

SPI Japan 2013 発表概要集

- ・セッション順番＞発表順番に、掲載されています。
- ・目次から、各発表の概要にジャンプすることができます。

目次

1A1「Software Product Line の実践-実装用ソフトウェア部品の開発と全社展開」中村伸裕(住友電気工業).....	2
1A2「Software Product Line の実践-プログラム開発の効率化を目指した設計資産の構築」川口晃史(住友電気工業).....	7
1A3「Software Product Line の実践-テスト資産の構築」服部悦子(住友電気情報システム).....	11
1A4「テスト自動化を現場に普及するための組織的な取り組み」吉田 麻紀(インテック).....	14
1B1「残業のいらぬソフト開発への第一歩」加藤貴裕(たのしいソフト開発研究所).....	18
1B2「テスト設計プロセス可視化の取り組み」伊藤由起子(パナソニック).....	23
1B3「トヨタ開発方式を利用したソフト開発の継続改善活動」比嘉定彦(アドバンテスト).....	26
1B4「CMMI ベースの1日型簡易診断の実施」伊藤裕子(東芝).....	33
1C1「アプリケーション運用・保守プロセス定着への取り組み」相澤武(インテック).....	38
1C2「全社活動によるプロセスの再改善・大改善」和良品文之丞(キヤノンソフトウェア).....	42
1C3「派生開発手法導入に見る現場起点のプロセス組織展開事例」赤松康至(オムロン).....	48
1C4「実機レス情報交換会の活動の紹介」平原嘉幸(東芝テック).....	51
2A1「パッケージ開発プロセス改善による品質向上と生産性向上」松浦豪一(富士通マーケティング).....	52
2A2「産学連携によるコードレビュー改善事例」佐藤美和(三菱電機).....	58
2A3「アジャイル開発における品質保証部門によるシステムテストのアプローチ」永田敦(ソニー).....	64
2A4「現場ですぐできる定量データ分析～予測モデルのゆるい作り方～」矢部智(NTT データ).....	70
2B1「業務の中で自然に学ぶ仕掛け「プロセスの自己履行検証とピア・レビュー」」竹下千晶(デンソークリエイティブ).....	74
2B2「レベル3達成組織における自律改善の推進」藤縄幾子(パナソニック).....	79
2B3「SaPID 実践事例より～改善推進役がやるべきこと／やってはいけないこと」安達賢二(HBA).....	83
2B4「「真のプロセス定着」への取り組み」宮川研二(ダイキン情報システム).....	87
2C1「チケット駆動開発によるプロセス改善」阪井誠(SRA).....	94
2C2「チケットを利用したタスク管理の試み」古石ゆみ(SRA).....	98
2C3「Android 端末中国ODM活用ノウハウ」林潔(パナソニック).....	102
2C4「文化背景の異なる拠点間におけるプロセス改善の取り組み」田村朱麗(東芝).....	107

1A1「Software Product Line の実践-実装用ソフトウェア部品の開発と全社展開」中村伸裕(住友電気工業)

<タイトル>

Software Product Line の実践

<サブタイトル>

実装用ソフトウェア部品の開発と全社展開

<発表者>

氏名（ふりがな）： 中村 伸裕

所属： 住友電気工業株式会社 情報システム部 情報技術部

<共同執筆者>

氏名（ふりがな）：

所属：

・ 要点

ソフトウェア資産の再利用はソフトウェアのQCDを同時に改善できる手段として昔から期待されており、ソフトウェア・プロダクト・ラインはソフトウェア資産の再利用をうまく進める為の方法である。組み込み系での事例は多く報告されているものの、基幹システムの報告は少ない。住友電工では1999年よりソフトウェア資産の開発を進め、全社レベルで再利用を進めている。本報告では、

- (a) 業務システム用の再利用性の高いソフトウェア資産は構築できるか、
 - (b) 全社レベルでソフトウェア資産の再利用は実施できるのか、
 - (c) 再利用率はどの程度まで上げられるのか、
 - (d) 部品化の制約により利用者の満足度が悪化しないか、
- について報告する。

・ キーワード

Software Product Line, 再利用部品,

・ 想定する聴衆

Software Product Line の導入を検討している開発者
ソフトウェアの再利用を進めたい開発者

・ 適用状況

- ☒ 多用されている段階、☐ 適用できる段階あるいは初めて適用する段階
- ☐ 適用するにはさらに検討を必要とする、☐ 着想の段階
- ☐ その他（ ）

・ 適用可能性に関する制限

- ☐ 汎用性がある、
- ☒ 類似プロジェクトにも適用可：具体的な類似点（業務システム）
- ☐ 自プロジェクトのみ

・発表内容

(1) 背景

住友電工では継続的にシステム開発のQCD改善に取り組んでおり、1995年頃からオブジェクト指向言語がソフトウェア部品の開発に役立つと考えていた。オブジェクト指向言語であるJavaは当初Appletとして普及したが、1999年頃からサーバー・サイドでの利用が始まり、手軽にオブジェクト指向言語が利用できるようになった。この技術革新を背景に当組織ではソフトウェア部品の整備を開始し、組立型のシステム開発をめざす取り組みを開始した。この取り組みがSoftware Product Lineの考え方とよくマッチすることからSoftware Product Lineの適用事例として報告する。

(2) 改善前の状態

Gold Fusionというスクリプト言語を利用してWeb型の業務システムを開発しており、ソフトウェアの部品化はほとんど進んでいなかった。
開発生産性も横ばいの状況が続いていた。

(3) 改善前の状態をもたらした原因（因果関係）

オブジェクト指向言語の導入コストが高く、また、習得も難しかった。
（環境の問題）

(4) 変更内容・対応策

オブジェクト指向言語の特徴を活かしたソフトウェア部品を開発し、社内の全プロジェクトに展開することで、開発コストの低減を図る。その際、部品開発の独立したチームを組織し、複数のアプリケーション開発チームと協業しながら全社的に再利用を進める。

(5) 変更や対応策の実施内容

・業務システムのソフトウェア部品抽出

当組織ではオンライン処理の割合がバッチ処理に比べ非常に高く、画面処理の部品化が必要であった。DB処理の部品化に比べ、画面の部品化は難易度が高い。例えば、図1に示す2つの画面は、データ項目のラベル、桁数入力方法等が異なるため、よく似た画面であっても個別に開発する必要があり、部品化が容易ではなかった。そこでデータ項目をオブジェクト化する仕組みを考案し（図2）、データ項目の違いを意識しないで動作する画面の部品化に成功した。この仕組みにより、これらの部品を組み合わせたより粒度の大きい部品も開発できるようになり、再利用率が大幅に向上した。

図 1. Web システムの画面例

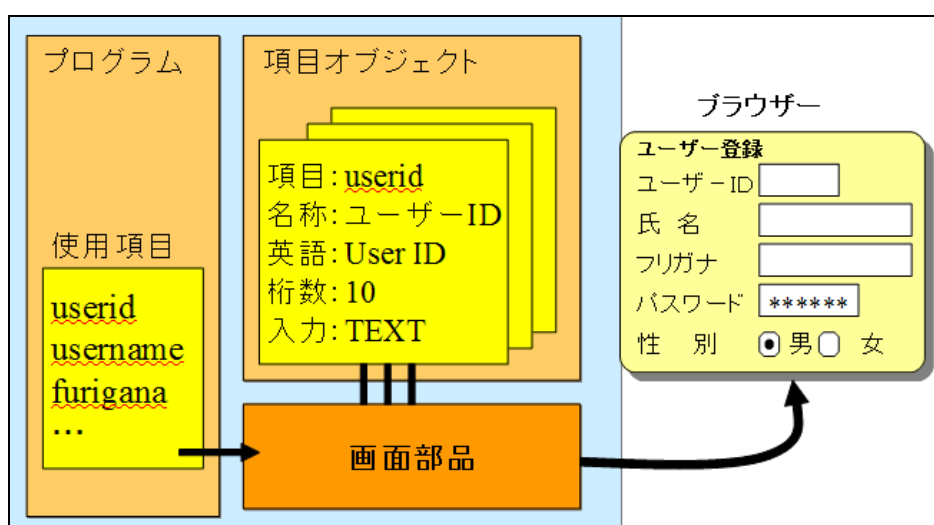


図 2. 項目オブジェクトのイメージ

・ソフトウェア資産の展開

一般的にソフトウェア部品の再利用では他人が作った部品は使用したくないという心理面での阻害要因が知られている。この問題を回避するため企画段階で開発チームのリーダーを巻き込み、部品化の検討を進めた。また、開発者がすぐに部品が使えるように演習付きの 3 日間の教育コースを定期開催し、開発プロジェクトのソフトウェア部品再利用の負担を軽減した。

このような取り組みの結果、ソフトウェア部品は社内の全開発プロジェクトで活用されるようになった。

(6) 変更・改善後の状態と効果

(a) 業務システム用の再利用性の高いソフトウェア資産は構築できるか

項目オブジェクトの考案により、画面部品やメインルーチンの部品化に成功し、業務システムのオンライン処理の多くの機能を部品化することができた。

- (b) 全社レベルでソフトウェア資産の再利用は実施できるのか

1999 年の開発されたソフトウェア資産は 2003 年にメジャーバージョンアップし 2013 年までの約 10 年間で 300 以上の社内全プロジェクトで再利用されており、全社的なソフトウェア資産の活用が実現できている。

- (c) 再利用率はどの程度まで上げられるのか

ソフトウェア資産を活用したプロジェクトで開発されたソースコードのライン数の分布を図 3 に示す。中央値 77 ステップ、平均値 164 ステップであった。比較対象として、JUAS ソフトウェア・メトリクス調査のデータから業務システムの 1 機能当りのライン数を算出すると 2084 行であった。平均値で 92% のソースコード量の削減が実現されている。再利用率 92% と考えることができる。

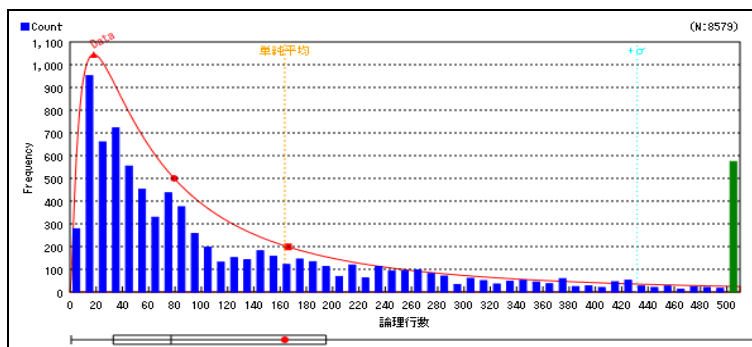


図 3. 開発したソースコードのライン数の分布

- (d) 部品化の制約により利用者の満足度が悪化しないか

ソフトウェアの部品化により部品の機能を拡張することで、各開発プロジェクトの開発コストを上げることなく、使い勝手のよい画面を利用者に提供することができた。図 4 に示す利用者の操作性に関するアンケート結果では“良い”、“非常に良い”という回答が 60% であり、利用者視点でもソフトウェア部品のメリットが確認できた。

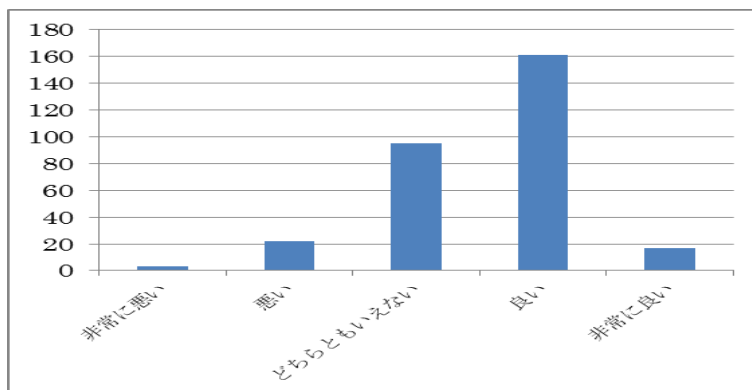


図 4. 利用者の操作性に関するアンケート結果

(7) 改善活動の妥当性確認

ドメイン開発とアプリケーション開発を分離し、協業しながらソフトウェア資産の再利用を進めるというソフトウェア・プロダクト・ラインの考え方は有効であり、効果があることがわかった。

- ・ 開發生産性の向上

今回の取り組みの目的は開發生産性の向上である。

再利用の目的である生産性の向上を評価するため、SEC データ白書のデータを使ってベンチマークを行った。当組織では 1500～5000FP のシステム開発が多いが、データ白書のデータの中央値の 3～5 倍の生産性を示している。

- ・ 今後の課題

今回の取り組みは、Software Product Line の実装の部分の取り組みに相当する。要求開発、設計、試験の領域でもドメイン資産を構築し、活用することでさらにコスト削減ができると考えている。

1A2「Software Product Line の実践-プログラム開発の効率化を目指した設計資産の構築」川口晃史
(住友電気工業)

<タイトル>

Software Product Line の実践

<サブタイトル>

プログラム開発の効率化を目指した設計資産の構築

<発表者>

氏名（ふりがな）： 川口 晃史（かわぐち あきふみ）

所属： 住友電気工業株式会社 情報システム部 情報技術部 システム技術グループ

<共同執筆者>

氏名（ふりがな）： 中村 伸裕（なかむら のぶひろ）

所属： 住友電気工業株式会社 情報システム部 情報技術部

・ 要点

住友電工では自社開発のフレームワークを用いて業務システムを開発しており、Software Product Line (SPL)のソフトウェアを部品化して再利用するという考え方を導入することで開発効率を上げることに成功していたが、導入の対象が開発工程に限定されていた。本発表では設計工程にも SPL を導入し、設計資産を提供することでプログラム開発効率を高める取り組みについて報告する。

・ キーワード

Software Product Line, 設計, 実装, 再利用,

・ 想定する聴衆

Software Product Line の導入を検討している開発者
ソフトウェア資産の再利用を進めたい開発者

・ 適用状況

☐多用されている段階、☒適用できる段階あるいは初めて適用する段階
☐適用するにはさらに検討を必要とする、☐着想の段階
☐その他（ ）

・ 適用可能性に関する制限

☐汎用性がある、
☒類似プロジェクトにも適用可：具体的な類似点（業務システム）
☐自プロジェクトのみ

・発表内容

(1) 背景

住友電工ではオブジェクト指向言語の Java を利用した自社開発のフレームワークを用いて業務システムを開発している。このフレームワークは業務システムでの画面や入力項目等の部品を持っており、部品を組み立てる形式で開発を行う。ソフトウェア資産の再利用を進めるという Software Product Line (SPL) の考え方を適用して構築されたもので、開発効率を上げることに成功していたが、SPL の適用範囲は実装工程に限られていた。そこで設計工程にも対象を広げ、設計資産を有効活用することで開発効率をさらに改善するべく取り組みを開始した。

(2) 改善前の状態

設計工程では画面レイアウトや遷移、DB への入出力等を記述した仕様書を Excel で作成し、開発工程ではソフトウェア部品の組み立てを定義ファイルで行い、業務ロジックを Java 言語で記述していた。部品組み立て定義ファイルの作成では仕様書と同一の内容を変換して記述しており、業務ロジックの作成では仕様確認や実装方法の調査などの付加価値を生まない作業が発生していた。

(3) 改善前の状態をもたらした原因（因果関係）

設計工程に対して Software Product Line を導入する取り組みができていなかった。

(4) 変更内容・対応策

Excel に記述していた設計情報をデータベース化して、リポジトリに登録する手法に変更し、設計情報を開発工程で有効活用することで開発工数の削減を図る。下記 3 点、

- (a) 設計工程の生産性を維持
- (b) 変換作業が必要な部品組み立て工程を廃止
- (c) 業務ロジック開発工数の削減

の達成を目指し、図 1 に示すようなりポジトリ設計情報で業務システムが動作する新しいフレームワークの開発に取り組んだ。

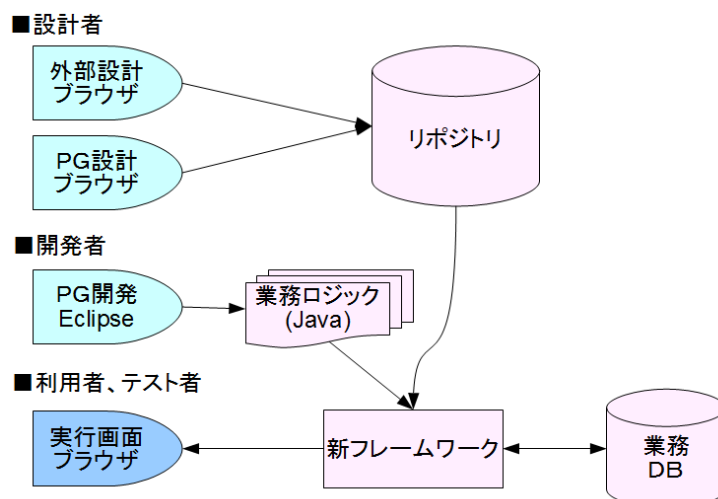


図 1 リポジトリを利用した開発イメージ

取り組みの中で浮上した下記課題について次節で記述する。

- ①外部設計段階での仕様変更対応
- ②業務ロジック開発者への仕様提供方法の改善

(5) 変更や対応策の実施内容

①外部設計での仕様変更対応

リポジトリでは異なる画面間で同一の入力項目がある場合、同一の項目部品を紐づける構造になっている（図2左）。そのため、ある画面の入力項目の仕様変更が、他の画面にまで影響が及ぶ構造になっていた。これは外部設計の初期段階では項目の整合性が取れるというメリットとなるが、外部設計の終盤においては、ユーザーレビューで合意され変更不可となった画面の項目まで影響が及んでしまうというデメリットを持っていた。

この問題に対して設計情報にリビジョン管理の考え方を導入して対応した。仕様にリビジョン番号とともに合意されているかを示す情報を持たせ、仕様変更を行う場合には合意したリビジョンの仕様は不変として、新しいリビジョンに変更後の仕様を保持する（図2右）。これによって合意済みの画面への影響なしに仕様変更の登録ができる。

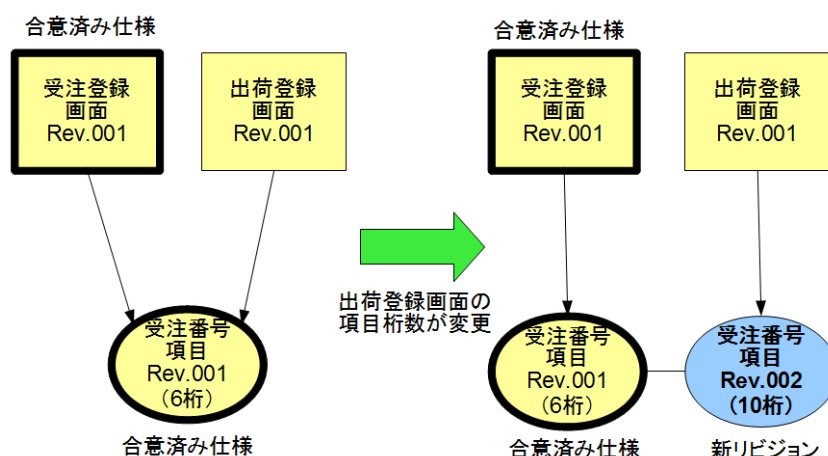


図2 合意済み仕様に対する仕様変更

仕様変更の際には、合意済み画面も含めて影響を受ける範囲の設計者にその旨を通知する機能を設けており、変更内容の確認を促す。さらにリビジョンの整合性のチェック機能によって成果物間でリビジョンの不整合を取り払う措置を取れるようにしている。

②業務ロジック開発者への仕様提供方法の改善

業務システムはその性質上データベースへの読み書きが大量に発生する。業務ロジックで記述する内容も多くがデータベース処理である。処理を記述する際には対象テーブルの項目や格納する値の仕様理解が必要であり、これまでは仕様書の中から該当部分を探し出して項目名称等を誤りなくロジックに記述しなければならず、探す手間と記述間違いによる不具合対応工数が発生する問題があった。

当組織で利用している Java 開発環境である Eclipse には、Java メソッド（他言語では関数、サブルーチンに対応）に付与した説明の表示や変数の入力補完などの補助機能（図3）がある。問題の対応としてこの機能を活用し、必要な仕様情報を必要な個所で提供できる機能を開発した。これによって各メソッドにカーソルを置くと対応する仕様が表示され、項目名称など書き間違いの発生しうる記述に対してはキーを押すことで入力補完が働き、一覧から選択入力できるようになった。

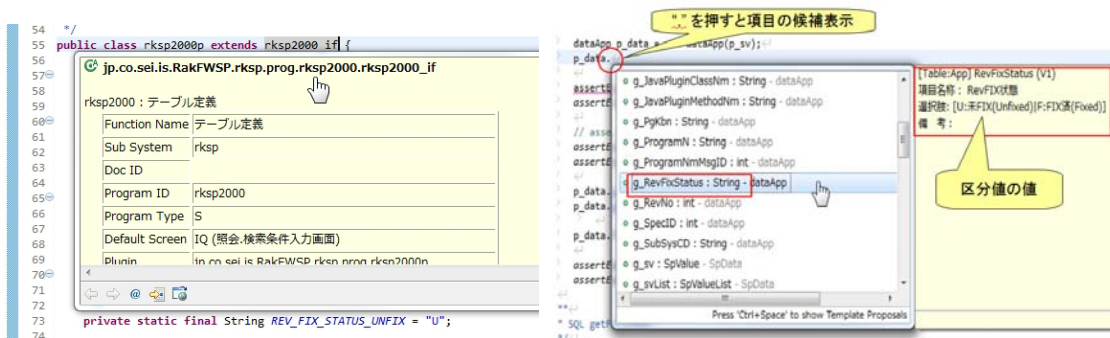


図3 Java 開発環境の補助機能

(6) 変更・改善後の状態と効果

開発した機能の効果を確認するため、ドメイン開発者で小規模な評価用のシステム開発を行った。その結果、当初の3つの目標に対して(a)設計工程が改善前と比較してほぼ同一かそれ以下の工数で完了すること、(b)新フレームワークにより部品組み立て定義なしで問題なく開発ができること、(c)仕様表示機能により業務ロジックの開発工程がスムーズに進められることを確認した。

(7) 改善活動の妥当性確認

現在、実プロジェクトでの試行が行われており、発表の際に試行結果を合わせて報告する。

1A3「Software Product Line の実践-テスト資産の構築」服部悦子(住友電気情報システム)

<タイトル>

Software Product Line の実践

<サブタイトル>

テスト資産の構築

<発表者>

氏名（ふりがな）： 服部 悦子（はっとり えつこ）

所属：住友電気情報システム株式会社 QCD改善推進部 品質改善推進グループ

<共同執筆者>

氏名（ふりがな）： 中村 伸裕（なかむら のぶひろ）

所属：住友電気工業株式会社 情報システム部 情報技術部

・ 要点

当組織では開発フレームワークのバージョンアップに伴いソフトウェアプロダクトラインで示されているテスト資産の提供に取り組むことになった。その中で時間のかかる作業であった自動テストの実現について、テストプログラム及びテストデータを自動生成することにより工数削減を実現し開発プロジェクトで利用できるよう整備した。本発表ではこの事例を紹介する。

・ キーワード

自動テスト、単体テストの改善、デグレードの防止

・ 想定する聴衆

基幹系システムの PM、PL、開発者

・ 適用状況

☐ 多用されている段階、☒ 適用できる段階あるいは初めて適用する段階

☐ 適用するにはさらに検討を必要とする、☐ 着想の段階

☐ その他（ ）

・ 適用可能性に関する制限

☐ 汎用性がある、

☒ 類似プロジェクトにも適用可：具体的な類似点（ ）

☐ 自プロジェクトのみ

・発表内容

(1) 背景

当組織では昨年より生産性向上活動を開始している。コーディングの生産性向上を目指し開発フレームワークのバージョンアップを始めた。その中で従来実施できていなかったソフトウェアプロダクトラインで示されているテスト資産の提供についても取り組むことになった。

(2) 改善前の状態

テスト資産を提供するにあたり単体テストについて現状調査するといくつかの問題点が見つかった。

開発フェーズでは単体テスト後の修正作業の後で全てのテスト項目を実施する人と一部しか実施しない人とがいる。一部しかテストしない人は、修正による影響範囲が正しく把握できていない場合、デグレードを発生させ欠陥流出が起きている。

IT フェーズになるとプロジェクトによっては開発メンバーが他プロジェクトへ移っていきたりしてプロジェクトに残っているメンバーで修正することになる。修正を担当することになったプログラマーは修正に関係すると判断した部分だけをテストする。他人の作成したプログラムを修正することになり開発フェーズ以上にデグレードの発生する確率が高い。

(3) 改善前の状態をもたらした原因（因果関係）

プログラムの修正が必要になった時、プログラマーは限られた時間の中で修正とテストをする必要がある。テストをするにはテストを実行する環境の準備やテストデータの準備が必要である。準備には時間がかかるので全てのテスト項目を実施することができず、修正に関係すると判断した項目だけ再テストする。結果その判断が間違っているとデグレードを検知できず流出させてしまっている。

(4) 変更内容・対応策

この問題を解決するには自動テストを実現する必要があると考えた。自動テストが実現できればプログラム修正後に現在と同じ工数で全ての単体テスト項目を実施することができ、デグレードが発生していても検知できる。

(5) 変更や対応策の実施内容

自動テストの実現に向け、①従来のテスト工数と同じ時間で自動テストを開発する（自動テスト開発の効率化）、②初めて取り組む自動テストについて効率よく習得させる（習得性の確保）が必要あり、以下のように取り組んだ。

◆ 課題①. 自動テスト開発の効率化

自動テストを従来のテスト工数内で開発するために、テストプログラム及びテストデータの雛形を自動生成するツールを開発することにした。

バージョンアップ中の開発フレームワークでは機能の設計情報がリポジトリとしてデータベース化されプログラムのインタフェース（メソッド名、引数）を保持している。インタフェースは15種類あり、種類に応じたテストプログラムの雛形を自動生成することができる。プログラマーは自動生成された雛形に必要なロジックを追加することでテストプログラムを開発することができる。

又、テストデータについても同様にリポジトリより画面項目を判断しテストに必要なデータの雛形を生成し使えるようにした。テストデータはテストプログラムから切り出し CSV ファイルとしたことにより、テストケースに応じて簡単にコピーして作成することができるようになり効率化できた。

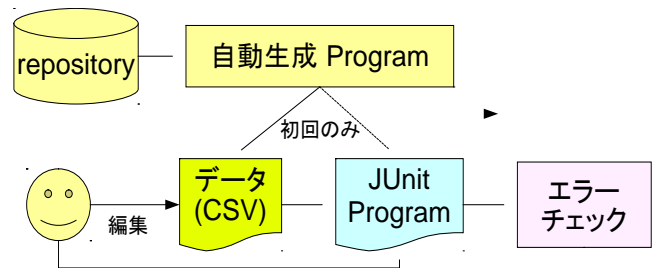


図 1. リポジトリを活用した自動生成プログラム

◆ 課題②. 習得性の確保

自動テストは新しく取り組むプロセスであり習得に時間がかかることが想定されるが、社内の開発プロジェクトに展開することを考え習得し易くしておく必要があった。

そこで、まずテストプログラム、テストデータの雛形を自動生成することで初動の取り組み易さを確保した。

次に開発者が必要な都度すぐにマニュアルを参照できるようテストプログラムからマニュアルへのリンクを作成することにした。マニュアルはエラーチェックや更新処理、画面表示の加工ロジックなど 15 のパターンに分類し、今編集しているテストプログラムの種類に適したマニュアルに飛ぶよう工夫した。

この 2 つの対策で習得性を確保した。

表 1. 課題と対策関連図

課題 \ 対策	テストプログラム 自動生成	テストデータ 自動生成	テストデータ CSV ファイル	マニュアル
自動テスト開発の効率化	○	○	○	
習得性の確保	○	○		○

(6) 変更・改善後の状態と効果

開発したツールをフレームワーク開発チーム内で試行評価した結果、

- ・ 従来のテスト工数とほぼ同等で自動テストプログラムを開発できた
- ・ テストファーストの実現で実装する業務ロジックの動作イメージが頭にできあがり、ロジックの構造（分岐・ループ）がイメージし易くなる

ということがわかった。

又、テスト実行時のカバレッジを確認できるのでテスト漏れ防止効果も期待できる。

(7) 改善活動の妥当性確認

現在、評価プロジェクト（実開発プロジェクト）が進行中である。

本発表の時には自動テストの実現とマニュアルの効果を確認し、経過とあわせて報告する。

以 上

1A4「テスト自動化を現場に普及するための組織的な取り組み」吉田 麻紀(インテック)

<タイトル>

テスト自動化を現場に普及するための組織的な取り組み

<サブタイトル>

<発表者>

氏名（ふりがな）：吉田 麻紀（よしだ まき）

所属：株式会社インテック 技術部

<共同執筆者>

氏名（ふりがな）：小林 麻美（こばやし あさみ）

所属：株式会社インテック 技術本部 技術部

氏名（ふりがな）：藤井 彩乃（ふじい あやの）

所属：株式会社インテック 技術本部 技術部

氏名（ふりがな）：山崎 春奈（やまざき はるな）

所属：株式会社インテック 技術本部 技術部

氏名（ふりがな）：加藤 康記（かとう やすのり）

所属：株式会社インテック 先端技術研究所 研究開発部

氏名（ふりがな）：西川 美紀（にしかわ みき）

所属：株式会社インテック 先端技術研究所 研究開発部

氏名（ふりがな）：高木 慎也（たかぎ しんや）

所属：株式会社インテック 先端技術研究所 研究開発部

・ 要点

- ・ テスト自動化を多くの現場に広く普及し、効果的に利用するためには多くの課題がある
- ・ 個々のプロジェクトや技術者の努力だけでは解決困難な課題も多く、生産環境、標準化、育成など多面的かつ組織的な取り組みが必要である
- ・ 本発表では、そのなかから「クラウド型テスト自動環境 TaaS」「テスト標準化の取り組み」などについて照会する。

・ キーワード

ソフトウェアテスト、テスト自動化、テスト設計、テストツール

・ 想定する聴衆

SEPG、プロジェクトマネージャ、ソフトウェアエンジニア

・ 適用状況

- ☐ 多用されている段階、☐ 適用できる段階あるいは初めて適用する段階
☐ 適用するにはさらに検討を必要とする、☐ 着想の段階
☒ その他（多用までいかないが適用され始めている段階）

・ 適用可能性に関する制限

- ☒ 汎用性がある、
☐ 類似プロジェクトにも適用可：具体的な類似点（ ）
☐ 自プロジェクトのみ

・発表内容

(1)背景

- ビジネス変化のスピードは年々加速し、それを支援する IT システムも迅速な対応が必要であり、開発・改修期間の短期化と生産性向上が迫られている。
- 一方、スマートデバイスや多種のブラウザなどの出現により、ユーザインタフェースのマルチデバイス化が進んでいる。それに伴い、アプリケーションが動作保証すべき環境も多種多様となり、また、それらの製品ライフサイクルも短くなっている。
- 従来の手動テストに頼っていたは、このような IT の多様化やスピードに対応するためのテスト工数が爆発的に増え、いずれ対応が困難になることは明白である。早期にテスト自動化技術を確立し、現場に普及・展開し、テストの生産性向上を実現する必要がある。

(2)改善前の状態

- 上記の背景もあり、社内においてテスト自動化への関心は高い。しかし、実際に自動化に取り組むプロジェクトは多いとは言えず、導入を試みても途中で断念したり、限定的な利用に留まるケースも多い。
- 大半のプロジェクトは従来どおりの手動テストから脱却できず、そのためにテスト漏れによるディグレードやテスト工数増によるプロジェクト遅延などが少なからず発生している。
- テスト自動化の導入が進まないのは、技術的なハードルが高いことや、多くの誤解や理解不足が存在することが理由であろう。テスト自動化は依然として特殊な技術であり、それが現場への普及を妨げている。

(3)改善前の状態をもたらした原因（因果関係）

- 2011 年から約 2 年間、社内システムの開発・保守プロジェクトにおいて、テストの標準化と自動化の試行に取り組んだ。その経験からテスト自動化技術を現場に広く普及し、効果的に活用するためには多くの課題があることがわかった。
- なかでも以下の課題は、個々のプロジェクトや技術者の努力だけでは解決が困難であり、組織的な取り組みが必要であると判断している。
 - (1) 自動化ツールは誰もがすぐに上手く使いこなせるわけではない
既製のツールを使いこなすには、相応の経験やスキルが求められ、ある程度の試行錯誤も必要である。最初から上手く使いこなすのは難しい。
また、ノウハウは経験者に閉じてしまいがちで皆が同じような壁にぶつかる。
 - (2) テストケース（設計）に曖昧さや属人的な部分が多い
テスト仕様書のテンプレートや項目はプロジェクトごとに決められ、取り決めが曖昧なため、その内容は担当者によってばらつき（抜け・漏れ・粒度）がある。自動テストの場合、テストの内容や手順を詳細にスクリプトとして記述する必要があり、曖昧さや属人性をできるだけ排除することが求められる。
 - (3) 自動化への高すぎる期待や誤解がある
テスト自動化により大きなコスト削減ができるだろう、という期待があるが、実際には必ずしもコスト削減にはつながらない。また、自分達の課題が明確でないまま、自動化すること自体が目的になってしまうケースもある。これらはテスト自動化に対する理解不足が根底にあると考えられる。

(4) 変更内容・対応策

- 上記の3つの課題を解決し、テスト自動化技術を現場に広く、速やかに展開するために以下の活動に組織的に取り組むことを計画し、順次実施している。
 - (1) 誰でもすぐに上手く自動化ツールを使いこなせるようにする
 - ・ クラウド型テスト自動化環境 TaaS (Test as a Service) の構築と利用
 - ・ TaaS に習熟したサポートチーム (問合せ・支援・改修など) の設置
 - (2) テスト計画・設計の曖昧さや属人性をできるだけ排除する
 - ・ テスト設計の標準化 (テスト観点や設計項目など)
 - ・ テスト方法・手法やテストプロセスの標準化
 - (3) 自動化への高すぎる期待や誤解を解き、正しい理解のもとに取り組む
 - ・ テスト自動化の基本的な考え方をガイド化し、マネージャやリーダーへ説明
 - ・ 診断にもとづく課題・期待の明確化と、テスト自動化戦略・計画の策定
 - (4) テスト自動化に伴うトータルコストをできるだけ下げる
 - ・ オフショア/ニアショアの活用、そのためのテストプロセスの見直し
 - ・ テスト資産の再利用

(5) 変更や対応策の実施内容

- 対応策のなかで主要な2つの取り組みについて詳述する。
 - (1) クラウド型テスト自動化環境 TaaS の構築と利用
市販製品・OSS・自社開発を組み合わせた社内向けのテスト自動化環境であり、次のような特徴をもつ。
 - ・ 社内の開発クラウドサービス (ezPlatform) 上に構築し、国内およびオフショア拠点から自由に利用できる
 - ・ 自社開発のテストスクリプトジェネレータにより、スクリプト作成・保守工数を削減する
 - ・ ezPlatform の仮想化基盤を利用し、複数の OS やブラウザに対する自動テストが可能である
 - (2) テストの標準化や自動化戦略策定・導入支援
個々の現場においてテスト自動化の導入を容易にするための標準化や、テスト自動化の戦略・計画策定などを支援する。
 - ・ テストの基本的な考え方やテストプロセスの認識統一、テスト観点の共通化、自動化を意識したテストケースの作成手順などの標準化である。
 - ・ このような取り組みは、テスト計画や設計の曖昧さや属人性を排除し、テスト自動化の導入を容易にする。さらにテスト漏れや抜けを防ぎ、テストそのものの質も高める。またテスト資産の再利用性やテストプロセスの視認性 (見える化) の向上にもつながる。

(6) 変更・改善後の状態と効果

- TaaS は、テスト自動化のハードルを下げ、テスト自動化が初めての場合でも比較的簡単に取り組むことを可能にした。これまで中小規模プロジェクトを中心に 10 数件の適用実績があり、効率面でも相応の効果が表れている。
- 大規模プロジェクトでのテスト自動化の適用実績はまだ少ない。テストの標準化と一体で取り組む必要があり期間や労力がかかる。従来できなかった回帰テストが部分的に実施できるようになるなどの改善成果は認められるが、品質・生産性の定量的な評価にはもう暫く時間を必要とする。

(7)改善活動の妥当性確認

- 本施策は、テスト自動化を社内の多くの現場に広く普及し、効果を上げることが目的であり、現在はその途上にある。目的達成のための課題がある程度明確になり、それらの課題への対策を取りながら組織的に推進・運営する体制はできつつある。
- 当面は、テスト自動化の効果が見える化し、それをもって各現場の理解を得ながら順次展開を図っていくことになるが、普及がある程度進んだ段階ではテスト自動化を梃子に現状のテストプロセスや体制などを大きく変革していくような判断も必要となるだろう。
- また、将来的にはテストの自動化だけでなく、その技術的な成果をもとに順次他の工程の自動化へと発展させていく計画である。自動化の追求により生産性を飛躍的に向上させることが目的の一つであるが、一方で単純・繰り返し作業をできるだけ IT に任せ、技術者が本来行うべき知的な業務に専念できるようにすることも大きな目的と考えている。

以 上

1B1「残業のいらないソフト開発への第一歩」加藤貴裕(たのしいソフト開発研究所)

＜タイトル＞

残業のいないソフト開発への第一歩

＜サブタイトル＞

4年間ほぼ残業なしを実現した取組の一つである工程作成法を紹介

〈発表者〉

氏名（ふりがな）： 加藤 貴裕（かとう たかひろ）

所属：たのしいソフト開発研究所

〈共同執筆者〉

氏名（ふりがな）：

所属：

· 要点

進捗遅延が当たり前のソフト開発ですが、工程表から正確な進捗がわからないことがその原因の一つと考えています。そこで工数の見積もり方法を変えて、進捗が明確にわかる工程の書き方をご紹介しますと考えています。

- ・ キーワード

イメージスケジューリング、プロセス設計、作業定義

- ・ 想定する聴衆

以下のような悩みをもつソフト現場技術者。

- ◆ソフト開発は工程通りにいかないのが当たり前であり、未来はわからないのだから工程表を書いても無駄ではないかと考えている人。
- ◆慢性的な進捗遅延をなんとかしたいと考えている人。
- ◆このまま作業すれば納期に間に合うと信じられる拠り所としての工程が欲しいと考えている人。

- ・適用状況

- ☒ 多用されている段階、☐ 適用できる段階あるいは初めて適用する段階
☐ 適用するにはさらに検討を必要とする、☐ 着想の段階
☐ その他（ ）

- ・適用可能性に関する制限

- ☒汎用性がある、
☐類似プロジェクトにも適用可：具体的な類似点（ ）
☐自プロジェクトのみ

・発表内容

(1) 背景

私がソフト開発業界に入ったのが 1995 年。その当時からソフト開発は残業が当たり前でした。当たり前というかむしろ、残業しなければ普通じゃないという環境でした。

残業するのが当たり前という環境を好んでいる自分もいました。

◆徹夜で仕事をして会社へ泊まり込み、翌日皆から驚きの表情で迎えられる。

◆普段スーツの会社で休日出勤時は普段着が許されていた環境で、日曜日休日出勤で徹夜した翌日、皆がスーツの中普段着でいる自分。

これらのことに誇りを感じていた自分がいました。

「残業するのは無能の象徴だ」

一部の人間は残業している環境を批判していましたが、そう批判している人々も残業していたことで、誰もこのような話に見向きもしませんでした。

1999 年ごろから CMM という言葉が聞かれ始めても、今仕事があり何も変える必要を感じなかったことで、この状況を変えよう、変えたいとは思いませんでした。

ところが、IT バブルが弾けて突然仕事がなくなります。つい一か月前には残業しなければならない状況だったのが、残業してはならない状況に変化します。

このような状況で(株)システムクリエイツの清水さんのホームページを目にします。

ホームページの記事から、

「必ずなにか目的がある状態で始まるソフト開発は、他の職業と比べてもっとも計画的に作業しやすいのでは？」

と考えるようになります。

それからホームページの文章を元に PFD や詳細スケジューリング、要求仕様書とは何かを学び、残業をしないようにするために試行錯誤の日々がはじまります。

周りが「残業のないソフト開発など夢だ」と相手にしない状況で周囲の協力など得られるはずもなく、スキルのない私がこれを実践していくのは無理があると感じます。

そこで、作業範囲が限定できる派遣技術者へ立場を変えるなどして、まず勉強の時間が確保しやすい環境をつくります。

そうして数年後にある程度やり方にぶれがないスケジュール法ができます。

その頃、周りには相変わらず残業に苦しんでいる人ばかりでした。

そこで「周りで身体を壊すまで残業している同僚に伝えるためにどうするか？」それには実際にやって見せることが一番と考え、現場で数年間残業をしないソフト開発を行います。

ですが、「あんただからできる」、「定時で帰るんじゃないかって余裕あるんだからもっと残業しろ」といわれるばかりで誰も「どうやっているんだろう？」と関心をもつことはありませんでした。

そこで、本発表の場をお借りして、私がどのように考えどんなことを実践していったのかを広く浅くお伝えすることで、「必ずしも残業は必要なさそうだ」ということに気付いていただきたいと考えています。

(2) 改善前の状態

曖昧にしても開発当初から目的とスケジュールが決まっているソフト開発。他の職業と比較しても決して計画しにくいとは思えませんが、実際は計画通りにいかず毎日深夜までの残業や休日出勤が当たり前になっています。

このような現状で現場の技術者は心身ともに疲れ切った状態になっています。

この計画通りにいかない原因の一つに工程表を作成する方法があると考えています。

私が実践してきた工程表の作成方法のポイントをお伝えすることで、自分が作った工程表を信頼し、進捗遅延の不安におびえて残業しなくてもよいと考え行動するキッカケになればと考えています。

(3) 改善前の状態をもたらした原因（因果関係）

プロセスつまり、納品物に至る成果物の連鎖関係があいまいであること。

それから、どんな作業を実施するのかあいまいなまま予想する工数で作成していること。

これらが、工程を立てているにも関わらず残業を余儀なくされている原因と考えています。

(4) 変更内容・対応策

作業期間を決まった作業フェーズ（設計、コーディングなど）の感覚的割合に応じて割り振るような工数見積もり方法から、納品物に至る成果物や作業の連鎖関係を明確にした上で具体的にした作業から工数見積もりを行うようにしました。

(5) 変更や対応策の実施内容

説明するスケジュール法は、これまでの工数見積もりとは違う方法で工数を見積もります。Process Flow Diagram (以下 PFD と称す) を使いプロセスを視覚化し、作業を階層化することで具体的な作業まで分解します。

説明するスケジュール法は、分解し具体的にした作業の文言から実際に行う作業をイメージし工数を予想します。工数を見積もる段階でイメージした作業を、実際に作業できるほどにイメージすることで、予想に対する現実の差が少なくなります。

また PFD と成果物、作業の定義を元にプロセスをシミュレートすることで、計画段階での予想外な出来事が大幅に減ります。

そして、想定した作業を実際に行うため予想と実際の成果物の差、つまり進捗が明確にわかるようになります。

進捗が明確にわかるということは、実際に問題になるよりも早い段階で、予定よりもどれくらい遅れているのか？どのくらい残業すれば遅れがなくなるのか？が、わかるということです。このため、プロセスを組み替えるなど問題になるタイミングよりもかなり早いタイミングで対策を検討、実行することができます。

尚、実際にお伝えしようと考えていることは別ファイルへ記載しましたので、そちらをご参照いただけたら幸いです。

【ファイル名：加藤貴裕 説明内容の案】

(6) 変更・改善後の状態と効果

2008 年から 2012 年にかけて従事したソフト開発プロジェクトに関して、実績と予想との差分をトレースし、リスケしていくことでほぼ残業なしの毎日を実現できました。

(7) 改善活動の妥当性確認

本活動を実施することで工程表から明確な進捗を把握して、早いタイミングで進捗遅延を把握、プロセスを検討しリスケなどの対策を打てるようになります。

また、納品物に至る成果物の連鎖関係つまり、プロセスを設計することで予想外のごとの発生をある程度は抑えることができます。

以上のことから、ソフト開発の範囲に限定すればご紹介する工程表の作成方法へ変えることで残業は確実に減らすことが可能です。

尚個人的に取組、残業なしで開発できるようになると以下の課題が浮上します。

◆残業するのが当たり前という職場の習慣にどう対処するのか？

残業するのが当たりの職場で、定時で帰ろうとすると必ずと言っていいほど「なまけもの」扱いを受けます。

このため、せっかく残業せずに開発できるようになっても努力が無駄になるため結局、元のやりかたへ戻ってしまいます。

◆残業して苦しんでいる同僚を後目に快く帰ることができるのか？

自分は残業せずに作業できるようになっても、他に残業で苦しんでいる人を無視して帰るというのは、ものすごく後ろめたいものを感じます。

1B2「テスト設計プロセス可視化の取り組み」伊藤由起子(パナソニック)

<タイトル>

テスト設計プロセス可視化の取り組み

<サブタイトル>

<発表者>

氏名（ふりがな）： 伊藤 由起子（いとう ゆきこ）

所属：パナソニック株式会社

<共同執筆者>

氏名（ふりがな）： 森下 直人（もりした なおと）

所属：パナソニックアドバンステクノロジー株式会社

・ 要点

これまで、テスト計画、テスト分析、テスト設計、テスト実装などのソフトウェアテストのプロセスが整備されてきたが、テスト分析・テスト設計の詳細な手法・手順の大部分は、ターゲットとなる商品毎に定められた担当者に任されている。

そのため、テスト分析・テスト設計のやり方は、熟練者のスキル、ノウハウにたよっている場合が多い。

本取り組みでは、PFD（Process Flow Diagram）を用いて各担当者のテスト分析・設計のやり方をプロセス視点で可視化し、ノウハウを共有した。

プロセス視点で設計資料を整理しなおすことにより、共通して実施すべき部分、異なっているが、異なっていてよい部分を議論でき、各人のよいやり方を学ぶことができた。

さらに、リーダがすべきことと、メンバがすべきことを議論、共有できた。

・ キーワード

テスト 検証 可視化 プロセス PFD Process Flow Diagram

・ 想定する聴衆

SEPG、テストエンジニア

・ 適用状況

☐ 多用されている段階、☒ 適用できる段階あるいは初めて適用する段階

☐ 適用するにはさらに検討を必要とする、☐ 着想の段階

☐ その他（ ）

・ 適用可能性に関する制限

☐ 汎用性がある、

☒ 類似プロジェクトにも適用可：具体的な類似点（ ）

☐ 自プロジェクトのみ

・発表内容

(1) 背景

これまで、テスト計画、テスト分析、テスト設計、テスト実装などのソフトウェアテストのプロセスが整備されてきたが、テスト分析・テスト設計の詳細な手法・手順の大部分は、担当者に任されている。そのため、テスト分析・テスト設計のやり方は、熟練者のスキル、ノウハウにたよっている場合が多い。

しかしながら、熟練者から人が入れ替わるときや、人を増やさなければならないときに、熟練者のノウハウを展開できることは、テストの質を保持していく上で重要である。

(2) 改善前の状態

テスト分析・テスト設計は、商品毎に決められた担当者がそれぞれの裁量でいろいろな工夫をして進め、資料化している。

その一方で、ターゲットとなる商品は異なるものの、テスト分析・テスト設計のノウハウとして共有すべきものもあると考えられる。

担当者毎に書き方が異なる部分については、どんな工夫をしているか第三者が把握することが難しく、ノウハウを共有し切れていない。

(3) 改善前の状態をもたらした原因（因果関係）

ソフトウェアが多様化する中で、テスト分析・テスト設計のやり方の大部分は、担当者に任されている。また、異なる商品をテストする担当者間では、テスト対象が異なるため、テストを実施する上で設計書を共有する必要性がない。

(4) 変更内容・対応策

PFD (Process Flow Diagram) を用いて各担当者のテスト分析・設計のやり方をプロセス視点で可視化し、比較・共有した。

アクティビティのような大きな粒度のレベルでプロセスを可視化するのではなく、プロセスをタスクレベルまで分割して PFD 化することにより、各担当者がもつノウハウを分析するとともに、横展開を実施して、よいやり方を組織的に活用できるようにした。

(5) 変更や対応策の実施内容

3つのプロジェクトのテスト分析・設計資料を参照し、それぞれのやり方を PFD 化した。そして、ヒアリングにより PFD を補完した。内容を分析するとともに、横展開を実施して、よいやり方を組織的に活用できるようにした。

PFD 化にあたっては、ノウハウを抽出するために、アウトプットとなる設計書単位の粗い粒度のプロセスではなく、何を題材に、どう考えて資料化したのかがわかる粒度までプロセスを分解した。たとえば、「変化点を抽出する」「新規・流用・改造を特定する」といった様に、どんな検討をしたのかが分かる粒度でプロセスを分解し、それをどんな順序で行ったのかがわかる形で可視化した。そのために、成果物として表がある場合には、表の列への項目の設定、表の行への項目の設定、マトリクスへの記入それぞれについて、何を目的に、何を参照し、どう考えて設定したのかがわかる形で可視化した。

設計資料には記述されていないが、頭の中で考えている部分についても、記述することにより、テストエンジニアの頭の中を可視化した。

議論にあたっては、共通して実施すべき部分、異なっているが、異なっていてよい部分を意識して議論を進めた。

さらに、PFD 化した内容を担当者別に分類・整理することにより、リーダとメンバの間でどのように分担してテスト分析・設計を進めているかを可視化した。

(6) 変更・改善後の状態と効果

プロセス視点で設計資料を整理しなおすことにより、共通して実施すべき部分、異なっているが、異なっていてよい部分を議論できるとともに、各人のよいやり方を学ぶことができた。

たとえば、表現方法については、文章よりも図や表で表現するとわかりやすいということのをあらためて認識した。競合条件の整理の仕方については効率のよいやり方を検討する必要があることがわかった。さらに、リーダがすべきことと、メンバがすべきことを議論、共有できた。レビューについては、何をレビューすべきか目的を明確にした上で、レビューア・対象成果物を目的に合ったものとする必要があるということがわかった。

結果として、テスト分析・テスト設計をどうすべきかについてプロジェクトメンバで共有することができた。

今後、よいやり方をベースにガイドライン化し、新規メンバがはいった場合にも共有できる様にしていくとともに、他プロジェクトへ展開する。

(7) 改善活動の妥当性確認

実施範囲外

1B3「トヨタ開発方式を利用したソフト開発の継続改善活動」比嘉定彦(アドバンテスト)

<タイトル>

トヨタ開発方式を利用したソフト開発の継続改善活動

<サブタイトル>

開発中の“待ち”や“遅れ”を、“不良(欠陥)”と同列に扱い共に改善する方法の提案

<発表者>

氏名(ふりがな): 比嘉 定彦(ひが さだひこ)

所属: 株式会社アドバンテスト 品質保証本部 ソフトウェアQA部

<共同執筆者>

氏名(ふりがな): 清海 光子(きようみ みつこ)

所属: 株式会社アドバンテスト 品質保証本部 ソフトウェアQA部

・要点

トヨタ開発方式は、開発中の“待ち”や“遅れ”を“不良(品質欠陥)”と同列に扱い(7つのムダの一つという点で同列)、改善対象とする方法論である。また継続改善を進めるうえで有用な方法である。

・キーワード

トヨタ開発方式
7つのムダ

・想定する聴衆

開発リーダの方
プロセス改善推進業務に従事している方

・適用状況

☐多用されている段階、☒適用できる段階あるいは初めて適用する段階
☐適用するにはさらに検討を必要とする、☐着想の段階
☐その他()

・適用可能性に関する制限

☐汎用性がある、
☒類似プロジェクトにも適用可: 具体的な類似点(装置の制御用ソフトウェアの開発)
☐自プロジェクトのみ

・発表内容

はじめに

本発表の内容は、弊社でトヨタ開発方式*1について調査/検討し、ソフト開発現場で実践した内容と結果をまとめたものである。

*1) 次の文献にて、著者が名付けた開発方式。

”トヨタ製品開発システム” James M. Morgan、Jeffrey K. Liker 著

補足) 著者は実務および研究の知見を基に、トヨタ自動車の開発部門に出向き調査した結果を 開発モデル/プロセス/教育/ツールと技術 の分野別に仕組と事例を関連付けて整理し、“トヨタ製品開発方式”と名づけた。

1. 背景

弊社では、1990年代よりソフトウェアプロセス(規模/工数/品質)を定量化する活動を行い、定量化の概念(メトリクス)が普及してから久しい。
メトリクスが普及した結果、品質は改善されてきたが工期の改善は課題という状態が続いた。

2. 改善前の状態

改善の対象として取り上げたのは、開発工期と品質を共に改善することが難しい現実があるためである。

現状：

品質目標は達成した一方で開発工期は延長されることが多い。
また、顧客納期を延長できない場合には要求機能を分納する等で対応している。

上記の現状を改善の対象とした理由は、開発工期と品質を共に改善できる方法こそ真の改善であると考えたためである。

3. 改善前の状態をもたらした原因(因果関係)

開発工期の延長や機能の分納が起きる要因を考察した結果は次の通りである。

現状の分析：

定量化(メトリクス)の普及によりソフトウェアプロセスは可視化され、デザインレビューが徹底されるようになったが、それだけでは開発工期と品質を共に改善していくことにはつながらない。

4. 変更内容・対応策

我々は『目標工期と品質の両者を共に達成できる取り組みが必要』という状況の中で、トヨタ開発方式に出合った。

トヨタ開発方式では不良(欠陥)だけでなく、開発工程の中で発生する “待ち” や “遅れ” の状態も改善の対象としている。“遅れ”や待ち”の状態を改善することは開発工期の改善につながるはずである。

我々は、トヨタ開発方式(原則3)にある「7つのムダ」と対応施策に着目し、対応施策(トヨタ開発方式)を社内のソフト開発で活動し易いようにカスタマイズし、運用することで現状を改善することにした。カスタマイズした結果を次に示す。

7つのムダ	製品開発での例	対応施策 (トヨタ開発方式)	社内用にカスタマイズ
作り過ぎ	機能別部門間での活動が同期していない	プロセスロジックとマイルストーン要求に基づく平準化と同期化	部署間の同期化と平準化を導出するマイルストーンを設定し維持する
在庫	作り過ぎのムダの結果、後工程のリソースが空くのを待つ状態が発生		
手持ち	成果物(技術情報を含む)や意思決定を待っている	品質／技術データベースから知識を引き出す	課題ばらし*(検討プロセス)の質を確保／向上する
加工	流用可能部品をゼロから設計、標準アーキテクチャから逸脱した設計,等	標準化したマイルストーンと詳細計画	マイルストーンを使って開発をブレイクダウンし、総量ばらし*／課題ばらしを実施。 課題ばらしした結果をもとに作成した作業計画は無駄を排除しチームの目標を網羅
動作	意思決定や仕様確定につながらない資料/報告書の作成や、会議への参加		
運搬	専門化されすぎた1つの部門から他の部門へ技術情報(技術チェックリスト等)を提供する	製品開発価値の流れの改善	マイルストーン完了遅れなどを分析し、開発の阻害要因を取り除く
不良	後になって発生する設計変更や手直し	リーン製品開発への転換	プロセスを継続的に改善

5. 変更や対応策の実施内容

本活動の実施内容を以下に示す。

1) 準備会

準備会を開催し、前回の開発終了時に行った改善検討結果が今回行う開発の計画へ反映されていることを確認した。

2) 開発管理状況の確認

マイルストーン期日のタイミングでF2Fを開催し、開発リーダー(課長)へ“待ち”や“遅れ”、“不良”の有無を確認した。あるプロジェクトで確認した例を次に示す。

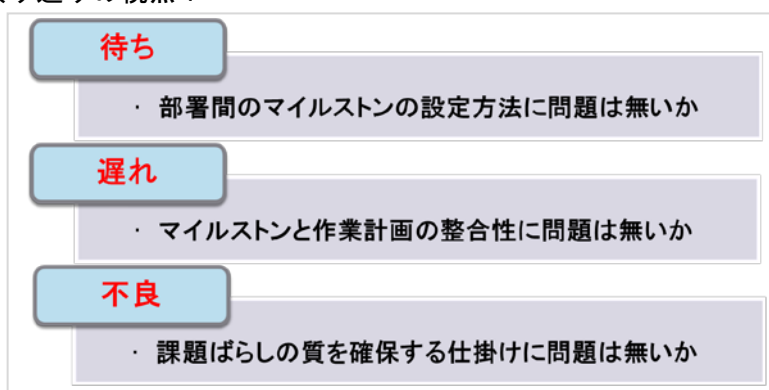
マイルストーン管理表〈プロジェクトB〉 確認結果

マイルストーン	期日(予定)	達成日	確認結果
M1.オフライン単体テストが完了しオンライン単体テスト開始	2012/10/19	2012/10/25	遅れ
M2.オンライン単体テストが完了しオンライン総合テスト開始	2012/11/02	2012/11/02	OK
M3.新規版対応のため、* *へオフライン環境を提供する	2012/11/06	2012/11/20	遅れ
M4.従来版で動作実績のあるデバイス実測評価が開始できる	2012/12/07	2012/12/27	待ち

3) マイルストーン毎の振り返り

終了遅延等が発生したマイルストーンについて、“待ち”や“遅れ”、“不良”の内容を確認し、カスタマイズした内容に基く視点を使って振り返りを実施した。

振り返りの視点：



次に、あるプロジェクトのマイルストーン終了時に 工期(“待ち”や“遅れ”)と品質(“不良”)について振り返り(YWT[2])を行った例を示す。

Y：やったこと、W：わかったこと、T：つぎにやること

・ マイルストンの達成遅れ(待ち)についての振り返り

Y：“動作実績のあるデバイス実測評価が開始できる”というマイルストーンが遅延した。

W：ハード開発から“キャプチャできる”という情報があつたが、ソフトウェアテストに耐えられる状態ではなかった。

T：ソフト開発側の期待値を明確にした部署間マイルストーンを設定する。

<改善例>

“動作実績のあるデバイス実測評価において、キャプチャが100%動作可能なハード状態で開始できる”

効果：開発部門間の活動の同期化による改善

- ・他部署から指摘された不良についての振り返り
Y：既存ハードと同じ設計で新規ハードへ対応した結果、特定のタイミングで正しく動作しなかった。
W：時系列処理に漏れがあった（ハードとソフトの制御単位的一致について不明確な所があったことが原因）。
T：時系列処理が不明確なタイミングの制御仕様について資料を作り、関係者で勉強会を行う。

効果：分析/設計効率の改善に伴う課題ばらしの質確保による品質向上

4) パフォーマンス計測

カスタマイズした内容の実施結果を、次のパフォーマンス計測指標で計測し、計測結果を基に今後の改善検討を行うようにした。

- ・“遅れ”や“待ち”を改善する指標 ⇒ マイルストーン遅延率（以降遅延と略す）
＝（遅延が発生したマイルストーンの数 / マイルストーンの総数）× 100[%]
- ・“不良（欠陥）”を改善する指標 ⇒ 他部署から指摘された不具合
＝ 工程終了後に他部署から指摘された不具合件数[件]

補足）通常の開発規模は1リリース当り10～100人月、期間は6ヵ月～1年

5) 改善検討

開発の終盤または終了時にパフォーマンス計測結果を基に現状を把握し、改善目標を達成するのに必要な課題を検討/設定する。
あるプロジェクト（プロジェクトA）での例を次に示す。

例1：現状の把握と今後の改善目標設定

現状の把握：

- マイルストーン遅延率（他部署要因を含む遅延）： $3/10 = 30\%$
- マイルストーン遅延率（自部署要因のみの遅延）： $1/10 = 10\%$
- 他部署から指摘された不具合：1件

今後の改善目標：

- 現状値を半減する（他部署要因含めて30→15%、自部署要因は10→5%へ改善）

例2：改善に必要な課題の設定

マイルストーン遅延率を半減するための課題（工期の改善）：

- ①外部要因（影響を小さくする）

- ・ハード部署が問題（課題）へ対応するスケジュールを監視する。

②自部署要因（発生リスクを低減する）

- ・開発/評価環境を整備するためのマイルストーンを設定する。

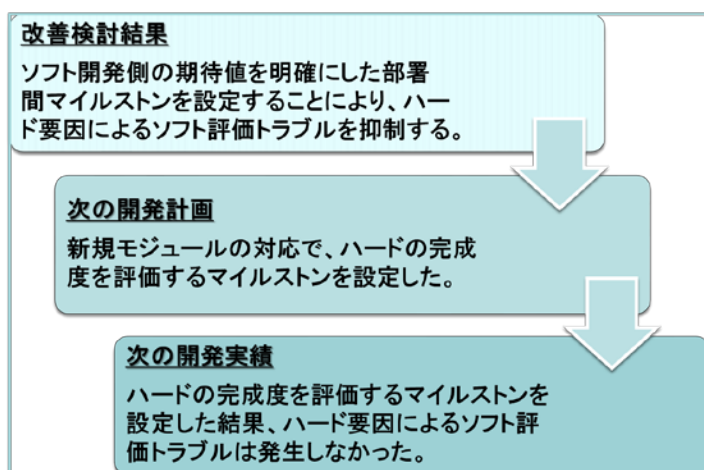
他部署から不具合を指摘されないようにするための課題（品質の改善）：

③設計要因（課題ばらしの質を向上する）

- ・不明確な部分の制御仕様について資料を作り勉強会を行う。

6) 改善検討結果の活用

改善検討を行った結果を基に次期開発の計画へフィードバックし活用した。
あるプロジェクト（プロジェクトB）での例を次に示す。



6. 変更・改善後の状態と効果

活動した初回のリリース版と活動2回目のリリース版（現在活動中）で取得したパフ
ォーマンス値を比較することにより、本活動の効果を確認する。

	【活動初回】			【活動2回目】		
・プロジェクトA						14 個中 10 終了
他部署要因を含む遅延	30%	(3/10)	->	0%	(0/14)	
自部署要因のみの遅延	10%	(1/10)	->	0%	(0/14)	
指摘された不具合	1 件		->	T B D		
・プロジェクトB						12 個中 9 終了
他部署要因を含む遅延	54%	(7/13)	->	0%	(0/12)	
自部署要因のみの遅延	38%	(5/13)	->	0%	(0/12)	
指摘された不具合	1 件		->	T B D		

プロジェクトA, B共に 現時点では遅延と不具合は発生していない。
現在迄の経過から活動2回目終了時の結果を推測すると、2つのプロジェクトが共に
工期と品質の改善目標を達成できる見込みである。

7. 改善活動の妥当性確認

パフォーマンスの向上実績が見込める状況になった要因を明確化するため、各プロジェクトが行った改善内容とその効果を確認した。

・プロジェクトA

- 補正データ転送の実現方法についてハード部門と事前に検討会を開催
(課題ばらしの質の確保により、手戻りを防止した)
- 補正データ転送のオンライン評価を前倒しで(実装の期間中に)実施
(マイルストーンに対する作業計画を網羅でき、遅延を防止した)

・プロジェクトB

- 実装以後の開発進捗は、機能1と機能2は別マイルストーンにより管理
(“待ち”や“遅れ”の伝播の防止により、開発遅延を抑制した)
- オンライン評価の開始条件(ハードの完成度調査)をマイルストーン化
(関係部門間の依存関係を把握し同期化することで、遅延を抑制した)

各プロジェクトが行った改善内容を基に今回の活動で改善されたことをまとめた結果は次のとおりであり、工期と品質を共に改善する効果があった。

今回の活動で改善されたこと：

- ・ 関係部門間の依存関係を把握する率が高くなり同期化が進んだ(“待ち”の改善)
- ・ 課題ばらしの質の確保が進んだ(“不良”の改善)
- ・ マイルストーンに対する作業計画の網羅性と整合性が改善された(“遅れ”の改善)
- ・ “待ち”や“遅れ”、“不良”の改善に必要な“T”(振返り結果)を出すようになった
(工期と品質を共に改善する方策の導出)

参考文献

- [1] James M. Morgan and Jeffrey K. Liker, The TOYOTA Product Development System
トヨタ製品開発システム(翻訳), 日経BP社, 2007
- [2] 中村素子, 勝田博明: 技術者・エンジニアの知的生産性向上, 日本能率協会マネジメントセンター出版, 2009

1B4「CMMI ベースの 1 日型簡易診断の実施」伊藤裕子(東芝)

<タイトル>

CMMI ベースの 1 日型簡易診断の実施

<サブタイトル>

ハードウェア設計開発部門における取り組み

<発表者>

氏名（ふりがな）： 伊藤裕子 （いとうゆうこ）

所属：株式会社 東芝 ソフトウェア技術センター

<共同執筆者>

氏名（ふりがな）：

所属：

・ 要点

ソフトウェア開発以外のシステム／ハードウェア設計開発部門において、診断の品質と期間のバランスを取り、短期間で効率良く実施可能な CMMI ベースの簡易診断手法を開発し、5 部門に対して適用した。

・ キーワード

アプレイザル、CMMI、診断、簡易診断、ハードウェア設計開発部門、短期間、小規模開発

・ 想定する聴衆

SEPG、アプレイザルメンバ

・ 適用状況

☐ 多用されている段階、☒ 適用できる段階あるいは初めて適用する段階

☐ 適用するにはさらに検討を必要とする、☐ 着想の段階

☐ その他（ ）

・ 適用可能性に関する制限

☒ 汎用性がある、

☐ 類似プロジェクトにも適用可：具体的な類似点（ ）

☐ 自プロジェクトのみ

・発表内容

(1) 背景

東芝グループ内では CMMI モデルをベースとした SPI 活動を全社的に推進しており、開発プロセス上の課題の発見、分析を行うための診断を活用し、改善を進めている。診断手法は、対象組織や改善のフェーズにより様々だが、SCAMPI による公式な診断手法や、SCAMPI を参考に 2 日間でゴールレベルを簡易的に診断する東芝手法などが用いられている。

また、東芝グループ内の製品開発においては組込系の開発が多く、SPI 活動を進める部門が増えるに従って、ソフトウェアだけでなくシステムやハードウェアを含めた製品開発全体のプロセス改善が求められてきている。それに伴いソフトウェア開発部門だけでなく、関連するシステム／ハードウェア設計開発部門を対象に診断をしたいという要望も増えてきている。

(2) 改善前の状態

東芝グループ内の組込系開発をしているシステム／ハードウェア設計開発部門 5 部門において、CMMI に沿った改善活動を行うこととなった。改善活動開始にあたり現状分析のために、半年で各部門に対して簡易診断を実施することとなったが、診断回数も多く、診断にかけられる時間も短かった。また、ハードウェア設計開発部門を対象とした簡易診断のプロセスはなかった。

(3) 改善前の状態をもたらした原因（因果関係）

従来の社内におけるゴールレベルでの簡易診断は 2 日間かかるものだったが、本事例では半年間で 5 回の診断を実施する必要がある、より短期間で実現したかった。

さらに、従来の簡易診断はソフトウェア開発部門向けだったため、ハードウェア設計開発部門向けの、診断方法が体系的に整理されておらず、参考にできるノウハウもなかった。

(4) 変更内容・対応策

インタビューと報告書作成は 1 日で完了するようにし、ソフトウェア以外のシステム／ハードウェア設計開発部門に対しても診断できるよう、簡易診断のプロセスを整備した。

(5) 変更や対応策の実施内容

(a) 1日で完了させるための活動

インタビュー範囲の見直しと、報告書テンプレートの改訂を行った。従来の診断では、ドキュメントレビュー、インタビュー、インタビュー結果の整理、報告書作成のプロセスで進めていた。診断の信頼性と効率性のバランスを考慮し、診断の各プロセスについて改善を行った。各プロセスの改善内容は、以下の表1にまとめた。

表1：改善前後の診断プロセスの比較

診断プロセス	改善前	改善後
ドキュメントレビュー	診断期間中に実施	事前に関連ドキュメントを受取り、診断期間前に実施
インタビュー	複数プロジェクトを対象とし、複数の立場の人からインタビューを実施	典型的な1プロジェクトのみを対象とし、プロジェクトリーダーと開発者に対して実施
インタビュー結果の整理	各インタビュー後、結果をワークシートへ記入し、整理	省略 (下記報告書作成と同時に実施のため)
報告書作成	全インタビュー完了後、各ワークシートをもとに報告書を作成	各インタビュー後、情報を取れた範囲から報告書に記入

・インタビュー範囲を見直し

短時間で診断を行うため、対象プロジェクト数とインタビュー数からインタビュー範囲を見直した。対象プロジェクト数を最低限の1プロジェクトとした一方で、部門内の典型的な1プロジェクトを選ぶようにした。また、プロジェクトリーダーと開発者の異なる立場の方から情報を得るようにしたことで、診断の品質を保てるようにした。さらに、事前にスポンサーへ組織課題についてヒアリングすることで、重視するプロセス領域を把握しインタビューを効率的に進められるようにした。

・報告書テンプレートを改訂

従来は各インタビュー後にワークシートへの記入を繰り返し行い、全インタビュー完了後に報告書をまとめていた。今回、従来の報告書にワークシートの機能も持たせた報告書のテンプレート（図1参照）に改訂したことで、インタビュー後にワークシートを利用せず、直接報告書に書きこめるようにした。

プロジェクト計画策定(PP)				
<ul style="list-style-type: none"> プロジェクトの作業を見積もり、計画を作成し、計画に対するコメントを得ること。論理的に見積もられているかがポイント 				
ゴール	状況	成果物	判定	
SG1 見積もりを確立する	強み: ・ 弱み: ・			
SG2 プロジェクト計画を策定する	強み: ・ 弱み: ・			
SG3 計画に対するコメントを獲得する	強み: ・ 弱み: ・			

プロジェクト計画策定(PP)				
<ul style="list-style-type: none"> プロジェクトの作業を見積もり、計画を作成し、計画に対するコメントを得ること。論理的に見積もられているかがポイント 				
ゴール	状況	成果物	判定	
SG1 見積もりを確立する	弱み: ・組織の見積り基準はなく、過去の類似プロジェクトの実績などの見積りも行っていない	-		×
SG2 プロジェクト計画を策定する	強み: ・開発計画書を作成し、スケジュール、体制、スキルについて計画している 弱み: ・担当毎の細かいスケジュールは作成していない	開発計画書、体制図、大日程工程表、リスク管理表		△
SG3 計画に対するコメントを獲得する	強み: ・担当者間で開発計画を調整し、合意している ・開発審査時に開発計画がレビューされ承認されている	開発審査説明資料、開発審査議事録		○

図 1：報告書テンプレートと記入例

(b) ハードウェア設計開発部門への適用

インタビュースクリプト(質問票)内のソフトウェア開発独特の用語や、理解が難しい用語に対して補足の質問を追加した。成熟度レベル2のプロジェクト管理のプロセス領域(REQM, PP, PMC, SAM)と、品質保証(PPQA)の範囲についてはソフトウェア向けの質問と同様の質問項目で対応し、追加は行わなかった。構成管理(CM)、測定と分析(MA)については質問を補足した。

・ ベースライン (構成管理(CM)に関連)

ソフトウェアの場合は、構成管理で利用されるツールに関連したビルドのルールなどを聞くことが多いが、ツールなどを活用していないことを考慮し、計画書や図面の正式版の管理について聞くことでベースラインに関する質問を補うようにした。

・ データの測定 (測定と分析(MA)に関連)

審査認定時の指摘件数や、レビュー指摘件数、不具合件数、工数の予実など具体例を挙げ、同様の活動を行っているか質問した。

(6) 変更・改善後の状態と効果

システム／ハードウェア設計開発部門 5 部門において、CMMI 成熟度レベル 2 の範囲の簡易診断を実施した。短期間の診断で受診部門に対して妥当な改善の機会を挙げることができた。診断後、診断を行った全部門において弱みに対するプロセス改善活動が開始された。

(a) 1 日で完了させるための活動

インタビューの範囲を見直し、事前にスポンサーから課題を把握したことで、診断の品質を確保し進めることができた。また、報告書テンプレートの改善によって、診断結果の整理と報告書作成工数を、従来の8時間から5時間に削減することができた。

(b) ハードウェア設計開発部門への適用

診断のインタビューでは、インタビュースクリプト内のソフトウェア特有の用語を補足することで、質問の意図が伝わり、システム／ハードウェア設計開発部門に対しても診断で必要な情報を取得することができた。

(7) 改善活動の妥当性確認

成熟度レベル 2 の範囲は基本的なプロジェクト管理のため、ソフトウェア開発部門と同様、ハードウェア設計開発部門に対しても診断が有効であることが分かった。また、ソフトウェア特有の用語の補足でハードウェア設計開発部門に対しても診断可能なことが分かった。

今後成熟度レベル 3 のエンジニアリングのプロセス領域など、対応できる範囲を広げていきたい。

1C1「アプリケーション運用・保守プロセス定着への取組み」相澤武(インテック)

<タイトル>

アプリケーション運用・保守プロセス定着への取組み

<サブタイトル>

<発表者>

氏名（ふりがな）：相澤 武（あいざわ たけし）

所属：株式会社インテック 技術部

<共同執筆者>

氏名（ふりがな）：小林 麻美（こばやし あさみ）

所属：株式会社インテック 技術部

・ 要点

保守プロジェクトに標準プロセスを導入し定着させる場合、開発プロジェクトとは異なる保守プロジェクト特有の特徴を踏まえた標準化戦略が必要である。

本発表では、当社が行った保守プロジェクトの特徴にあった標準化戦略について紹介する。

・ キーワード

標準化、プロセス改善、アプリケーション運用・保守プロセス、見える化

・ 想定する聴衆

SEPG 初心者

・ 適用状況

☐多用されている段階、☐適用できる段階あるいは初めて適用する段階

☐適用するにはさらに検討を必要とする、☐着想の段階

■その他（多用まではいかないが適用されている段階）

・ 適用可能性に関する制限

■汎用性がある、

☐類似プロジェクトにも適用可：具体的な類似点（ ）

☐自プロジェクトのみ

・発表内容

(1) 背景

当社では、お客さま満足や収益の確保などプロジェクトの成功率を高めるために「良いプロセスが最上の成果、高い生産性をもたらす」という考え方のもと全社の業務プロセスの標準化に取り組み、展開を図ってきた。標準プロセスの展開後8年が経過し、標準化による相応の効果は見られたが、開発プロジェクトの失敗やアプリケーション運用・保守プロジェクト（以下保守プロジェクト）における重大障害なども少なからず発生している。これらの失敗や障害発生の中には、標準プロセスをしっかりと守っていれば防止できたケースもある。このような背景を踏まえて、今一度、標準プロセスの重要性を再認識し、その利用を徹底するために、「標準プロセスを効果的に活用する」ための施策を実施している。今回は近年その重要性がますます高まってきている保守プロジェクトに着目して活動に取り組んだ。

標準プロセスを効果的に活用することで、「アプリケーションの運用・保守は出来上がったシステムの維持管理を行うもの」といったイメージを払拭し、システムの安定運用のみならず、運用・保守を起点とした、お客さまの経営戦略の迅速な策定・遂行や、ITシステムの全体最適の実現に貢献することを目指す。

(2) 改善前の状態

保守プロジェクトの課題として一般的にあげられる

- ・ 属人化による作業のブラックボックス化
- ・ 保守作業の過剰サービスや過小サービス
- ・ 維持管理作業に注力しがちで改善サイクルの意識が疎かとなりがちになる

は、当社においても保守プロジェクトの課題であると仮説を立てた。

(3) 改善前の状態をもたらした原因（因果関係）

今回の施策を実施するうえで、まず社内内の主要部門に対して、保守プロジェクトの現状についてのヒアリングと(2)であげた仮説の確認を行った。その結果、保守プロジェクトには、開発プロジェクトとは異なる次のような特徴があることが確認でき、(2)の課題を解決するためには、これらの特徴を踏まえた保守プロジェクト向けの標準化戦略が必要であると考えた。

- ・ プロジェクトの規模は1名体制から数十名体制までさまざまなタイプがある
- ・ 長年慣れ親しんだ独自のプロジェクトルールが確立されている場合が多い
- ・ 有期性がないこともあり、維持だけで改善サイクルの意識が疎かとなる場合が多い

(4) 変更内容・対応策

(3)で述べた保守プロジェクトの特徴を踏まえて標準化戦略・手法・進め方を考案し、それをアプリケーション運用・保守プロセス標準化ガイドにまとめた。この標準化ガイドは、保守プロジェクトにおいて、アプリケーション運用・保守プロセスを適用する際のテーラリングガイドとして位置付けている。以下に標準化ガイドの特徴をあげる。

① 保守タイプの設定

当社のアプリケーション運用・保守サービスにおける代表的な契約パターンを5つに分類し、そのパターン毎に適用手順を記載した。契約パターン毎に保守タイプを設定することで、特に規模が小さい保守プロジェクトでは、最低限、どのレベルまでを実施すればよいかを明確にした。

② サービスメニューとのマッピングの考え方を導入

長年にわたり保守契約を実施してきたプロジェクトにおいては、既にプロジェクトの中で独自のルールを確立しているところもある。このようなプロジェクトに対しては、新たに作成した標準プロセ

スに沿ってプロジェクトのルールを作り直すというよりも、自分たちの実施してきたことが、標準プロセスのどの部分に該当するのかといったマッピングを行い、既の実施できている部分は、今までのやり方を踏襲し、足りない部分があれば追加するという考え方を導入した。

③計画に基づくプロジェクト運営

保守プロジェクトの場合、開発プロジェクトのような有期性がないため、プロジェクト計画時の定量的な目標の設定やプロジェクト完了時の定量的な評価が難しく、維持管理作業をこなすだけになってしまうことが多い。標準化ガイドの中では、以下にあげる保守プロジェクトにおけるプロジェクト運営のPDCAサイクル(図1)を定義し、計画書に基づくプロジェクト運営を意識させるようにしている。

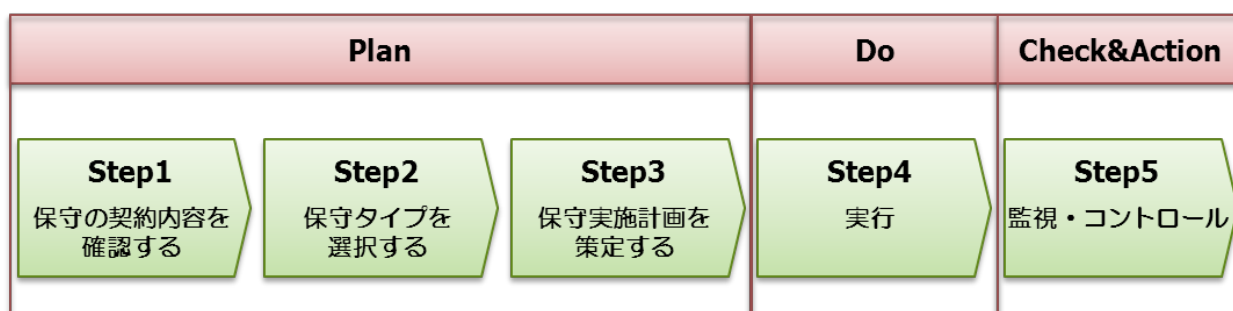


図 1.プロジェクト運営におけるPDCAサイクル

(5) 変更や対応策の実施内容

アプリケーション運用・保守プロセス標準化ガイドの活用を促進するために以下の施策を行った。

①標準化ガイドの内容をベースにしたeラーニング教材の開発

知識習得が必要なメンバが必要とするタイミングで学習できるように、日時を指定した集合研修ではなく、eラーニング教材化し、必要なタイミングにいつでも学習できるようにした。

②作業工程コードの見直し

標準化ガイドの中で定義している作業種類ごとに、作業実績が記録できるように、作業工程コードの見直しを行った。

③先行部門の取組み事例を社内広報誌(月刊)で紹介

先行部門以外への啓蒙のため、先行部門の取組み事例を社内広報誌に掲載し紹介した。

(6) 変更・改善後の状態と効果

現時点で、(2)であげた課題に対して以下の効果が確認できている。

①保守作業が見える化することができた

誰が、どの作業に、どのくらいの時間を費やしているのかといった、従来はブラックボックス化されていた部分が見える化できるようになった。見える化されたことで、本来実施すべき作業ができていなかったり(過小サービス)、保守契約範囲外の作業まで実施してしまっている(過剰サービス)などがわかった。

例えば、あるプロジェクトでは、1ヵ月間の保守の作業種類ごとの作業工数を分析し、計画以上に時間を費やしている作業(このプロジェクトの場合は、サービスデスク、業務運用が該当した)を洗い出し、その作業に対して改善のための施策を打った。

②メリハリをつけた保守プロジェクト運営

保守プロジェクトは、ともすると維持管理だけに意識が向いてしまい、プロジェクトにおける改善サイクルを回すという点が疎かになりがちであった。今回保守プロジェクトにも計画に基づくプロジェクト運営の考え方を取り入れたことで、保守作業の見える化から改

善すべき点が浮き彫りになり、そこに対して改善策を検討し対策を打っていくといった、保守プロジェクトに携わるメンバに改善サイクルを回すという意識が徐々に芽生えてきた。

③標準化ガイドを利用することで標準プロセスの適用が容易になった

標準プロセスだけの提供ではなく、標準プロセスをどのように自プロジェクトに適用すればよいかの手順を標準化ガイドで示したことで、従来に比べて標準プロセスの自プロジェクトへの適用が容易になった。

これらの効果は、先行部門の取り組み事例を紹介する社内広報誌（月刊）の中でも、複数の部門から改善効果として紹介されており、利用部門からは一定の評価を得ていると考える。

(7) 改善活動の妥当性確認

従来は、プロセスを充実することばかりに労力を注ぎすぎており、プロセスを作った後の現場へのフォローという面が十分に実施できていなかったが、今回の取り組みにより、課題であった「標準プロセスを効果的に活用する」ための一つの成果が得られたと考える。

今後も今回の事例を参考にして、他の業務の標準プロセスについても、「効果的に活用される標準プロセス」にするための取り組みを実践していきたい。

以上

1C2「全社活動によるプロセスの再改善・大改善」和良品文之丞(キヤノンソフトウェア)

<タイトル>

全社活動によるプロセスの再改善・大改善

<サブタイトル>

既存の強みの継承と抜本的見直しの両立

<発表者>

氏名(ふりがな): 和良品 文之丞(わらしな ぶんのじょう)

所属: キヤノンソフトウェア株式会社

<共同執筆者>

氏名(ふりがな): 石井 康博(いしい やすひろ)

所属: キヤノンソフトウェア株式会社

・要点

事業のニーズに基づいてプロセスを確立し、改善ニーズによって継続的にプロセスを改訂しても、環境の変化や組織の成熟度・体制の変化によってさらなる見直しが必要である。

経営トップの強力なリーダーシップの下、全社的な改善活動を期間限定で立上げ、既存のQMS(品質マネジメントシステム)をより効率的なQMSに再定義した。QMSの大幅な改造に際しては、抜本的に体系を見直すことと、既存の強み活かして不足を補うことの、両面のアプローチが有効であった。また強制する部分、共通にする部分と、事業に自由度を持たせる部分の切り分けも重要である。

当社が実施した去年のプロセス改善委員会の事例を元に、具体的な取組み、工夫点を報告する。

・キーワード

プロセス改善、改善活動、全社活動、改善委員会

・想定する聴衆

SEPG、品質保証の方、全社の改善活動の推進役の方

・適用状況

☒ 多用されている段階、☐ 適用できる段階あるいは初めて適用する段階

☐ 適用するにはさらに検討を必要とする、☐ 着想の段階

☐ その他 ()

・適用可能性に関する制限

☒ 汎用性がある、

☐ 類似プロジェクトにも適用可: 具体的な類似点 ()

☐ 自プロジェクトのみ

・発表内容

(1) 背景

2006 年から大型受託プロジェクトの管理強化に取組み、受注前から完成後までの一貫性を持つ活動を定義し、規程（ルール）とプロセスを QMS として確立した。その後、改善ニーズに基づき継続的に改訂を行ってきた。

2012 年、経営トップの強力なリーダーシップにより、プロジェクト管理のより一層の強化と品質向上に取り組むよう指示があった。

そこで、既存のプロセスを見直すこととし、改善活動の目的を、以下の 2 点に定めた。

- 全社共通で、プロジェクトリスクを適正にコントロールする
- 使い勝手が良く、効率の良い QMS を再構築する

(2) 改善前の状態

既存の QMS は全社に適用されていたが、以下の課題があった。

- プロジェクト管理精度・リスクが事業の成熟度に大きく依存する
 - 対顧客、対協力会社との契約リスクが増大している
 - 開始時・実行中・出荷時の管理精度にばらつきがある
- 複雑で使いづらいプロセスになっている
 - 適用条件が複雑で、資料も散在しており、わかりづらい
 - 現場の実情に合わず、適用しづらい部分が増えている

(3) 改善前の状態をもたらした原因（因果関係）

当部門は、大型受託案件の管理ルールに基づき、受注リスクの事前審議やプロジェクトの監視を実施している。現場から寄せられる改善提案や問合せ、提出された書類の不備を踏まえて部門によるブレインストーミングを行い、前述の課題が以下に起因していると考えた。

- 全社でルール化している範囲が狭い
プロジェクトの受注前のリスク管理、プロジェクト実施中のプロジェクト管理にルールやプロセスが集中しており、開発プロセスや品質管理の多くが事業に委ねられている。
- 2006 年以降改定を重ね、つぎはぎになっている
プロジェクトの経験や改善ニーズに基づき、随時改訂を重ねたために、適用条件や例外事項が入り組み複雑になっていた。また、具体的なテンプレートやフォーム類も都度追加したため、イントラでの掲載形態の一貫性が崩れ、ガイド、テンプレート、サンプルなどが散在し、参照しづらくなっていた。
- 事業の統廃合・再編により開発が多様になっている
組織の再編、グループ企業を含む事業の統廃合により、自社商品も受託開発も多様になっているが、開発形態の多様化にプロセスやルールが追従できていなかった。

(4) 変更内容・対応策

QMS として規程とプロセスを整備し、審議体等のチェック体制を整えたことにより、当社にはプロセスを遵守する文化ができています。そこで改善の主体を QMS の見直しとし、以下の施策を考案した。

- グループ会社を含めて、プロジェクトの管理規定を統一する
プロジェクトの管理に関係する会社規程を、グループ会社間で共通の取組みとなる

よう、グループ会社で改訂を行う。

- 規定を守るためのグループ共通ガイドラインを作成する
既存の QMS の中で上位概念となる部分を実施基準として制定し、基本的な活動の定義、用語の統一を図る。また現在の強みを残しつつ、開発工程や品質管理に必要な活動を盛り込む。
- 事業に合った QMS を再定義する
既存の QMS から各事業の特性に合った QMS を再定義し、適用し易く使い勝手の良い QMS を確立する。

(5) 変更や対応策の実施内容

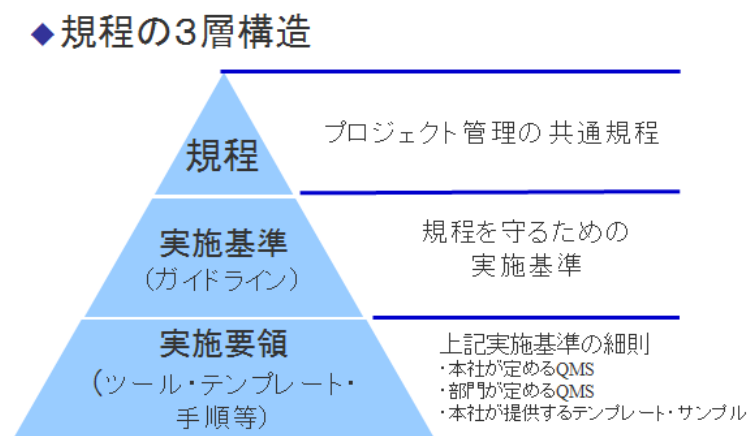
上記施策を実施するため、経営承認を得た上で、全社的な委員会を発足した。委員会の特徴は以下のとおりである。

- 委員長は役員とする
- 委員は、開発系の事業責任者で構成する
- テーマ別の分科会を立上げ、現場に合った内容を検討する
- 分科会メンバは、各委員に自部門の実務者を推薦していただく

活動の推進及び QMS の再定義に際して、工夫した点を以下に示す。

- 分科会メンバの役割を明確にし、事務局が積極的に作業を行うことで現場の実務者の負荷を軽減する
 - 分科会メンバは、開発の実情を踏まえた判断を示す・意見を出す、現場で使っているテンプレートを提供する、分科会成果物のレビューを行う
 - 事務局は、集めたテンプレートや意見を踏まえて QMS の原案を作成し、次回の会合でメンバにレビューを依頼し、QMS 作成の実作業を行う
- 抜本的な構造の見直し 1：
規程・実施基準・実施要領の三層構造で QMS を再定義した【図 1】
 - 規定により強制力を持たせる
 - 基準以下は、柔軟な改訂を可能とし、変化に強くする
 - 要領は本社制定分＋事業別の QMS として、共通部分と事業独自部分を明確にする

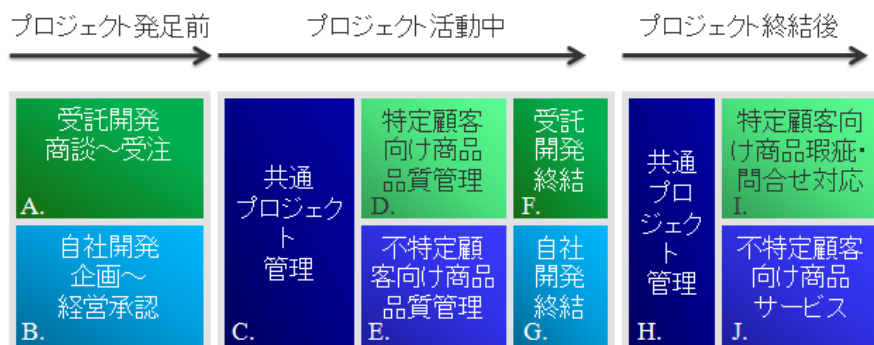
図1. 規程の階層図



- 抜本的な構造の見直し 2 :
- プロジェクトのライフサイクルと、案件の区分によるブロックの構成とした【図 2】
- プロジェクトのライフサイクルは、発足前、活動中、集結後とした
 - 案件の区分は、受託開発と自社開発の受注形態と、特定顧客向けと不特定顧客向けの商品の形態である

図2. 共通プロセスの概要

- ◆ プロジェクト・ライフサイクルと受託／自社開発、特定顧客／不特定顧客で区分けする



- 強みの継承 1 :
- 既存の QMS で有効に機能している強みの部分は残し、不足を補う【図 3】
- 既存の活動を前述の区分けされたブロックにマッピングした
 - 各ブロックの中で、事業が実践しているベストプラクティスや、品質向上に必要な活動を共通ルールとして組入れた

図3. 共通ルールの新設(赤枠の部分)



- 強みの継承 2 :
 全社共通と事業固有の QMS を共存させた
 - 管理水準引上げ、本社による監視に必須のものは全社共通
 - 事業の経験値や教訓がノウハウとして残るものは事業固有
 - 上記に基づき、文書を指定書式・推奨書式・自由書式に区別

(6) 変更・改善後の状態と効果

上記の活動により、全社の標準プロセスを再構築し資産化できた。2012 年 10 月 1 日に初版をリリースし、11 月および 2013 年 4 月、6 月に改訂を行っている。

- 共通ガイドラインの例
 - 実施基準
 - 実施要領 ブロック A 「受託開発 商談～受注」
 - 実施要領 ブロック B 「自社開発 企画～経営承認」
 - 実施要領 ブロック C・H 「共通プロジェクト管理」
 - 実施要領 ブロック D 「特定顧客向け商品 品質管理」
 - 実施要領 ブロック E 「不特定顧客向け商品 品質管理」
 - 実施要領 ブロック F 「受託開発 終結」
 - 実施要領 ブロック G 「自社開発 終結」

課題であった対顧客・対協力会社の契約リスクに対しては、ブロック A の「見積仕様書作成」とブロック C の「プロジェクト計画レビュー」により条件を明文化することで、コントロールができるようになった。

また、ブロック構成としたため、案件に応じた柔軟な経路が取れるようになった。具体的には、受託開発、自社商品開発以外の経路、「商品の受託開発」である。これは受注時には受託案件としてリスクを管理し、開発段階では不特定顧客向け商品として品質を作り込み、出荷は特定顧客に対して行う、というものである。

上記の他、移行判定や出荷判定のルールを共通化したことで品質担当者による品質状況の確認がしやすくなったことや、現場が実践している活動を取り入れたために普及や遵守が円滑に行えたことも、今回の改善活動の効果である。

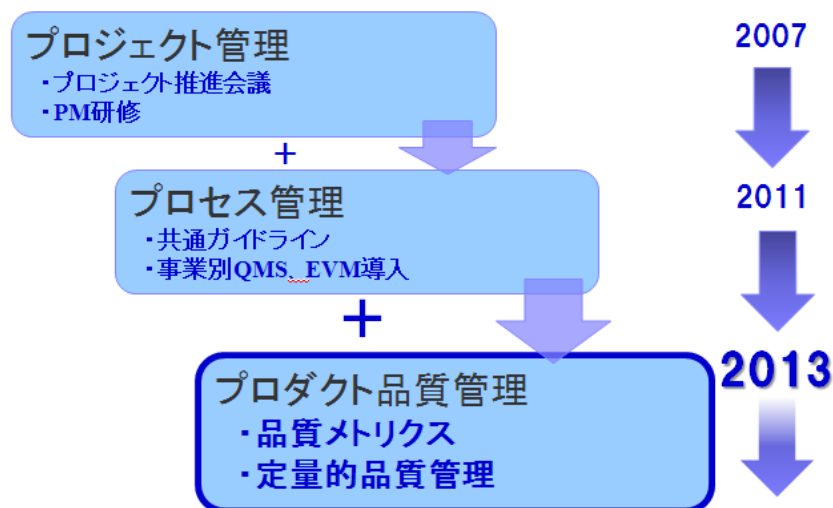
(7) 改善活動の妥当性確認

前述の効果の他、特筆すべき副次的な効果は、以下のとおりである。

- 改善意識の向上
 新しく再定義した QMS の適用を分科会メンバにて検討する過程で、プロジェクトの脅威となるリスクの軽減や品質の向上のために、何をすべきか、どこまでやるべきか、前向きで活発な意見が出された
- 現場の実務者同士のコミュニケーションの促進
 事業を超えて実務者同士が意見交換をし、互いの開発現場の実情や課題を知り得たことで、互いに面識ができ親近感が醸成された
- 説明会の実施を通じた委員会事務局と現場の交流
 抜本的な QMS の再構築による変更点及び新たに追加した活動内容を周知徹底するため、各現場の部課長以上を対象とする説明会を、それぞれの現場に出向く形で計 14 回行い、現場の生の声を聞く良い機会となった

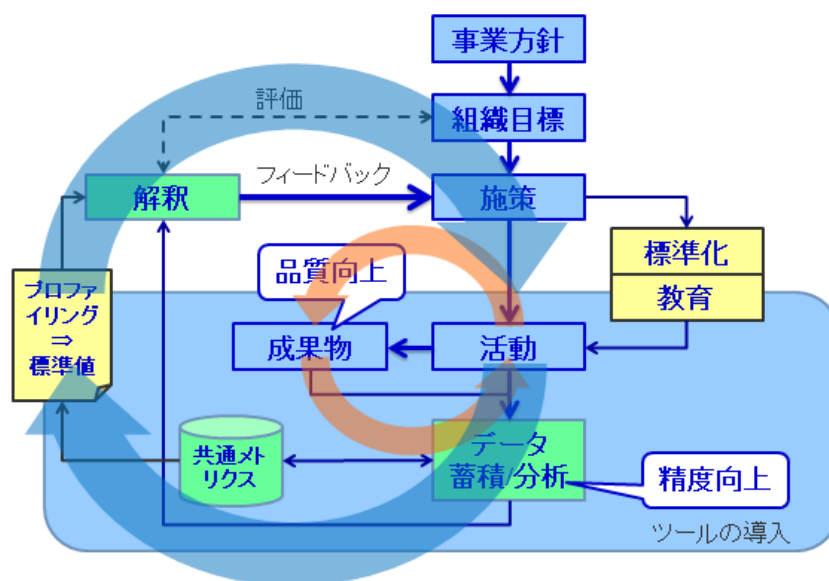
プロセス改善委員会は2012年末に解散したが、2013年2月より、新たに「品質改善委員会」を発足し活動を開始している。主な目的は品質向上であり、その手段は、プロダクトに対する品質メトリクスの導入及び定量的品質管理である。これまでの取組みの道筋は以下に示すとおり、CMMIにおける組織の成熟度の進化に重ねることができる。【図4】

図4. 2013年度の品質改善活動の位置づけ



品質改善委員会の活動形態は前述のプロセス改善委員会と別なアプローチを採っており、事業を主体としている。また活動の結果を全社のプロセスへ組入れ、品質向上を継続的な活動としたいと考えている。【図5】

図5. 品質メトリクス導入概念図



以上

1C3「派生開発手法導入に見る現場起点のプロセス組織展開事例」赤松康至(オムロン)

<タイトル>

派生開発手法導入に見る現場起点のプロセス組織展開事例

<サブタイトル>

<発表者>

氏名（ふりがな）：赤松康至（あかまつやすゆき）

所属：オムロン株式会社

<共同執筆者>

氏名（ふりがな）：筒井賢（つついけん）

所属：オムロンソフトウェア株式会社

・ 要点

現場起点のプロセス改善・組織展開の一事例

・ キーワード

現場起点、現場主体のプロセス導入

・ 想定する聴衆

SEPG、ソフトウェアエンジニア

・ 適用状況

☐多用されている段階、☒適用できる段階あるいは初めて適用する段階

☐適用するにはさらに検討を必要とする、☐着想の段階

☐その他（ ）

・ 適用可能性に関する制限

☐汎用性がある、

☒類似プロジェクトにも適用可：具体的な類似点（ ）

☐自プロジェクトのみ

(1) 背景

本件が対象とする開発部門の主軸事業は、顧客から品質、とりわけ信頼性が求められる中、弊社において比較的大規模なソフトウェアを開発している。また、事業ライフサイクルが成熟期のピークにあり、今後の事業環境上、品質確保は当然のことながら、さらなる開発効率化が求められている。

また、

- ・組織標準プロセスが確立されており、品質重視の比較的「重厚」なプロセスである。
- ・典型的な開発スタイルは、ベースとなるソフトウェアから改修を実施して製品を作り上げる、いわゆる派生型ソフトウェア開発であるが、組織プロセスと若干ギャップがあり、その隙間は現場で補う必要がある。具体的には、大きく変更される機能仕様の重点的レビュー、ソースコードの部分変更による影響範囲の見極め、などである。
- ・派生型開発により、ベースとなるソフトウェアに度重なる改造を加え、複雑化し、影響範囲の特定・品質確保の困難性が増大してきている。

(2) 改善前の状態および課題

本件の取り組みにおける課題は以下の通りである。

- ・一般的にありがちなトップダウン型な導入スタイルは極力避けること。
- ・品質、効率性の両面で、管理層および現場双方の納得性が得られ、自発的な継続となること。
- ・組織への導入・展開を着実にを行い、成果・達成感に結びつけること。

(3) 変更内容・対応策

・当該部門において、派生開発に特化した方法論である、XDDP(eXtreme Derivative Development Process)に着目し、導入を試みた。XDDPは(株)システムクリエイツの清水吉男氏が提唱している派生開発に特化した開発アプローチで、ベースとなるソースコードが存在することを前提としたプロセスである。影響範囲特定、抜け・漏れを防止する工夫や、上流工程での仕様の明確化に効果があり、当該部門が実現を目指している品質確保と開発の効率化の両立に有効と考えた。

上記取り組みの中で課題を解決するにあたり、以下の対応策を講じた。

- ・従来のトップダウン型に加え、ボトムアップ型のアプローチを併用。
- ・開発プロジェクトで活躍し、現場課題を熟知するリーダを現場推進リーダに任命。
- ・既存資産を熟知し、現場設計者に影響力のあるアーキテクトを中心に導入時の諸課題解決を現場と一体として取り組む。

従来の導入アプローチに上記対応策を $+\alpha$ で講じることにより、推進力が増加し、確実な組織展開が図れるとの仮説をもって XDDP の導入にあたった。

(4) 変更や対応策の実施内容

導入ステップ：

手法の導入は、主に以下の3ステップから構成した

- Step1：導入準備、試行プロジェクト実施、効果測定
- Step2：特定製品群への展開、組織プロセスへの反映
- Step3：組織展開

組織運営：

- ・ステアリングコミッティを組織化し月次レベルで状況の共有化、関係者の巻き込み、方向性の指示・確認を行った。継続して効果を確認し、目標・狙いの妥当性をこのステアリングコミッティにて確認した。
- ・展開にあたっては、組織プロセスへの反映、委託先を含めた関係者へのトレーニングを行い、落とし込みを徹底した。

加えて、工夫した点は以下の通りである。

- ・計画時に ODSC 分析を実施し、目的(Objectives)、成果物(Deliverables)、成功基準(Success Criteria)に分類することで、狙いや目標を明確にした。目標設定にあたっては、事業の目標・狙い、現場課題と結び付け、測る指標を定義した。
- ・比較的大きなプロセス変更を伴うため、極力納得性の得られるような方法で効果を確認できるようにした。具体的には、試行フェーズでは同一案件を同等のプロジェクト体制、同等スキルレベルの要員で並行して実施させ、効果の差異が極力純粋に手法による違いになるようにした。
- ・手法導入の初期ターゲット製品のアーキテクトが検討段階から参画し、早期から積極的な支援者となった。その結果、従来ありがちな「抵抗勢力」という高い障壁をなくし、また、このアーキテクトが他製品群のアーキテクトにも働きかけ、勉強会の実施などを通じて現場レベルで支持を獲得した。
- ・現場に近い推進リーダが実施状況・結果のヒアリングを行い、率直な意見を収集、対応することでさらに現場の納得感、支持を得た。

上記はいずれも現場に使い推進リーダが主体的に実行したことであり、SEPG など改善推進者は目標連鎖、組織プロセス定義、組織展開時のオリエンテーションなどプロセス改善の最低限のテクニカルな支援で活動を実現した。

(5) 変更・改善後の状態と効果、妥当性の確認

改善前の課題に対しては以下のように対応できた

- ・現場に近い推進リーダ、影響力のあるアーキテクトを含めた、現場主体の改善活動として進められた。
- ・事業課題である品質、開発効率面での定量的な効果を、ステアリングコミッティを通じて管理層に随時報告し、理解が得られた。
- ・勉強会、アーキテクトの技術指導、適用におけるヒアリング・アンケート実施と対策など実際に使う際のきめ細かい対応を通じて現場の納得が得られた。
- ・展開を計画した全てのプロジェクトで、プロセス順守面での指摘がほとんどなく手法が適用された。
- ・従来手法と比較した結果、開発規模あたりのテスト不具合件数が約 50%減、開発規模あたりの工数が約 20%減と、品質、効率両面で期待通りの効果が出た。

本件を通じて、現場に近い推進リーダおよび影響のあるアーキテクトに強い改善の意思を持っており、ボトムアップ型のアプローチも併用することで、自発的に改善のサイクルが回る可能性があることが分かった。あとは SEPG など改善推進者が少し背中を押すだけで良いという理想的な活動に一步近づいたと確信できた。

また、別施策においても同様のアプローチを実践し、確実な技術導入・プロセス改善に繋げていく。

1C4「実機レス情報交換会の活動の紹介」 平原嘉幸(東芝テック)

東芝テックでは、全社ソフト開発力強化活動の一環として、2009年から実機レス情報交換会を開始し、MFP 開発部門や POS 開発部門での実機レス開発環境の取組みの情報共有や今後についての議論を行っている。

本発表では、実機レス情報交換会の活動を紹介すると共に、MFP 開発部門や POS 開発部門の実機レス開発環境の利用シーンを紹介し、実機レス開発環境とはどういうものか、どう使われるのか、MFP 開発・POS 開発でどのような効果があったか等について紹介する。

<タイトル>

実機レス情報交換会の活動の紹介

<サブタイトル>

実機レス開発環境整備による開発効率化の実践とその効果の紹介

<発表者>

氏名（ふりがな）：平原 嘉幸（ひらはら よしゆき）

所属：東芝テック（株）

<共同執筆者>

氏名（ふりがな）：山平 喜文（やまひら よしふみ）、箕浦 悠介（みのうら ゆうすけ）、
稲垣 泰広（いながき やすひろ）

所属：東芝テック（株）

2A1「パッケージ開発プロセス改善による品質向上と生産性向上」松浦豪一(富士通マーケティング)

<タイトル>

パッケージ開発プロセス改善による品質向上と生産性向上

<サブタイトル>

品質データからのアジャイルに関する考察

<発表者>

氏名（ふりがな）： 松浦豪一（まつうらひでかず）

所属：株式会社富士通マーケティング ソリューション事業本部 GLOVIA 事業部 開発部

<共同執筆者>

氏名（ふりがな）：

所属：

・ 要点

アジャイル開発のプラクティスを取り入れ、短期間の繰返型で開発を進めると1回あたりの開発規模が減少することにより障害作込み件数が減少し、品質向上すると判断していた。

しかし、アジャイルのプラクティスを継続的に実施していくなかで、データを分析することで、品質向上や生産性を向上させる仕組みが判明した。

・ ソフトウェア開発スキルが一定レベル以上であれば、短期の繰返型で開発することにより、障害作り込みは減少し、開発スピードは向上する。

・ 開発スキルや設計手法に問題があると、予定期間内に終わらず、異常と検出できる。

・ 開発プロセスの教育、テストコードの共有、ソースコードのリファクタリングを継続的に取り組むことにより、プログラムの構造化が進み生産性が向上する。

その結果、開発者のスキルが多少低くても、分散開発であっても効果があるということについて品質データをもとに説明する。

・ キーワード

繰返型開発、品質向上、生産性向上、レガシーコード、構造化プログラミング、アジャイル

・ 想定する聴衆

SEPG, SE, 品質保証の方、ソフトウェア開発リーダー

・ 適用状況

☒ 多用されている段階、☐ 適用できる段階あるいは初めて適用する段階

☐ 適用するにはさらに検討を必要とする、☐ 着想の段階

☐ その他（ ）

・ 適用可能性に関する制限

☒ 汎用性がある、

☐ 類似プロジェクトにも適用可：具体的な類似点（ ）

☐ 自プロジェクトのみ

・発表内容

(1) 背景

私の所属する部署ではERPパッケージ(会計、固定資産、人事、給与)を20年以上提供するパッケージ部隊で、6カ月に1回のタイミングで定期的にパッケージをエンハンスしている。一般的にパッケージ開発は、継続的なエンハンス開発であるがゆえに長年の経験・ノウハウを持つSEに依存しやすい傾向にある。また、エンハンスごとに開発ボリュームが変化するため、常に同じ要員を確保することはできない。このような状況下において、当部門ではQCDに対するお客様の期待に応えつつ、環境の変化に耐えうる開発プロセスを構築する必要があった。

(2) 改善前の状態

過去の品質データを期間別に分析した結果、開発項目ごとの期間が1カ月以上になると、作業時間の変動が大きくなり、納期遅延が発生することが判明した。また、品質面では、クリティカルパスにかかわる開発項目において品質不良が発生することにより、全体の品質低下を引き起こしていることが分かった。さらに、クリティカルパスにかかわる開発項目において品質不良が発生する原因分析を行った結果、パッケージ開発の経験が浅い要員の開発部分で発生していることが判明した。設計時間をかけているにもかかわらず、設計を原因とする障害が発生している。

(3) 改善前の状態をもたらした原因（因果関係）

- ・過去1年間の開発項目のうち遅れが発生した項目を調査した結果、すべて開発期間が1ヶ月をこえる案件であった。開発期間が短い項目でも遅れは発生するが、予定より早く開発が完了する項目があった。
- ・1ヶ月をこえる案件について、テスト工程で発生して障害の検出すべき工程が、前工程であった。

(4) 変更内容・対応策

GLOVIA/MI V01L40の開発で以下の対応策を実施した。

タイムボックス制約付インクリメンタル開発を実施した。

★プログラミングからテストまでの期間を5日にした、繰返型開発に変更した。

★開発スキルの高いメンバを投入した。

実施後の結果（V01L30とV01L40の比較）表1参照

- 1) 開発スピードが1.4倍になった。
- 2) 開発中の障害検出率が0.2件/KS増加し、顧客発生障害が4.4%減少した。
- 3) 1人月あたりの開発ポイントが1.5倍になった。

成功者の共通点からプロセス化を実施した。

- ・開発項目を1人月以下に分解している。
- ・設計時に状態遷移図、要因分析表を作成している。
- ・テストセットを作成し、繰り返し利用している。

(5) 変更や対応策の実施内容

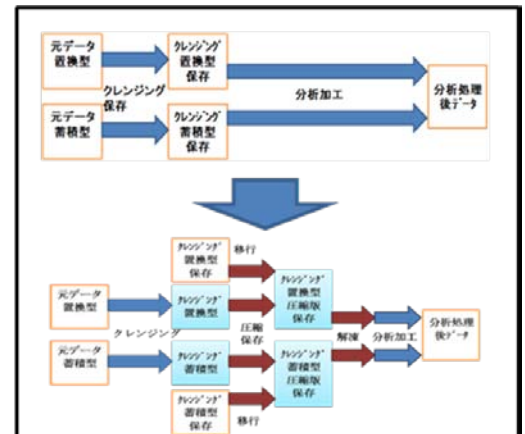
- 1) 成功者の共通点を開発プロセスとして規定した。
- 2) 定期的な進捗管理で状況をチェックするようにした。
- 3) 開発スキルの高いメンバを放出し、新規メンバを投入した。

【実践例】状態遷移図と要因分析表

「データの取込み機能に圧縮機能を追加する」を例にして説明する。開発項目に合わせで状態遷移図を変更した場合、以下の単位で開発作業を分割することができる。(図3参照)

- データの圧縮コマンド
- データの解凍コマンド
- 置換型データクレンジング処理の切り離し
- 蓄積型データクレンジング処理の切り離し
- 置換型データ圧縮保存
- 蓄積型データ圧縮保存
- 置換型データ解凍
- 蓄積型データ解凍
- 移行ツール

次に、設計時にテスト項目設計書である要因分析表を作成する方式に変更した。本方式により、品質向上させる方法について説明する。組合せパターンの検証するために、要因分析表を作成し、プログラミング前に検証するようにした。シミュレーションすることで組合せパターンの漏れがないことを検証できるため、品質が向上する。また、要因分析表では最小の組合せパターン設計し、プログラミングで共通化するように指導した。(図4参照) 最小パターンで設計した場合、A-2, B-1, C-2 と A-2, B-2, C-1 はテストが不要となる。最小組合せで実現するために、圧縮処理を共通化するようにプログラミングすることを促す。設計作業が定着し、プログラムの構造化が進む。結果的に小規模で機能追加できるようになり、生産性が向上する。



管理番号		機能名				
要因	項目	A	B	C	D	E
		コマンド	データ型	圧縮処理	解凍処理	
因子	1	取込みコマンド	蓄積型	正常終了	正常終了	
	2	移行ツール	置換型	異常終了	異常終了	
	3					

組合せ条件(要因/因子の番号をカンマで区切って1条件/1行で記入。複数因子は"/"で区切って記入例: 要因Aと要因Bと要因Cを組合せる→A,B,C、要因Aの因子1,3と要因Bを組合せる→A-1,3-B)

1	A-1,B,C-1,D
2	A-1,B-1,C-2
3	A-2,B-1,C-1
4	A-2,B-2,C-2

図4 要因分析表

【実践例】スケジュール管理

クリティカルパスの問題により、納期遅延や品質不良を発生させないために、スケジュールを組み換える方式を考案した。優先度設定とスケジュールの組み換えについて説明する。

前述した機能について開発者3名で開発する場合、機能単位や階層単位で並行開発をする。(図5参照) 例えば、機能単位で開発した場合、共通化しないと同一機能を重複して開発し、規模を増加させるリスクがある。開発中に共通化しようとするれば、処理の待ち合わせが必要になり、開発効率が低下するリスクがある。また、移行ツールのように他機能の実装に依存する機能は、設計当初には仕様の完成度が低く、並行開発した場合に初期段階の開発部分に品質不良を作り込むリスクがある。

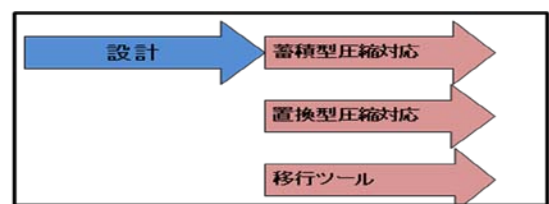


図5 ボトルネックが発生しやすい並行開発

上記のようなリスクをなくし、効率良く開発する方式を考案した。状態遷移図で分割した項目の内容を加味して優先度を設定してスケジュールする。（図6参照）ボトルネックになることが予想される共通機能は、最初に開発する。例では、圧縮・解凍コマンドである。逆に他の開発項目が決まってから開発した方が良い機能は着手をできるだけ遅らせる。（参考文献2）具体的には移行ツールである。状態遷移図と開発項目の関連を見極めたスケジュールとすることで、開発中のリスクを低減し、納期遅延や品質不良を減少させる。

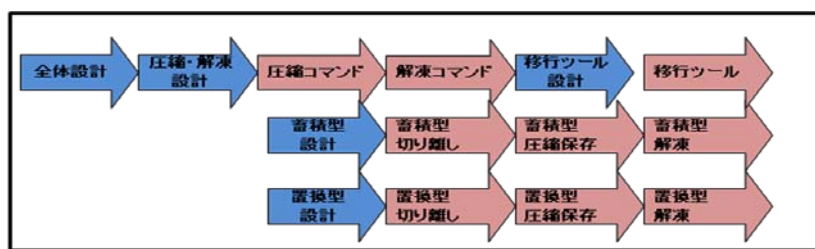


図6 クリティカルパスを優先したスケジュール

(6) 変更・改善後の状態と効果

実施後の結果（V01L30とV01L50の比較）表1参照

- 1) 開発スピードが0.9倍になった。
- 2) 開発中の障害検出率が8.9件/KS増加し、顧客発生障害が6.4%減少した。
- 3) 1人月あたりの開発ポイントが1.5倍になった。

本方式により、生産性が向上し、出荷後検出障害が減少した。GLOVIA/MIのV01L30とV01L50を比較した効果を算出すると、3,680万円のコスト削減になる。（表1参照）

【障害減少】開発規模(72.6KS)×出荷後検出障害率の差異(0.30件/KS-0.11件/KS)=14件

【コスト削減】ポイント(354pt+14pt)×1ptあたりの開発工数差異(0.29人月/pt-0.19人月/pt)×100万=3,680万円

※計算の基礎情報 障害1件あたりポイントは1pt、1人月あたりの単価を100万円と仮定する。

表1 GLOVIA/MIの開発品質データ

VL	Pt*1	開発規模	開発工数	速度	工数/pt	作込障害率	出荷後検出障害	テスト率
V01L30	167pt	59.0KS	49.1人月	1.2KS/人月	0.29人月/pt	12.8件/KS	0.3件/KS	150件/KS
V01L40	272pt	89.7KS	52.4人月	1.7KS/人月	0.19人月/pt	13.0件/KS	0.17件/KS	188件/KS
V01L50	354pt	72.6KS	69人月	1.1KS/人月	0.19人月/pt	21.7件/KS	0.11件/KS	449件/KS

*1 ポイントはリピーテーションの回数を表す。

(7) 改善活動の妥当性確認

★考察1 開発中の障害検出が8.9件/KS増加した。

予定どおりに進まない開発案件があり、障害検出率が高いことがわかった。

10KS ソースレビュー検出障害38.7件/KS

14KS ソースレビュー検出障害28.7件/KS

本案件が、他開発と同等な10件/KSであれば、全体として9.3件/KSとなる。

逆にソースレビューで検出できなかった場合、顧客で1%の障害が検出されると、

0.17件/KSになり、V01L30と同等の品質になっていた。

また、以下の特徴があることがわかった。

- a) 上記の案件は、設計工程のコスト比重が高く、プログラミングに記述が多くかかれていた。状態遷移図がきちんと書かれていないため、影響範囲を考慮した設計

が不足していた。開発項目の分割も適切でなかった。

→設計工程は、見積工程の25%になるようにスケジュールし、それ以上かかっているようであれば、品質に問題があると判断して、品質改善策を実施するようにマネジメントしている。

- b) テスト設計において、境界値を設定したテストが不足していることが判明した。
→要因分析表の作成方法の教育を実施し、開発中に作成した分析表をレビューすることで、設計段階の品質向上をはかっている。

【考察1 結論】

→品質の悪い開発項目を早期に検出し、対策を実施したことで顧客発生障害を未然に防ぐことができた。

→問題分析により、開発手法の教育及びOJTの方式を確立することができた。

★考察2 開発スピードは低下しているのに、開発ポイント率（生産性）は向上している
V01L30(0.35KS/pt)/V01L40(0.32KS/pt)に比べ、V01L50(0.20KS/pt)は、1pt 開発するための開発規模が減少している。

→V01L40の開発でのリファクタリングにより、プログラムの構造化が進んだ。

※V01L10/V01L20/L30の障害がV01L40出荷後減少している。（過去レベルの品質が改善している）

V01L30 V01L10:0.04 件/KS V01L20:0.02 件/KS

V01L40 V01L10:0.00 件/KS V01L20:0.00 件/KS V01L30:0.03 件/KS

V01L50 V01L10:0.00 件/KS V01L20:0.00 件/KS V01L30:0.00 件/KS

【考察2 結論】

→設計手法の統一、プログラム構造の改善により、生産性が向上している。

→リファクタリングにより、過去障害を減少させることにより、品質が向上している。

★考察3 設計手法の改善、テスト自動化の効果

品質データの変異のみで検証した場合、設計手法の改善やテスト自動化の効果を検証することは難しい。他の変動要素を排除することができないからである。

開発中に検出できなかった、顧客発生障害を分析し、プラクティスとの関連を考察した。

要因分析表を使っても、組み合わせパターンの漏れの障害が検出された。

開発時点の問題は以下のとおりであった。

- 操作画面の機能追加テストにおいて、変更前の動作を保証するテストを設計していた。

→画面の操作環境を因子としてあげていたが、具体的な選択項目の内容が記載されていなかった。見る人によって捕らえ方のばらつきができてしまい、テスト項目のレビューで指摘できていない。設計者と作業員においても同じように捕らえ方にばらつきが発生してしまう。

- 操作画面には、リグレーションテスト（回帰テスト）が用意されていなかった。
→旧機能と同じ動作をすることがテストされていなかった。手動でテストしている。

顧客発生障害が発生していないコンポーネントにおいては、具体的な値を指定してテストを実施しており、結果確認までを自動化している。その結果、同様の問題が発生していないのである。発生している障害にはそれぞれ理由があるが、施策が機能していないこ

とが明確になる。逆説的だが検出件数が減少しているのは、設計手法、テスト自動化の効果であると推測することができる。

【考察 3 結論】

- 設計手法の改善、テスト自動化は品質向上に寄与している。
- あいまいな分析や手動のテストは、作業者によってばらつきが発生しやすい。
自動テストはコストを改善するのではなく、品質を改善する。

(8) アジャイル開発に関する考察

- ・分散開発でも適用可能
→3拠点での開発を実施しているが、生産性、品質に効果がある。
- ・開発スキルを育成しながらでも進められる
→マネジメント、設計、開発手法を統合して進めることで、高スキルでないメンバーであっても育成しながら進めることができる。繰返型なので、ふりかえりをする
ことで成長を促すことができる。
- ・品質不良を未然に検出することが可能であり、マネジメントが働きやすくなる。

(9) その他

今回の説明範囲にはないが施策が大きく2点ある。

- ・CMMI とアジャイルの融合
当プロジェクトは、CMMI Dev1.2 において、レベル3の範囲をカバーしている。
アジャイルとCMMI を組み合わせてやることによって、品質向上している部分がある。
- ・開発項目選定における狩野モデルの適用
品質に関して、障害がある・なしの範囲だけでなく、製品の魅力的品質や当たり前品質
を分析することにより、満足度を向上させる手法をあわせて適用している。

2A2「産学連携によるコードレビュー改善事例」佐藤美和(三菱電機)

<タイトル>

産学連携によるコードレビュー改善事例

<サブタイトル>

<発表者>

氏名（ふりがな）： 佐藤 美和（さとう みわ）

所属：三菱電機株式会社

<共同執筆者>

氏名（ふりがな）： 細谷 泰夫（ほそたに やすお）

所属：三菱電機株式会社

氏名（ふりがな）： 吉岡 克浩（よしおか かつひろ）

所属：三菱電機株式会社

氏名（ふりがな）： 白川 智也（しらかわ ともや）

所属：三菱電機株式会社

氏名（ふりがな）： 森崎 修司（もりさき しゅうじ）

所属：静岡大学

・ 要点

検出したいコードの問題の性質に着目し、いつ、どのような問題を検出したいかの合意形成と対応するレビュー手法について検討しガイドライン化を行った。

・ キーワード

産学連携、コードレビュー、アジャイルインスペクション、ウォークスルー、インスペクション

・ 想定する聴衆

組織マネージャ、プロジェクトマネージャ、QA エンジニア、SEPG、開発者

・ 適用状況

☐ 多用されている段階、☒ 適用できる段階あるいは初めて適用する段階

☐ 適用するにはさらに検討を必要とする、☐ 着想の段階

☐ その他（ ）

・ 適用可能性に関する制限

☒ 汎用性がある、

☐ 類似プロジェクトにも適用可：具体的な類似点（ ）

☐ 自プロジェクトのみ

・ 発表内容

(1) 背景

SPI Japan 2011 で発表した「産学連携によるデザインレビュー改善事例」で紹介したレビューガイドラインは組織においても設計ドキュメントのレビュー手法として適用を進めている。更に改善の範囲をコードに広げるために、前回の設計ドキュメントのレビュー手法と同じ枠組みで産学連携を行いコードレビューガイドラインを作成した。

本発表では、以下について事例紹介を行う。

(a) 産学連携の継続的な取り組み

(b) コードレビューの改善

(2) 改善前の状態

各プロジェクトでは、限られた時間でコードレビューを実施しているが、検出容易な指摘に偏って、重要な欠陥の見逃しが発生したり、指摘された改善事項が修正コストの大きさから修正されないケースがあり、コードレビューの投入コストに見合う効果を得ていなかった。

(3) 改善前の状態をもたらした原因（因果関係）

エラー！参照元が見つかりません。は実際のプロジェクトにおけるコードレビューの指摘を分類した結果であるが、指摘の大半がコーディング規約、違反に関するものである。このような傾向のコードレビューをコーディング終盤に実施した場合、指摘の大半の修正が見送られる可能性が高くなる。

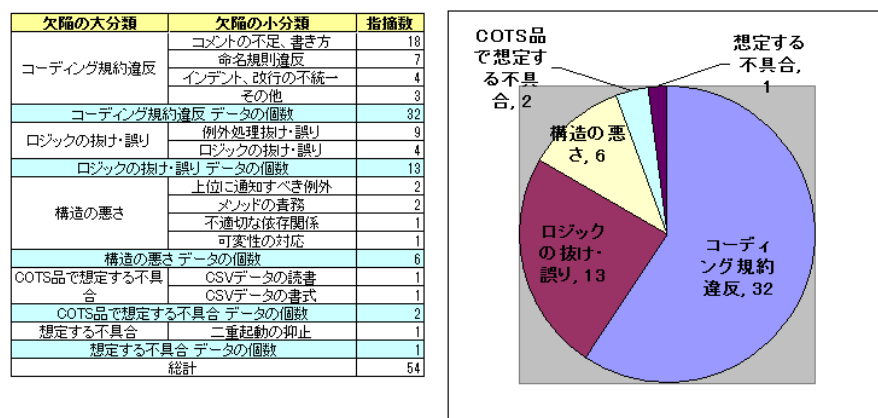


図 1 コードレビューで指摘した欠陥の分類

また、過去の不具合や COTS 品の知識を持つ有識者がコーディング規約、違反のような分類の指摘に終始してしまい本来指摘してほしい重要な欠陥検出に至らないケースがある。このような事象によりコードレビューの投資対効果が低下している。

(4) 変更内容・対応策

設計ドキュメントのレビューガイドライン^{*1)}と同様の時系列でレビュー技法を使い分ける。**エラー！参照元が見つかりません。**で示すように、コード作成前に、改善可能あるいは検出可能な時期にレビュー観点・形態を明確にしてレビューを実施する。レビューを計画してか

ら、実施することで、レビュー効率と価値を高めることができる。

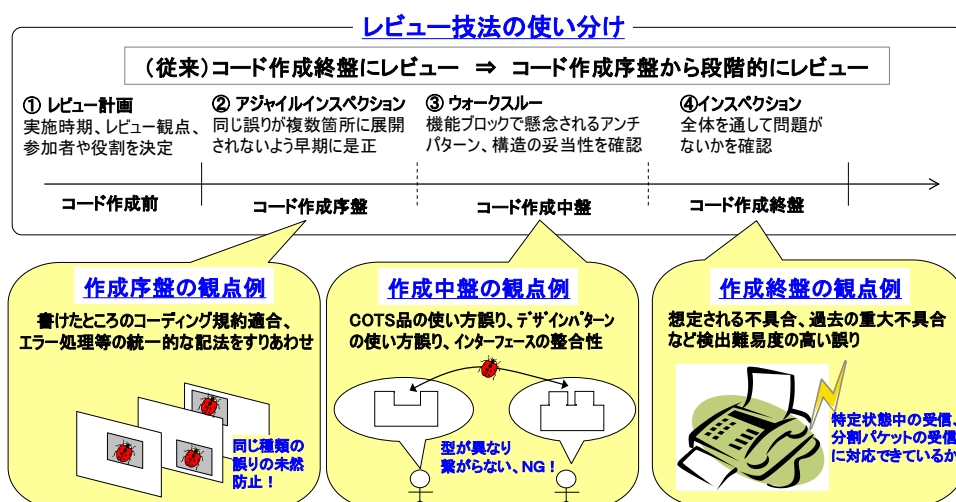


図 2 手法を用いたコードレビュー計画例

レビュー観点については、エラー！参照元が見つかりません。に示すように、指摘すべき欠陥を「誤りか、誤りでないか」「粒度の大きさ」という観点で四象限に分類した。レビュー観点を分類することによって、指摘すべき欠陥の性質を共有し合意することが可能になる。

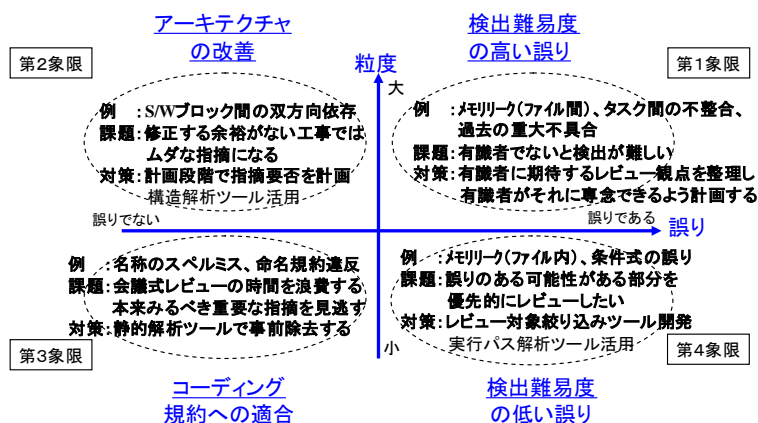


図 3 指摘すべき欠陥の分類

これらレビュー観点の実施時期については、第1象限、第4象限の欠陥は不具合動作を起こす原因となるものであり、ある程度作り込まれたコードから検出される場合が多いため、コード作成中盤から終盤にかけて行うことが望ましい。

また、第2象限、第3象限についての欠陥は命名規則違反など、品質や保守性の観点で影響があるものなので、コード作成の序盤から中盤にかけて検出し、コードに反映させていくことが望ましい。

このように、いつどのような性質の欠陥を検出するかを合意することにより、それぞれのレビューでの目的を明確にすることができる。

- i. 文書作成序盤では、これから記載する箇所に展開可能な指摘を抽出し、文書の内部品質を構造させるレビュー
- ii. 文書作成中盤では、設計が不明確な部分、重要な部分などピンポイントでレビュー対象を設定、内容を擦り合わせることによる設計の質を高める議論中心のレビュー
- iii. 文書作成終盤では、レビューアに「ユーザ」、「テスト」、「後工程担当者」などの役割を設定し、さまざまな観点で網羅的に誤りを検出するレビュー

				レビュー計画入力			
NO	シナリオ名	実施要否の判断基準	象限	実施 要否	序盤のアジャイル インスペクション	中盤のウォークスルー	終盤のインスペクション
1	想定する不具合	確認すべき不具合パターンがある。	1	○			○
2	COTS品で想定する不具合	使用しているCOTS品で確認すべき不具合パターンがある。	1,4	○		○	
3	構造の悪さ	・構造の悪さを修正するリソース、開発期間がある。 ・保守性を向上すべき要求がある。 例) 複雑さに起因する品質問題、効率の悪化、 今後の流用頻度が高い	2,3	○	○		
4	設計ポリシー違反	SPL(ソフトウェア・プロダクト・ライン)等の明文化された設計ポリシーがある。	2	○		○	
5	仕様との不整合	・実施すべき。 ・テストとの住み分けも考慮する。	1,4	○			○
6	ロジックの抜け・誤り	・実施すべき。	1,4	○		○	
7	コーディングルール違反	・実施すべき。	3	○	○		
8	変更影響箇所の漏れ・誤り	・流用開発である。	1,2,3,4	○		○	

図5 レビューシナリオを用いたレビュー計画

(6) 変更・改善後の状態と効果

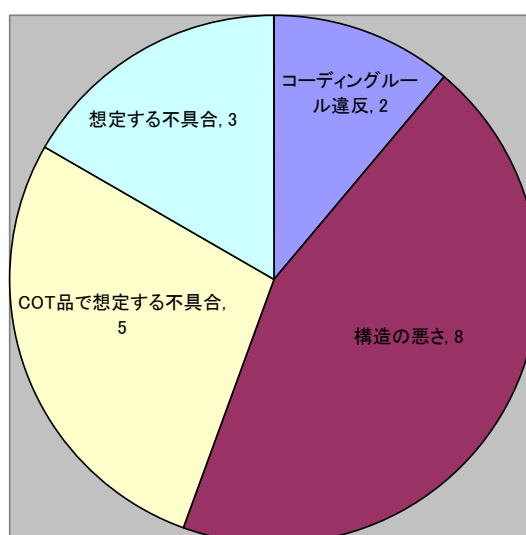
(7) 改善活動の妥当性確認

手法を試行プロジェクトに適用し、有効性の評価を行った。試行プロジェクトはC#の新規作成のツールでライン数は約1KLである。コード作成の序盤から本コードレビュープロセスに従って3回のレビューを実施した。図6はレビュー毎に適用するシナリオに従った具体的なレビュー観点をレビュー実施前にレビューイ、レビューアで議論し抽出した結果である。

NO	タイミング	シナリオ	レビュー観点
1	序盤	コーディングルール違反	・命名規則 ・クラスとファイルの関係 ・その他記載方法
2		構造の悪さ	・複雑さ ・巨大なクラス ・可視性の適切さ ・責務の適切さ ・定石でない
3	中盤	COT品で想定する不具合	・市販ライブラリの使い方 ・Excelオブジェクトの使い方 ・CSVの扱い
4	終盤	想定する不具合	・文字コード、改行コードの不整合 ・ファイルフォーマットとコードの不整合 ・データの量が増加することによる不具合 ・2重起動したときの動作
5		構造の悪さ	・全体の中で複雑なコードを確認

図6 試行プロジェクトにおけるレビュー観点

試行の結果、それぞれのレビューにおいて計画したシナリオ、レビュー観点到に合致する欠陥を指摘することが出来た。各シナリオについての考察としては、「コーディングルール違反」のシナリオの欠陥数が従来のプロジェクトに比べて大きく割合が減少した。このシナリオは、1つのクラスを作成したタイミングでレビューを実施し、クラスの命名規則について欠陥を2件指摘した。その結果、以降で作成されたクラスに関しては開発者の命名規則逸脱が無くなり、狙い通り欠陥の作り込みを減らすことが出来た。「構造の複雑さ」に関しては、2つクラスが作成された段階でレビューを実施し、エラー処理の実施方法の欠陥が複数見つかった。これはもっと早い段階でエラー処理方法のレビューを行うことで更に減らすことが出来たと言える。また、従来は検出数が少なかった「COT品で想定する不具合」、「想定する不具合」に属する重大な欠陥が全体の50%近くを占めており、価値の高いレビューを実施できたと言える。



2A3「アジャイル開発における品質保証部門によるシステムテストのアプローチ」永田敦(ソニー)

<タイトル>アジャイル開発における品質保証部門によるシステムテストのアプローチ

<サブタイトル>アジャイルチームに寄り添う品質保証部門の考え方

<発表者>

氏名 永田 敦（ながた あつし）:

所属：ソニー株式会社

<共同執筆者>

氏名（ふりがな）:

所属：

・ 要点 従来の考え方，プロセスを監査するとか，出された要求仕様書をもとにシステムテストをして品質を保証するという考えでは，アジャイル開発に対してどうもしっくりこない。しかし，アジャイル開発の品質に対する考えを理解し，アジャイル開発チームに入り込むようにしていく，そして開発チームもそれを受け入れたとき，QCDに対してより大きな効果があることがわかった

・ キーワード

アジャイルテスト，アジャイル開発，スクラム，プロダクトオーナー，チーム，スプリント

・ 想定する聴衆

アジャイル開発に興味のある方，アジャイルテストに興味のある方，ソフトウェアテストエンジニア，品質保証の方，ソフトウェアエンジニア、

・ 適用状況

☐多用されている段階、☒適用できる段階あるいは初めて適用する段階

☐適用するにはさらに検討を必要とする、☐着想の段階

☐その他（ ）

・ 適用可能性に関する制限

☐汎用性がある、

☒類似プロジェクトにも適用可：具体的な類似点（ ）

☐自プロジェクトのみ

・発表内容

(1) 背景

アジャイル開発が現場に導入され始めている。開発現場は、何とか変えていかなければならないという策の一つとしてアジャイル開発を選択している。アジャイルプラクティスについての知識はある程度勉強してツールや環境を用意し、今回はアジャイル開発をやりますと宣言して開発をはじめてしまう。

(2) 改善前の状態

しかし、アジャイル開発における品質に対する考え、特にシステムテストレベルの品質についてはあまり意識されていない。設計者に聞いてみると概念的には理解をするが、それを実際のプラクティスでどのように行動するのかということが明確になっていない。アジャイルプラクティスは、品質を良くしていくという仕組みがある。しかし、その仕組みを積極的に使うモチベーションは持たず、そのプラクティスを行うことがアジャイルであるという程度で、実行していた。そのような場合、アジャイルで開発をやっていると言っているが、イテレーションごとのテストがあまりできていないことが多いため、イテレーションが終わったところでシステムテストで行うと多くのバグを出してしまう。そのため、バグを直すイテレーションが始まるが、もし結合度の高いコードであれば逆にデグレードを起し、手戻りが手戻りを呼んでイテレーションが止まらなくなって出荷が伸びてしまう。

(3) 改善前の状態をもたらした原因（因果関係）

このようなことになる原因は、アジャイル開発の品質に対する考え方を理解していないところにある。アジャイル開発を進める人たちが最も重視するのは、“ソフトウェアの価値”である。そして、アジャイルプラクティスを調べてみると、品質を高めるための行動、もしくはその行動の改善の実践を行う仕組みを持っている。スタンドアップミーティングも毎日改善するためのきっかけになると考える。価値を生むためには、その仕組みを理解して実践していかなければならない。また実践してすぐにはじめから成果が出るものではないので訓練が必要である。品質は価値を生む一つの要素のとしてあらわしているが、アジャイル開発の中心にあるものと強調されてはいない。従って、アジャイルプラクティスは、開発を進捗をさせる道具として行われる。

もう一つは、QA のアジャイル開発に対する知識と認識の不足である。なお、QA の仕事がプロセス監査である組織がある。これも品質に大きく影響するが、今回は、QA でテストするケースについて述べていきたいので、プロセス監査は議論のスコープからは外す。

Lisa Crispin が書いているアジャイルテストングでは、システムテストはイテレーションの終わった後に位置付けてある。内部品質が悪い場合、(2)で述べたようにイテレーション後に多くのバグを出してそれを修正することになる。実はそれでは、QA においては今までのウォータフォールとなんら変わらない。そしてさらに困るのが、アジャイル開発にしたために、テスト分析／設計をするためのテストベースの情報が“もらえないこと”である。その“開発の輪に”入っていかないと進捗さえ分からなくなってしまう。受け身で待っていても必要な情報は入ってこない。従来のままでは、品質保証としての評価は難しくなってくる。

一方で、設計側も品質保証部門に対して壁を作ってしまうことがある。設計側からしてみると、QA はいろいろな彼らにとっては必要のない仕事と思えるような情報の要求を次々に出してくる部署と感じている。よく考えれば、その要求は結局、設計活動の改善には有用なものかもしれないものでさえも、設計活動とは結びつかない余計な仕事を要求してきたと思い、なるべく仕事を増やさないように QA に対してガードをしてしまう振る舞いをしてしまう。ますます壁が高くなってしまふ。

(4) 変更内容・対応策

- ① QA と設計チームが一緒に行動する様、両者をマインドセットをした。

両者の壁を取り除くためには、まずマインドセットが必要と感じた。QA チームに対してはアジャイルテストにおける役割を説明した。設計チームが考えている価値観を説明し理解してもらうようにした。

- ② バグの管理方法を変えた

QA は以前は独自のバグ管理ツールを持っていて、設計にそれを使わせていた。

バグの管理を設計のバックログ管理を話してしまうと、バグの情報がバックログに反映しにくくなってしまう。しかしバグを直す行為は設計のタスクであり、結局はデバッグ行為で工数を使い、予定されていたバックログが消化できなくなる。品質が悪ければ、バグは予定している“価値”を組み込むべきバックログバックログとは桁違いの数で増えていき、設計者は一生懸命働いているが実質的な進捗が滞ることになる。つまりバグもバックログとして、機能を組み込むバックログと同等として扱い管理していかないと、納期を守れないリスクが早期にわかりにくくなる。

- ③ QA におけるプロダクトバックログへのフィードバック

プロダクトバックログを QA がいじるということとはあまり考えられていなかった。しかし、QA は顧客観点で同じドメインの製品を評価しており、プロダクトバックログのもとになるユーザーストーリーを補完する詳細なシナリオや例外条件の情報を持っている。それを設計の仕様であるプロダクトバックログに反映することによって、より高い品質をはじめから設計で組み込むことが期待できる。

- ④ QA の見積もりへの参加

アジャイルプラクティスを共有する一つとして、QA がプランニングポーカーにも参加した。見積もりも本来は設計だけでやるものだと考えられている。その作業に対するリスクをどう見積もるかも見積もりの大きな要素となる。QA も障害事例の経験から設計とは違うリスクを見つけ出すことにより、設計だけの見積もりを補完することが期待できる。

(5) 変更や対応策の実施内容

- ① QA と設計チームが一緒に行動する様、両者をマインドセットをした。

テストチームは、テストによって、ステークホルダーに対して有用な、製品の品質情報を提供をする組織である。つまりサービス業である。QA が評価をする場合もそれは変わらない。QA は設計に対して有用な品質情報をフィードバックする役割を持っている。その手段として、アジャイル開発の場合は、設計チームの中に入り込んだ法が効率がよい。チーム内では、あたかも高速道路のように早い情報のやり取りが行われている。それは前に述べたように暗黙知の共有とそれをフックするツールやプラクティスに支えられている。QA チームもそれに身をゆだねて、設計と同じ世界、同じスピードで活動する。ツールやプラクティスを設計と同じように使い、チームの一員として情報を共有していく。朝会、振り返りやスプリントミーティングにも参加する。やろうと思えばいいやというほどの情報がそこには行き交っている。それを必要な情報を“積極的に”かつ”自立的に”“テストベースとして獲得していけばよいのだ。それをもとに、設計の活動と同期を取りながら、できるところから評価を行って、どこまで設計が品質を埋め込んだかを設計に対して正確にフィードバックしていくことが役割である。これがアジャイルテストである。ここで一番大切なのは、自律性と、目的を持ったフィードバックを行う姿勢である。フィードバックは、相手がどのようなフィードバックを求めているのかを理解することと、こちらが出す情報（フィードバック）はさらに相手からどのようなフィードバックを返してほしいかという目的を相手と合意していることが必

要である。そうでなければ、お互い期待したフィードバックをやり取りすることはできない。つまりコミュニケーションができないのである。

一方で、設計に対しても、QAがチームの一員として入っていくという姿勢を理解してもらい、それを受け入れるよう促した。

QAは、チームとして入り込むことをマネージャも含めて理解し合意した。しかし、それによるその行動を設計に説明していくことがうまくできなかった。どうしても今までのQAと設計との関係における言葉で表現してしまうので、その表現だけで設計は反発してしまった。設計の方は、“今回はアジャイル開発なので、QAが要求するいつものプロセスやイベントには合わない”という表現になり、アジャイル開発の本質的な考えをまだ理解していないQAにとっては、議論がかみ合わない状態になった。

② バグの管理方法

開発ツールとバグ管理ツールを別々にするのではなく、今回は、設計がアジャイル開発管理ツールとして使っているTFSを使うことを進めた。その狙いは、バグをバックログとして扱うことである。バグを分析して直していく活動も結局設計にとってはタスクであり、設計コーディングをする活動と同等に見えなければ実質的な仕事量がだせない。バグのバックログが増えていけば、消化するべきバックログが増える。チームのベロシティが変わらないならば、早くバグを直そうとすれば、本来の機能を埋め込むバックログはできなくなってしまい、次のイテレーションにまわる、つまり負債が増えることになる。バグが別のシステムで管理されていると、それが見えにくくなり、実質的な進捗の遅れの兆候を見落とし、手遅れになるリスクがある。バグをバックログとして扱い、情報を一元化することにより、リアルタイムにフィードバックをかけて手を打つことができる。アジャイル開発はその対応に適応して、時にはプロセスを変えることもおこなっていった。

③ プロダクトバックログへのフィードバック

アジャイル開発では、要求がユーザーストーリーから降りてきて、どうしても要求の詳細が仕様化されていないもしくは仕様として決まっていない場合が多い。今回も仕様がつめられていない部分はしばしばあった。一方で、今回はQAがドメイン知識とかなりの製品の振る舞いについての知識を、前のバージョンでの評価によって持っていた。今回開発する製品は、基本的には前のバージョンと振る舞いの仕様は同じものであるため、QAとしてはこの知識を使って評価を行っていく。それならば、その評価の情報をプロダクトバックログに反映させて、プロダクトバックログの仕様の品質を上げていくように促した。また、どのように評価するかという評価仕様を設計に説明することにして、評価観点の情報を設計に初めから盛り込むことを合意した。

④ QAの見積もりへの参加

アジャイルプラクティスを共有する一つとして、QAがプランニングポーカーにも参加した。

(6) 変更・改善後の状態と効果

① QAと設計チームが一緒に行動する様、両者をマインドセットをした

設計と同じ部屋で作業を行う。

朝会、スプリントミーティング、レトロスペクティブのほか、仕様のレビューにおいて、チームの一員となっている。QAと設計はお互いの仕事のアプローチを互いにrespectをもって理解し、信頼関係を結ぶことができた。それにより、以下の施策を自然に活動に取り込むことができた。

② バグの管理方法

QAがバグのバックログの管理プロセスをTFSでどのように行うかを提案した。開発のプロセ

スとの整合性を取った上で、開発のプロセスの一つとして組み込まれる形になった。QA は設計になるべく負担をかけないようにバグ管理プロセスをシンプルにした。その一方、必要なバグの情報を設計からフィードバックしてもらえるように、テンプレートを QA が提案し、バグのバックログの書き方のルールを決めて設計と合意した。

管理ツールを設計と QA で統一することにより、バグのバックログの増加によるストーリーポイントへの影響をリアルタイムにメトリクスとして取ることができる。これにより、ベロシティに対してバグを処理する割合と予定されていた機能と品質を埋め込むタスクを処理する割合が見えてくる。別の言い方をすれば、機能を埋め込むタスクに対するベロシティがわかり、真の進捗が見えてくる。それによって、そのベロシティを上げる努力、つまりバグを減らす改善と設計コーディングの改善の目標とモチベーションを持つことができる。アジャイルはその改善の対処をすぐ行えることが利点であるが、今回の活動でその仕組みづくりができた。

③ プロダクトバックログへのフィードバック

特に仕様に対しては、テスト観点での情報をまず口頭で共有してバックログへの反映をしたり、設計に促している。これは、テストをする前に、その情報を手締めから設計に組み込ませることであり、自然にWモデルの実践を行っていることと同等である。つまり、QA をアジャイルにうまくからませると、Wモデルが自然にできてくる。

設計と QA との信頼関係が出てお互いにそのフィードバックをリスペクトする。

さらに、口頭でのコミュニケーションも自然に出てくることで、暗黙的な情報が増えていき、仕様の変更などについての理解や、設計の進捗の理解が飛躍的に早くなる。つまり、設計者と同じコミュニケーションスピードを持つようになる。一方で QA はテスト仕様書に対しては、ドキュメントとして準備をすることにより、不明確性、曖昧性や漏れをなくしている。

④ QA の見積もりへの参加

QA がプランニングポーカーにも参加するようになった。見積もりをする際に、そのバックログの概要の説明があるので、QA は製品の動きと、そのバックログで果たそうとしている価値を理解することができる。QA はコード内部の詳細は分からなくても、同じドメイン製品の評価知識から、機能の振る舞いや評価観点からみたバックログの規模とリスクをつかむことができる。QA において、出した見積もりについての説明責任があるので、バックログにたいして QA なりに考えることになる。その際にバックログに対するリスク分析や、テスト分析を考えることになるので、QA にとってはよい訓練である。

また、その状態でプランニングポーカーを引いてみると、設計と同じ値が出たり、違ったりする。違っている場合は、説明を要求される。その議論に QA が入ることで、設計が考えているそのバックログの大きさ、設計のアプローチ、複雑さがわかる。また、QA がその値を出したことの評価観点での理由を話すことにより、設計者は新たなリスクを認識できたりしている。より多くの視点、視座からバックログの見積もりから得たストーリーポイントは、今後それをベースに開発規模やベロシティを出していく上において、設計、QA お互いに納得できるものになった。また、見積もりの際の議論を通じてバックログのリスクを設計と QA が共有することができたことは重要である。

⑤ メトリクスや評価判定基準の見直し

できるだけ設計と QA に負担をかけずに品質と進捗の情報をフィードバックするために、SQA チームがメトリクスの取得分析と報告を行うことになった。管理ツールが設計と QA で統合したことで、SQA でも情報が取得できるだけでなく、それを自動化していくこともされようとしている。

(7) 改善活動の妥当性確認

今まで、イテレーションが終わった後、つまり開発が終わった後に行っていた QA によるテストの 8 割を、イテレーション内で前倒しで行う計画にすることができた。

定量的にはとれていないが、QA がバックログや使用に対する指摘を設計に対して行い、W モデルの実行が自然に行われていた。

QA と設計との関係がよくなったことにより、職場に活気が出てきた。

ツールの共有化とリアルタイムに見えることが出来てきたことにより、改善をして品質と効率をあげ、ベロシティをあげていくことに集中するモチベーションが出てきた。これにより、費用対効果はまだプロジェクトが終わっていないので詳細には説明できないが、アジャイル開発の利点を生かした、改善活動を伴った開発活動が期待される。

課題としては、もしバグが多く発生した場合、機能（勝ち）を入れ込んでいるバックログがバグのバックログに埋もれて見えにくくなってしまい、進捗管理がやりにくくなる可能性がある。

以上

2A4「現場ですぐできる定量データ分析～予測モデルのゆるい作り方～」矢部智(NTT データ)

<タイトル>

現場ですぐできる定量データ分析～予測モデルのゆるい作り方～

<サブタイトル>

<発表者>

氏名（ふりがな）： 矢部 智（やべ さとし）

所属：NTT データ

<共同執筆者>

氏名（ふりがな）：木暮 雅樹（きぐれ まさき）

所属：NTT データ

<共同執筆者>

氏名（ふりがな）：大鶴 英佑（おおつる えいすけ）

所属：NTT データ

・ 要点

プロセス改善における定量データを用いた予測モデルの構築において、離散値を活用した事例と効果を照会する

・ キーワード

予測モデル、定量分析、プロセスの測定、現場の巻き込み

・ 想定する聴衆

SEPG

・ 適用状況

■多用されている段階、□適用できる段階あるいは初めて適用する段階

□適用するにはさらに検討を必要とする、□着想の段階

□その他（ ）

・ 適用可能性に関する制限

■汎用性がある、

□類似プロジェクトにも適用可：具体的な類似点（ ）

□自プロジェクトのみ

・発表内容

(1) 背景

ソフトウェア開発における定量データの利用方法の一つとして、開発プロセスの早い段階で得られた情報を元に後半または完了時の状況を予測する数式モデル（以下、予測モデルという）の構築がある。

NTT データではこれまでのプロセス改善の取り組みの中で品質やコストなどのパラメータに対して様々な予測モデルを構築してきた。

（SJ 2010 発表「CMMI 高成熟度における品質・工数予測モデル構築手法」）

予測モデルを利用することで今後起きうる問題について早い段階でアクションが打てるため、プロジェクトの運営がより安全になり、ビジネス上の成果も上がってきている。

(2) 改善前の状態

しかし、予測モデル構築については以下のような問題点があり、全面的な普及には至っていない。

1. 予測モデルの概念の理解
2. 分析に適した過去データの収集
3. モデル構築に不可欠な仮説の検討

このうちとくに時間がかかっているのが3の仮説検討であり、過去の取り組みでは予測モデル構築までに短い組織で8ヶ月、長いところでは2年近くかかっている。これではいくら予測モデルが役に立つと言っても、費用対効果で問題がある。

(3) 改善前の状態をもたらした原因（因果関係）

仮説検証が長引く原因としては以下のものがある。

1. 予測モデルで何を予測したいのかの問題認識の統一
2. 実データによる予測モデルの検証

1については検討の時間を短くするだけでは十分な議論ができず関係者の納得感が得られないというトレードオフがあるため、一律に短くすることは得策ではない。

2については短くしても質が落ちないのであれば可能な限り短くすることが望ましい。そこで我々はどのような仮説の立て方であればデータによる検証が短時間で終わり、なおかつ予測モデルとしての利用価値も保てるか、ということを検討し、実践したので報告する

(4) 変更内容・対応策

基本的な統計による予測モデルとしては、回帰分析がある。データから関連を見つけようとするためまず散布図を書いて、回帰分析を試してみるのではないだろうか。

我々の事例でも、いくつかの回帰式を活用しており、インプットとなる数字を入れるとリニアに予測値が出てくるのでプロジェクト管理に対して非常に相性がいいと評判である。

その一方で、目的変数や説明変数の組み合わせによっては、回帰分析の当てはまりが悪いこともある。仮説の筋はいいのに、なかなかモデルが構築できず、時間ばかりが過ぎてしまうこともあった。長い時間がかかるという印象を与えてしまうのは予測モデル普及の妨げにもなり、二重の意味で好ましいことではない。

そこで基礎統計の中でも離散値を使用した予測モデルの積極的な活用に着目した。

よくあることであるが、現実のプロセスで仮説をヒアリングしてみると、連続値の他に

「レビューアが有識者または初心者の場合でエラーの数が異なる」

「実装する機能の難易度が高い場合はエラーが出やすい」

連続的な数値以外の表現（離散的な値）を説明変数に含んだ形でかえってくることも多い。

また、予測した値も必ずしも連続値だけが歓迎されるわけではなく、ある事象の有無（たとえば「サービス開始後のバグの発生」）が予測できればよい、というニーズもあるため、目的変数が離散値のモデルも有用である。

そこでこれらの離散的な表現を予測モデルに使うため、グループ間の検定（ t 検定、分散分析等）やロジスティック回帰を仮説に取り入れることにした。

本発表では実際に離散値を用いて分析した予測モデルのケースを複数紹介する。

皆さんの予測モデル構築のヒントになれば幸いである。

(5) 変更や対応策の実施内容

ここでは4つのケースについて予測モデルの定義とプロジェクトにおける利用方法について解説する。

ケース1：設計書レビュー工数密度とシステムテストのバグ数

前半の工程である設計書のレビューにかかる工数密度（＝1 ページあたりのレビュー時間×レビューアの数）が多いグループと少ないグループで、システムテストのバグ数の傾向に違いがあるかを検定。

より厳しい品質を求める場合は多いグループのベースラインを、リスクが低いと見なせる場合は具区内グループのベースラインを使用してプロジェクトの工数を見積もるようにした。

品質に対するコストの取り方が計画時点で明確にできるというメリットがある。

ケース2：工程ごとの生産性とサービス後のバグの発生有無

過去にサービス後のバグを1件以上発生させたグループと、ゼロ件のグループについて、開発工程ごとの生産性（FP/人月）の分布に違いがあるかどうかを検定。

組織目標としては、ゼロ件を目指していたため、ゼロ件のデータを元に作成したベースラインでプロジェクト計画（工数、スケジュール）を策定するようにルール化した。

ベースラインを守るだけではもう一つの目標である生産性向上には不足するため、一部のテスト工程を自動化し、本質的な品質は担保したまま、工数削減を達成した。

ケース3：設計書レビュー密度とシステムテストのバグ密度

ケース1とよく似ている。検定の方法は同じだが使い方は異なっている。

品質を確保するため、バグが少ない方のグループから算出したレビュー密度（1 ページあたりの人数×時間）ベースラインよりも少なかった場合は、別途「レビューの質チェックリスト」を用いてレビューの内容が妥当であったかを検証し、不十分と判定したものは強化レビューを実施するというルールにした。

強化レビューの対象となったレビューは、組織における有識者から見ても納得のいくものであり、予測モデルは品質の向上ツールとして有効に機能している。

ケース4：設計書執筆者の製造経験の有無と製造工程で見つかる設計起因の手戻りエラー数

人員のスキルを連続値で表すのは難しいし、ランク付けをして数値化しても実績が必ずしも数値に比例しないことがある。ここではもっと簡単に製造経験の有無を使用して検定を行い、客観的に人員配置の根拠として使うようにした。

しかしスキルの数値化について考えると、人間をランク付けするのが本来の目的ではなく、有識者

が持っている知識を周囲に伝えることこそが大事である。どのような知識が品質に影響を与えているかについての考察も開始しているところである。

この組織では最初から離散値を意識したモデル構築を推奨したため、モデルの検証は2ヶ月で終わった。

(6) 変更・改善後の状態と効果

- ・ 予測モデルの構築時間が短縮され、最短で2ヶ月となった
- ・ 定性的な要素を説明変数にできることでメンバーの予測モデルに対する納得感が高まった
- ・ 予測値そのものを使う場合に加えて、予測値をもとに予防的なアクションが取られることが増えてきた
- ・ 予測精度は検定で有意が確認できているので、おおむね良好である

(7) 改善活動の妥当性確認

今後の課題：

- ・ 離散値を取り入れた場合のデータ収集や分析のノウハウのとりまとめを行い、教育コンテンツとして展開したい
- ・ 全体の期間は短くなったが、統計ソフトへのデータ投入と検定をする手間がかかる。ツール化を検討したい。
- ・ 目的変数と説明変数の組み合わせは手探りで行っているので、効率的なパターンを見つける手法を研究したい。

2B1「業務の中で自然に学ぶ仕掛け「プロセスの自己履行検証とピア・レビュー」」竹下千晶(デンソークリエイト)

<タイトル>

業務の中で自然に学ぶ仕掛け「プロセスの自己履行検証とピア・レビュー」

<サブタイトル>

—

<発表者>

氏名（ふりがな）：竹下 千晶（たけした ちあき）

所属：株式会社デンソークリエイト

<共同執筆者>

氏名（ふりがな）：木下 靖裕（きのした やすひろ）

所属：株式会社デンソークリエイト

氏名（ふりがな）：池永 直樹（いけなが なおき）

所属：株式会社デンソークリエイト

氏名（ふりがな）：山路 厚（やまじ あつし）

所属：デンソーテクノ株式会社

・ 要点

余分・やらされの活動を転換し、普段の業務の中で自然に若手技術者に技術を伝承して効果的に育成することで、組織全体の技術力を強化・向上する。

・ キーワード

スキル向上、人材育成、トレーニング指向、プロセス履行検証、ピア・レビュー

・ 想定する聴衆

プロジェクトマネージャ、開発リーダー、技術者育成を担当している方

・ 適用状況

☐多用されている段階、☐適用できる段階あるいは初めて適用する段階

☐適用するにはさらに検討を必要とする、☐着想の段階

■その他（適用が広がりつつある段階）

・ 適用可能性に関する制限

■汎用性がある、

☐類似プロジェクトにも適用可：具体的な類似点（ ）

☐自プロジェクトのみ

・発表内容

(1) 背景

以前より「人が育つこと」を第一義とした「トレーニング指向アプローチ」の下、プロセス改善に取り組み、プロセスの整備とそれに基づいた活動は定着してきた。「それで人は育ったのか？」という、近年、意に反して「若手技術者が育っていない」という現場の声を耳にするようになった。仕事のやり方は改善されてきているが、その中で若手技術者が学び育つようなやり方にはなっていないのではないかと気づき、業務の中で「人が育つ」やり方となる“仕掛け”が必要であると考えた。

(2) 改善前の状態

若手技術者を育成する方策の 1 つとして、若手技術者がエンジニアリングプロセス（以降、EP と略す）をもとに自己で履行検証する「EP 自己履行検証」に取り組んできた。「EP 自己履行検証」は、プロセスの履行を確認することに加え、上司や識者のノウハウや経験が埋め込まれたプロセスの理解を深め、自分のプロセスとして身に付けることでスキル向上する「自己トレーニング」を狙った活動である。しかし、現場では EP 自己履行検証はプロセスに書かれている作業を「やった／やっていない」で「○／×」を付ける表面的な確認に留まっており、自己トレーニングには至らず、若手技術者の育成に繋がっていなかった。

(3) 改善前の状態をもたらした原因（因果関係）

若手技術者にとってのプロセスは、既に用意されていることがほとんどである。そのため、仕事をする上で与えられた“ルール”や“手順”となっており、プロセスに書かれていることを実施することが目的となりやすい。また、「EP 自己履行検証」も、上司から「やれ」を言われて実施しており、やらされ感が強く余分な活動・仕組みと感じている若手技術者もいた。ルール・手順であるプロセスをなんとなく実行し、目的もよく分からずやらされ感いっぱい EP 自己履行検証を行うため、自己トレーニングになっていないのではないかと考えた。

(4) 変更内容・対応策

現場で若手技術者の育成のために実施される OJT をより効果的な育成の場とするために、ピア・レビュー（以降、レビューと略す）に着目し、レビューを OJT の中心に置いた ORT (On the Review Training) の定着に取り組んできた（SPI Japan2013 で発表）。EP 自己履行検証を自己トレーニングとなる活動に変化させる方策を検討しているうちに、レビューとの親和性に気付き、EP 自己履行検証をレビューの準備として位置づけることが、“仕掛け”になるのではないかとひらめいた。レビューに生きる・レビューの準備となることで、“余分”な活動ではなく、“やらされ感”を払拭でき、表面的な活動から狙いを満たす活動に変化させられるのではないかと考えた。

以降、レビューと EP 自己履行検証それぞれがスキル向上に繋がるメカニズムを(a)、(b)で述べ、両者の親和性から EP 自己履行検証をレビューの準備と位置づけた根拠を(c)で述べる。

(a) レビューによるスキル向上（ORT）

レビューの場では、レビューア（上司・識者）は自身の知識・経験・スキル等を活かして、様々な想定・仮説、つまりシミュレーションをすることで欠陥等を見つけて指摘する。若手技術者は、そこから自分にとっては未知・未経験の知見に触れることで疑似体験することができる。疑似体験を通して若手技術者は、レビューアのスキルを盗み取り、スキル向上する。

しかし、指摘が体裁等の軽微な欠陥や作成者自身でも見つけられるようなものばかりのレビ

ユーでは、レビューアの知見を発揮するようなシミュレーションは行われておらず、若手技術者のスキル向上には繋がりにくい。レビューを通して若手技術者が学ぶにはレビューアによるシミュレーションが不可欠である。そこで、レビューでシミュレーションが行われるには2つの条件があると考えた。

条件①：作成者自身が除去できる欠陥・不備が残っていないこと（少ないこと）

軽微な欠陥がレビュー時点で残存していると、そこに意識・エネルギーが集中してしまいシミュレーションに至らない。

条件②：作成者が設計成果等の結果に至った経緯・根拠を簡潔に正しく説明できること

レビューでは、作成者が、何をもとにどのように考えて解にたどり着いたのか、その考えられている範囲や経緯をレビューアが自分の知見と照らし合わせながらシミュレーションする。その題材が適切に・簡潔に説明されなければ、よいシミュレーションができない。レビューアはイラつき、発散しやすくなる。

これらの条件を満たすには、レビュー開始以前の取り組みが必要である。それができるようにORTを定着させる仕組みとして展開してきたが、実際には、これらの条件が満たせず、レビューでシミュレーションができていない場面が散見された。

(b) 自己トレーニングとなる EP 自己履行検証

現場でよく見かける EP 自己履行検証は、プロセスに書かれている作業に対して「やった／やっていない」を成果物の有無や記憶に頼って「○／×」を付けているだけになっている。しかし、EPは作業を羅列したものではない。プロセスの目的にそった活動を実行し、その目的を満たした成果物ができたかどうか、言い換えれば、「やったと言えるのか」という十分性・妥当性を作成自身が問い確認することが真の EP 自己履行検証と考えている。真の EP 自己履行検証が実施できれば、以下のような理由から自己トレーニングに繋がるはずである。

- ・ 作成者自身が実施するため、記憶が新鮮なうちに実施できる。また、他人に指摘されるのとは異なり真摯に取り組める。
- ・ 対象がエンジニアリングである。技術的要素が中心のため技術スキルとして取り込みやすい。
- ・ 十分性・妥当性をするために技術的な根拠・理屈を考えることになる。

(c) EP 自己履行検証をレビューの準備と位置づける

EP 自己履行検証はレビューの前に実施されることが多く、プロセスの実施者、つまり、対象の成果物の作成者自身の活動であるため、もともとレビューとの関係性は強い。前述のような自己トレーニングとなる EP 自己履行検証であれば、シミュレーションが行われるレビューとなる条件を自然に満たすことになり、親和性が高く、相乗的に効果を生み出せる活動となる。条件①への貢献：作成者自身が「成果物を作ったかどうか」ではなく「内容が適切かどうか」という視点で成果物を見直す。この過程で、作成者自身が除去可能な欠陥を減らすことができる。

条件②への貢献：プロセスの目的を解釈し成果物の適切さを見直す過程で、自分が何を考えたのか、なぜこうしたのか、を振り返ることになり、作成の経緯や技術的な根拠が自然に整理される。

この親和性により相乗効果を生むメカニズムを現場に持ち込むことにより、今までの“余分”“やらされ感”といった EP 自己履行検証のイメージを“レビューの準備になる・役立つ”に転換できると考えた。バラバラだった仕組み・活動を相互に関連付け、それぞれの問題を補完し、互いの仕組みの良さを引き出すトータルの仕組みとなり、「人が育つ」という最終的なゴールに効果を発揮できる“仕掛け”になると考えた。

(5) 変更や対応策の実施内容

EP 自己履行検証がレビューの準備にもなり、レビューと共に相乗的な効果を発揮する活動となるためには、知識として理解した上で実践する力が必要と考え、2つの仕組みを構築した。

① トレーニング

EP 時履行検証の実施者である若手技術者向けと、指導する立場である上司向けのトレーニングを用意した。

若手技術者向け：EP 自己履行検証がスキル向上に繋がり、レビューの準備にもなるメカニズムと具体的な取り組み方を学ぶトレーニング（集合教育）を企画・開発・実施。

指導者向け：PM 層を対象に、若手技術者の EP 自己履行検証の取り組み方を確認し、プロジェクトとしての活用の仕方を診断・改善する対面式のコンサルティングを定期的実施。

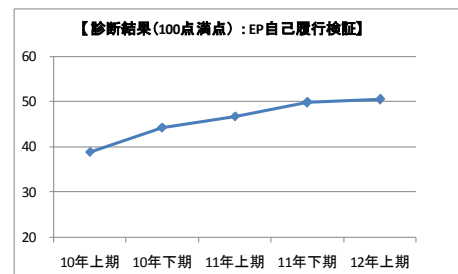
② 実施支援

若手技術者がトレーニングで学んだことを業務の中で実施することを、現場に入り込んで支援する「フィールドサポーター方式」を考案し、導入。各組織長に選出された対象者に対し、6ヶ月間、EP 自己履行検証とレビューへの取り組みの改善を支援し、EP 自己履行検証とレビューでスキル向上できる仕事のやり方への転換と効果の体感（成功体験）による定着をはかった。

(6) 変更・改善後の状態と効果

① EP 自己履行検証が「やった／やっていない」ではなくレビューの準備となる活動に変化した。

EP 自己履行検証の目指す姿を表した「成長モデル」を用いて PM 層向けのコンサルティングを行っている。「成長モデル」の評価値の平均値が、コンサルティング開始から2年間で約11%向上した。（図1）

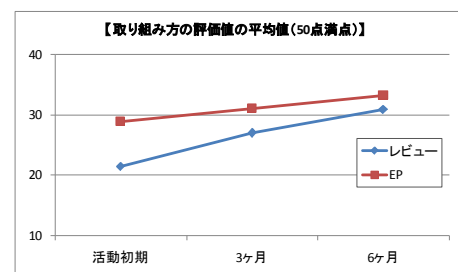


【図 5】

② 若手技術者の EP 自己履行検証およびレビューの取り組み方が改善された。

フィールドサポーター方式による実践支援では、EP 自己履行検証およびレビューの取り組み方をそれぞれの「評価モデル」で確認している。

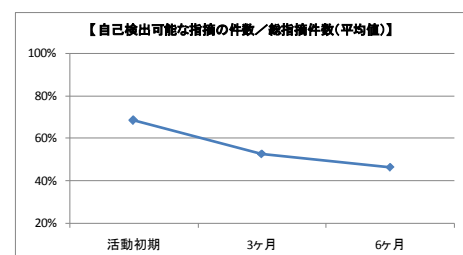
実践支援経験者10名の平均値が、6ヶ月の支援期間を経て、EP 自己履行検証が約10%、レビューが約24%向上した。（図2）



【図 6】

③ 若手技術者がレビューに持ち込む成果物の品質が向上したことから、スキル向上が認められる。

「レビューで受けた指摘に占める作成者自身で検出可能な不具合・欠陥が減る」ということは、EP 自己履行検証により若手技術者個人の開発および確認のスキルが向上し、かつ、レビューでシミュレーションがしやすくなり、レビューによるスキル向上に繋がっていると推測できる。



【図 7】

実践支援経験者 10 名のレビューで受けた指摘に占める自己検出可能な不具合・欠陥の割合が 6 ヶ月の支援期間を経て約 22%向上した。(図 3)

(7) 改善活動の妥当性確認

(6) で挙げた効果から、PM が中心となってプロジェクトとして EP 自己履行検証をレビューの準備として活用する風土は出来つつあり (①)、若手技術者自身も EP 自己履行検証とレビューの繋がりを意識して取り組むことができるようになり (②)、その結果、若手技術者自身がより良い成果物を開発できるスキルが備わってきたと判断している。よって、今回構築し実施してきた仕組みは、若手技術者が別物ではなく業務を通して学ぶ有効な仕掛けとなっていると評価している。

今後の課題として次の 2 点について、取り組むことを考えている。

- ① 業務の質や体制の変化から、若手技術者が EP 自己履行検証の実施やレビューを受ける機会が少ないケースが増えてきている。そういった場合の EP 自己履行検証やレビューの代替となる取り組みを創出する。
- ② 業務を通じた育成は、本来支援組織主導ではなく各現場で自然に行われるべきであり、そうでなければ形骸化の危険もある。現場主導の永続的な活動として定着させる。

2B2「レベル3達成組織における自律改善の推進」 藤縄幾子(パナソニック)

〈タイトル〉

レベル3達成組織における自律改善の推進

〈サブタイトル〉

～やらされ感の改善から自律改善への転換～

〈発表者〉

氏名(ふりがな): 藤縄 幾子 (ふじなわ いくこ)

所属: パナソニック株式会社

〈共同執筆者〉

氏名(ふりがな): 安倍 秀二 (あべ しゅうじ)

所属: パナソニック株式会社

・要点

組織で定義されたプロセスを、テラリングをうまく使うことにより、開発現場が自ら気づき、検討して、自らが継続的にプロセス改善を行っていきけるような改善活動への転換のアプローチ。

・キーワード

プロセス改善、自律改善、SPINA3CH

・想定する聴衆

SEPG、ソフトウェアエンジニア

・適用状況

- ☐ 多用されている段階、☒ 適用できる段階あるいは初めて適用する段階
- ☐ 適用するにはさらに検討を必要とする、☐ 着想の段階
- ☐ その他()

・適用可能性に関する制限

- ☒ 汎用性がある、
- ☐ 類似プロジェクトにも適用可: 具体的な類似点()
- ☐ 自プロジェクトのみ

・発表内容

(1)背景

モデルベースでの改善を SEPG が推進し、組織レベルでプロセスが定義されていても、プロセス実施や改善に対する開発部門の当事者意識は弱く、やらされ感が強い。このため、開発部門やプロジェクトの実態に適合させるための自律的なプロセスの改善活動の実現が難しい。

(2)改善前の状態

組織では過去から SEPG を中心としてプロセス改善に取り組み、CMMI をベースに組織のプロセスを構築し、CMMI レベル3も達成している。近年は車載プロジェクト用に機能安全規格 (ISO26262) に対応したプロセスも導入してきてきた。

開発プロジェクトは、組織で定義されたプロセスを負担と感じながらも”実施すべきもの”として取り組む傾向が強い。この際に、プロセスの意図を正しく理解せずに“やらされ感”で、定義されたままに活動を行っていることが多い。更に、定義された活動そのままではプロジェクトの実態に適合しない場合には、プロセスを実施することへの”不満”を感じることも少なくない。

また、この“やらされ感”や“不満”は、非効率なプロセスの実施や、その妥当性を十分に判断することのないプロセスへの非遵守を発生させてしまい、品質問題を発生させるリスクや現場のモチベーション低下等を起こす可能性を含んでいる。

更に、プロセス改善も SEPG のリードにより実施され、開発者自らがプロセスを改善しようという意識は低く本質的に開発プロジェクトに効果の大きい改善が実施されることが多くはない状況である。

近年、車載分野プロジェクトでは機能安全規格 (ISO26262) への対応が必要となり、従来定義したプロセスだけでは不十分となり、ますます、プロセスは膨らむ一方である。

このような状況では、開発プロジェクトにとっては“やるべきこと”は増える一方で有り、プロセスの負担が一層増加し、プロセスを正しく実施できないばかりかプロジェクトの破綻を招きかねない。開発者自らが、何が必要であるか、何が課題であるかを“自分達で気づき”、“自分達で改善して”くことにより、本質的に各開発プロジェクトに適切な“やるべきこと”を定義し、実施し、効率的な仕事へと繋げなければいけないと考える。

(3)改善前の状態をもたらした原因(因果関係)

組織のプロセス導入はトップダウンで行われるのが通常であり、開発現場では当事者意識が弱くなり、導入側 (SEPG) 任せになりがちである。その上、プロセスの意図を十分に理解しないまま盲目的に実施することが多い。

また、導入側 (SEPG) も現場毎に適合したプロセス構築の支援を実施することは可能であるが、更に、やらされ感を増長させ、“してもらおう”ことが当たり前になり、自らの改善意識が遠くなりがちである。

(4)変更内容・対応策

各開発プロジェクトの“自律的改善”を促進する。

そのために SPINA³CH 自律改善メソッドの考え方を応用し「課題の気づき(課題抽出)」「課題分析」「課題の絞り込み」「改善検討」「振り返り」を各開発組織自らが実施し、その結果を各開発プロジェクトのプロセスに活かせるようにし、開発者が当事者意識を持って継続的に”自律改善”を行うようにする。

(5)変更や対応策の実施内容

SPINA³CH(Software Process Improvement with Navigation, Awareness, Analysis and Autonomy for Challenge)自律改善メソッド(以降、SPINA³CH メソッドと呼ぶ)は、“ソフトウェア開発の「仕事のやり方」を現実に合わせて改善し、向上させるためのツール”で、開発現場の気づきを始まりとして状況と課題を分析し、課題を絞り込んで改善検討を行い、改善を進める、ことを自律的に推進するために IPA/SEC から公開されている手法である。それは、1)課題ベース及びモデルベースでの改善の両方の良さを持っている、2)スタートは課題ベースによりプロジェクトで起こっている問題(事実)を明らかにする、3)改善テーマを絞り込む、4)複数の道具を使用する、といった特徴がある。SPINA³CH メソッドは、プロセスが未整備である組織の改善に向いており、このため今回このメソッドの道具のすべては使用しない。

実施すべき活動が既に定義された「組織のプロセス」が改善のスタートとなる。これまでは、開発内容が類似しており、また、お客様のプロセス順守要求が強いので、組織のプロセスのテーラリングの幅はとても小さかった。それを、プロジェクトによるテーラリングの裁量を拡大し、プロセスに内装されるアクティビティを基本に、それをどう実施するか、やテンプレートも含めてテーラリングを可能としている。”テーラリングガイドライン”を活用し、開発プロジェクトでのテーラリング実施を支援する。

開発プロジェクトは、プロジェクト開始時に「組織のプロセス」の中のどのプロセスを実施するかを選択することから始まる。プロセスのマージや分割も可能にしている。また、プロセスのアクティビティをどう詳細に実施していくかも検討し、自プロジェクトに適合した実施すべきプロセスを決定する。この時点で決定したプロセスがプロジェクトの遵守すべきプロセスとして定義される。このテーラリングしたプロセスの妥当性は、QA 部門によって、プロジェクト計画を確認する品質ゲートにおいて、確認される。

エンジニアリング系の作業成果物のテンプレートは、どうしても、それを埋める作業を実施しがちなので、テンプレートの全面的な使用の要求をやめ、作業者の自主性に任せた。作業成果物を自由に作成させることにより、自らが何を記載すれば、自らの設計を効果的に説明できるのかを考えさせるようにした。プロセスの目的を達成させ、かつ、記載内容の高位平準化を実現するために、チェックリストを準備し、単なる項目の羅列では無く、作業成果物の構成を意識した、階層的な構造とした。

プロジェクトが完了時には、プロジェクトで定義されたプロセスを元に、プロジェクトの”振り返り”を実施する。この中で「課題の気づき(課題抽出)」「課題分析」「課題の絞り込み」「改善検討」を進める。

”振り返り”は、プロジェクトリーダー、プロジェクトメンバーが参加して開催する。テーラリングしたプロジェクトの定義されたプロセスに対する良かった点やプロジェクトで起こった問題を抽出し、詳細化し、真因分析を行うと共に、問題やその元になる原因全体を俯瞰してみる。この結果、対策の効果や必要性、実現性等を考慮して対策する対象を決定し、具体的な改善策を検討・決定する。

上記のテーラリングやここでの改善策は、「組織のプロセス」で定義された各プロセスの意図を逸脱してはならないことは自明である。このため、テーラリングや「改善検討」のためにはそのプロセスで求められる活動の目的を正しく理解していることが重要となる。これを支援する方策として「組織のプロセス」の理解度を向上するための応用研修を準備し、効率的で適切な改善策が立案されることを期待している。また、SPINA³CH メソッドの「改善検討シートワークシート(ヒント集)」での“ソリューション例”を参照することも効果的であると考えている。

決定した改善策は次の“プロジェクトの定義されたプロセス”に実装する。また共通的に「組

「組織のプロセス」に反映すべき良かった点や改善すべき点は組織資産として取り上げて蓄積していくことも必要である。

開発組織は、更に次のプロジェクト完了時の「振り返り」で改善策の結果・効果を確認し、更に次の改善を行っていくことで「自律改善」を継続的に回していくことができる。

図1にこれらの関連を示す。

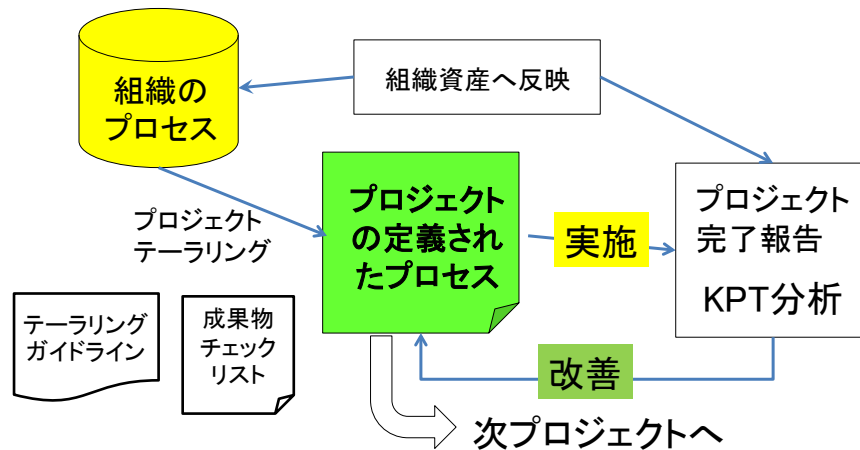


図1 自律改善・継続改善の重要性

(6)変更・改善後の状態と効果

現時点では、プロジェクトは、上述したテラリングを実施し、プロジェクトの定義されたプロセスに基づき活動中である。本発表までには、複数のプロジェクトがふり返しを実施予定で、その段階で、本活動の成果を報告予定である。

(7)改善活動の妥当性確認

組織で定義されたプロセスは、多くの組織に対応することも考慮され、特に、車載開発においては、お客様のプロセス要求に加えて、機能安全規格(ISO26262)の要求もあり、開発プロジェクトの規模や特徴によってはそのままの実装では重くなりがちであることも否めない。

だからこそ、開発プロジェクトは自ら実施すべきプロセス(活動)を各自に最も適合した形で効率的・効果的に実施することが求められる。

このためには、各開発組織自らの”自律改善”を継続的に実施していくことが最も効果を発揮するといえる。

2B3「SaPID 実践事例より～改善推進役がやるべきこと／やってはいけないこと」安達賢二(HBA)
〈タイトル〉

SaPID 実践事例より～改善推進役がやるべきこと／やってはいけないこと

〈サブタイトル〉

現場が自らの一步を踏み出すために

〈発表者〉

氏名（ふりがな）：安達 賢二（あだち けんじ）

所属：株式会社 HBA 経営管理本部

〈共同執筆者〉

氏名（ふりがな）：なし

所属：－

・要点

現場が自ら改善実践する一步を踏み出し、最終的には自律改善を回すようになるまでに改善推進役がやるべきこと／やってはいけないことを、実際の改善実践中に発生するさまざまな局面・事象への対応事例とそれら全体から言えること、そしてそのことを実現・実践するために必要な改善推進役の能力・身に着けておくべきノウハウとして共有します。

（5/23 第5回トワイライトフォーラムでお話しさせていただいた「自律改善の実現を目指す改善推進役の役割」と「改善推進役に求められるスキル」をさまざまな局面の事例であきらかにする試みです）

・キーワード

自律改善・SaPID・改善推進役

※改善推進者としていないのは、中小企業などでは役割として兼務することが想定されるから、そして最終的には管理者やリーダーが自ら推進役を担うのが理想と考えているからです

→一般的には“改善推進者”として認識されている場合は“改善推進者”としていただいて問題ありません。

・想定する聴衆

中小IT組織の管理者やリーダー、業務・プロジェクトチームのリーダー、メーカー・ベンダーなどのSEPG・SQA・PMO、組織の改善推進者 など

・適用状況

■多用されている段階、□適用できる段階あるいは初めて適用する段階

□適用するにはさらに検討を必要とする、□着想の段階

□その他（ ）

・適用可能性に関する制限

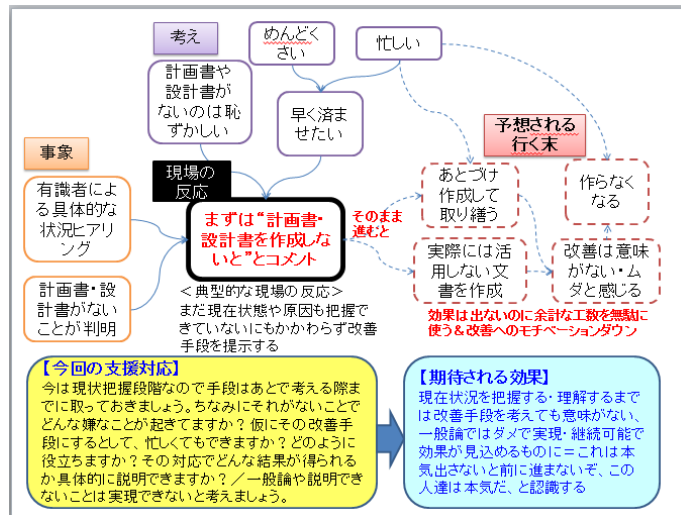
■汎用性がある、

□類似プロジェクトにも適用可：具体的な類似点（ ）

□自プロジェクトのみ

・発表内容

以下(1)～(6)の実証実験対応過程で発生したさまざまな問題事項に対して改善支援チームが行った対応とその意味を考察し「改善推進役がやるべきこと／やってはいけないこと」を明確化する、そして関係者で共有したい、というのが今回の内容である。
個別の対応事項とその意味の考察結果表現形式は「発生事象」「現場の考え」「現場の反応」「予想される行く末」「今回の支援対応」「期待される効果」とする。

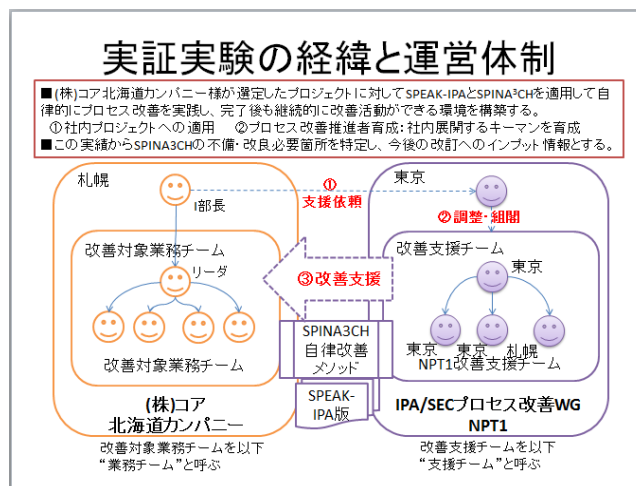


事例：個別の対応事項とその意味の考察結果表現（当初ヒアリング時）

この対応により改善推進役の役割「IT技術者やその管理者がエンジニアライフ・マネージャライフを発見的に歩むのを支援する＝IT技術者や管理者が自らの背景や経緯、さまざまな事情や境遇／現状を理解して自ら前に進む答えを出して歩んでいくのを支援すること」を果たすためにはどう対応するべきか、を関係者と共有することを目指す。

→※実証実験の経緯と詳細は「添付スライド」を参照／以下にそのポイントを記す。

(1) 背景＋(2) 改善前の状態



当時 IPA/SEC プロセス改善 WG では SPINA3CH 自律改善メソッド・SPEAK-IPA 版の普及と有効性確認（改良箇所の特定）を目指して実証実験組織を募集していた。

そこに(株)コア北海道カンパニー様の特定顧客向けに映像編集機能付きシステム開発／継続的メンテナンスを行っている部門のI部長から改善支援依頼があった。

＜当初＞

●依頼者 I部長の当初の意向：

今後の組織内横展開を目指す上で選定したチームに対して以下の事項を解決するために改善を実施したい。

①見積もり精度を高めたい

②標準化によりプロジェクトを進められるようにしたい

支援I質問：「顧客に対する成果や現場の問題など何か解決したいことはないですか？」
→障害を減らし顧客に信用される運営をしたい

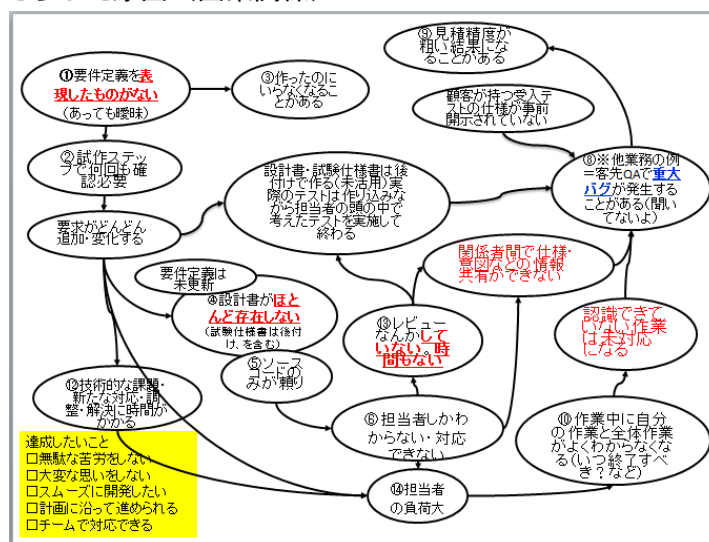
●改善対象業務チームリーダーの当初のコメント：

見積りが大きな課題の一つ。顧客とのトラブルは現時点ではない。

→※実際には納品後障害などが発生しいろいろと困っていたことがあとで判明

※注意：実証実験上では当案件を「SPINA3CH&SPEAK-IPA 版を併用したアプローチ」で対応することとしたが、実務的には当初の問題構造構築までの過程は SPINA3CH の道具を使わず SaPID で行う“普段各メンバーが感じている問題点を提供してもらう”で対応／SPINA3CH の改善検討シートも未活用など SaPID アプローチを多用したため、実質的には「SaPID&SPEAK-IPA 版を併用したアプローチ」と言える内容である。

(3) 改善前の状態をもたらした原因（因果関係）



問題構造は図の通り。(改善着手後 1.5 か月経過時点で完成していた図)

この図から、どの領域を、どのような改善手段で打ち崩し、何をを目指すのかを検討した。しかしその後 3.5 か月間は、どの領域に何をしてよいのか現場メンバーが自ら答えが出せずに、そして早く終わりたいがゆえに改善支援チームには知らせずにそれっぽい改善手段（実施前からうまくいかないのでは？と薄々気づいていた手段）を実施してうまく行かず、さらに困る・・・など、迷走した。

(4) 変更内容・対応策＋(5) 変更や対応策の実施内容

結局改善手段したのは以下の 3 項目。

- (1) 窓口一本化による対応要求事項の内容と規模・量のコントロール
- (2) 朝会で(1)と作業の状況共有（繁忙期は1日2回）／難易度高い事項は有識者と検討
- (3) テスト仕様の事前作成

“多少なりとも効果が期待できる自分達でもやれること”を選択したため、実施は比較的容易であった。むしろ最も苦労したのはこの3項目に決めるまでの過程である。

(6) 変更・改善後の状態と効果 + (7) 改善活動の妥当性確認

- 1) 品質 開発規模2倍・期間および要員1/2以下 に対して
修正依頼件数：改善前に比べ56%減 障害件数：改善前に比べ48%減
- 2) 生産性
単純比較はできないが、仕様変更、障害対応消化率が飛躍的に向上。
要求～2日以上滞留することはほとんどなかった。
- 3) 顧客満足度
顧客側担当は今回初めての協働。
改善活動により品質・生産性の両面で実績を残した上に、ある機能については前倒し
ができ、信頼関係が築かれる結果に。

そして特記すべき事項として、改善開始時にはあまり元気がなく笑顔がほとんどないリーダーとメンバーが、最後はイキイキと「やってよかった!」「これこれが反省点、今後の課題です～次は是非この改善をしたい!」と自ら宣言したこと。

(このあとどうするなどはこちらから一切要望していないにもかかわらず)

2B4 「「真のプロセス定着」への取り組み」 宮川 研二(ダイキン情報システム)

<タイトル>

「真のプロセス定着」への取り組み

<サブタイトル>

やらされ感の払拭による全員のレベルアップ

<発表者>

氏名（ふりがな）： 宮川 研二（みやがわ けんじ）

所属：ダイキン情報システム株式会社

<共同執筆者>

氏名（ふりがな）： 谷口 善弘（たにぐち よしひろ）

所属：ダイキン情報システム株式会社

<共同執筆者>

氏名（ふりがな）： 木下 あすか（きのした あすか）

所属：ダイキン情報システム株式会社

・ 要点

社内のプロセスの成熟度評価には、モデルを用いたプロセス構築や評価を行うことが多い。しかしとかくレベルという便利な数値に惑わされ、レベル達成に目標がすり替わってしまい、開発現場では何も変わっていないことが多い。つまり、形式的なプロセス構築は終えたものの、実態の業務改善が伴っていない場合が散見される。

当社が苦労した「真のプロセス定着」について、工夫と取り組みを報告する。

・ キーワード

プロセス改善、プロセス定着、教育、やらされ感

・ 想定する聴衆

SEPG、品質保証

・ 適用状況

☒ 多用されている段階、☐ 適用できる段階あるいは初めて適用する段階

☐ 適用するにはさらに検討を必要とする、☐ 着想の段階

☐ その他（ ）

・ 適用可能性に関する制限

☒ 汎用性がある、

☐ 類似プロジェクトにも適用可：具体的な類似点（ ）

☐ 自プロジェクトのみ

・発表内容

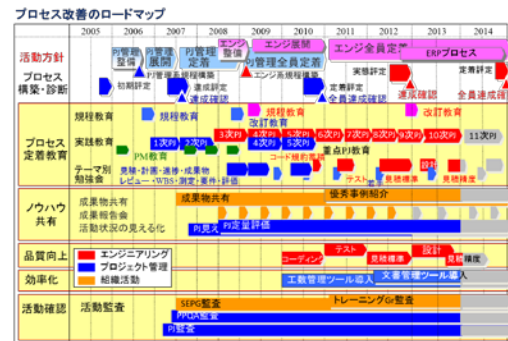
(1) 背景

生産性の向上ならびに品質の向上のための業務改善を目的として、2006年から開発プロセスの改善に取り組んできた。特に開発のケジメが取れずに最終的に開発が混乱することを避けるために、改善の活動を展開してきた。

CMMIをはじめとするモデルに照らし合わせて社内の開発状況を診断し、そこから出た強み・弱みを基に社内プロセスを再定義した。そしてモデルに対する診断と成熟度による評価で、組織の状況を把握してきた。2007年には最初の目標レベルであるプロジェクト管理の展開は達成したものの、開発グループの間には開発業務が楽になったという実感はなく、一部のプロジェクトが成し得た結果としか思えなかった。

その時点で、より上位のレベル到達を急ごうとしたものの、それよりも全社員がプロジェクト管理をできるようになること（最悪のプロジェクトであっても無理なくプロジェクト管理がこなせること）が大切だという判断のもと、業務改善のために「真のプロセスの定着」を実現することを目指して取り組んできた。

本書では、開発グループに社内標準を理解・納得させて、「やらされ感」を払拭しプロセス改善を進めたかについて、教育とノウハウ共有についての当社のアプローチを紹介する。



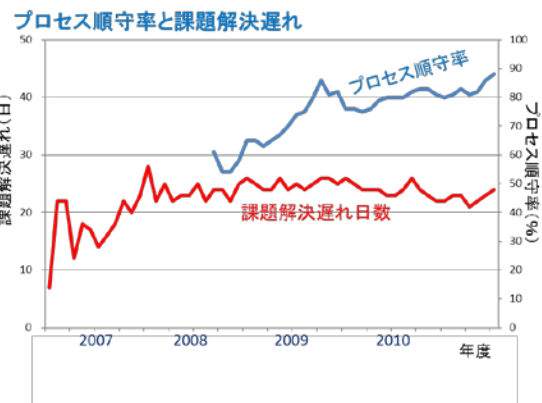
(2) 改善前の状態

業務改革を目指し「ケジメの取れた開発」に関する社内標準を整備し、改善のための体制を作って、開発業務の改善を行ってきた。

しかし、形式的には社内標準を守れているように見えても、開発グループにとっては締め付けられてやっているだけで「やらされ感」が漂っていた。このことは、社内標準に対するプロセス順守率の向上に対して、監査等での指摘課題の解決遅れが一向に縮まらないことでも裏付けられる。また決められた標準を愚直に守るがために、無理のある開発が行われていたのも事実であった。

開発現場からの声は、以下の2点に集約される。

- プロセスの構築は行われたものの、開発のやり方が変わった、あるいは楽になったという感触がない。一部のトップランナーでの取組であって、組織全体の活動になっていない。
- 社内標準プロセスを実施するに当たり、開発者には「やらされ感」が強く、社内標準を守った開発の必要性は頭で判っているものの、実践の場になると活用ができていない。特に開発業務が忙しくなれば楽な方向（プロジェクト管理の欠如）になりがちである。



そこで「真のプロセス定着」には、これら「やらされ感」の払拭こそが大切であると意識し、この課題を克服するために取組みを開始した。

(3) 改善前の状態をもたらした原因（因果関係）

「やらされ感」の払拭に当たり、この業務改善の活動を全員が当事者意識を持ち取り組むことが重要であり、組織全体の業務改善活動とするために、

「組織全員」がケジメの取れた開発をできる

という目標を掲げた。つまり、最低のプロジェクトであっても社内標準に順守したケジメの取れた開発をできるレベルまで到達するといった厳しい目標設定を行った。

それを実現するために、現状の業務分析を行った結果、以下の4点の課題が浮かび上がった。

- ① 開発現場での業務改善に対する意味が理解されていない。上級管理者の改善活動に対する認識も希薄である。
- ② ベテランから若手へのノウハウの伝授ができていない。
- ③ 社内標準作りへの開発者自身の参画意識がない。
- ④ プロジェクトが持つノウハウの他プロジェクトへの展開がない。

そこでこの4つのそれぞれの課題について、解決してくための対策案ならびに実施内容を次章以降で述べる。

(4) 変更内容・対応策

開発者にプロセスの意味を理解し、納得したうえでの活動を行うためには、それに対するしくみ作りも大切であるが、何よりも開発者に活動の意味を理解させ、自らが必要と感じて開発活動を行うようになることが必要と判断した。依然根強い「やらされ感」をいかに払拭するかに取り組んだ。

そのために、以下の4つの側面から教育ならびにノウハウ共有を行うこととした。これまでもプロセス展開のための全社一律の教育は行ってきたが、これからは必要なところに必要な教育を「的を絞って」実施していくことにした。

① 重点プロジェクト教育

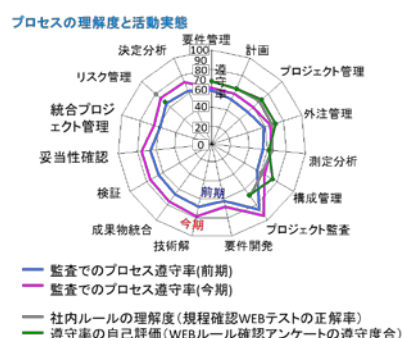
社内標準については頭では理解しているものの、実践の場で活かすことができないのが開発の実態であった。

そこで今までの全員への一律の教育を改め、主体をプロジェクトマネージャに定めた。そのうえで、プロジェクトの開発進捗に沿ってその時点で行うべき開発活動について、診断と相談をプロジェクトマネージャに対して実施した。また、これまでの診断と指導が中心の教育から、開発上でやりにくい個所の議論へとすることで、プロジェクト運用上のプロセス面での悩みごとの相談の場に教育内容をシフトしていった。

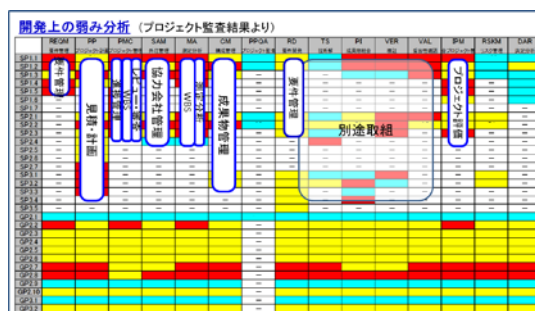
また教育対象であるプロジェクト(プロジェクトマネージャ)についても、これまでの教育網羅性を追求することよりも、改めて教育を必要とする対象者の絞り込みを行った。合わせて、プロジェクト(プロジェクトマネージャ)ごとに教育到達のレベルを設定し、教育完了の判断を明確にした。

② テーマ別勉強会

前述の重点プロジェクト教育の結果、ならびに日頃からプロジェクトに実施し



ている活動監査の結果から、組織共通の弱みと思われる8テーマを選定した。ただ座学による教育をしても実開発での応用は困難であったために、各自の開発手法を披露し、他人の良いところからヒントを得るためのディスカッションを中心とした教育を設定した。

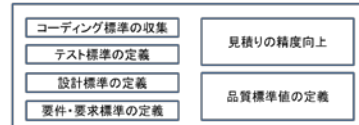


③ 技術標準ディスカッション

品質向上のための技術標準の見直しにおいて、この部分はとりわけプロジェクトマネージャの資質によるところが強かった。今回特に特に見積標準値、テストプロセスなどを、見直しが必要なテーマとして選定した。

プロセスの見直しを行うに当たり、まずプロジェクトの実態を把握し、弱みと強みを見出すことが必要であったが、その際にも前述のテーマ別勉強会で行ったディスカッションによるプロジェクトの実態把握の手法を活用した。

技術標準の見直し



④ ノウハウ共有

プロジェクトの振り返りを行いプロジェクトのノウハウを組織で共有するための全社での報告会は以前から実施していたが、漠然と結果の報告が行われていたにすぎなかった。そこで発表のポイントを明確にし、それで生ずるノウハウを他プロジェクトに訴える報告に変えることが必要であった。

そのために、発表プロジェクトの選定に当たっては、組織が年度で定めた改善取組テーマに対する工夫を取り入れたプロジェクトを選出し、発表するようにした。

(5) 変更や対応策の実施内容

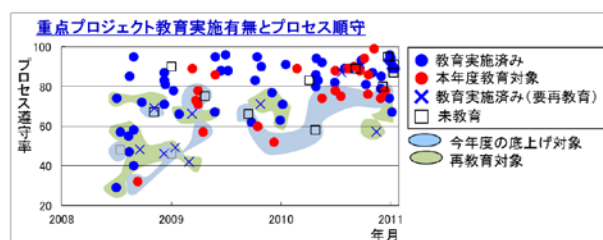
以上の4つの取組みについて、具体的な展開を述べる。

① 重点プロジェクト教育

プロジェクトの進捗に合わせて活動状況の社内標準に対する診断と不足箇所の説明を行う教育はこれまでも行っていたが、今回は教育の対象・レベルを明確にする、以下の変更を行った。

- 教育が必要な要員と達成レベルを SEPG から個人名で指名し、開発マネージャに教育の参加を要請した。
- SEPG 主導の診断中心の教育から、開発グループ主体のやりにくい個所の相談の場に、教育自体の進め方を変えた。

プロジェクトに関与しながら未教育である者の中には、画一的な開発工程の進め方しかできず、苦勞してプロジェクトを進めていることが散見された。また、教育完了者のなかにもその後の開発で社内標準に準拠した活動ができている者も見られた。SEPG は、それらの者を要教育対象者として選定し、開発マネージャに教育の参加を要請した。同時に開発マネージャに対して、今回選出した教育対象者を極力次期プロジェクトのマネージャ的な役割に配置することを依頼した。



教育内容もプロジェクトの診断を行ったうえで、開発のケジメが取れていない箇所に対しては、どこがやりにくいのかを聞き出し原因の深堀りを行い、他のプロジェクトの事例や他社の開発の進め方を踏まえながら、うまく考えていく場にしていった。

二、マ別勉強会

教育を開催するに当たり、以下の点を準備した。

- 机上の説明だけではイメージできずに消化不良になることを恐れ、実際の開発グループで使った成果物を使用し、参加者で論議することにした。その際、成果物の記載内容への理解の違いを避けるために、似た業務を行っている者を集めた。

③ 技術標準ディスカッション

重点プロジェクト教育の実施有無

教育対象

PMとして教育済み
メンバーで教育済み

PMとして要教育
尚し再教育済

PMとして要教育
メンバーで要教育

PMとして教育未
メンバーで教育未

数値は達成レベル

部門	Gr	課長	GL	メンバー (PM経験)	メンバー (PM未経験)
開発1部	G+SCM	PM	PM		PM
	化学	PM	PM		PM
	経営	PM		PM	PM
開発2部	生産	PM		PM	PM
	新生産			PM	PM
	関連企業	PM	PM	PM	PM
開発3部	油種・特種	PM	PM	PM	PM
	営業	PM	PM	PM	PM
開発4部	物流	PM	PM	PM	PM
	サービス	PM	PM	PM	PM
	技術情報	PM	PM	PM	PM
開発5部	設計情報	PM	PM	PM	PM
	デバイス	PM	PM	PM	PM

勉強会実施内容		
テーマ	事前準備	時間構成
第1回 見直し	開発プロジェクト計画書 工程見直しシート	0:15 前回テーマの行動目標の実施確認 0:30 一般説明、社内規程
第2回 品質管理 成果物管理	進捗会議議事録 成果物管理ライブラリ	1:10 チェックステップ 0:05 今回テーマの行動目標の立案
第3回 協力会社管理	RFP	
第4回 レビュー 審査	レビュー記録 審査記録	
第5回 WBS	WBS	
第6回 測定分析	測定分析シート	
第7回 案件管理	案件定義書 変更管理票	
第8回 プロジェクト評価	プロジェクト評価報告書	

[illegible]

さず話してもらう必要があった。

テスト設計、見積作成など、実務担当者を集めて、そのやり方を聞き出すところから始めた。ただし、テーマに応じて適切な参加者選定した。

- SEPG が主体となってまとめる技術標準については、ヒヤリングから強み・弱みを抽出し、標準にまとめあげる。
- 開発グループが主体となってまとめる組織標準値に関しては、SEPG から開発グループに対する実態のヒヤリングと考え方の整理を行う、その結果を開発グループ側でまとめて文書化したものを、両者でレビューし標準値を確定していく。

テスト・設計などエンジニアリング系の実態把握には、開発の実態をフリーディスカッションで聞き出し、開発者には互いのプロジェクトの良い点を盗んでもらった。一方 SEPG としては、優れた活動を組織標準に盛り込むとともに、組織共通の弱みについては技術標準としてのプロセスを定義した。

それに対して組織標準値の設定においては、開発実態を聞き出すまではエンジニアリング系の技術標準と同様であるが、SEPG は考え方を整理することを援助した。例えば見積標準値に関しては、開発グループが工数見積をする際に考える要素(難易度、要員スキル等)を SEPG が聞き出し、実データとの整合性をみながら考え方の整理と確認を取ることを行った。それを基に開発グループが作成した見積ロジックを改めてレビューし、使えるものに仕上げていった。

④ ノウハウ共有

社内でのノウハウ共有は、プロジェクトの最終工程で行う振り返りのためのプロジェクト評価報告書の作成と、半期で開催する改善成果報告会であった。それらに対し、以下の改善を行った。

- 評価報告書に、課題の抽出、原因の確定、次回の対応の3要素が表現できるようにフォームを変更する。
- プロジェクトが行う評価報告書作成の初期レビューの場に、第三者の眼として SEPG が参加する。
- 改善成果報告会では発表テーマを定め、それに沿った報告を行う。そのテーマは年度の組織改善テーマのなかから選ぶ。

まず、漠然と作成されていたプロジェクトの振り返りのための評価報告書であるが、次へのアクションを示しそれを次のプロジェクトで実行するといった改善サイクルが回るように、「次回プロジェクトでの対応策」を記載することを指導の観点とした。そのためには客観的なデータから予実の乖離を判断し課題を抽出する。

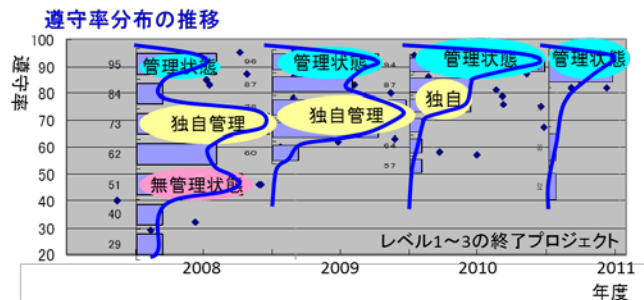
- (1) 客観的なデータから予実の乖離を判断し課題を抽出
- (2) 抽出された課題から、対処可能なレベルまで原因深掘りをし、真の原因の確定
- (3) 原因に対して次プロジェクトではどう対応するか、あるいは他プロジェクトに何を訴えるかの行動記載

をフォームとして定めた。またその指導のために、開発グループが行うプロジェクト評価の場の初回のブレインストーミング時に、SEPG が参加した。その際、広く多くのプロジェクトを見渡すことのできる SEPG が、プロジェクト自身で気付くことの少ない強みについても助言を行った。

改善成果報告会については、年度の組織改善テーマに基づいたプロジェクトからの報告とすることで論点が明確になるとともに、プロジェクト側も人に訴えるという意識が芽生えた結果、具体的事象の提示などの工夫が行われるようになり、他プロジェクトの聞き手が真似をすべきか否かの判断ができる効果も生まれた。

(6) 変更・改善後の状態と効果

これらの取組みを行った結果、開発グループの開発に対し社内標準で求める活動の理解は確実に向上した。とりわけ、以前は社内標準で定める活動に関して意識の低かった者について、底上げができたことが大きいと考える。



(7) 改善活動の妥当性確認

前述のように、本活動によって組織全体の底上げができたと考える。また、全プロジェクトを対象にした社内アプレイザルにおいても、最悪プロジェクトであっても目標レベルが達成したことを確認でき、目標としていた「全員でのケジメ取り」の第1弾は成し遂げたと判断する。

「やらされ感」の払拭はある程度の効果は出たが、業務が一旦混乱をきたすと管理系の業務が疎かになるのも事実であり、「やらされ感」が完全に払拭されたとは断言できない。「データや成果物の本当の意味での活用」等の次の課題に向けて、今後も業務改善を実行し続けていく。

また今回は国内の開発拠点に対する取組であったが、海外グループ会社とのシステム開発業務に関する改善情報の交換を始めており、互いに改善の方向性と取組み手法の共有を図り、次なる組織のレベルアップに向けて進んでいきたい。

2C1「チケット駆動開発によるプロセス改善」阪井誠(SRA)

〈タイトル〉 チケット駆動開発によるプロセス改善

〈サブタイトル〉 現場重視、管理重視、それとも情報共有重視

〈発表者〉

氏名（ふりがな）： 阪井 誠（さかい まこと）

所属： 株式会社 SRA

・ 要点

チケット駆動開発はツールの有効利用として注目されているだけでなく、プロセス改善を目的として多くのプロジェクトで実践されるようになった。しかし、その方法は多種多様であり、知見はあまり整理されていない。本報告では現場重視、情報共有重視、管理重視の3種類の代表的なプロジェクトの経験とメリット・デメリットを整理し、得られた知見を報告する。

・ キーワード

チケット駆動開発、プロセス改善、プロジェクトファシリテーション、モチベーション

・ 想定する聴衆

プロジェクトリーダー、サブリーダー、SEPG 担当者、ソフトウェア技術者

・ 適用状況

- ☒ 多用されている段階、☐ 適用できる段階あるいは初めて適用する段階
- ☐ 適用するにはさらに検討を必要とする、☐ 着想の段階
- ☐ その他（ ）

・ 適用可能性に関する制限

- ☒ 汎用性がある、
- ☐ 類似プロジェクトにも適用可：具体的な類似点（ ）
- ☐ 自プロジェクトのみ

・

発表内容

(1) 背景

ITS (Issue Tracking System) の障害票に相当するチケットを障害だけでなく、課題、作業などを構成管理と関連付けて管理するチケット駆動開発 (TiDD) [1] が注目されている。近年は、チケット駆動開発が広く知られるようになり、多くのプロジェクトで実践されるようになった[2] [3]。

チケット駆動開発は、チケットをプロジェクトの運営情報の中心とするシンプルな情報共有が特徴である。しかし、その方法は多種多様であり、知見はあまり整理されていない。本報告では現場重視[4]、情報共有重視、管理重視の3種類・4分類の代表的なプロジェクトの経験とメリット・デメリットを整理し、得られた知見を報告することで、チケット駆動開発がより有効に活用されることを期待している。

(2) 改善前の状態

現場重視プロジェクト

プロジェクトがどんどん大変になったが、全貌がわからないのでメンバーが守りに入っていた。

情報共有重視プロジェクト

段階的な計画を実現するとともに、保守に必要な情報の蓄積と効率化が必要だった。

管理重視プロジェクト

- 1) トレーサビリティの確保が必要とされたので、作業を容易にしたかった。
- 2) オフショア開発のコミュニケーションの効率化を図りたかった。

(3) 改善前の状態をもたらした原因（因果関係）

現場重視プロジェクト

類似システムの開発経験者が少なく、課題が多かった。

情報共有重視プロジェクト

要件が決まっておらず、順次仕様を詳細化して開発する必要があった。また、継続的な保守が必要だった

管理重視プロジェクト

- 1) 顧客の要件と実施作業のトレーサビリティを確認するうえで必要だった。
- 2) ファイル共有とメールによるコミュニケーションでは混乱が予想された。

(4) 変更内容・対応策

現場重視プロジェクト

チケット駆動開発を導入して、気づいた課題を順次チケット化して備忘録の代わりにした。

情報共有重視プロジェクト

段階的な開発を実施し、開発期間ごとに作業チケットを発行して管理した。構成管理とも関連付けた。

管理重視プロジェクト

- 1) 要件のチケットと作業のチケットを関連付けて、トレーサビリティを管理した。
- 2) オフショア先とのコミュニケーションフローを定義して、チケットに集約した。

(5) 変更や対応策の実施内容

現場重視プロジェクト

タスク一覧用のレポート（一覧画面）を作成し、チケットで作業指示も行った。

情報共有重視プロジェクト

チケット駆動開発の経験者だったので、自主性を尊重した。

管理重視プロジェクト

- 1) 管理用のレポート（一覧画面）を作成して随時、関連付けの確認（棚卸しをした）。
- 2) 現場での作業が容易なように、権限の設定を柔軟にした。

(6) 変更・改善後の状態と効果

現場重視プロジェクト

やらないといけないことが見える化され、作業漏れが少なくなったほか、メンバーのモチベーションが高くなった。

情報共有重視プロジェクト

作業が効率化された。目に見えた生産性の向上はなかったが、朝会も効率化できた。

管理重視プロジェクト

- 1) チケットの関連でトレーサビリティを管理することができた。タイトルに項番が必要になるなど、作業の負担は大きかった。
- 2) コミュニケーションとその履歴がチケットに集約されて、作業が効率化された。

(7) 改善活動の妥当性確認

チケット駆動開発を導入することで、今後の予定、現在の状況、過去の履歴を表示・検索することができるので、プロジェクトの情報共有や保守作業に有効だった。報告した3種類・4分類のプロジェクトは、それぞれの目的をもってチケットの利用方法を工夫した（表1）。

現場重視のプロジェクトでは、気づいた課題をチケット化することで漏れなく作業が実施できた。作業のチケット化により、いつまで続くかわからない状況から、大変だけれどもゴールが見える状況になり、モチベーションの向上によってチームが一丸となることができた。また、危機感と目的意識があったので、チケットによる情報共有を有効に活用できた。

情報共有重視のプロジェクトでは要件が固まっていなかったが、チケットで順次管理することで段階的ながらも計画的に管理することができた。また、チケットと構成管理を関連付けていたので、過去の経緯を容易に確認できた。チケットの運用はチケット駆動開発の経験者の自主性に任せたので、チケット駆動開発に対する負担は最も少なかった。

管理重視のプロジェクト-1では、トレーサビリティが確保されている事を、チケットの一覧でチェックできた。その反面、開発メンバーに不要な要件チケットがあり、見やすくソートして一覧化するには、タイトルや追加した属性に項番を入れる必要があった。要件変更によって、一覧の順序変更が必要になった際には修正が大変だった。

管理重視のプロジェクト-2では、1) ファイルの上書きやロックなどのファイル共有に関連する問題、2) 検索性が悪く、個人の管理能力に任せられるので、情報がうずもれてしまうメールに関連する問題、の2つが大きく改善された。実施1か月後および半年後に、日本側の担当者にヒアリングしたところ、「もっと早く導入すればよかった」「なぜ入れなかったのだろう」との感想が得られた（その後、話す機会があった際にも同様の感想を述べていた）。

表 1 事例のまとめ

	現場重視	情報共有重視	管理重視	
目的	作業漏れ防止	計画、保守性向上	1) トレーサビリティの確保	2) オフショアの管理
チケット利用法 (障害管理以外)	備忘録・情報共有	進捗、履歴	トレーサビリティ・進捗	課題管理、QA
プロジェクト数	2	3	2	1
チケットの経験	障害管理のみ	あり	あり	一部
モチベーション向上	大	中	小	大
管理面の効果	中	中	大	大
コミュニケーション向上	大	中	中	大
強制	なし	なし	あり	あり
負担感	小	なし	大	小
結果	作業量が明確になり、モチベーションが向上	管理が容易。必要十分な記録を自主的に実施	効果あり。要件変更時の負担大	ファイル共有とメールの混乱が解消、
感想	危機感と目的意識があり、有効に使えた	報告も簡素化され、作業が進めやすくなった	面倒。負担軽減の方法があったかもしれない	もっと、早く導入すればよかった

これらのプロジェクトを比較すると、現場の要望や自主性に任せたプロジェクトではチケット駆動開発の負担が少なく好評であった。その反面、管理を目的とした場合は、現場のメンバーには効果よりも負担が大きく感じられた。チケットの親子関係をサポートするプラグインや、親子チケットを標準でサポートしている Redmine を用いるなど、実現することだけでなく作業効率も考慮すると、もっと評価が良くなったと考えられる。

これらの経験を生かして、管理の負担をなるべく減らして現場作業の容易性を重視したオフショア開発の管理では、管理と現場のバランスをとることができた。チケット駆動開発は現場作業の改善の中で生まれ、普及した。導入の目的に関わらず、現場の意見をうまく取り入れることがより良い改善の秘訣なのかもしれない。

チケット駆動開発には基本的なプラクティスはあるものの、標準的なプロセスや認証制度、資格制度といったものはなく、目的に合わせて様々な実践方法がある。本報告はそのような中から 3 種類・4 分類を示した。これらは実践例の一部に過ぎず、より多くの知見を集めることで多様な問題の解決が容易になり、効率的で快適なプロジェクトが実現できると考えられる。

参考文献

- [1] 小川, 阪井, “チケット駆動開発”, 翔泳社, 2012.
- [2] 岡, 三宅, “本当に使える開発プロセス”, 日経 SYSTEM, 2012.
- [3] 前川, 西河 誠, 細谷, “わかりやすいアジャイル開発の教科書”, ソフトバンククリエイティブ, 2013.
- [4] 阪井, “チケット駆動開発によるプロジェクトの活性化 ―見える化と運用ポリシーがプロジェクトを変えた”, SPI Japan 2010, SPI コンソーシアム, 2010.

2C2「チケットを利用したタスク管理の試み」古石ゆみ(S R A)

<タイトル>

チケットを利用したタスク管理の試み

<サブタイトル>

小規模プロジェクトでの自律的管理のために

<発表者>

氏名（ふりがな）： 古石 ゆみ（こいし ゆみ）

所属：(株) S R A 産業第一事業部

・ 要点

外的要因によりマスタスケジュールの頻繁な変更、FaceToFace の時間がとれない、各自が複数の小さなタスクを掛け持ちしている…そのような状況下で、プロジェクトメンバー同士がお互いのタスクを自律的に管理し共有するために「チケット」の利用を試みた。

発生したタスクを1つのチケットとして登録し、プロジェクト全員に公開することで、お互いの作業負荷、作業進捗の見える化が可能になる。また、過去のメールを探すことなく、得たい情報にたどり着くことができる。

本報告は、「プロジェクト管理はプロジェクトマネージャが」という発想から脱却し、メンバー対メンバーのフラットな情報共有、自律的なプロジェクト管理を実現するために、チケット管理の導入を自ら試みたものである。状況が日々変化していくプロジェクトにおいて、マスタスケジュールと週報だけでは乗り切れない、と、悩んでいる方にとって、少しでもヒントになれば幸いである。

・ キーワード

タスク管理、構成管理、タスクの見える化、チケット駆動開発、トラック、小規模、チーム内情報共有

・ 想定する聴衆

S E P G 初心者、小規模プロジェクトのマネージャ、プロセスエンジニア、ソフトウェアエンジニア

・ 適用状況

☐ 多用されている段階、☐ 適用できる段階あるいは初めて適用する段階

☒ 適用するにはさらに検討を必要とする、☐ 着想の段階

☐ その他（ ）

・ 適用可能性に関する制限

☐ 汎用性がある、

☒ 類似プロジェクトにも適用可：具体的な類似点（小規模プロジェクト）

☐ 自プロジェクトのみ

・発表内容

(1) 背景

弊社では以前から、プロジェクトがイントラネット上で Trac や Subversion を利用できる開発支援環境が提供されている。

本報告で取り上げるプロジェクト（＝報告者自身のプロジェクト）の業務は、ソフトウェア開発ではなくコンサルティングであるが、成果物（ドキュメント）を作成→社内レビュー→顧客レビューという流れは、ソフトウェア開発業務と変わらない。

当プロジェクトは、約1年間で3～5名の参画のサイクルを数年間、同じ体制で実施してきた。

過去2年間、外的要因により、マスタスケジュールやWBSには表れにくい詳細な計画変更が頻繁に発生していた。このような状況の中で、作業分担の変更など柔軟な対応ができるよう、プロジェクトメンバー同士がお互いの作業や成果物を共有する必要があった。

(2) 改善前の状態

お互いの作業状況を共有するために、過去2年間、下記の試みを実施した。

- ・2年前（2010年度）：週報をメールで共有

- 書くのも読むのも時間がかかる。時間をかけた割には作業負荷の問題が見えにくい。

- ・1年前（2011年度）：週次ミーティングを開催

- ミーティングの度に議論になり、時間がかかる。

両者とも、並行しているタスクが多くさばききれなかった。このような状況により、結局、お互いの作業を共有することが難しかった。

(3) 改善前の状態をもたらした原因（因果関係）

上に述べた過去2年間の試みが、なぜ「お互いの作業を共有することが難しい」状態をもたらしたのか分析した結果、以下の原因が考えられた。

- 1) 「ある課題にフォーカスした検索」に向かない

プロジェクト内での「長期にわたる課題」について、その経緯を調べ、対策を検討したい場面がある。その場合に過去のメールや週報という膨大な文字情報から検索することは、ほぼ不可能に近い。週報や週次ミーティングは、その瞬間のスナップショット的な状況を把握するのには有用だが、ある課題にフォーカスして追跡するような検索には向かないと思われる。

- 2) 資料のレビュープロセスとバージョン管理の問題

あるタスクで作成した資料は、社内レビュー→顧客レビューというプロセスを経てクローズしていく。資料は、修正の都度、日付を付加した別のファイルとして保存していた。つまり修正するごとにファイルの数が増えていく。どのファイルが社内レビューの最終版か、現在そのファイルがどのような状況なのかを判別しにくい。

(4) 変更内容・対応策

そこで、プロジェクト内の情報共有の方法として、「チケットによるタスク管理」を導入することにした。弊社には、(1)で述べた Trac と Subversion が使えるため、これを利用することにした。

(5) 変更や対応策の実施内容

Trac や Ticket をどのように利用したかを、以下に述べる。

- 1) タスクのチケット化

マスタースケジュールレベルの進捗は従来通りの管理を踏襲したが、これまで週報やミーティングで共有していた、各自のタスクの1つ1つを、チケットとして登録した。チケットは、例えば「ある資料を完成させる」というタスクに対して1枚発行することにした。

- 2) チケットのカテゴリやステータスのカスタマイズ
チケットシステムは、元々ソフトウェア開発者が利用することを前提に作られていたため、当プロジェクトのタスクやステータス管理には合わない面もあった。そこで、まずチケットのカテゴリやステータスのカスタマイズを行った。
- 3) 資料を Subversion へ登録
作成した資料は Subversion に登録し、1種類の資料は基本的に1ファイルで管理、Revision 管理をすることで過去のバージョンも取り出せるようにした。
- 4) チケットと Subversion の連動
これは、システムで提供されている機能であるが、チケットから関連する資料へのリンクを貼ることができる。また、Subversion の各 Revision から関連するチケットへのリンクを貼ることができる。このように、タスクと関連する資料を関連づけられるようにした。
- 5) チケット単位にログを残す
1枚のチケットで扱うタスクについての議論は、メールではなくなるべくチケット内で行うようにした。チケットに書かれた内容は、関係者にメールで届くように設定することもできる。これにより、メールにも残せるようになる。逆にメールのみのやりとりはチケットに残らないため、チケットに転記し、タスク単位のログをまとめて閲覧できるようにした。

(6) 変更・改善後の状態と効果

「チケットによるタスク管理」により、以下の改善が得られたと思われる。

- 1) チケット番号や Revision 番号で話ができるように
〈Before〉
「〇〇案件の××作業について、△△書のレビューをお願いします。ファイルの場所は□□です」
→レビューは文字列を頼りに作業内容を思い出し、ファイルの場所をたどり、資料を探す。
〈After〉
「チケットの××番について、Revision△△番のレビューをお願いします。」
→レビューはチケット番号をクリックし、さらにチケット内の Revision 番号をクリックして資料を見るだけ。
- 2) チケット一覧で、タスクの状況が分かるように
ある特定の人タスク、あるステータスのタスク（例：未着手）、ある案件内のタスクなど、見たいチケットの絞り込みが容易にできるようになった。
- 3) 「ある課題にフォーカスした検索」が可能に
チケット単位にログを残していくため、ある課題にフォーカスした検索が可能になった。
- 4) 確実なレビュープロセス
チケット単位にレビュー状況を追跡できるようにしたため、各タスクについてのレビュー状況やそれに関連づけられた資料の状況について、情報を共有できるようになった。

(7) 改善活動の妥当性確認

「チケットによるタスク管理」については以下の課題があり、さらに検討が必要と思われる。

1) 運用の定着に関する課題

これまで、全ての手続きや連絡をメールで行ってきたメンバにとって、「やり方の変更」を受け入れるまでには時間がかかる場合もある。実際、変化に対応しにくく、どうしても従来の方法からしばらく抜けられないメンバもいた。「やり方の変更」を行う際には、その効果など「納得のいく説明」を十分に行うべきであった。

2) 予実管理には不向き

詳細なタスク管理や共有は可能になったが、「予定に対してどのくらい進んでいる／遅れている」は、見えにくい。予実管理についてはチケットは不向きであり、やはり従来のマスタスケジュールで管理するのが現実的と思われる。

3) 小規模プロジェクトならでは

このようなタスク管理は、3~5人程度での情報共有の手法としては効果的であると思われる。しかしそれ以上の規模になると、お互いの情報を共有するというより、グループを分け、グループごとに管理者を立て、階層的に管理するという、従来型の管理が妥当と思われる。

4) 成果の量を可視化できないか？

各チケットのタスクをクローズさせることにより、なんらかの成果が発生するはずである。各チケットの「成果の量」が可視化されていると、例えば「私は今月、〇〇円分の成果を出しました」などと言えるようになり、プロジェクトメンバの達成感も向上するのではないか。チケット管理により、このような副次的効果も期待できる。

以上

2C3「Android 端末中国 ODM 活用ノウハウ」林 潔(パナソニック)

<タイトル>

Android 端末中国 ODM 活用ノウハウ

<発表者>

氏名（ふりがな）：林 潔（リン ケツ）

所属：R&D本部 クラウドソリューションセンター

<共同執筆者>

・ 要点

弊社は、昔から生産委託を中心に海外 ODM を多く活用してきている。今後、Android 搭載のスマート端末が急速に普及し、Android 端末の開発で中国 ODM を活用することが必要となると思われる。Android 搭載の商品の素早い開発サイクルという特徴にあわせて、今までウォーターフォール型の開発プロセスとは違った、中国での開発に向けたやり方で開発を行う必要があると考えている。その開発方法を学ぶため、Android 搭載の商品の試行開発を、中国 ODM へ委託し実施した。試行中に発生した課題と、その原因、注意すべきことをまとめたので報告する。

・ キーワード

中国 ODM、Android 端末、海外オフショア、海外委託

・ 想定する聴衆

海外推進担当、海外委託のプロジェクトリーダー、オフショア開発

・ 適用状況

☒ 多用されている段階、☒ 適用できる段階あるいは初めて適用する段階

☐ 適用するにはさらに検討を必要とする、☐ 着想の段階

☐ その他（ ）

・ 適用可能性に関する制限

☒ 汎用性がある、

☒ 類似プロジェクトにも適用可：具体的な類似点（Android 開発端末）

☐ 自プロジェクトのみ

・発表内容

(1) 背景

弊社は、昔から生産委託を中心として海外ODMを多く活用してきている。昨今、Android搭載のスマート端末が急速に普及し、Android搭載の商品の開発に、中国のODMを活用する機会が増加している。Androidはプラットフォームのバージョンアップが頻繁で、ソフトウェア規模も大きいという特徴があり、日本でもなかなか開発がうまくいかないことが多い。さらに、中国ODMでAndroid搭載商品を開発する場合は、さらに多くの課題が生じる。

今回、うまく中国ODM活用するためのノウハウを取得するため、Android搭載端末の中国ODMでの開発を試行した。今後、得られたノウハウを実際のAndroid搭載の商品開発を行う事業場に展開する予定である。さらに、その試行を通じて、中国ODMの開発の「速さ」と「安さ」についても知見を得た。

(2) 試行プロジェクト概要

開発体制：日本の委託側は少人数（2名）で管理

開発テーマ：Android搭載タブレット試作

開発方針：

- ・中国のやり方を尊重し、できるだけ日本側から干渉しないようにする
- ・客観的に本来の実力を評価するめに、開発要件のみを提示する
- ・開発途中での課題発生時の支援は最小限に留める

評価項目：

- ・どれだけの短期間で開発できるか
- ・ソフトベンダーから主体的にEMS、部品ベンダーをコントロールできるか
- ・日本側から要求せずに、どこまで品質レベル達成できるか
- ・どれだけの費用でできるか

(3) 全体取組概要と考察

今回の中国ODMを利用した開発試行の、基本的な流れを以下に示す。



① 中国ODM選定

最初は、どのODMを選択すれば良いかの情報収集を行った。基本的に中国で有名なAndroid開発会社のリストを作成して比較表にまとめた。その中から選定した会社を訪問して、より詳細な調査を行った。

② ODM現地訪問

ODMへ訪問する場合、日本と中国ではかなり違うことを意識しておく必要がある。日本人は「性善説」で相手の話を基本的には全部信じる傾向がある。一方、中国企業は仕事を取るために一生懸命良いことをアピールする。自分達が一部しかできないことも、全部でできると宣伝するなど、良い面だけを強調して見せる。またできないことは隠して、わざわざ言うことはしない。

よって、会社訪問では正確に相手の実力を見破れる能力が必要となる。一つの方法としては、技術的な議論を徹底的に行い、相手の実力を把握することである。日本人は、初めてODMへ訪問する時は、遠慮して深く技術的な質問をせず、安易に仕事を委託して、後で大変な目にあうことがある。

又、中国企業は、打ち合わせにおいては、優秀なメンバーだけを集める、あるいは他社が使っているエビデンスを利用し、それを自社の成果として、日本側に紹介する場合もある。その場合は、本当かどうかの判断が極めて難しい。一つの対策として、開発現場を確認することが有効である。開発現場は嘘をつかないので、大体どのようなレベルの開発を行っているか、どれだけ忙しいかなどが分かる。情報セキュリティを理由に開発現場を見せない会社があるが、そのような会社は警戒した方がよい。

③ 見積もり交渉

見積もり交渉も重要である。人月単価が安いと思って委託するのは大間違いである。ODMに長年勤めていた副社長クラスの人によると、実際の開発では、見積もりした人月より半分しか人を投入していないことも場合も多いそうである。見積もりした人月を全部投入すると儲からないということである。一番理想な委託方法としては、人月で委託するよりも、プロジェクト全体を全て委託する方が、トータルで安い費用で実現できる。この場合、事前に要求する品質の水準を明確する必要がある。見積価格の交渉においては、今回の1回の開発だけの話ではなく、将来の委託の可能性や相手にもメリットがある話をするで見積もり交渉はしやすくなる。

④ 契約

実際の契約の際に、明確にしておかなければならないことは、全ての成果物（ソースコード、電子回路図、CADデータなど）が提供されるかということ。委託元は、全ソースコードを貰わないと、品質を向上するための修正作業を委託先のODMでしか実施できなくなる。一般的には、ODMが今回の委託で開発したソースコードは提供するが、ベースとなったソースコードを提供しない場合が多い。使っているチップベンダーから提供されたソースコードもODMから直接委託元に渡せないことも多い。

また、もう一つ明確にしておかなければならないのは、開発中で発生した特許権が誰に属するかである。その他、商品の出荷後の品質保証や為替リスクなど契約書で事前に明確すべき事項は多い。後で何か揉めた場合は、契約書を参照することで解決することも多い。

⑤ プロジェクト管理

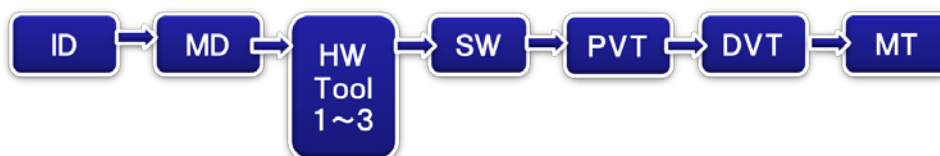
プロジェクト管理については、基本的に日本と同じでリスクや納期、品質の管理をしなければならない。注意しなければならないことは、小さい会社に委託すると、会社が倒産するリスクが大きくなることである。常に、その会社の倒産リスクを考えながら管理しなければならない。中国には、スマートフォンやタブレットの開発会社が2000社あり、特に、中国深セン市では、企業の入替りのスピードが早く、倒産も非常に多い。世の中のトレンドに合った企業を早く立ち上げ、失敗したらすぐ倒産させて、次の新しいチャンス

を掴むことが当たり前である。長く信頼関係を構築する日本企業の考え方とは違うことに注意が必要である。今回の試行、調査の間にも、従業員4000人規模の基板製造会社や2000人規模のEMS会社が売却されたり、会社のオーナーが交代することを良く目にした。基本的に中国ODMコントロールの一番良い方法はお金であり、最初に開発費の全額を払ってしまうと、コントロールが効かなくなる。最初は、全体の1/3の費用を払って、途中で品質の確認ができれば1/3払って、最後に品質水準を全部クリアできたら残金を支払うのが良い。

(4) 評価結果

・どれだけの短期間で開発できるか

- 基本的なタブレット開発フローは、以下の図の通りで、最初からデザインからハードウェア生産（金型3回回す）、ソフトウェア、プロトタイプ検証テスト、商品検証テスト、量産テスト工程である。今回試作までPVTまで行って、基本仕様を与えてから基本機能の動作まで4ヶ月掛かった。ハードウェアの試作納品を3回行って、徐々に性能が良くなった。最初の1回目の試作品は素早く納品されたが、Androidの基本画面起動できたレベルで、カメラ機能やタッチパネルがノイズ残る状態だった。それが2回目、3回目と試作納品を繰り返すことにより、品質が向上した。



開発のやり方も日本では主流のウォーターフロー型ではなく、フィーチャー駆動開発で、開発は「機能」を単位として、機能ごとにチームを作って進めていく。設計書などドキュメントよりも、動作するソフトウェアを重視し、作ってみてバグ発生したらすぐ修正する方法である。

・ソフトベンダーからチップベンダーやEMS全部コントロールできるか

- 基本的には、各部品メーカーからドライバを提供する。部品メーカーのサンプルソースコードと部品メーカーのエンジニアと一緒に実装とチューニングをすることとなる。部品メーカーの技術力とサポート力が、商品納期に影響する。小さいODMであれば、優先順位が下がり、納期が遅れる可能性が大きい。今回、プリント基板を大きな製造メーカーに頼んだが、量が少ないため、正式ラインではなく試作ラインで生産することになった。さらに順番待ちのため1ヵ月待たされることになった。そのため、ODMと一緒に、製造メーカーに我々の商品ビジョンと、将来大量生産の可能性があることを説明して、早く生産することができた。

・日本側から要求せずに、どこまで品質レベル達成できるか

- 日本から要求した品質レベルは達成できなかった。黙っているだけでは、中国側は品質には気をつかわないため、日本側がチェックしないとやらない。単機能のバグがあればすぐ対応するが、複数機能に跨るバグがあれば、バグの原因深く分析することができず、問題を解決する時間が掛かった。また、顧客優先

の意識が弱くて、途中成果物での品質は気にしなかった。顧客満足を実現するため、優先順位を決めて対応することができなかった。

・費用が日本と比べてどのくらい安くできるか？

- － 人件費だけから見ると半分安く実現できたが、日本側から現場の管理、技術指導行って、トータルではおおよそ日本の1/3の費用を削減することできた。ただ、今までやったことがない技術、特別な部品を利用する場合などは、日本で開発したほうが安い。

(5) ODM活用の考察

中国ODM活用に向けた方法は以下と考える

- ・ 数量の多い商品を開発する場合は、中国で一流のODMに委託する方が安全、安心
- ・ 試作プロジェクトあるいは要件が決まらない場合は、中小規模のODMへ委託する方が、仕様変更に柔軟に対応してもらえる
- ・ 基本的には、中国ODMが開発してきた資産をそのまま利用し、特別な注文（特殊仕様など）を可能な限りしない場合の方がメリットは大きい

(6) これから予定

まとめたノウハウを社内の事業場の実際の商品開発に適用し、ノウハウを展開していく

2C4「文化背景の異なる拠点間におけるプロセス改善の取り組み」田村朱麗(東芝)

〈タイトル〉文化背景の異なる拠点間におけるプロセス改善の取り組み

〈サブタイトル〉文化差異によるコミュニケーション問題に対する解決ステップの提案と実践

〈発表者〉

氏名（ふりがな）： 田村 朱麗（たむら しゅれい）

所属：株式会社東芝

〈共同執筆者〉

氏名（ふりがな）：小島 昌一（こじま しょういち）

所属：東芝ソフトウェア・コンサルティング株式会社

・ 要点

海外拠点のプロセス改善の推進は、一般的な推進上の問題に加えて、海外拠点特有のコミュニケーション上の問題により、改善推進が計画通りに進まないことが多かった。後者の問題を解決するために、文化差異によるコミュニケーション問題に対する解決ステップを考案した。本ステップの特徴は、ホフステードの文化的次元を導入し、優先付けして利用している点である。本ステップを、ある海外拠点のプロセス改善推進に適用することにより、その有効性を確認した。

・ キーワード

プロセス改善、海外拠点、ホフステードの文化的次元、コミュニケーション問題

・ 想定する聴衆

海外拠点のプロセス改善活動の推進に携わる方

・ 適用状況

☐多用されている段階、☒適用できる段階あるいは初めて適用する段階

☐適用するにはさらに検討を必要とする、☐着想の段階

☐その他（ ）

・ 適用可能性に関する制限

☐汎用性がある、

☒類似プロジェクトにも適用可：具体的な類似点（海外拠点のプロセス改善推進）

☐自プロジェクトのみ

・ 発表内容

(1) 背景

筆者らは、プロセス改善の専門家として、国内外の当社グループ会社のプロセス改善を推進する立場にある。海外拠点のプロセス改善の推進は、一般的な推進上の問題に加えて、海外拠点特有だと思われる問題のために、改善推進が計画通りに進まないことが多かった。

(2) 改善前の状態

海外拠点のプロセス改善の推進は、計画よりも改善の進捗は遅れ、かつ、筆者らの推進工数も増大する傾向にあった。(プロセス改善を計画通りに進めるとともに、筆者らの推進工数も、計画工数内に収めたかった。)

(3) 改善前の状態をもたらした原因 (因果関係)

図 8 に示すように、海外拠点のプロセス改善活動を推進する上での問題を抽出し、その因果関係を整理した。そして、この中で、手を打つことができることで、かつ、解決すると最も効果が見込める『文化・背景の違い』によるコミュニケーションギャップに着目することにした。

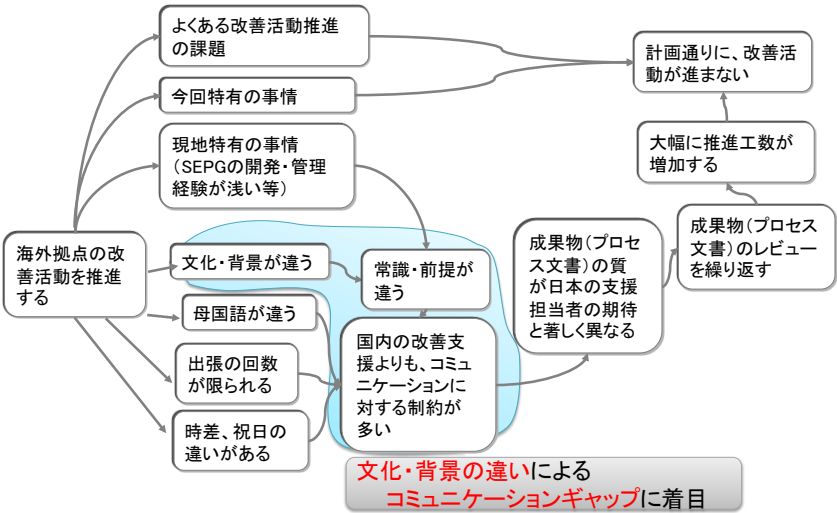


図 8 海外拠点のプロセス改善を推進する上での問題

上記を整理するに当たっては、改善手法として知られている SPINA3CH 自律改善メソッドの手順やツール類[1]を利用した。

(4) 変更内容・対応策

(3) で着目した問題を解決するために、表 1 に示す文化差異によるコミュニケーション問題の解決のステップを考案した。このステップは、『Step2 コミュニケーション上の問題の要因分析』に、文化差異の理解のフレームワークとして広く知られるホフステードの文化的次元[2][3][4]を導入し、優先順位付けして利用することを特徴としている。文化的次元には、『権力格差』『個人主義対集団主義』『男性らしさ女性らしさ』『不確実性の回避』『長期志向対短期志向』『気ままさ対自制』がある。

表 1 文化差異によるコミュニケーション問題に対する解決ステップ

Step1	コミュニケーションにおける現象と問題の明確化
Step2	コミュニケーション上の問題の要因分析 ① ホフステードの各文化的次元について、対象国の特徴および対象国と日本との文化差異をベースに、Step1 で明らかになったコミュニケーション上の問題の要因を分析する ② 着目する文化的次元を絞り込む
Step3	解決方針の決定
Step4	解決策の立案
Step5	解決策の実施

(5) 変更や対応策の実施内容

ここでは、(4)で考案したステップを、ベトナムのソフトウェア開発拠点のプロセス改善推進上の問題解決に適用した事例を示す。

＜組織の特徴＞ 技術者の平均年齢が若く、ソフトウェア開発・管理経験が浅い

＜改善計画＞ 高品質ソフトウェアを開発するために、CMMI 成熟度 3 達成を目指す

＜改善推進体制＞ 開発拠点の SEPG(以下、被支援側)を、筆者ら(以下、支援側)が支援する

Step1: コミュニケーションにおける現象と問題の明確化

被支援側と支援側のやり取りの大部分は、被支援側が作成した成果物を支援側がレビューすることであった。支援側は、被支援側に対して『開発拠点の事情を考慮した上で、コメントを取捨選択して反映すること』を期待していた。しかし、当時、被支援側は、支援側である筆者らからのアドバイスやコメントを、ほぼ無批判に受け入れていた。そのため、被支援側がレビューコメントを開発拠点の事情を考慮せずに成果物へ反映した後に、支援側が再びコメントするというサイクルを何度も繰り返していた。これは、工数の増大、作業の遅れを引き起こしていただけでなく、支援側・被支援側双方にとってストレスになっていた。

Step2: コミュニケーション上の問題の要因分析

筆者らが直面していたコミュニケーション上の問題と、ホフステードの各文化的次元との関係性を調べ、着目する文化的次元を上位者と下位者¹の権力格差の程度を表す指標である『権力格差』に絞り込んだ。『権力格差』により、支援側が直面していた状況を次のように解釈することができた。

図 9 の通り、ベトナムは、日本よりも上位者と下位者の権力格差が大きい。支援側と比べ、被支援側であるベトナムの改善推進者は、相対的にスキル・知識・経験が浅く年齢も若い。このことから、被支援側から見た支援側と被支援側の関係は、図 9 のベトナムにおける上位者と下位者の関係に類似していると考えた。現地の技術トップ(ベトナム人)も、この解釈は妥当という意見であった。



図 9 日本とベトナムの上位者と下位者の権力格差

¹ 上位者と下位者：ここでいう上位者と下位者とは、上司と部下、発注者と受注者、先生と生徒、年長者と年少者など、権力の強い者と権力の弱い者を指している。

Step3: 解決方針の決定

解決方針としては、『一方が他方に合わせる(支援側が被支援側に合わせるまたは、被支援側が支援側に合わせる)』『双方が歩みよる』がある。

支援側としては、改善目標を計画通りに達成するためには、被支援側であるベトナムの開発拠点の SEPG がプロセス改善の推進やそのための知識・スキル習得に専念してほしかった。なので、今回のケースでは、Step2 で絞り込んだ『権力格差』の支援側と被支援側の文化差異に基づく解決するために、解決策の実行による被支援側への負荷が最も少ない『支援側が被支援側の価値観や思考パターンに合わせる』という方針を採用した。

Step4&Step5: 解決策の立案とその実施

Step3 で採用した解決方針に基づき、被支援側と支援側のやり取りの大部分を占めていた被支援側の作成した成果物に対するレビューコメント方法を表 2 のように変更した。また、表 2 の変更を円滑に実施するために、開催頻度が高くかつ被支援側が作成した成果物のレビュー結果について議論することが多かった Web 会議の運営方法を表 3 のように変更した。

表 2 レビューコメント方法の変更

変更前	気づいたことは重要度にかかわらずコメントしていた。
変更後	レビューコメント等は重要なことに絞るなど、「言い過ぎない」ように配慮した。 特にプロセス改善に関する重要なポイントについては、質問や議論を促すなど、被支援側の理解を促進する工夫をした。

表 3 Web 会議の運営方法の変更

変更前	Web 会議は、可能ならベトナム側の SEPG が主導するように促していた。
変更後	影響力の強さを有効に活用する 1 つの手段として、Web 会議では日本の支援担当者が主導した。

(6) 変更・改善後の状態と効果

(5) の実践事例において、文化差異によるコミュニケーション問題に対する解決ステップを適用することにより、支援側と被支援側の関係を客観的に捉え、対策を立てることができた。これにより、双方のコミュニケーション上の問題を解決し、効率的なプロセス改善が可能になり、ストレスも軽減した。その結果、この組織は、当初の計画通りに、CMMI 成熟度レベル 3 を達成した。

(7) 改善活動の妥当性確認

実践事例において、考案したステップを適用することにより、被支援側が作成した成果物のレビューコメント対応のための後戻り作業が減少し、支援側の改善推進工数を削減できた。後戻り作業の減少は、改善前後の『成果物当たりのレビューコメント対応のための改訂回数』を比較することで確認した。改善推進工数の削減は、改善前後の『改善推進工数の超過率』²を比較することで確認した。その結果、改善活動の遅れを挽回し、当初の目標を達成することができた。なお、考案したステップにより導出された解決策の立案や実行準備にはほとんど費用がかからなかった。

今後は、考案したステップを、必要に応じて、ベトナムにおけるコミュニケーション問題や、中国、インド等、ベトナム以外の国とのコミュニケーション上の問題の解決に適用する。事例を蓄積することで、ステップが、汎用的に有効であることを示していく。

² 工数超過率 = (実績工数-計画工数) / (計画工数)

(8) 参考文献

- [1] SPINA³CH 自律改善メソッドガイドブック、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
- [2] 経営文化の国際比較—多国籍企業の中の国民性、Geert Hofstede、産業能率大学出版部、1984
- [3] 多文化世界、Geert Hofstede、有斐閣、1995
- [4] Hofstede's Cultural Dimensions: <http://geert-hofstede.com/dimensions.html>