクノロジーの急速な進化，③ツールが毎回異なること，④スケジュール作成の困難性，⑤面白さの推測の困難性，⑥利害関係者の指示による手戻りの6つが挙げられる（Keith, 2010; Schreier, 2017）．

確かに，ゲーム開発プロジェクトに限らず，どの分野のプロジェクトであっても，成果物の独自性を考えれば，管理の不確実性は高くなる傾向にはあるだろう．しかし，ゲームの要件の中でも，多分に主観的な要素である，心理的な「面白さ」はプロジェクトの不確実性を高いものにしている．

「面白さ」は主観性の高い要件ではあるが，これまで，行動心理学や認知心理学，情報学などの分野において，コンピュータゲームにおける「楽しさ」や「面白さ」などの心理的要因に関する研究は存在してきた．それらの研究では，「面白さ」の具体的な特徴への詳細化（たとえば，爽快感など）や尺度の定義づけが試みられている[1)]．しかし，現時点では，やはり客観的な評価は困難であることに変わりはないだろう．

「面白さ」については，当然パブリッシャー（広報や販売を担当する企業で企画も行う）と開発者の間で話し合いは行われ，修正を要求されることもあるが，最終的には何らかの合意が得られているはずである．しかし，その合意が本当に守られているかは，ゲームをプレイして確認する必要がある．これはパブリッシャーが，開発者側が合意を守っていることを疑っているからではなく，両者間の合意に何らかの誤解が存在していないかを確認するためである．また，プレイしてみた結果，パブリッシャー側が考えを変えるということも起こりうる．もし，プロジェクトがかなり進行してから，重大なゲーム要件の変更が必要になれば，それまでのゲームアセット（ステージに配置するオブジェクトなどのゲーム内の素材）はすべて無駄になり，その分のプロジェクト・コストも無駄になってしまう．

このようなゲーム開発プロジェクトの不確実性に対処するために，どのようなPMを採用すべきであろうか．ビジネス・ソフトウェア開発では，Schwaber & Beedle（2001）をはじめとして，また，近年でもPMI（2017）など多くの文献が指摘しているように，従来のウォーターフォール方式に見られるような逐次的なPMではなく，スクラムを中心としたアジャイル方式が導入されている．ゲーム開発プロジェクトへのアジャイル方式の適用に関する事例は，ゲーム開発の複雑化が，比較的最近になってからということもあり，ビジネス・ソフトウェア開発に比べると多く確認できないが，CEDECなどのゲーム開発者のためのカンファレンスにおいては，田中（2012），田口（2014），畠山（2019）での報告がみられる．本稿においても，アジャイル方式のゲーム開発プロジェクトについて検討したい．

1）たとえば、藤江他（2004），Yee（2006），Poels et al.（2007），Komulainen et al.（2008），林他（2016）などの研究がある．

3．ソフトウェア開発プロジェクトにおけるアジャイル

3.1　Takeuchi and Nonaka（1986）および平鍋・野中（2013）におけるスクラム

アジャイル（方式による）開発とは，1986年に発表された竹内・野中稿の「The New New Product Development Game」で発表されたスクラムの概念を Ken Schwaber 氏が2003年に著書『Agile Software Development with Scrum』において紹介したことをきっかけに，米国のソフトウェア開発業界において広まり，現在でも数多くの企業が活用している PM 手法である．同論文においては，新製品開発方法で，速さと柔軟性が求められる場面では，成果物を紙に書き，それを壁越しの別のチームに渡すようなリレー形式ではなく，様々な専門性を持った人が一つのチームを組み，ラグビーのように開発の最初から最後まで一緒に働くことが求められるという内容が，米国の Nasa や日本の富士ゼロックス，キヤノン，本田の開発手法を例にとる形で解説されている（Takeuchi and Nonaka, 1986, pp.137-146; 平鍋・野中, 2013, p.199）．

野中教授は1986年の論文では想定していなかった、ソフトウェア開発業界におけるスクラムの活用を鑑み，平鍋・野中（2013）において，オリジナルのスクラムをソフトウェア開発の文脈において再考している（平鍋・野中，2013，p.197）．

野中教授は，ソフトウェア開発におけるスクラムは，オリジナルのスクラムの特徴，すなわち，①プロジェクト・チームは自ら組織化する②不安定な状態を保つ③開発フェーズを重複させる④「マルチ」学習⑤柔らかなマネジメント⑥学びを組織で共有するという 6 項目を依然として有しているが，「スプリント」や「エクストリーム・プログラミング」，「ユーザー・ストーリー」といった実践的なプラクティスを通じて，ソフトウェア開発に特化したものになっていると述べている（平鍋・野中，2013，p.201）．

ただし，オリジナルのスクラムには存在する，「多層学習」や「他能力学習」などをチーム外部へ浸透させる仕組，報酬システムや人事評価などのマネジメント手法をどのように援用するかなどについては，いまだソフトウェア開発におけるスクラムでは，整備されておらず，発展途上にあるとしている（平鍋・野中, 2013, p.212-216）．

3.2　PMBOK と Agile Practice Guide におけるアジャイル

PM の国際的な団体である PMI（Project Management Institute）は，同団体が2013年に発行した PM の標準知識体系である「The Guide to the Project Management Body of Knowledge」（以降 PMBOK）の第 5 版，および，同じく2017年にアジャイル方式を採用したプロジェクトに従事した実務家のグループである Agile Alliance® と共同で発行した『アジャイル実務ガイド』（以降，実務ガイド）において，アジャイル方式を扱っている．

実務ガイドにおいて，アジャイル方式は，ソフトウェア業界の思想的リーダーらが，2001年に提唱した『アジャイルソフトウェア開発宣言』を発端として始まった運動が基になっており，価値によって定義され，原則によって導かれ，様々な実務慣行を通して明らかにされるマインドセットであると記されている（PMI, 2017, p.8-10）．また，アジャイルは，図表 1 に示しているように，様々なフレームワークと方法を対象とする包括的用語であり，「価値に焦点を当てた」，「小さいバッチサイズ」，「無駄の排除」などリーンの概念をカンバンと共

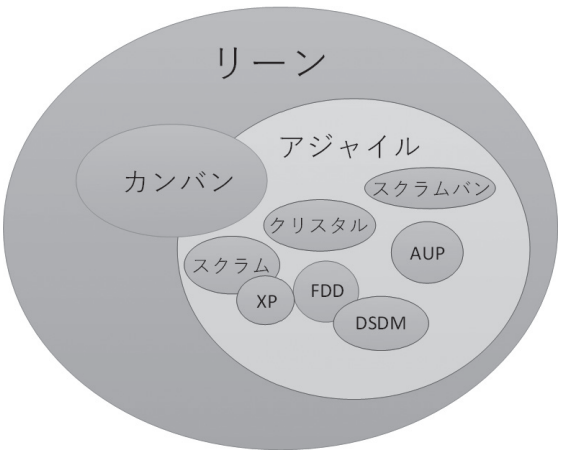
有したリーン方式のサブセットとして位置づけられている（PMI, 2017, p.11）.

また、実務ガイドでは，プロジェクト・ライフサイクルを引き渡しの頻度および変更の程度にもとづいて，予測型，反復型，漸進型，アジャイル型という連続的な 4 つの形態に分類しており，アジャイル型ライフサイクルはその中でも，引き渡しの頻度，変更の程度の両方において最も高いレベルのライフサイクルとして説明されている（PMI, 2017, p.19）．ただし，すべてのプロジェクトに完璧なライフサイクルがあるわけではなく，各プロジェクトは，その状況に応じた特性の最適なバランスを提供する位置を連続線上で見つけるべきとしている（PMI, 2017, p.19）.

アジャイル型ライフサイクルでは，価値のあるプロダクトを早期にかつ継続的に引き渡しすることで，顧客満足度が向上する．アジャイル型ライフサイクルでは，高度な変化に対応し，より多くのプロジェクト価値を提供するために，反復型アプローチと進捗型アプローチの両方が組み合わされている（PMI, 2017, p.25）.

実務ガイドでは，アジャイル型ライフサイクルを「反復ベースのアジャイル」と「フロー・ベースのアジャイル」に分類している．前者では，チームは完成したフィーチャーを引き渡しするのにイテレーションで作業する．チームは最重要フィーチャーに取り組み，チームとして協力してそれを完了する（PMI, 2017, p.25）．それに対し，後者では，チームは反復ベースのスケジュールではなく，作業を開始する遂行能力に基づいてバックログから開発するフィーチャーを引き出す．チームは，タスク・ボード上の列を使用してワークフローを定義し，各列の進行中の作業をマネジメントする．各フィーチャーを完了するのにかかる時間はそれぞれ異なるが，チームは，進行中の作業のサイズを小さく保ち，変更が必要な場合も手直しを減らせるように，課題を早期に特定する（PMI, 2017, p.25）.

このように，市場ニーズや技術に関しての不確実性の高低により，設定すべきプロジェクト・ライフサイクルおよび適用すべき PM 手法は異なる．様々な分野の中で，ソフトウェア開発プロジェクトは市場ニーズや技術の両方において高い不確実性があり，それが，アジャ



図表 1　多数のアプローチを総称する用語であるアジャイル

出典：PMI（2017），p.11

図表２　ライフサイクルの４つのカテゴリーの特性

特性				
手法	要求事項	活動	納品	目標
予測型	固定	プロジェクト全体で1回実行	1回の納品	コストのマネジメント
反復型	動的	是正されるまで反復	1回の納品	ソリューションの正しさ
漸進型	動的	増分毎に1回実行	頻繁で小さな部品	スピード
アジャイル型	動的	是正されるまで反復	頻繁で小さな部品	頻繁な納品とフィードバックを通した顧客価値

出典：PMI（2017，p.18）

イル宣言後のアジャイルの発展をもたらしたものであるといえる．もともと，PM が誕生したのは，1960年代のアメリカであり，公共，宇宙，軍事産業において用いられていた手法である．つまり，ソフトウェアというよりもハードウェアに重点を置いていた時代に発展を遂げてきたものである．そのため，PM の標準的な手法としてアジャイルが登場したのも近年になってからである．PMBOK において，第５版で初めて登場したこともアジャイルが比較的新しい手法であることを示している．

本稿で取り上げるゲーム開発プロジェクトはソフトウェア開発プロジェクトの一つであるため，アジャイル方式との親和性は高いと思われる．ただし，ゲーム開発に存在する不確実性には，先述した「面白さ」やキャラクターデザインやアートという主観的な要因があるため（Chandler, 2014, p.45），ビジネスにおけるソフトウェア開発プロジェクトよりも管理上，難しい側面があると思われる．

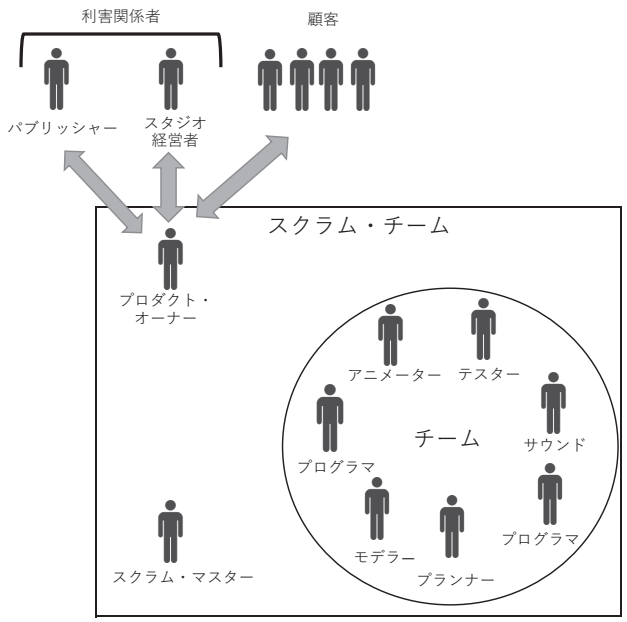
４．スクラムによるゲーム開発プロジェクト

近年，スクラムやカンバン方式によるゲーム開発を実践，啓蒙しているのが，Clinton Keith である．彼は，防衛産業でのプロジェクトを経て，90年代半ばより家庭用ゲーム機のゲーム開発プロジェクトに携わるようになり，それらのプロジェクトにおいて，アジャイルの概念やプラクティスを取り入れたゲーム開発を多く手掛けてきた（Chandler, 2014, p.43）．Keith（2010）におけるアジャイル方式のゲーム開発プロジェクトでは，ゲームのデザインや技術等の各要件を漸進的に決定していくコンセプト・フェーズ，およびプリ・プロダクション・フェーズにおいてもっともスクラムを活用し，プロダクション・フェーズにおいてはカンバン方式を併用する方法をとっている．本節では，Keith（2010）をもとに，一般的なアジャイル方式のソフトウェア開発に関する文献等も援用し，アジャイル方式によるゲーム開発について検討する．

4.1　スクラム・チーム

スクラムを実行するには，まず，スクラム・チームを編成する必要がある．スクラム・チームは，Schwaber and Beedle（2001）等のスクラム関連の諸文献が示している通り，プロダクト・オーナー，スクラム・マスター，そして様々な役割を担う専門家によるチームで構成されている．プロダクト・オーナーは，ゲームのビジョンをチームに伝え，プロダクト・バックログに入れる項目（PBI）の決定と優先順位付けを行い，ゲームの投資収益率を最大化する責任を負っている（Keith, 2010, p46, 訳書, p.56）．PBI とは，開発しなければならないフィーチャーに，優先順位をつけ，待ち行列に並べたものである（Schwaber and Beedle, 2001, p.32 訳書, p.37）．また，PBI に関しては，スクラム特有の概念であるユーザー・ストーリー（以降 YS）で表現される．YS は，顧客の視点からフィーチャーを端的に表現したものであり，〈誰として〉，〈何をしたいか〉，〈何故なら～だからだ〉というテンプレートを有している（Keith, 2010, p.85-88, 訳書, p.105-107）．

パブリッシャーやスタジオ経営者等の利害関係者は，ゲーム開発プロジェクトにおいて進行の可否や PBI を決定する上で，重大な決定権を持っている．また，スクラム・マスターはスクラム特有の管理上の役割であり，スクラムの価値，プラクティス，ルールが決められ，守られていることを保証する責任を持つ（Schwaber and Beedle, p31, 訳書, p.36）．いわば，スクラムのリーダー的存在だが，あくまでも，サーバントリーダー，あるいは，ファシリテーターとしての役割を担う．主な役割としては，①障害や問題が解決されていることを見守る，②進捗状況の計測，③計画，レビュー，レトロスペクティブ（振り返り）のファシリ



図表３　スクラム・チームと利害関係者

出典：Keith（2010）, p.45, 訳書, p.55

テーション，④継続的な改善の奨励と促進，⑤利害関係者とチーム間のコミュニケーションの支援が挙げられる（Keith, 2010, p.47, 訳書, p.58）.

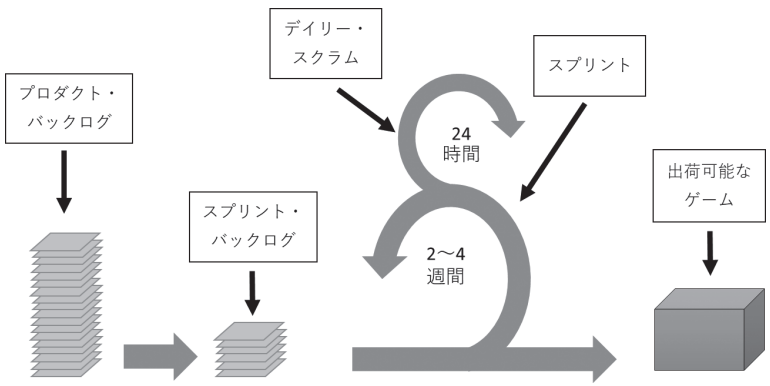
4.2　リリース計画とスプリント計画

スクラムで開発されるゲーム開発プロジェクトは，6～10人の機能横断的なチームにより，2～4週間ごとの固定されたタイム・ボックスが設定されたスプリントと呼ばれるイテレーションにより進行する（Keith, 2010, p.38, 訳書, p.47）.

逐次的なウォーターフォール方式のプロジェクトでは，コンセプト設計からゴールドマスターまでのプロセスに関する詳細な計画が事前に策定され，リリースも1度だが，漸進型のスクラムでは，顧客に動くソフトウェアを提示するためにリリースを繰り返す．そのため，数か月ごとに，マイルストーンでもあるリリース計画を並行して策定し，スプリントの計画と実行を進める（Keith, 2010, p.117-120, 訳書, p.146-150）．Schwaber and Beedle（2001, p.80-81, 訳書, p.91-92）では，リリース計画の重要性が実例とともに説明されている．すなわち，顧客はいつソフトウェアがリリースされるかについて常に心配しており，もしリリース計画が提示されるならば，たとえフィーチャー，コスト，品質間のトレードオフが必要になったとしても，顧客は納得する傾向にあるということである.

リリース計画では，プロダクト・バックログや（あれば前回のビルド）をもとに，利害関係者やスクラム・チームなどが議論し，①リリース・ゴールの設定，②リリース・ゴールを達成するためのYSとその優先順位付け，③各YSのストーリー・ポイントの見積もり，④各スプリントの長さの決定とスプリント・ゴールの決定，④リリース日の設定を順番に行う（Keith, 2010, p.117, 訳書, p.147）.

スクラム・チームは，リリース計画の範疇において，スプリント計画を策定する．スプリント計画では，利害関係者やスクラム・チーム内外の専門分野の技術者の参加のもと，スプリント・バックログに収めるべきPBIの決定およびPBIを達成するのに必要なタスクの決



図表4　スクラムの全体像[2)]

出典：Keith（2010）p.39, 訳書, p.48

2）Schwaber et al.（2001, p.8, 訳書 , p.9）にもほぼ同じ図が掲載されており，スクラムを取り扱う多くの文献にも同様の図が挙げられている.

定を行う（Keith, 2010, p.59-61, 訳書, p.72-p.75）.

スプリント・バックログの編成では，現在のチーム構成を考慮して達成できると思われる優先順位の高い PBI を選択するが，選択される PBI は 1 回のスプリントで開発できる程度に小さく，タスクを作成するのに必要十分な詳細情報を持たなければならない．そのため，計画の立案作業は，最も優先度の高い PBI の詳細を定義することに費やされ（Keith, 2010, p.110, 訳書, p.137），優先度が低い PBI に関しては，プロジェクトが進むまで詳細を定めない．

一つ一つの PBI を表現する YS にはその規模を表すストーリー・ポイントが設定され，スプリント計画で各 PBI のストーリー・ポイントが見積もられる．ストーリー・ポイントは開発の規模を数値化したものだが，あくまでも相対的な尺度であり，個々のポイントの設定には数分しかかけないため，その PBI をいつ完成させるかについてはコミットされない（Keith, 2010, p.115, 訳書, p.143）．ウォーターフォール型のプロジェクト開発では，当初の予定に盲目的に従い，プロジェクト終盤まで進捗を信頼性ある形で測定できるようにならないが，スクラムでは，各スプリントで達成したストーリー・ポイントを合算することで，ベロシティ（PBI を消化する速度）を計測し，いつまでにどのくらいの規模のフィーチャーを開発することができるのか，もしくは，一定の規模を開発するにはどれくらいの期間が必要なのか予測できる（Keith, 2010, p.111, 訳書, p.139）.

プロジェクトの初期ではベロシティの見積もりの精度は低いものの，プロジェクトの進行が 3～4 スプリント進むにつれ，精度が高まるため，スクラム・チームの開発スピードの実態が分かるようになる（Rasmusson, 2010, p.125, 訳書, p.162）.

PBI の選択が終われば，PBI をタスクに分解する．たとえば，「プレイヤーがジャンプできる」という項目を抽出したのであれば，それをスプリント計画中に「アニメーター担当: ジャンプ・アニメーションの開発（10時間）」，「プログラマ担当: ジャンプ機能の制御（5 時間）」，「プログラマ: ジャンプの操作性の調整（16時間）」，「プランナー: ジャンプの調整（6 時間）」というタスクに分解することができる（Keith, 2010, p.39, 訳書, p.48）．タスクの実行に必要な時間は，1 時間単位，もしくは30分単位で見積もられるが，その際に，割り込みで入る作業や問題の発生を取り除いた純粋な作業時間で見積もられる（Keith, 2010, p.62, 訳書, p.77）.

4.3　進捗の計測

スプリント中，チームは進捗状況に関する情報を共有し，スプリント・ゴールの障害の特定と，除去を行う必要があるが，その際，進捗状況の管理に有用な，バーンダウン・チャートやタスク・カード等のスクラムの標準的なプラクティスを活用する（Keith, 2010, p.68, 訳書, p.83）.

タスク・カードは行うべきタスクが書かれたカードであり，それらのカードをタスク・ボードという掲示板に貼ることで，未着手，作業中，完了の各タスクを把握することができる.

チームは日々，残作業の見積もり時間の合計値を更新して，スプリント・ゴールへの進捗状況を計測し，バーンダウン・チャートと呼ばれるグラフに，この日々の計測をプロットする（Keith, 2010, p.69, 訳書, p.84）．このグラフでは，スプリント・ゴール達成までに残っている理想の時間を 1 日あたりの減少時間をもとに算定し，その傾向線を引く．図表 5 の例で

は，スプリント・ゴールが達成できない状況下にあることを把握できる．

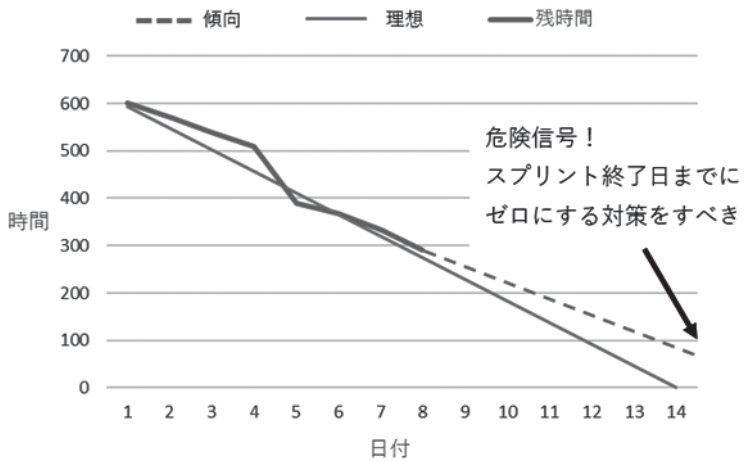
4.4 デイリー・スクラム（DS）

チームは毎日集まり，15分を上限としたDSで，先述したタスク・ボードやバーンダウン・チャートを用い，①全員の作業の同期化，②スプリント・ゴールに向けたチームのコミットメントの再確認，③障害の識別，④メンバー同士の認識のずれの解消をはかる（Keith, 2010, p.74, 訳書, p.89–90）．具体的には，①前回のDSから何をしたか，②次回までに何を達成するか，③問題や障害があるかについて報告し，話しあうが，DSで問題を解決することは重要視されない（Keith, 2010, p.74, 訳書, p.90）．

4.5 スプリント・レビュー

スプリント・レビューは，スプリントの最終日に行い，スクラム・チームと利害関係者が一緒にゲームをプレイし，開発した成果について議論する（Keith, 2010, p.75, 訳書, p.91）．スプリント・レビューを行うことで，チームと顧客，利害関係者の間で，広範囲に及ぶコミュニケーションが促進され，ゲームのビジョンが改善され，利害関係者が実際にプレイする機会が増える（Keith, 2010, p.76, 訳書, p.92）．

また，近年の大規模なゲーム開発では，複数のスクラム・チームで構成される大規模なスプリント・レビューも行われるが，プロジェクトのスタッフ全員が進捗状況を把握できる点，チーム間の統合やビルド方法の習慣の改善を促進する点，利害関係者の時間的負担軽減などの利点がある一方で，小規模なスプリント・レビューにおける利害関係者と開発者との1対1のコミュニケーションが阻害されるなどの欠点がある（Keith, 2010, p.77, 訳書, p.94）．



図表5　バーンダウン・チャートの例

出典：Keith, 2010, p., 訳書, p.85の図に理想の進捗線を加筆した

4.6　振り返り（スプリント・レトロスペクティブ）

スプリント・レビュー後には，チーム・メンバー全員とスクラム・マスターの参加による「振り返り」が行われる．振り返りでは，①やめるべきことは何か②新たに始めるべきことは何か③続けるべきことは何かという3つの質問についてチーム・メンバーが自問自答することで，開発そのものの改善が図られる（Keith, 2010, p.78-79, 訳書 , p.95-96）．

スクラム・マスターは3つの質問に対する回答を全て記録し，振り返りのあとで，その結果をチーム・エリアに貼り出し，次のスプリントで実行できる項目をチェックする（Keith, 2010, p.80, 訳書, p.97）．

振り返りの利点として技術的負債の削減が挙げられる．技術的負債とは，先延ばしされると，後に開発チームにとって負担となる，技術的問題を表現した比喩であり，放置すると開発のベロシティが落ちてしまう（貝瀬他, 2015, p.91）．ベロシティの低下はプロジェクトが予定期間内に終了しないリスク，およびプロジェクト・コストが予算を超過するリスクを意味している．振り返りは，仕事の進め方の改善が目的であるが，振り返る過程で技術的負債の存在を明らかにして，早期に解決することも肝要であると思われる．

4.7　プロダクション・フェーズ

開発フェーズがプロダクション・フェーズにまで進むと，技術面や要件に関する不確実性は低下する傾向にあり，アジャイル方式のスクラムの適用の仕方を調整する必要がある（Keith, 2010, p.133, 訳書, p.167）．プロダクション・フェーズでは，コア・メカニクスには極力変更を加えずに，生産効率と着実な改善に集中する（Keith, 2010, p.131, 訳書, p.165）．

スクラムにおけるタスク・ボードはスプリント中のタスクを可視化するが，アセット制作で必要となるタスク同士の依存関係や制作の速度を可視化しないという問題が存在する（Keith, 2010, p.137, 訳書, p.172）．この問題により，制作ライン上で手待ちが発生してしまうため，多くのプロジェクトでは，プロダクション・フェーズに入った段階でスクラムが放棄されているそうである（Keith, 2010, p.139, 訳書, p.174）．この問題を解決するためにトヨタのかんばん方式を利用したプラクティスの一つであるカンバン・ボードで開発フローの可視化と平準化が図られる（Keith, 2010, p.140-155, 訳書, p.176-194）．

カンバン・ボードでは，区切られた列によりワークフローの各ステップと作業の許容量が示される．列内のカードは制作中のステージを示しており，各段階の作業が完了すると，左側の列から次のステージのカードを引くことができる（Keith, 2010, p.141, 訳書, p.177）．カンバン・ボードを準備したら，制作フローのばらつきを平準化するために，制作フローの各ステップにタイム・ボックスを決め，許容量としてカードの枚数（例えば，図表6上部の「プロダクト・バックログは5枚のカードの配置が許容されている.）を設定する（Keith, 2010, p.142-146, 訳書, p.178-182）．

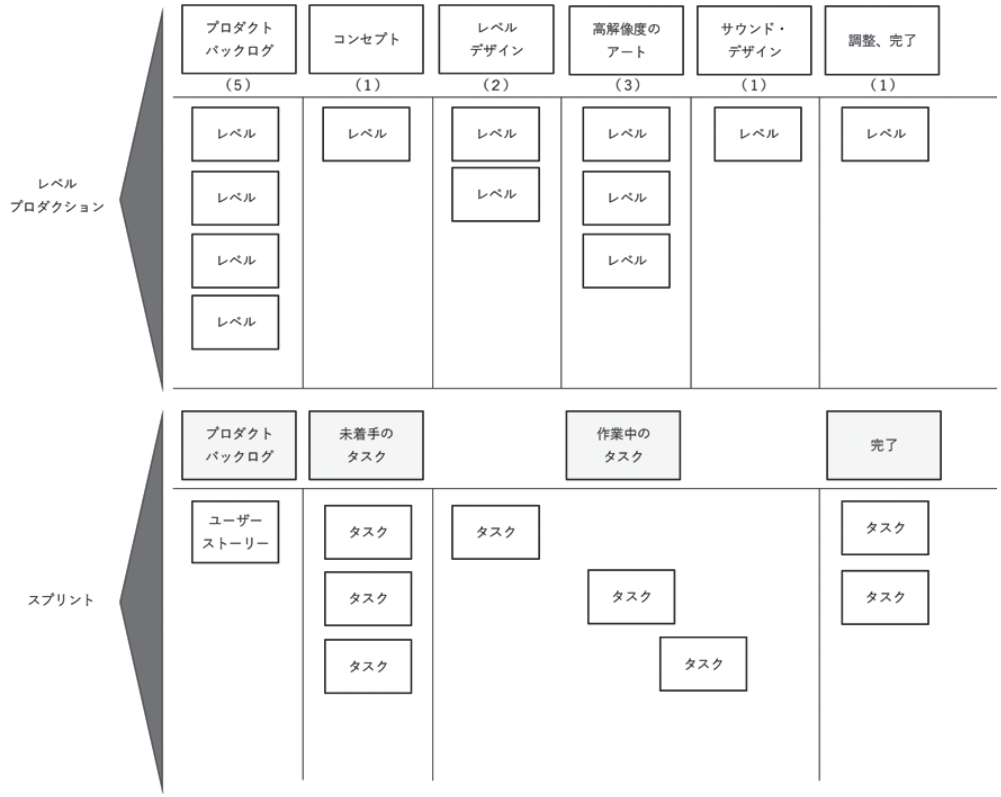
また，プロダクション・フェーズにおけるステージ制作は，このフェーズにおける予算の50%以上を必要とする程の高コストが発生するため，ステージ・オブジェクトの品質要件がコストどのように影響を及ぼすかについて，プリプロダクション中のスクラムにおいて情報の精度をあげるように改善する必要がある（Keith, 2010, p.229, 訳書, p.284）．

4.8　テスト・フェーズ

Keith（2010, p.251, 訳書, p.307）によれば，スクラムでは，従来行われるようなテストは行われず，スプリントによる繰り返しサイクル内でテストを行う．従来のゲーム開発では，プロジェクトが終盤に差し掛かり，デバッグはされていないが，フィーチャーは全て実装されているα版が完成するまでテストは行わず，α版で初めて，多くのテスターを雇うため，開発中のゲームに関する必要最低限の教育だけしか行えず，結果的に最高の品質保証ができなくなってしまう（Keith, 2010, p.250, 訳書, p.306）．スクラムでは，QA（Quality Assurance：品質保証）はプロジェクト期間中を通して，チームメンバーにより協働的に行われ，致命的な欠陥の早期解決が図られる．

5．アジャイル方式の PM における管理会計の役割

アジャイル方式の PM は，スクラムによる漸進的なプロジェクトの進め方やカンバン方式によるプロジェクトの停滞と遅延を防ぐ方法により，ウォーターフォール方式に見られる逐次的な PM の問題点を防ごうとしている．そのような，アジャイル方式による PM において



図表 6　スプリントの循環を追加したカンバン・ボードの例

出典：Keith（2010）, p.154, 訳書, p.193

管理会計はどのような役割を担うべきだろうか．坂手（2012）で考察したように，PM の国際標準である PMBOK の提唱する10の知識エリアでは，プロジェクト・タイム・マネジメントにおいて活動基準予算が，プロジェクト・品質マネジメントにおいて品質コストが扱われていたが，アジャイル方式の PM においては，特にコスト・マネジメント領域における役割期待が大きいと思われる．具体的には，プロジェクト計画段階でのコストの見積もりと予算化，および，プロジェクト・コストのコントロール，そして PM 手法の効果測定である．

5.1　見積り情報の提供

プロジェクト・コストの見積もりに関しては，不確実性の高いプロジェクトを対象とし，頻繁な変更が必要となるアジャイル方式の PM では，PMI（2017, p.92）が指摘しているように，簡易的な見積もりで済ませる方が合理的であると考えられる．

一つの方法として，Lacey（2016）における見積もり方法を使うことで，コストを含めた納期やフィーチャーについて事前にプロジェクトの要件について見積もることができる．

まず，プロジェクト期間を 1 スプリントの期間である 2 週間で除すことで許容スプリント数を算出する．スプリントあたりの人件費（チームの時間当たり人件費×１日あたりのプロジェクト従事時間×10日間）を乗じれば，許容スプリントに必要なコストを見積もることができる．これをプロジェクト予算と比較することでコストの余裕，あるいは不足がおおまかに把握できる．また，総予算を 1 スプリントの人件費で除すことで，予算上では何スプリント許容されるかが判明し，許容スプリント数と比較することで期間的な余裕や不足を把握することができる．さらに，許容スプリント数にベロシティを乗じることで許容期間内に消化できるストーリー・ポイント数が判明する．消化できるストーリー・ポイントと必要とされる PBI のストーリー・ポイントを比べて，プロジェクト目標の達成が無理であれば，コストや期間を調整する必要性を利害関係者に示唆する必要がでてくる．

開発の初期段階においては，簡易見積もりが合理的ではあるが，Keith（2010）によれば，ゲーム開発プロジェクトで，プロダクション・フェーズという最もコストのかかるフェーズでは，詳細なコストの見積もりが必要となる．なぜなら，進捗状況との兼ね合いで，フィーチャーやステージ数を削除したり，出荷日を伸ばすなどの意思決定を早期に行う必要性が生じた場合，フィーチャーやステージのコストを事前に把握しておく必要があるからである（Keith, 2010, p.135，翻訳, p.171）．このプロダクション・フェーズに関するコスト見積もりは最初から可能ではなく，プリ・プロダクション・フェーズにおいて，ゲームのデザインや技術，アートやサウンドに関する要件の詳細が確定するにしたがって段階的に行っていくものである（Keith, 2010, p.135, 訳書, p.171）．

また，PMI（2017, p.59）では，アジャイル方式の痛点として，プロダクト・バックログ項目が非効率に並べられてしまうことが挙げられ，それを解決するために，CD3（Cost of Delay Divided by Duration）という方法が提唱されている．Black Swan Farming（2020）によれば，CD3 では，遅延コスト（フィーチャーが遅延することで発生するコスト）を開発からデリバリーまでの見積もり時間で割った CD3 スコアによって PBI を優先順位付けする．この遅延コストの見積り情報の提供もアジャイル方式では，管理会計の役割と言えるのではないだろうか．ただし，ビジネス・ソフトウェア開発においては，フィーチャーごとに対価

を受取る場合，合理的に遅延コストを見積もることが可能だが，ゲーム開発プロジェクトの場合は，パブリッシャーからフィーチャーごとに対価を受け取る契約を結ばなければ，フィーチャーごとの対価を見積もることは困難であると考えられる．

また，プロジェクト中の変更要求に応じるためには，コストに余裕を持たす必要があり，この余裕に関しては，プロジェクト開始前に変更量の最大値を想定して見積もり，利害関係者との間で合意をしておくことが肝要であるとしている（英他 , 2014, p.117）．このコストの余裕に関しては，Leach（2005, 2014）では，ゴールドラットのクリティカル・チェーンにおけるプロジェクト・バッファをコスト管理に適用しており，タスクごとではなく，プロジェクト全体で 1 つのコスト・バッファを設定すべきであると述べている（Leach, 2005, p.205）．

5.2　進捗管理のための情報提供

従来のプロジェクト・マネジメントにおいて，プロジェクトの進捗管理に EVM（アーンド・バリュー・マネジメント）が用いられることは幅広く知られている．EVM では，スケジュールされた作業に割り当てられた認可済み予算であるプランド・バリュー（PV）と完了した作業に関する予算であるアーンド・バリュー（EV）との差額であるスケジュール差異，および，EV と実コスト（AC）との差額で表したコスト差異（CV）を測定し，そこから，EV ／ PV で計算するスケジュール効率指数（SPI）や EV ／ AC で計算するコスト・パフォーマンス指数（CPI）を導き出し，プロジェクト全体の進捗を管理する（PMI, 2017b）．

PMI（2017a）では，EVM で用いるベースラインは長期的な見通しを表す予測の成果物であるが，アジャイル方式では，チームが実際に引き渡したものを測定するという経験的，かつ価値ベースの測定が用いられることが指摘されている．そのため，通常は先述したバーンダウン・チャートを用いて，コストではなく，チームのベロシティと完了したフィーチャーのストーリー・ポイントの測定をスプリントごとに行う．しかし，スプリントごとのアーンド・バリューの計測が必要である場合は，図表 7 に示した EVM に基づいたバーンアップ・チャートを用いることが可能である（PMI, 2017a, p.68）．

PMI（2017）におけるアジャイル方式の EV 分析では，SPI を完了したストーリー・ポイントを計画したストーリー・ポイントで除して計算しており，コストの業績測定と別に行っている．この点が従来の EVM との違いとなっている．

ゲーム開発プロジェクトは従来の予測型プロジェクト・ライフサイクルで開発することは難しく，プロジェクトの途中でパブリッシャーからの要求や途中の成果物であるゲームのプロトタイプをプレイして面白みが感じられないなどの理由で，要件が変更される可能性が高い．その場合，要件の変更にともなってベースラインそのものが変動してしまうため，差異の解釈については慎重に行う必要がある．

また，先述したように，プロダクション・フェーズにおいてフィーチャーの追加等の要件変更が必要となった場合に，プロダクト・オーナーはフィーチャーの追加がコストへ与える影響を判断する必要がある．その際，適切な意思決定を行うために，プリ・プロダクション・フェーズ中に 1 ステージだけ制作して，アセットのコストを把握するなどの方法をとる必要がある．このように，アジャイル方式では，プロジェクト開始時に精度の低いコストの予測に時間をかけるよりも，変更への意思決定を迅速に行うために，プロジェクト期間中に短い

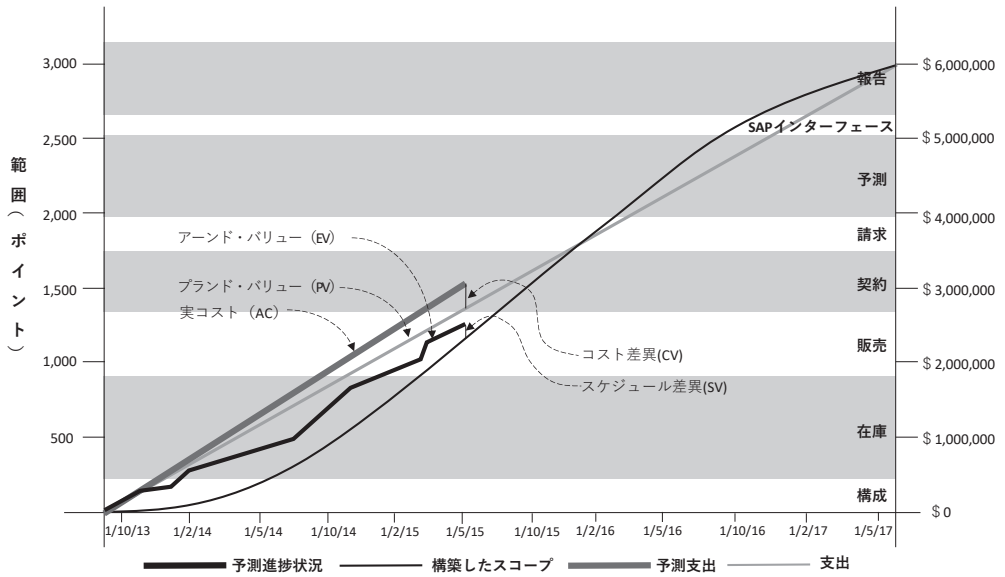
サイクルでコスト情報を収集し続けることが重要になる．

5.3　プロジェクト・ライフサイクルの設定と PM 手法の効果測定

プロジェクト目標を達成するためには，まずは，プロジェクト開始前に最も適切なプロジェクト・ライフサイクルを設定する必要がある．その設定を間違うと，プロジェクトの後半で重大な手戻りが発生する可能性が高くなり，結果，予算に対する超過コストを招くことになってしまう．そして，プロジェクト・ライフサイクルの設定後にはそのライフサイクルに最も適した PM 手法を選択すべきである．

ゲーム開発プロジェクトにおいては，これまでに類をみないような大型のゲームを開発する場合もあれば，追加コンテンツという小規模で，不確実性の低いプロジェクトも存在する．後者の場合は，ウォーターフォール方式や一部をアジャイル方式にするというハイブリッド方式でも管理可能である（Chandler, 2014, p.46）．いずれにしても，プロジェクト・ライフサイクルの設定と PM 手法の選択を行う際には，費用対効果についての情報が必要となる．例えば，英他（2013, p.227–236）では，ウォーターフォール方式を一部アジャイル方式に変更した際に，当初計画に対して工数がどの程度削減できるかを予測し，削減効果を測定している．そして後に，プロジェクト開始後の変更対応により増えた工数を測定し，コストが顧客と合意したコストの余裕分内に収まっているかどうかを測定したうえで，アジャイル方式の PM の費用対効果について判断している．このような費用対効果の判断に必要な，コスト情報の提供も管理会計への役割期待であると考えられる．

アジャイル方式では，これまで，種々のプラクティスが生み出されてきた．そうしたプラクティスの特徴として，インクリメンタルな成果物を提供し，不確実性から生じるリスクを



図表 7　アジャイル方式におけるアーンド・バリュー

出典：PMI（2017a, P.69）

軽減するためにあえて非能率的なプラクティスを活用している点がある．たとえば，スクラムで活用されるエンジニアリング技法にエクストリーム・プログラミング（XP）があり，具体的な XP のプラクティスの例として，テスト駆動開発やペア・プログラミングが存在する．

Beck（2003）で示されているように，テスト駆動開発は最初からコードにテストコードを組み込んでおく方法であり，ゲームそのもののコードを書く前にテストコードを書くため，最初のテストは必ず失敗するようになっている．そうすることで失敗するコードに後々まで気づかぬままプロジェクトが進行し，手戻りのコストが発生するリスクを防ぐことができる．また，ペア・プログラミングは二人で一台の PC を使い，プログラミングとプログラミングの改良を同時に行っていく手法である（Beck, 2003, p.42, 訳書, p.40）．こうすることで，知識を共有したり広げることが可能になったり，コードの品質を保つことが可能になる（Keith, 2010, p.212-214, 訳書, p.262-264）．このような XP の手法はスクラムにとって多くのメリットをもたらすものではあるが，より多くのコストも必要となり，このコストを上回るベネフィットをもたらすものでなければならない．そのような各々のプラクティスが生み出すコストとベネフィットをスプリントのふりかえり時やプロジェクト後の自己検分時に比較できるようなコスト情報を日頃から収集することも管理会計の役割であると考えられる．

6．まとめと今後の課題

本稿では，不確実性の高いプロジェクトにおけるアジャイル方式の PM についてゲーム開発プロジェクトを例にあげて検討し，管理会計にどのような役割期待があるかについて考察した．アジャイル方式は要件変更が頻繁に行われることを受け入れ，インクリメンタルな成果物を確認することで見当違いな方向へプロジェクトが進むことを防ごうとする．そのような PM 手法では，計画指向による厳密な要件定義とタスク定義にもとづくコストの見積もりに注力しすぎずに，バーンダウン・チャート上において，YS の消化状況の確認とベロシティの計測を頻繁に行い，コストの変動をプロジェクト中にコントロールすることが重要であると思われる．そのためには，スプリントを回す中で，フィーチャーやプラクティスのコストへの影響をプロジェクト中に探索的に把握する必要がある．そのような情報収集と収集した情報にもとづく意思決定は，スクラムにおけるプロダクト・オーナーを中心としたチーム・メンバーのように，開発に直に関わっている者でなければ困難であろう．別言すれば，不確実性の高いプロジェクトでは，プロジェクト・コストについて，マネジメントによる計画重視の管理を緩め，開発現場での当事者によるコントロールを重視した管理を行うほうが結果として高いパフォーマンスを発揮できることを示唆していると思われる．

また，Chandler（2014）が指摘しているように，アジャイル方式はいわば急がば回れとでも表現できるようなプラクティスを採用することが多く，ゲーム開発の内容によっては，アジャイル方式によらない従来方式の PM を採用しても問題がない場合もある．そのような場合，アジャイル方式の PM のコスト対効果を再考する必要もあるだろう．

アジャイル方式は，顧客価値に重きをおき，顧客に頻繁にその価値を確かめてもらうところに特徴があるが，ゲーム開発ではソーシャルゲームであれば，可能であるものの，AAA

といわれる大作を中心とした据え置き型ハードのゲームでは，ゲーム内容が事前に漏れてしまうこともあり，初期から顧客に関わってもらうことは困難である．そのため，顧客価値の向上をどのような形で把握するかについては課題が残るだろう．

本稿は，様々な文献等をもとにアジャイル方式における管理会計の役割について考察したものであり，実際にゲーム開発会社において，プロジェクト・ライフサイクルの設定を決断させる要因とは何か，どのようにPM方式をライフサイクルに合わせて選択しているか，およびプロジェクト・コストの見積もりにどの程度注力しているのか，プロジェクトの進捗状況の確認方法についてコスト情報をどの程度重視しているか等について，インタビューや質問票調査による分析が必要であり，今後の研究課題である．

参考文献

馬場保仁，山本貴光著．2008.『ゲームの教科書』筑摩書房.

Beck, K et al. 2001. *Manifesto for Agile Software Development.* https://agilemanifesto.org/iso/en/manifesto.html（参照2020年6月8日）.

Beck, K. 2003. *Test-Driven Development: By Example.* 2nd edition. Addison-Wesley Professional. 長瀬嘉秀監訳. 2003.『テスト駆動開発入門』ピアソン・エデュケーション.

Beck, K., and Cynthia A. 2004. *Extreme Programming Explained: Ebrace Change.* Addison-Wesley Professional. 角征典訳. 2015『エクストリームプログラミング』オーム社.

Black Swan Farming. 2020. *Cost of Delay Divided by Duration.* https://blackswanfarming.com/cost-of-delay-divided-by-duration/（参照2020年7月14日）.

Chandler, H. M. 2014. *The Game Production Handbook.* 3rd edition. Jones & Bartlett Learning.

Cohn, M. 2006. Agile Estimating and Planning. Prentice Hall. 安井力，角谷信太郎訳. 2009.『アジャイルな見積りと計画づくり』マイナビ出版.

Dring, C. 2020. *What is happening with video game sales during coronavirus.*（28th March 2020）. https://www.gamesindustry.biz/articles/2020-03-28-what-is-happening-with-video-game-sales-during-coronavirus（参照6月1日）.

藤江清隆，馬場章．2004.「ゲームの面白さとは何か―テレビゲームのプレジャラビリティをめぐって―」『日本バーチャルリアリティ学会誌』8（1）: 15-19.

Grenning, J. 2002. *Planning Poker or How to avoid analysis paralysis while release planning.* https://mail.renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf（参照2020年6月15日）.

Gu, Tianyi. 2020.「新型コロナウイルスによるモバイルゲーム市場への影響：プレイヤーエンゲージメント急増により2020年の市場規模は770億ドル超へ」（2020年6月8日） https://newzoo.com/insights/articles/mobile-games-market-engagement-revenues-covid-19-gaming-jp/（参照2020年7月3日）.

畠山彰秀．2019.「ゲーム開発におけるスクラムのすゝめ ～失敗は成功のもと～」（2019年9月5日）CEDEC 2019. https://cedil.cesa.or.jp/cedil_sessions/view/2113（参照2020年6月15日）.

林志修，馬場章．2016.「デジタルゲームにおける共感尺度の開発」『日本デジタルゲーム学会　2016年　夏季研究発表大会　予稿集』49-52.

Hight, J, and Jeannie Novak. 2008. *Game Project Management.* Delmar Cengage Learning.

平鍋健児，野中郁次郎．2013.『アジャイル開発とスクラム』株式会社翔泳社.

株式会社角川アスキー総合研究所．2022.『ファミ通ゲーム白書2022』株式会社角川アスキー総合研究所.

貝瀬岳志，原田勝信，和島史典，栗林健太郎，柴田博志，家永英治．2015.『スクラム実践入門　成果を生み出すアジャイルな開発プロセス』技術評論社 .

Keith, C. 2010. *Agile Game Development with Scrum.* Addison-Wesley Professional. 江端一将訳. 2012.『アジャイルなゲーム開発　スクラムによる柔軟なプロジェクト管理』SB クリエイティブ .

Komulainen, J., J. Takatalo, L. Mikka and N. Göte. 2008. *Psychologically structured approach to use experience in games, Proceedings of the 5th Nordic conference on Human-computer interaction.* Building brideges Pages: 487-490.

Koster, R. 2018. *The Cost of games.*（17th January 2018）https://www.raphkoster.com/2018/01/17/the-cost-of-games/（参照2020年7月4日）.

Lacey, M. 2012. *The Scrum Field Guide.* Addison-Wesley Professional. 安井力，近藤寛嘉，原田騎郎訳．2016.『スクラム現場ガイド　スクラムを始めてみたけどうまくいかない時に読む本』マイナビ出版.

Leach, P. 2005. *Lean Project Management: Eight Principles for Success.* Advanced Projects, Inc. 小林英三監訳．2007.『リーンプロジェクトマネジメント』ラッセル社.

Leach, P. 2014. *Critical Chain Project Management.* 3rd edition. Artech House.

西村直人，永瀬美穂，吉羽龍太郎．2013.『スクラム・ブート・キャンプ　ザ・ブック』株式会社翔泳社.

Poels, K, De Kort, Y, and W. Ijsselsteijn. 2007. *It is always of fun!" Exploring Dimensions of Digital-Game Experience using Focus Group Methodology, Pro-ceedings of the International Academic Conference onthe Future of Game Design and Technology : Future-Play 2007, November 15-17, 2007, Toronto, Canada. Association for Computing Machinery: 83-89.*

Project Management Institute. 2014.『プロジェクトマネジメント知識体系ガイド（PMBOK®ガイド）第5版』Project Management Institute.

Project Management Institute. 2017a.『アジャイル実務ガイド』Project Management Institute.

Project Management Institute. 2017b.『プロジェクトマネジメント知識体系ガイド（PMBOK®ガイド）第6版』Project Management Institute.

Rasmusson, J. 2010. *The Agile Samurai.* Pragmatic Bookshelf. 西村直人，角谷信太郎監訳. 2011.『アジャイルサムライ―達人開発者への道』オーム社.

坂手啓介．2012.「プロジェクトマジメント再考」『大阪商業大学論集』第163号 : 37-53.

Schreier, J. 2017. *Blood, Sweat, and Pixels: The Triumphant, Turbulent Stories Behind How Video Games Are Made.* Harper Paperbacks. 西野竜太郎訳 . 2019.『血と汗とピクセル：大ヒットゲーム開発者たちの激戦記』合同会社グローバリゼーションデザイン研究 .

Schwaber, K and Mike Beedle. 2001. *Agile Software Development with Scrum.* Prentice Hall. 長瀬嘉秀・今野睦監訳. 2003.『アジャイルソフトウェア開発スクラム』ピアソン・エデュケーション .

田口昌宏 . 2014.「スクラムフレームワークがゲーム開発へもたらすもの」札幌 CEDEC 2014.（2014年

11月20日）https://www.slideshare.net/taguchimasahiro/ss-41791550（参照2020年6月5日）.

Takeuchi, H, and I Nonaka. 1986. The new new product development game. *Harvard Business Review* January-February: 137-146.

田中宏幸. 2012.「アジャイル開発を使ったゲームの作り方」Game Community Summit 2012（2012年1月19日）https://www.slideshare.net/swiftnest/gcs-13381463?from_action=save（参照6月15日）.

Yee, N. 2006. Motivations for play in online games. Cyberpsychology and Behavior 9: 772-775.