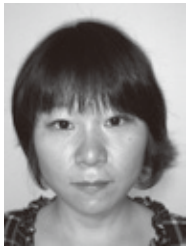
システム価値向上を目的としたScrumの試行・評価






中村　伸裕[†1,†2,†3]　服部　悦子[†2]　永田　菜生[†1]　楠本　真二[†3]

住友電気工業（株）の情報システム部では主として企業内で利用する事務処理システムを開発している．従来からＱＣＤの改善を継続しており，2011 年に CMMI レベル 5 を達成している．今回，利用部門に提供するシステム価値の向上を目的として Scrum の試行・評価を行った．その結果，設計品質の向上，開発者のモチベーション向上，開発プロセスの改善などの効果が得られた．

Evaluation of Scrum to improve value of systems

Nobuhiro Nakamura[†1,†2,†3], Etsuko Hattori[†2], Nao Nagata[†1] and Shinji Kusumoto[†3]

Abstract

Information System Department at Sumitomo Electric Industries Inc. designs and develops enterprise systems. We have been improving our software processes and achieved CMMI Level 5 in 2011. In this paper, we evaluate Scrum, one of the agile software development methods, to improve value of systems that we provide for user departments. The results show the following effectiveness of Scrum: improvement of design quality, increase of developers' motivation and improvement of development processes.

1. はじめに

日本情報システム・ユーザー協会 (JUAS) の『IT 動向調査』[1] によれば経営層のＩＴ部門への期待は，システム構築や安定稼働については高い割合で応えられているものの，ビジネスモデルやビジネスプロセスの変革については十分応えられていないことが示されている．この期待に応える一般的なアプローチは要件開発プロセスの改善であるが，その一つの手段として，システムの利用部門と密度の高いコミュニケーションをとるアジャイル型開発が考えられる．

アジャイルの基本コンセプトは 2001 年に行われた Kent Beck らによるアジャイルソフトウェア開発宣言 [2] で示されている．それから 10 年以上経過し，アジャイル型のシステム開発は欧米を中心に普及してきている [3][4]．アジャイル開発手法の中でも Scrum[5] を利用した開発は多く報告されており [3]，Scrum と XP を組み合わせて利用しているケースも多い．Scrum は中小規模のソフトウェアを 5 ～ 9 名のチームが 1 ヶ所に集まって開発を行うのが基本であるが，より大規模なソフトウェア開発やグローバルに分散した拠点での開発への適用可能性に関する報告も行われている [6]．

また，プロセス改善の視点では Scrum と CMMI（プロセス改善モデル）を効果的に組み合わせる研究 [7][8] や

【脚注】
† 1 住友電気工業株式会社　Sumitomo Electric Industries, Ltd.
† 2 住友電工情報システム株式会社
Sumitomo Electric Information Systems Co., Ltd.
† 3 大阪大学　Osaka University

CMMI Level 5 を達成したプロセスに Scrum を組み合わせることで生産性を向上させた事例が報告されている [9].

一般的に Scrum 導入の効果として (1) 事業目的により整合した機能のリリース，(2) 開発者のモチベーションの向上，(3) 短期間のリリースサイクルによる利用者の利便性向上，(4) 生産性の向上などが報告されている．

一方，住友電気工業 (株) の情報システム部門ではソフトウェア部品の再利用を進めるなどしてソフトウェアの開発生産性を改善し，プロセス改善によりソフトウェアに含まれる欠陥を低減してきた．結果として，2011 年には CMMI Level 5 を達成した．残された課題の一つが，システムの価値の向上である．今回，高い生産性と品質を実現する既存プロセスと Scrum を組み合わせることで，より価値の高いシステムが構築できるかどうかを試行・評価した．その結果，従来のウォーターフォール型の開発に比べ，Scrum はより価値が高いシステムを構築できる要素を持っていることがわかった．本論文では試行から得られた (a) 設計品質向上，(b) ソフトウェアプロセスの改善，(c) 開発者のモチベーション向上，(d) 教育効果の結果と考察を示す．

2.Scrum 概要

Scrum は竹内 弘高氏，野中 郁次郎氏が日米の製造業の設計・開発を調査した文献 [10] が起源となっており，Ken Schwaber 氏，Jeff Sutherland 氏がソフトウェア開発へ適用したものである．以下，文献 [11] を基に Scrum の概要を説明する．

Scrum は，複雑で変化の激しい問題に対応するためのフレームワークであり，可能な限り価値の高いプロダクトを生産的かつ創造的にリリースするためのものであり，役割，成果物，イベントが定義されている．

(1) 役割

ソフトウェア開発に携わる人について，次の (R1) ～ (R3) の 3 つの役割が定義されている．(R1) プロダクトオーナー：ソフトウェアの開発順序を決める権限を持ち，価値を最大化する責任を持つ要求者である．(R2) 開発チーム：ソフトウェアを開発チームであり，通常 5 ～ 9 名で構成される．(R3) スクラムマスター：開発チームに Scrum を正しく理解させ，開発チームが成果を上げるために支援や奉仕を行う．一般的なプロジェクトマネージャーとは役割が異なる．

(2) 成果物

次の (P1) ～ (P3) の 3 種類の成果物が定義されている．(P1) プロダクトバックログ：プロダクトオーナーが作成するソフトウェア要件の一覧である．要件には優先順位が示されている．(P2) スプリントバックログ：次回リリースする機能をプロダクトバックログから選択したもので開発チームが作成する．(P3) インクリメント：スプリントバックログの機能を既存の（前回リリースした）ソフトウェアに追加実装したもので，動作して，かつ，リリース可能なものである．

(3) イベント

次の (E1) ～ (E5) の 5 種類のイベントが定義されている．図 1 に Scrum の 1 サイクルを示す．(E1) スプリント計画ミーティング：2 つのパートに分かれており，Part 1 ではプロダクトオーナーが作成したプロダクトバックログをもとに何をすべきか理解する．Part 2 では今回の開発で作成すべき機能とそのための作業を計画する．(E2) スプリント：計画した機能を開発する．スプリントの期間は通常 2 週間から 1 ヵ月とされている．(E3) デイリースクラム：毎日行う 15 分以内のミーティングで進捗の評価と次に行うタスクの調整を行う．(E4) スプリントレビュー：開発チームが開発した機能のデモをプロダクトオーナーに対して行う．プロダクトオーナーはデモを確認して，完了しているかどうかの判断を行う．(E5) スプリントレトロスペクティブ：人・関係・プロセス・ツールの観点から今回のスプリントを点検し，改善計画を作成する．

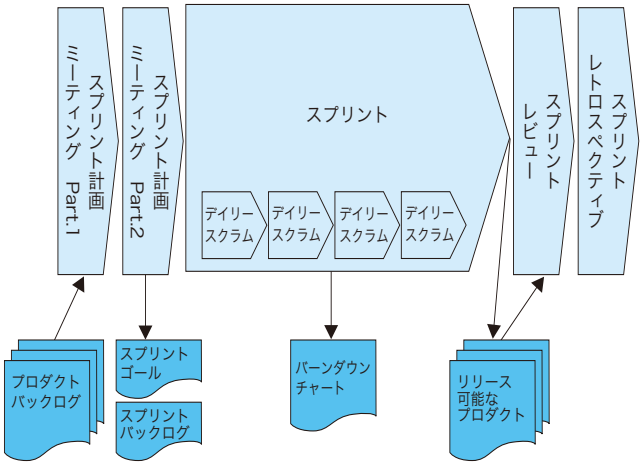


図 1 .Scrum のイベントと成果物

3.Scrum 導入の背景と狙い

住友電気工業 (株) の情報システム部門では主として自社で利用する事務処理システムを開発しており，システム開発の品質・納期・コストの改善も継続的に実施している．技術面ではデータ中心設計の採用により上流工程の品質を高め，自社開発フレームワーク (楽々 Framework II[12]) によるソフトウェア部品の再利用により高い開発生産性を実現している．また，CMMI を活用したプロセス改善も進めており，2011 年にレベル 5 を達成している．品質管理では管理図 (JIS Z 9021) を使ったプロセスの監視と制御を行っている．今後の課題の一つとして，利用部門の経営層や利

用者にとってより価値の高いシステムを提供することがあげられている．

経営者にとってのシステム価値は投資対効果などの指標で示すことができる．事務処理システムの構築では業務設計が投資対効果を決める重要な工程であるため，今回の試行とは別に改善活動を継続している．

利用者にとっての価値は業務効率が大きな要素である．当組織の事務処理システムは (a) 業務活動を記録する，(b) 担当者へ業務上の指示を伝達する，(c) 業務上の意思決定に必要な情報を提供する，(d) 業務管理に必要な情報を集計する機能を持つ．(a),(b) については，操作性や理解性が業務効率に関係する．(c),(d) については人の判断が伴うため担当者の経験や知識によって判断に必要な情報が異なることも多く，より多くの担当者が正しい意思決定を行える情報提供がシステムの価値となる．利用者のニーズとシステムが提供している機能のギャップは改善要望として現れてくるため，改善要望の少ないシステムがより価値の高いシステムといえる．

しかし，当組織では外部仕様書を利用部門と合意し，合意された外部仕様書どおりにシステムを開発することが基本的な考え方になっており，外部仕様書確定後にシステム開発に参加する開発者はより価値の高い機能を提供しようというモチベーションは低い．また，一部の外部設計担当者は利用部門と合意することに気を取られ多様な利用者の考慮が不十分であることもある．今回の試行では Scrum が設計プロセスやモチベーションの観点で価値向上の要素をもっているかどうか評価する．主な評価項目は以下のとおりである．

(a) 設計プロセスの改善効果

(b) 開発者の価値向上に対するモチベーション向上

(c) 設計プロセスの教育効果の有無

また，アジャイル開発に対する不安要素である下記の点も評価する．

(d) リリース時に含まれる欠陥の増減

4.Scrum 試行の準備

4.1. パイロットシステムの選定

パイロットシステムは失敗するリスクも考え，情報システム部門内部で利用するシステムとし，当時，開発が計画されていたタスク管理システムを対象とした．タスク管理システムはエンドユーザーからの問合せ，開発・保守部隊への技術支援，計画的な技術調査，改善活動などのタスクを管理するシステムであり，従来システム化されていなかった業務の効率化を目的としている．

4.2. 試行体制

スクラムガイドによれば開発チームは５～９名とされている．開発チームは標準的な７名とした．そのうちプロジェクトマネージャーの経験のある１名がスクラムマスターを兼任した．なお，開発チームには新人が２名含まれており，開発生産性は組織の平均値よりも低い状態であった．また，別の２名は時短勤務者であり会議開催の時間帯の制約があった．開発チームとは別に情報システム部門からプロダクトオーナー１名が参加し，さらに Scrum の評価，ツール提供などの支援を行う改善推進部門の担当者が１名参加した．試行期間は３ヵ月に設定した．

4.3. スプリント期間の設定

スプリント期間は２週間から１ヶ月とれされており，開発するソフトウェアの特性やチームの実力に合わせて設定することが求められている．プロダクトオーナーの視点ではスプリント期間は短い程，利用者により早いタイミングで機能が提供でき，また，開発する機能の軌道修正がしやすくなる．一方，開発者の視点では期間が長い程，進捗を調整する余裕ができ，スプリントゴールの達成確率が増加する．今回はスプリントをなるべく多く実施できるように２週間に設定した．

4.4. 作成する成果物作成の変更

当組織では外部仕様書，プログラム仕様書，ソースコード，統合テスト仕様書を成果物として作成している．Scrum では開発チーム全員が要求を聞き，外部仕様決定に関わるため，開発すべきプログラムの機能を理解することになる．そのため，機能の詳細を記述したプログラム仕様書の必要性は低く，作成しないこととした．外部仕様書は保守のために従来通り残すことにしたが，Scrum では開発中でも仕様変更が高い頻度で発生するため，プログラム開発後に，最新の状態に修正することとした．また，統合テスト仕様書もプログラム開発後に作成することとした．

4.5. ミドルウェア・OS

ミドルウェア・OS は，組織の標準に従っている．具体的には，自社製フレームワーク (楽々 Framework II[12]), Java, Tomcat, PostgreSQL，OS は Linux を使用した．

4.6. 教育

Scrum の経験者は社内にいなかったが，文献 [11] やインターネット上の資料を参考にしながら，半日の説明会を実施した．その後，全員が文献 [11] を読み，具体的な進め方を開発チームメンバーで議論した．社外の教育は受けなかった．

4.7. オフィス

話がしやすいよう隣向かいの８席を確保し，一ヵ所で開発した．席の横と後ろに金属製のキャビネットがあり，

チャートなどを張ることができる環境であった．

4.8. ツール

Scrum では進捗管理に残作業の工数の推移を示すバーンダウンチャートを使用するのが一般的である．

図２にバーンダウンチャートの例を示す．縦軸にタスクの残工数，横軸に日付をとり，計画と実績をプロットしていく．実績が０になることを“着地”と呼ぶ．最終日に残工数が０になっているのが望ましい．今回の開発では残工数がわかるスプレッドシートを作成した．バーンダウンチャート自体はスプレッドシートが示す数値をもとに手書きでプロットし，キャビネットに貼り付けた．

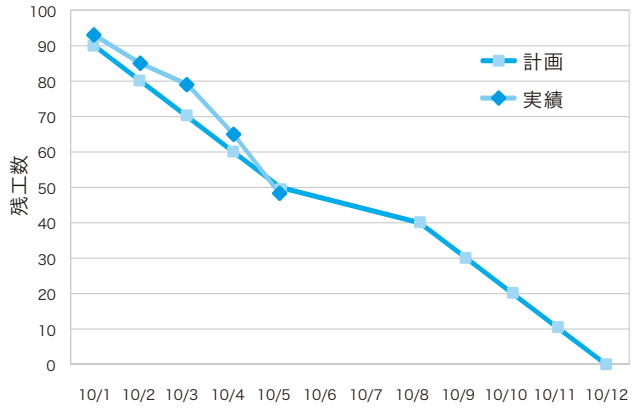


図２．バーンダウンチャートの例

5.Scrum 試行

5.1. プロダクトバックログの作成

プロダクトバックログはプロダクトオーナーが作成する．文書作成はウォーターフォール型の開発で標準として採用されている自社開発の文書管理ツールを使用した．本文書管理ツールは文書の属性が定義でき，文書一覧で優先順位，開発の状況 (完了 / 未着手等) が把握できるようになっている．プロダクトバックログは，業務上の施策ごとに１つの文書として作成した．主な記述内容は，目的，業務フロー図，各業務での利用方法，想定される機能の概要，影響が予想される機能である．画面イメージや詳細な機能の記述はなく，開発チームが検討する．開発中に出てくる利用者の要望にあわせてプロダクトバックログに新たな文書を追加したり，優先順位の見直しをしたりする．初回のスプリントは Scrum の経験者がいないこともあり，重要度が高く実装が簡単なものの優先順位を最上位に設定した．

5.2. スプリント計画ミーティング Part.1

スプリント計画ミーティング Part.1 の目的は開発チームが何を作るか正しく理解することであり，プロダクトオーナーが次回スプリントで開発してほしい機能を中心にプロダクトバックログの説明を行った．優先順位に変更があった場合は，変更内容の説明も行う．

5.3. スプリント計画ミーティング Part.2

Part2 では，今回のスプリントで実装する機能を仮決めし，タスクを分解し，計画を策定する．最終的には２週間の期間で開発できる作業量に合わせて実装する機能を確定し，プロダクトオーナーに連絡する．スクラムガイドによれば Part2 のタイムボックスは２時間であるが，実際にはこの作業の前に２～３日必要であった．作業量の見積もりを行うためにはプロダクトバックログに示されている要求を外部仕様（画面イメージを含む）に変換する必要があり，この作業に時間がかかっていた．Scrum では既存システムの変更が主として想定されている [14] ため，システム化の範囲を拡張する新機能が中心となるシステム開発では乖離が発生するものと考えられる．スプリントバックログの計画の粒度はウォーターフォール型のプログラム開発とほぼ同じ粒度であった．

5.4. スプリント（開発）

プログラム開発に関するプロセスで計画的に変更したものはプログラム仕様書作成の廃止であったが，ソースコードのチームメンバーによるコードレビュー，単体テストは従来どおり実施した．Scrum の実践で変化した点は，単体テストの際にも外部仕様の改善の相談が行われていることである．仕様どおり動作するかといった検証 (Verification) だけではなく，最終利用者にとって適した仕様になっているかといった妥当性確認 (Validation) の観点が含まれている点が大きな改善点であった．

5.5. 品質管理

通常，品質管理は管理図を使用してプロセスの異常を検出する方法を採用しており，Scrum でも同様の方法を利用しようとしていた．しかし，Scrum では要件単位の開発になるため，別の目的で新規開発した機能に今回のスプリントで機能追加するケースも多い．１つのプログラムが複数のスプリントで段階的に機能追加されることになる．そのため従来使用していたソースコードのライン数に対する欠陥数の指標が適切ではなく，プロダクトバックログ毎の開発工数に対する欠陥数を新たな欠陥密度の指標として管理図を使用することにした．過去の開発実績から工数ベースの欠陥密度の分布（プロセス実績ベースライン）を作成すると従来のライン数ベースの欠陥密度と同じように管理図による管理ができることがわかった．

図３に工数ベースの u 管理図の例を示す．縦軸は欠陥数 / 開発工数で計算される欠陥密度であり，横軸は開発順にプロダクトバックログの番号を示している．プロットした点が規

模によって変化する階段状の管理限界を超えた場合，対策を検討する必要がある．

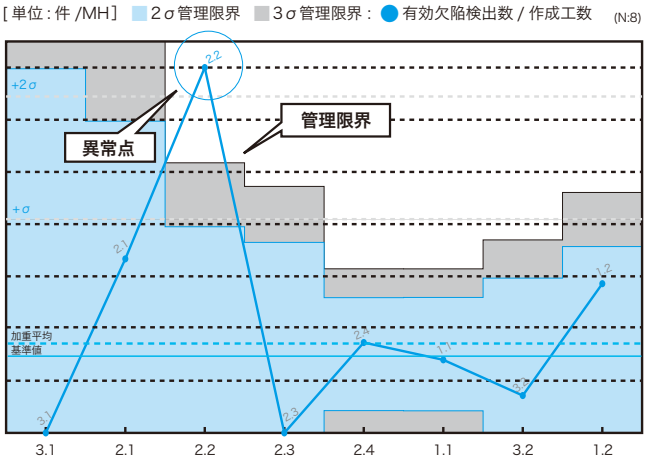


図３．工数ベースの u 管理図の例

5.6. デイリースクラム

(1) 開発状況の共有

今回のチームは勤務時間の制約があったため，午後１時からデイリースクラムを行った．より正確に実績を把握するためにデイリースクラムの直前にバーンダウンチャートを作成した．タスク計画は１日単位（その日の終業時刻）で立案したが，実績集計は昼間に行ったため半日分の差異が発生する．バーンダウンチャートの実績のプロットを半日分左にプロットすることで，計画と実績の差をより直感的にわかるようにし，開発チーム全員で進捗状況の認識を共有した．

(2) タスクのコントロール

Scrum 開始当初は計画見直しの観点が弱く，効果的に運用できなかった．しかし，第３スプリントからはプロジェクト計画から遅れているタスクを抽出して印刷し，担当者に作業状況や問題点を確認することにした．作業遅れの状況がより明確にわかり，バーンダウンチャートが着地できるよう，作業の進んでいる開発者が遅れている開発者のタスクを引き取ったり，支援したりして自発的に協力できるようになった．

5.7. スプリントレビュー

スプリントレビューでは，作成した機能のデモを行い，プロダクトオーナーがゴール達成の判断を行った．今回の試行では，関係部署の３人の課長に対するプロジェクトの状況報告の場としても利用し，欠陥の発生状況や残予算の状況などを共有した．参加者の都合により次回スプリントレビューの日程が決まった為，実質的にスプリントレビューの日付がスプリントの期間を決定することになった．

5.8. 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は振り返りの場であり，継続すべき良かった点 (Keep)，問題 (Problem)，対策 (Try) を開発チーム全員で抽出する KPT と呼ばれる手法 [15] を使って実施した．開発チームの関心事の１つは最終日にバーンダウンチャートが着地することである．着地できなかった場合は対策を検討した．問題意識の共有とプロセス改善の目的の共有ができ，Scrum を効率的に実施するために必要なプロセスであることがわかった．以下，改善例を示す．

(1) 計画漏れタスクの削減

１回目のスプリントでは計画漏れのタスクが多く発生し，バーンダウンチャートがなかなか下がらないという現象が発生した．計画した作業は予定どおり消化しているものの，必要なタスクの計画漏れが明らかになり，残作業の合計時間が計画どおり減らなかった．２回目のスプリントではこれらのタスクを計画に盛り込むことで計画漏れタスクは大幅に減少した．

(2) 仕様の検討時間

２回目のスプリントでタスクの計画漏れは大幅に減少したものの，納期までにすべてのタスクを完了させることが出来なかった．原因を分析した結果，システム価値を高めるための仕様改善の相談時間に全体の約２割の工数が使用されていることが判明した．この時間は設計品質を向上させるためのもので削減すべきものではない．アドホックに行われるため計画が難しく，持ち時間の８割でタスクの立案をすることにした．その結果，３回目のスプリントで初めて期間内にすべてのタスクを完了させることができた．

5.9. ヒアリング調査

Scrum による開発を評価のために開発者全員に対して個別にヒアリング調査を行った．調査内容はウォーターフォール型開発との比較に関する６項目の質問と自由な感想である．ヒアリングの所要時間は 15 分程度であった．

6.Scrum の評価と考察

6.1. 試行結果

表１に今回開発したシステムの改善要望とほぼ同一メンバーで前回開発したシステムの改善要望を示す．今回の規模は前回の規模の 1.1 倍であった．改善要望の合計件数は 88 件から 14 件に減少している．改善要望を操作性，理解性，機能性の観点で分類した．操作性はより少ない操作で業務を完了させる為の要望である．理解性は，操作方法や表示されているデータ，メッセージなどの意味が正確に伝わらない点の改善要望である．機能性は必要な機能がない，必要なデータが表示されていない問題に対する改善要望で

ある．すべてが Scrum の効果とは言い切れないが，理解性，機能性に関する改善効果があると考えられる．なお，Scrum 以外の要因としては開発者が時間とともにシステムの利用実態をより詳細に理解する，開発技術のスキル向上により使い勝手のよいインターフェースが実装できるようになる，といったことが考えられる．

表１．改善要望の件数

	合計	操作性	理解性	機能性
前回	88	27	27	34
今回	14	8	4	2
削減率	84%	70%	85%	94%

6.2. 価値の最大化

アジャイル宣言では最初に顧客価値を最優先することが示されている．ここでは，今回の試行でどのように価値が向上できたのか考察する．

(1) スプリントの効果

従来の開発では，１つの要求若しくは対象業務に対して１人のＳＥが外部仕様書を作成し，チーム内でピアレビューを実施していた．しかし，Scrum ではスプリント開始時から全開発者が参加するため，スプリントの初めに行う外部設計を全員で行うことになる（プログラマもコーディングの作業が出来ないため）．今回は，プロダクトオーナーの要求を全員で聞き，その後３～４名の２つのチームに分かれ，ホワイトボードを使って画面イメージや機能を検討した．従来，１機能に対する設計工数は１名で約 10 人時であったが，今回は３名で合計約 14 人時に増加している．さらにプログラム開発に着手した後も仕様の見直しに約５人時の工数が使われていた．合計約 19 人時，1.9 倍の工数が投入されており，価値向上の要因になっていると考えられる．なお，設計開始から設計終了までの期間は従来の 1/2 であった．

また，当組織では１つ要求に対する設計を一人で行っていたため，通常設計案は１つであった．Scrum では前述のとおり１つ要求に対する設計を３～４名のチームで行う．異なる価値観を持つ複数の技術者が設計を行う為，複数の案が出てきてどちらが利用者にとって良い案かを議論するケースが増えている．解の統合，選択が行われることでより画面に表示すべきデータの抽出，必要な機能の抽出，メッセージやデータラベルのわかりやすさ，操作性といった点で設計品質が向上していると考えられる．

しかし，スプリントごとに開発チームの能力が向上したため，前半のスプリントに比べ，後半のスプリントの方がより操作性，機能性，理解性の点で優れたユーザーインターフェースが設計・実装できている．前半のスプリントで開発した機能は後半に比べやや設計品質が劣っており，システム全体からみれば一貫性の確保が不十分であった．

(2) 段階リリースの効果

今回のシステムは従来システム化されていない業務を対象としていたため，システム企画段階ではより効果を高めるための積極的なアイデアは余り出なかった．しかし，システムを自分自身の業務に適用し，自分達の実データが画面に表示されると具体的な要求が出てくるようになった．要求を明確にできない状態でプロダクトバックログに 20 件の要求を書き出していたが，結果として 15 件が実装された．残り５件分の工数は新たな３件の要求の実現と開発した機能の改善に使われた．段階リリースにより有効性の低い要求が廃棄され，より価値のある機能に置き換わった．

6.3.Scrum によるプロセス改善効果の考察

今回の試行で特に変化があった開発プロセスを CMMI のプロセスエリアに分解して報告する．

(1) 要件開発 (Requirement Development)

従来の開発では要件開発はシステム全体の効果と開発費を見積もることが重要な課題であり，個々の要件を深掘りすることは多くなかった．しかし，今回の試行では複数人で設計を行ったことから設計案の選択の際，何のためにこの機能が必要なのか，この要求の本質は何かといった議論が行われ，潜在的な要求がより多く引き出された．

(2) 技術解 (Technical Solution)

従来の開発では，解の選択は個人の頭の中で行われており，解の選択が行われた記録が残ることが少なかった．Scrum では前述のとおり，解の選択が頻繁に行われるようになり，設計プロセスの品質が向上した．

(3) 決定分析と解決 (Decision Analysis and Resolution)

決定分析と解決は重要な決定（案の選択）を行うプロセスである．事前に評価項目とその重み，評価方法等を設定し，案の選択を行う．当組織ではこのような決定プロセスの記録があまり残っていない状況であった．今回の試行ではこのプロセスを完全に実施する機会はなかったが，案の選択プロセスが頻繁に実施されるようになったため，制度化，定着化が比較的容易に行えると考えられる．

(4) 妥当性確認 (Validation)

従来の開発では単体テスト，統合テストは仕様に対する不整合を確認する検証 (Verification) の観点のみであったが，今回の開発では単体テスト実施時も作成したプログラムが利用者にとって使い勝手がよいものかといった妥当性確認の観点が入っている．要求者の声を直接聞き，仕様検討の経緯を知っている開発者がプログラムを作成しているためあらゆるフェーズで妥当性確認ができるようになっている．

(5) プロジェクト計画 (Project Planning)

Scrum の開発チームはバーンダウンチャートの着地が 1 つの目標になっており，計画精度向上の動機づけが行われる．２週間ごとに計画作業が発生するため，成熟度の向上を加速させる効果がある．

(6) プロジェクトの監視と制御 (Project Monitoring and Control)

プロジェクト計画同様バーンダウンチャートの着地が 1 つの目標となっているため，チームの多くのメンバー計画に対する進捗の把握に関心が高く，またチーム全体で進捗の遅れを取り戻そうという意識があり，成熟度を向上させる効果がある．

(7) プロセスと成果物の品質保証 (Process and Product Quality Assurance)

当組織では通常，プロジェクト管理グループによるプロセスと成果物の評価がプロジェクトごとに月 1 回の頻度で行われている．今回プロジェクトは試行ということもありこの評価は行わなかった．今後実施する場合，以下の点が課題となる．(a) スプリントの期間が 2 週間と短いため評価のタイミングが難しい．(b) レトロスペクティブで毎回プロセスが改善されるため，定められた手順に従ってシステム開発が行われているかの評価が難しくなる．

6.4. モチベーション

(1) ヒアリング調査結果

Scrum を実践した 7 名の開発者に対して行ったヒアリング調査の結果を図 4 に示す．６つの質問に 5 段階で答えてもらい，平均値をプロットしている．いずれの質問に対しても Scrum の方がウォーターフォール型より良い回答が得られた．また，ヒアリングでは，“ 一人で悩んでいるよりも気軽に相談できるのがよい ”，“ 従来担当できなかった外部設計に参加できたのがよい ” といった感想が得られた．

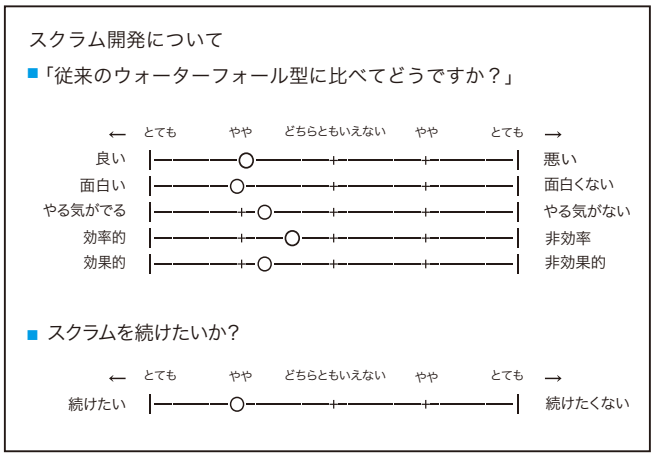


図４．Scrum 実践者へのヒアリング結果

外部からの観察では，(a) バーンダウンチャートを着地させる，(b) 約束した機能をより良い機能で提供するといったチームメンバー全員で共有した目標がチームワークを高め，モチベーションアップに寄与しているように感じられた．また，ウォーターフォール型の開発では初期に進捗が遅れると，プロジェクト終了まで進捗遅れのプレッシャーが続くが，Scrum ではスプリントが終わる度に再計画となり，こういったストレスから解放される．ヒアリングでは “ 休みが取りやすかった ” といった意見があった．

(2) 楽しさに関する考察

大林伸安氏の著書 “ 仕事が楽しくなる！２５のルール ” [16] に仕事が楽しくなる以下の５つのキーワードを使って楽しさに関する考察を行う．５つのキーワードと Scrum の要素を対応させてみると以下の様になり，Scrum 自体が楽しくなる要素を含んでいることがわかる．

(a)「ありがとう」と言ってもらえる

- スプリントレビューで自分が開発したものの評価を直接聞くことができる

(b)「なぜ，なんのために」かがわかっている

- プロダクトオーナーから要求を直接聞き，理解する
- レトロスペクティブで問題を共有し，プロセスを改善する

(c)「ゴール」が見える

- 開発する機能（スプリントゴール）をチーム全員で決める
- バーンダウンチャートでゴールが見える化される

(d)「昨日より今日が」が前進している

- レトロスペクティブにより継続的に改善活動が行われる
- 今まで担当していなかった新人が外部設計に参画

(e)「おめでとう」が言い合える

- バーンダウンチャート着地の喜びを共有

6.5. 教育効果

(1) 要件開発・外部設計の能力

ウォーターフォール型の開発では若い開発者はプログラム開発フェーズからプロジェクトに参加することが多く，外部設計の場を体験する機会が少ない．一方，Scrum では全員が設計に関与する．新人も積極的に外部設計に参加していた．スプリントを繰り返すことで基本スキルが習得できると考えられる．

(2) プロジェクト管理能力

今回の試行では，スプリント計画策定時に計画の達成可能性がより強く意識され，スプリントを繰り返すことで計画能力が向上していることが確認できた．開発中は，バーンダウンチャートが着地できるかといった観点から現状を評価し，問題があればすぐに対応策が実施されている．このような環境で経験を積むことで特に中堅社員の監視・制御能力が改善

した．また，２週間のサイクルで計画，監視，制御が繰り返されることも成熟度を上げる要因になっている．

開発者からは「30 分の時間が貴重に感じ，限られた時間で何をすれば価値の最大化ができるか考えるようになった．仕事の進め方が変わった．」という意見もあり，品質・納期・コストに対するより高い意識がうかがえる．

(3) プロセス改善能力

試行の結果，短い開発期間で開発計画どおりシステム開発するためには単に開発生産性の平均値を上げるだけではなく，開発プロセスを安定させ，ばらつきを小さくする必要があることがわかった．例えば２ヵ月の開発期間で計画どおりシステム開発できるチームであっても２週間の単位に分解すれば生産性が平均よりも高い期間と低い期間が存在する．この差が大きければ，スプリント計画どおり開発できるスプリントとそうでないスプリントが発生することになる．Scrum でスプリント期間終了までに計画した全作業を完了させる努力は，プロセスのばらつきを少なくし，安定化させる能力を身に付けさせる効果がある．

6.6. 品質（リリース時に含まれる欠陥）の評価

Scrum で開発した機能の欠陥でリリース以降に発生した欠陥は 12 件であった．ただし，第 4 スプリントで欠陥が 6 件発生しており，その他のスプリントでは 0 件または 1 件であった．第 4 スプリントは通常の開発で利用しない特殊な外部システムとの連携があり，インタフェース上の問題が発生した．第 4 スプリントを含めた欠陥密度は組織の基準値の 115% であった．第 4 スプリントの特殊要因を除いた 7 件では 67% であった．後者の方が今後の開発を予想する値として適切であると考えられる．

欠陥密度が従来より 33% 低くなった要因は，開発者が設計段階から参加していることにあると考えられる．プログラムの欠陥は，(a) 作成すべきプログラムの仕様を誤って理解する，(b) 理解した仕様を誤ってコーディングする，の２つの要因に分類することができる．前者は本人が実施する単体テストで見つけることができないため，次工程に流出する可能性が高い．しかし，Scrum で要件を聞くところから開発者が参加しているため，このミスを起こす可能性が低くなると考えられる．

7. まとめ

今回の試行だけで Scrum の効果を評価することは危険であるが，我々の組織では CMMI レベル５に適合するウォーターフォール型の開発プロセスに比べ，より価値の高いソフトウェアを提供できる可能性が高く，開発者のモチベーションも高くなることがわかった．さらに，従来のプロセス資産と Scrum を組み合わせることで，要件開発，外部設計，妥当性確認，決定分析と解決，プロジェクト計画，プロジェクトの監視と制御のプロセスが改善できることがわかった．

なお，今回の試行では自社開発のフレームワーク (楽々 Framework II) を使用し，既存部品の組み合わせでプログラムを開発しているため，IPA/SEC が公開している開発生産性 [13] に比べ２倍以上の開発生産性を実現している．そのため，２週間のスプリントでの開発が可能になっている．一般的な開発では３～４週間のスプリントが適切と思われるが，今回の評価に対する影響は少ないと考えられる．

今後の課題は外部設計の期間の確保である．２週間のスプリントでは外部設計に３日間程しか割り当てることが出来ない．設計では通勤途中に良いアイデアが出てくることもあり，投入工数が同じでも期間が長い方がよい設計ができる可能性が高い．より設計品質を上げるためには１つ前のスプリントで次のスプリントで機能を考え始めるなどの工夫が必要である．

アジャイル型開発への関心は近年非常に高まっており，今後導入する企業が増加すると予想される．我々の試行と評価が少しでも役に立てば幸いである．

【参考文献】

[1] 日本情報システム・ユーザー協会 , “2010 年度版企業 IT 動向調査 2011”, JUAS, 2011.

[2] Kent Beck , “Manifesto for Agile Software Development”, http://www.agilemanifesto.org/, 2001, 参照 2012-10-01.

[3] Version One, “2011 State of Agile Development Survey Results”, http://www.versionone.com/pdf/2011_State_of_Agile_Development_Survey_Results.pdf, 参照 2012-10-01.

[4] IPA/SEC, “ 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 ”, http://www.ipa.go.jp/sec/softwareengineering/reports/20120611.html, 参照 2013-09-24.

[5] Ken Schwaber, Mike Beedle, "Agile Software Development with Scrum", Prentice Hall, 2001, (和訳 " アジャイルソフトウェア開発スクラム ").

[6] D. Caivano et al., “Scrum Practices in Global Software Development: A Research Framework “, PROFES 2011, LNCS 6759, pp.88-102, 2011.

[7] J.Diaz, J.Garbajosa, and J.A.Calvo-Manzano, “Mapping CMMI Level 2 to Scrum Practices: An Experience Report”, Springer Berlin Heidelberg, vol. 42, no. 42, pp. 93-104 , 2009.

[8] C.R.Jakobsen and J.Sutherland, “Scrum and CMMI Going from Good to Great”, in 2009 Agile Conference, pp. 333-337, 2009.

[9] Sutherland, J., C. Jacobson, et al. “Scrum and CMMI Level5: A Magic Potion for Code Warriors! “, Agile 2007, Washington, D.C., IEEE., 2007.

[10] H.Takeuchi and I.Nonaka, "The New New Product Development Game", Harvard Business Review, January-February, 1986.

[11] Ken Schwaber and Jeff Sutherland, “ スクラムガイド - スクラム完全ガイド : ゲームのルール ”,http://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum%20Guide%20-%20JA.pdf, 2011, 参照 2012-08-16.

[12] 住友電工情報システム（株）,“ 楽々 Framework II 製品紹介 “, http://www.sei-info.co.jp/products/products_fw_top.html, 参照 2012-10-20.

[13] IPA/SEC, “ ソフトウェア開発データ白書 2012-2013”, IPA/SEC, 2012.

[14] Ken Schwaber, “SCRUM Development Process,” http://www.jeffsutherland.org/oopsla/schwapub.pdf, p.3, 参照 2012-08-29.

[15] Alistair Cockburn, "Agile Software Development", Addison-Wesley Professional,2001,(和訳 “ アジャイルソフトウェア開発 ”,P.256)

[16] 大林伸安 , “ 仕事が楽しくなる！２５のルール ”, ダイヤモンド社 , 2009