研究コース４（AQA グループ）

アジャイル開発における段階的品質の積み上げによる品質保証

Quality Assurance by quality stepwise refinement in Agile development

リーダー：伊藤　潤平（ウイングアーク 1st 株式会社）
研 究 員：山口　繁（日本ユニシス株式会社）
　　岡崎　一洋（サイボウズ株式会社）
　　横須賀　信介（テックスエンジソリューションズ株式会社）
　　木本 和伸（富士通株式会社）
　　山中 美穂（株式会社東芝）
主　査 ：永田　敦（サイボウズ株式会社）
副 主 査：山口　鉄平（ヤフー株式会社/一般社団法人アジャイルチームを支える会）
アドバイザー：細谷　泰夫（三菱電機株式会社）

研究概要
アジャイル開発は欧米を中心に国内ベンダーにおいても普及しているが，アジャイル開発における品質保証プロセスはまだ世の中に確立されていない．動くソフトウェアを継続的に提供し，顧客のニーズに素早く対応するのがアジャイル開発の手法であるが，イテレーティブにリリースしたプロダクトにおいて品質が保証されたものかの判断が難しいと言える．本研究ではアジャイル開発に対する品質保証活動において，戦略的なテスト計画を策定し，イテレーティブにリリースしたプロダクトの品質説明ができる状態，また，プロダクト品質を段階的に積み上げることによって総合的に品質保証出来るフレームワークを提案する．

Abstract
As agile development has been spreading mainly in Europe and the United States and in domestic vendors, but the quality assurance process in Agile development has not yet been established in the world. Agile development method is to continually provide moving software and quickly respond to customer's needs, but it can be said that it is difficult to judge whether the quality is guaranteed in the product released for iterative. In this research, in the quality assurance activities against agile development, we propose a state where we can develop a strategic test plan and provide quality explanation of products released to iteratively. We also propose a framework that can comprehensively guarantee quality by stepwise refinement product quality.

1. はじめに

2001 年に「アジャイルソフトウェア開発宣言」が発表されて以降，アジャイル開発は欧米において開発手法のスタンダードになりつつある[1]．また，国内においてもアジャイル開発を導入する企業が増えている傾向にある[2]．

「アジャイルソフトウェア開発宣言」では「動くソフトウェア」が重要な尺度といった説明はあるが，特に品質保証に対する定義はない．スクラムによるアジャイル開発で部分

的な品質保証活動を説明している例[3][4] もあるが,総合的な品質保証活動に関するフレームワークはまだ世の中に確立されていない.

本研究チームの研究員の多くは品質保証に関する組織に所属しており，プロダクト開発途中においてステークホルダーに品質を説明することが求められる.

本論文では，アジャイル開発を行う際に開発プロセスと品質保証プロセスを明確に分けて，戦略的なテスト計画を策定し，徐々に品質を確保することにより確保した品質が積み上がっていることを確認するためのフレームワークを提案する.

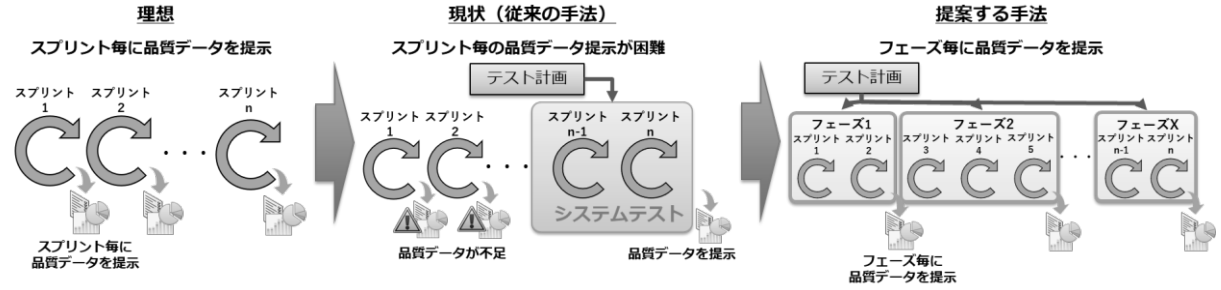
これにより，最終的な成果物で品質を判断したときに後戻りが発生するというリスクを軽減でき，また，プロダクトオーナー（PO）が投資継続判断をすることができる.

以降，２章では提案するフレームワークについて述べる．３章ではフレームワークの有効性を検証するために実際の開発現場で試行した結果とその考察を述べる．４章では検証結果受けて，抽出された今後の課題について述べる.

2. 提案するフレームワーク

アジャイル開発における品質保証プロセスの理想は，スプリント毎に品質データを提示することによって開発のリスクが明確になり，プロダクトのリリースが可能かどうか判断できることと考える．しかし現状では，非機能を含むシステムテストは機能が実装完了する開発後半で行っているため，スプリント毎に品質データを提示することが難しい.

そのため，本研究ではスプリント毎ではなく，フェーズという単位を用いて，そのフェーズ毎に品質データを提示することで徐々に品質を確保する仕組みを品質保証プロセスのフレームワークとして考案した．本フレームワークは，ウイングアーク１ｓｔ社の品質保証プロセス[5]や，見通しのよいテストの段階的詳細化の手法[6] を参考として，研究チームとしてアレンジを加えたものになっている．以降の節に考案したフレームワークの実施概要について述べる.



イメージ図 1　提案するフレームワーク

2.1 開発プロセスと品質保証プロセスの定義

開発プロセスがウォーターフォールかアジャイルかに関わらず，プロダクトの品質保証をする上では品質保証プロセスを定義して明確に開発プロセスと分けて考える.

開発プロセスはモノづくりにおける活動に重点を置く．一方で品質保証プロセスは品質の計測および分析の活動に重点を置き，製品の生産活動でどのように品質が確保されるのかを確認するためのプロセスと位置付ける.

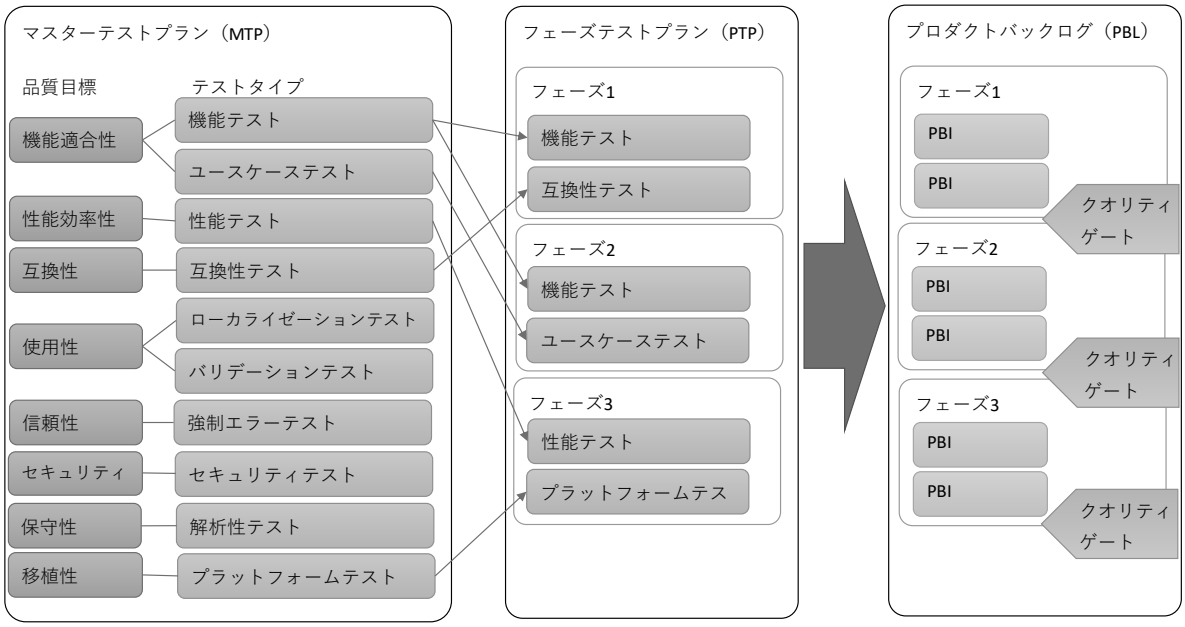
2.2 戦略的テスト計画の作成

プロジェクト開始時に，JIS X 25010:2013 のシステム/ソフトウェア製品品質の品質特性[7]を利用し，マスターテストプラン（MTP）を作成する．MTP では，品質特性の副特性毎にプロダクトの品質目標を定義する.

次に，フェーズテストプラン（PTP）で各フェーズにて達成すべき品質目標を細分化し，品質目標を実現するためのテストタイプを定義する．これにより，品質要求として定義さ

れた品質目標をフェーズ毎に段階を経て実現するための戦略的なテスト計画ができる．

開発プロセスがスクラムの場合，プロダクトバックログ（PBL）には優先度順でプロダクトバックログアイテム（PBI）が並べられている．PTP の作成において PBI の内容を参照するが，優先順位が低い PBI の内容は粒度が荒い状態のため，プロジェクト後半のフェーズに対しては，PTP で定義する品質目標があいまいな状態になる．このような場合，PBI の内容が明確になった時点で PTP も明確化していく．

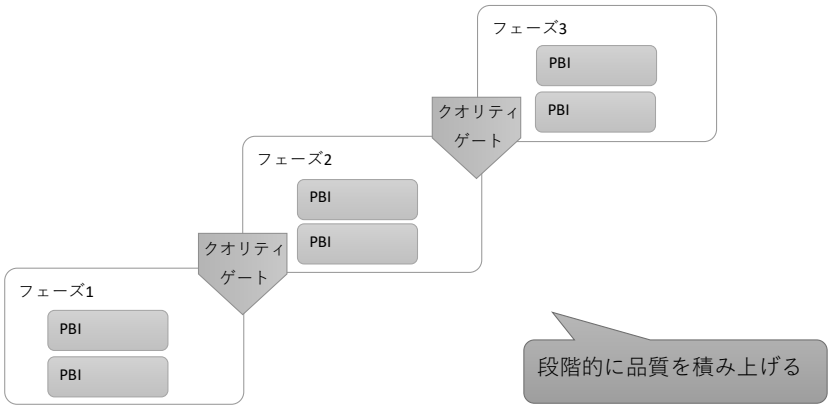


イメージ図 2　戦略的なテスト計画のイメージ

2.3　フェーズとクオリティゲートの設置による段階的品質の積み上げ

各フェーズにはクオリティゲートを設置し，終了基準（Exit Criteria）としてフェーズ内のテスト完了を目的に，フェーズ内で実施されたテストからサンプリングとしてある程度のテストケースを抽出して実行する．テスト実施後に定性分析や定量分析を行い，他に致命的な不具合がないか，類似の不具合が存在しないかを確認し，問題がなければクオリティゲート通過の判断とする．このクオリティゲートをパスすることにより，少なくとも PTP で計画した品質特性を網羅する品質目標が実現可能であることが明確になる．また各フェーズで設定した品質目標をクリアすることにより，最終的に MTP で計画した総合的な品質目標が確保されることが明確になり，プロダクトの品質が段階的に積み上がることがわかる．

また，2.2 節で述べたように，プロジェクト後半のフェーズに対する PTP はプロジェクト開始時には明確に定義できない場合がある．PTP が明確でないと，Exit Criteria の定義もあいまいになるため，PTP が明確になった時点で，Exit Criteria も明確化していく．



イメージ図 3　フェーズとクオリティゲートの設置による段階的品質の積み上げイメージ

3.　研究成果

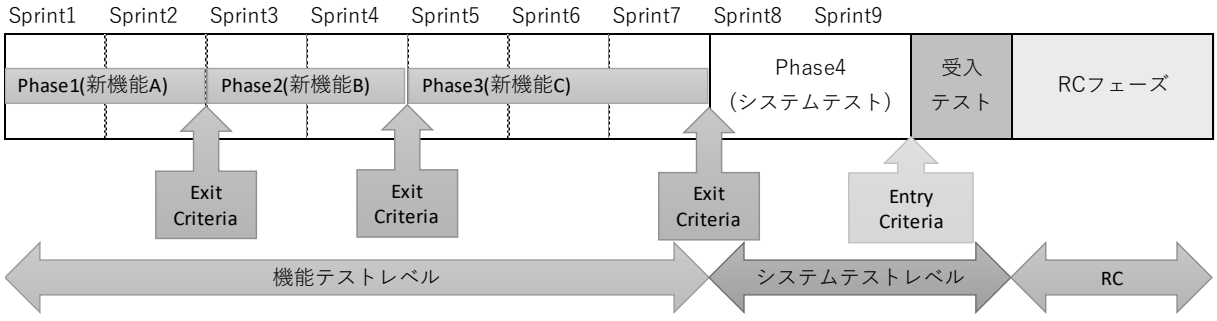
3.1　ウイングアーク１ｓｔ社製品の事例

ウイングアーク 1st 社のパッケージソフトウェア製品「SPA」の開発プロジェクトに適用した事例を紹介する．このプロジェクトは初期版をリリースした後の派生開発プロジェクトである．この製品は初期版リリース時に開発プロセスと品質保証プロセスを分けて実施しており，開発チームはウォーターフォールから初めてアジャイルに変更し，QA チームは従来のウォーターフォールで取り組んだ．結果的に既存の手法ではプロジェクト後半に品質が安定していないことが判明したため，品質保証プロセスを開始出来ず，リリース遅延のリスクが発生した．そのため，次期バージョンでは QA チームの品質保証プロセスをよりアジャイル開発に近づけ,プロジェクト開始時から品質を判断できる本手法を適用した．

プロジェクト計画時には戦略的なテスト計画として MTP を作成し， JIS X 25010:2013 のシステム/ソフトウェア製品品質の品質特性をベースとして品質目標を定義した．また，定義した品質目標を確保するためのテストタイプをマッピングすることにより，全体的なプロダクト品質が確保される仕組みを見える化した（「付録１．マスターテストプランの品質目標の定義とテストタイプのマッピング」参照）．

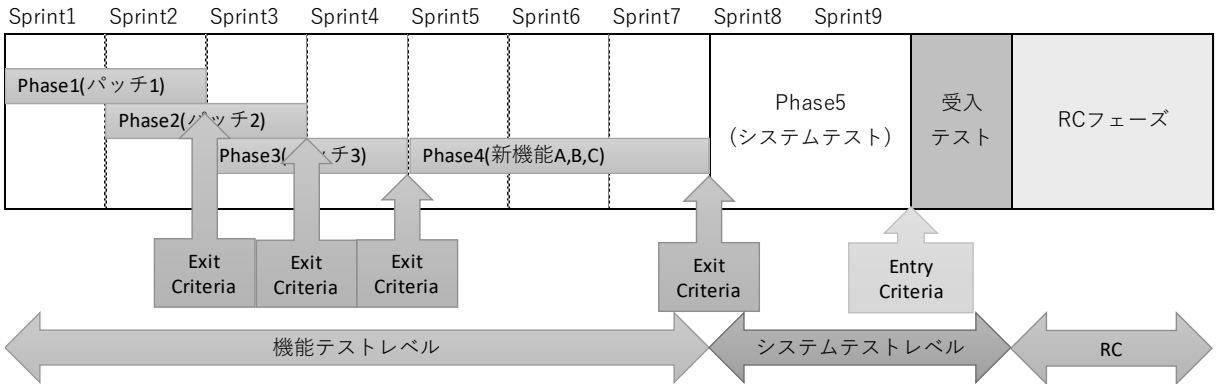
MTP 作成後は， PTP を作成し，各フェーズにて達成すべき品質とその品質を確保するためのテストタイプをマッピングする．テスト十分性については，全ての副特性に関するテストを該当するフェーズで必ず実行するように計画した．

ただし，プロジェクト開始時に全ての PTP の作成は困難であった．アジャイル開発は要求仕様がユーザーストーリーとして記述され，市場投入の価値の高いものから優先順位付けされる．そのため，プロジェクト計画時にはどの機能がいつ実装され，どのフェーズで対象となる機能がテスト可能になるかの判断ができない．以下のイメージ図 4 はプロジェクト計画時の PTP に反映されたスケジュールである．プロジェクトが予定通りに進行した場合は 3 つの新機能がフェーズ毎に実装され，リリース可能な状態になる．



イメージ図 4　プロジェクト計画時の PTP イメージ

実際に本プロジェクトでは最終的に全ての PTP が作成できたのは Sprint3 の Phase2 が完了するタイミングであった．このプロジェクトでは新機能実装といくつかの既知不具合修正の要件があり，プロジェクト前半から新機能実装のタイミングで既知不具合を修正する計画がされていた．ただし，新機能実装はある程度進んではいるものの，プロジェクトが進むにつれて振り返りや割り込みの不具合修正が多くなり，計画は常に変更されている状態になった．最終的に既存顧客から要望されている不具合対応は優先順位が高くなり，新機能実装は優先順位が低くなった．そのため Phase1～3 は不具合修正を対応してパッチリリースのタイミングでフェーズ分けし，新機能については，実装はプロジェクト前半で着手していたが Phase4 の中でテスト可能な状態にし，テストを集中的に行った．以下はそのスケジュールのイメージ図である．



イメージ図 5　PTP の最終的なスケジュールイメージ

このように PTP はプロジェクト計画時は直近のテスト計画以外はあいまいな状態であり，プロジェクトが進行するにつれて明確化していく．これはスクラムの PBL と同じで，優先順位で並んでいる PBI とほぼイコールで，プロジェクトの進行と共に PBI の優先順位の明確化とテストフェーズが明確になる状態であることが分かった．

リリースする計画とフェーズをマッピングし，フェーズをクリアする条件としてクオリティゲートを設置し，Exit Criteria として「ピックアップテスト」と称するサンプリング検証に問題がないことをクオリティゲート通過の判断とした．ピックアップテスト実施のタイミングはフェーズの最終日もしくは翌日にフェーズ内で予定されたテストが全て完了した時点で開始される．ピックアップテスト中は開発は並行して次のフェーズのタスクに着手するが，もし不合格になった場合は不合格部分を最優先 PBI として次フェーズ初期段階で対応する．また，合格するまで再ピックアップテストは実施される．もし条件付き合格になった場合は，条件部分を PBI として妥当な優先順位で対応する．PTP の最終版で定義されたフェーズと各フェーズで実施されるテストタイプと Exit Criteria は「付録 2．フェーズテストプランとクライテリア」を参考のこと．

フェーズテストの結果として，Phase 1～3 は不具合の修正に対するテストのため順調にクオリティゲートを合格し品質の積み上げに成功した．Phase4 では本来 7 スプリントで完了するはずの新機能を，最終的にプロジェクト後半の 3 スプリントで集中的にテスト可能状態にし，予定されたテストを完了させた．Phase4 の PTP では新機能に対するテストが実行されるため，テスト範囲を明確にし，1 つのテストケースで複数機能を網羅できるようなテストタイプとのマッピングを実現し効率の良いテスト消化を目指した．また，性能テストについては開発者の実装環境で単体テストとして性能に影響がないことを確認し，システムテスト内で性能テストを実施することにより効率化を目指した．

結果的にクオリティゲートでの確認時にはレビュー不足やコミュニケーション不足があり，4 回も不合格になっている．Phase4 で 4 回不合格になった遅延部分はプロジェクト

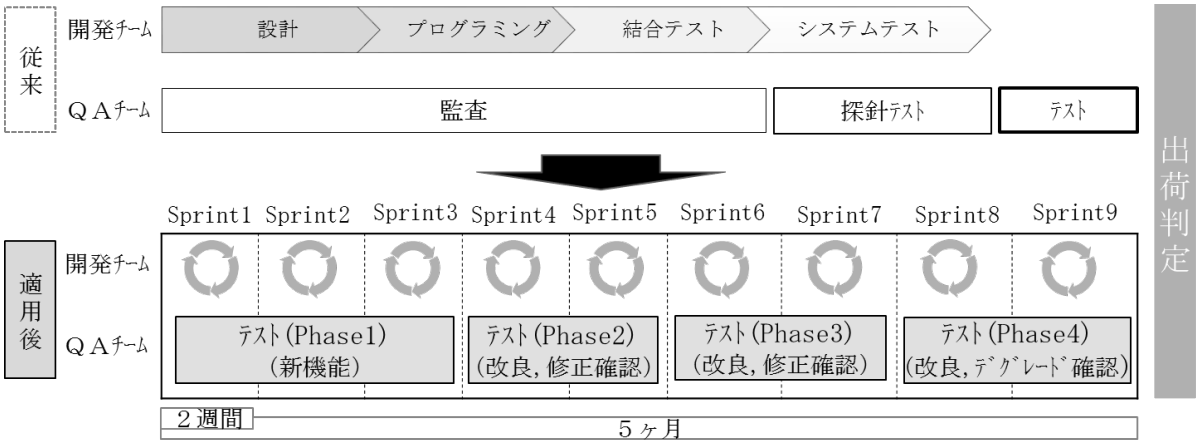
内のシステムテストフェーズでリカバリーされ，結果的にスケジュール遅延のリスクは解消された（「付録 3．テストフェーズ毎のクオリティゲート通過結果」参照）．

このように各フェーズでクオリティゲートを設置し，Exit Criteria を満たした上でゲートの通過を判断すれば次のフェーズへと進むが，不合格の場合はクオリティゲートを通過せず，Exit Criteria を満たすまで検証が行われる．ただし，Exit Criteria はあくまでも品質判断の材料として使われるため，クオリティゲート通過の判断をする際に定性分析を行い，条件が満たされていない場合でも品質的に問題ないことを QA チームが判断した際は「条件付き合格」として，条件を提示した上でクオリティゲートを通過することもある．また，役割として QA チームは品質データを提供し，クオリティゲート通過の判断をある程度行うが，最終的な判断は PO に委ねられる．

3.2　品質リスクを早期に発見・フィードバックし，予定どおり出荷できた事例

統合運用管理ソフトウェアの開発プロジェクトに適用した事例を紹介する．このプロジェクトは，社内サービス運用部門（PO）の要望を受け，Web 画面(利用者向け)の新機能追加，及び画面操作性を改良する開発である．

開発チームがウォーターフォールからアジャイルに変更したのに合わせて,QA チームは下流工程（新機能に対する探針テスト，出荷前）でテストするのではなく，動くソフトウェアに対し計画的なテストを行う本手法にて，品質を段階的に確認できるか実験した．



イメージ図 6　従来のプロセスと適用後のプロセス

QA チームは，達成すべき品質を重点にスピーディに検査するため，検査対象とする品質副特性を最大 3 つに絞ることを提案し MTP を作成した．開発チームへ MTP を説明し，Web 画面(利用者向け)の新機能追加，及び画面操作性を改良する開発目的から，(このプロダクトでは 2 つとなったが)「機能適合性」と「使用性」を検査対象にすることを合意し，フェーズごとで行う PTP と品質目標を細分化して定義した．

4 段階に分けたフェーズでテストを実施した結果，Phase1，2 でそれぞれ欠陥 1 件(メジャー)を検出した．1 件は Web 画面改善を実施（条件付き合格の判定）．もう 1 件はマニュアルどおりに動作するか確認が行われていない開発プロセスの問題であったことから，開発部門に改善を促した（不合格の判定）．開発部門は，Done の定義を見直すと共に，プログラム全体への品質対策(マニュアルとプログラムの整合確認にて 19 件の欠陥を新たに検出）を実施し，Phase3 に不具合修正が行われた．Phase3 以降新たな欠陥は検出しなかった（合格の判定）ことから，予定どおり出荷した．

この実験により，内在する品質リスクを早い段階で発見でき，また品質が段階的に積み

上がっていることが確認できたことから，有効な手法であると考える．

	評価対象スプリント	テスト項目数	インシデント・欠陥			不具合修正・改善	
			クリティカル	メジャー	マイナー	プログラム	マニュアル
Phase1	1～3	76	0	1	25	0	0
Phase2	4, 5	18	0	1	8	1	2
Phase3	6, 7	16	0	0	3	1	19
Phase4	8	152	0	0	5	0	0
合計		262	0	2	41	2	21

・**クリティカル**
基本機能が動作しない
改善が必須
・**メジャー**
一部機能が動作しない
改善が必要
・**マイナー**
改善、影響は微小
将来検討で可

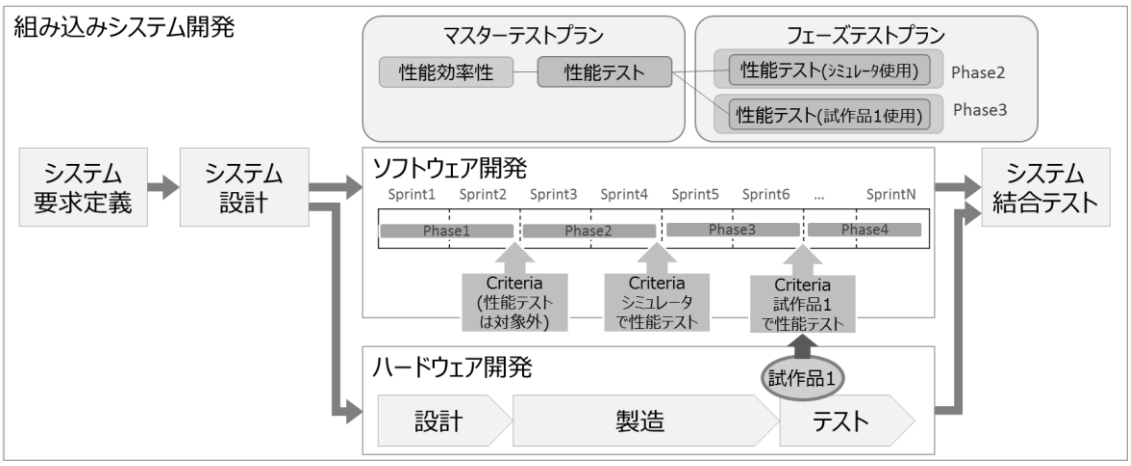
イメージ図 7　フェーズ毎のテスト結果と不具合修正数

3.3　考察

3.1 節および 3.2 節の 2 つの事例において，スプリント毎にリリースすることはできないが，フェーズ毎に戦略的なテストを実施することによって，フェーズの単位でリリース可能な品質かどうかの判断ができることが確認できた．本手法を用いることで品質の積み上げが可能となり，ステークホルダーに対して品質を説明するために有効であると考えられる．

一方で，今回適用した事例はいずれもパッケージソフトウェア開発であり，他の製品分野への適用可否や課題については未検証である．汎用性の確認は今後の課題とするが，本研究チームの一部の研究員は組込みシステム開発に関わっているため，組込みシステム開発への適用可否について机上で検討した．

組込みシステム開発では，ソフトウェアとハードウェアの開発は並行して行い，両者を組み合わせたシステム結合テストはプロジェクトの後半まで行うことができない[8]．そのため開発途中のソフトウェアのテストは，ハードウェアシミュレーターや試作ハードウェアを用いて行う必要があるが，これらはソフトウェアテストの手段の 1 つであるため，MTP の定義には違いはないと考える．例えば，品質特性の「性能効率性」において「性能テスト」を行うことに違いはなく，実際の性能テストにおいて，テストの手段として試作ハードウェア等を用いたテストを行うことになる．これは，PTP におけるテストタイプの定義に影響する．性能テストのテストタイプとして「性能テスト（シミュレーター使用）」「性能テスト（試作品 1 使用）」のように定義することが考えられる（イメージ図 8）．このように，組込みシステム開発においても本手法は適用できると考えられる．



イメージ図 8　組込みシステム開発への本手法の適用

4. 今後の課題

本研究で考案した品質保証プロセスの検証は前述したパッケージソフトウェアに限定しており，汎用性の検証はできていない．今後は組み込みやサービスなど，他の分野で本プロセスの有効性や改善点，課題などを確認したい． また，ウイングアーク１ｓｔの事例では担当者間でのコミュニケーション不足を起因とするクオリティゲートの不合格で追加テストが発生していることから，品質コストの削減についてのプロセス改善も課題としていきたい．

5. 謝辞

本研究活動において，ご支援，ご協力いただいた一般財団法人・日本科学技術連盟の方々，分科会主査，副主査，アドバイザー，研究員の活動をご承認いただいた各研究員所属企業の上司の方，すべての方々に深く御礼申し上げます．

参考文献

[1] VERSIONONE, State of Agile Survey 12th annual , https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report （2019 年 1 月 16 日現在）
[2] 独立行政法人情報処理推進機構 技術本部 ソフトウェア・エンジニアリング・センター， 非ウォーターフォール型開発の普及要因と適用領域の拡大に関する調査報告書 （非ウォーターフォール型開発の海外における普及要因編），http://www.ipa.go.jp/sec/softwareengineering/reports/20120611.html#L1（2019 年 1 月 16 日現在）
[3] SQuBOK V3 研究チーム，“アジャイル品質保証の動向“， ソフトウェア品質シンポジウム 2016，2016
[4] 永田敦，“なんたって DevQA アジャイル開発と QA の合体が改善を生む“， アジャイル冬の陣 2017，2017
[5] 伊藤潤平，“クオリティゲートの通過判断として品質特性を利用した受入テストの導入と効果“，ソフトウェア品質シンポジウム 2017，2017
[6] 吉岡克浩，水野昇幸，西康晴，“見通しのよいテストの段階的詳細化の手法 －テストの網羅性確保の提案－“，ソフトウェアテストシンポジウム 2013，2013
[7] 日本工業標準調査会，“ソフトウェア製品の品質要求及び評価（SQuaRE）-システム及びソフトウェア品質モデル”，JIS X 25010，2013
[8] 独立行政法人情報処理推進機構 ソフトウェア・エンジニアリング・センター，“組込みソフトウェア向け開発プロセスガイド ESPR ver.2.0，2006