

プロセス改善セミナー 事例紹介



独立行政法人 情報処理推進機構
技術本部 ソフトウェア・エンジニアリング・センター

我々は、2006年より独立行政法人情報処理推進機構（IPA）ソフトウェア・エンジニアリング・センター（SEC）エンタプライズ系プロセス改善ワーキンググループ（以降、プロセス改善WG）の中で、ソフトウェア開発組織の能力向上を目的に、各社のソフトウェア開発プロセス改善活動（以降、SPI活動）から得られたベストプラクティスを広く紹介する活動を実施しています。これは、成果の確認された事例を知ること、一人でも多くのソフトウェアエンジニアの方が、さらに効率よく品質の高いソフトウェアの開発に向けた取り組みにチャレンジしていただけるのではないかと、という考えから始めたものです。

最初の取り組みとして、2008年に10社のSPI活動の事例を、「プロセス改善ナビゲーションガイド ～ベストプラクティス編～」として出版しました。これは、書店はもとよりIPA/SECのイベントなどを通じて、延べ5,000冊がみなさまのお手元に届きました。この冊子に詰まったSPI活動のエッ

センスを取り出し、みなさまの組織やソフトウェア開発プロジェクトへ適用し、また継続して、さらに進化させることで確実にSPI活動の成果が得られる

はじめに

という事実を、多くの方に知っていただけたと認識しております。

しかしながら、実際にSPI活動に取り組んで成果を得るためには、プロセス面や技術面などの知識を獲得した上で、それぞれの組織に適応させる必要があります。いわゆるSPI活動のテーラリングです。組織への適応が上手くできず、活動が頓挫して失敗する事例がしばしば見受けられます。当プロセス改善WGでは、一方向の情報提供ではなくインタラクティブなやりとりを通じて、必要な知識に加えて実践的ノウハウを習得するための仕掛けや活動が必要であるとの結論に至りました。

そこで、2009年から各社で成果の上がった事例を実際に推進した方にご紹介いただき、その活動を進める上で特に注意した点や苦労した点、今後の展望などについて、参加者のみなさまとワークショップ形式で進めるセミナーを開催しています。

本セミナーでは、ワークショップをより充実したものにするため、次のような工夫も行いました。

- ・参加申し込み時に、議論したい内容をポジションペーパーとして提出
- ・セミナー開催前に、メーリングリストにて講師を交えての意見交換
- ・セミナー終了直後に、講師と参加者による意見交換会の開催
- ・セミナー後に、メーリングリストでの質疑応答と情報交換
- ・1か月後に、活動状況のフォローアップ

2時間という限られた中でのセミナーですが、参加していただいた方々には冊子を読んだだけでは得られない「多くの生きた情報」を得ていただけたと確信しております。

このたび、それらの記録を「プロセス改善セミナー 事例紹介 ～ベストプラクティスワークショップ編～」としてまとめました。このガイドは、次のような構成になっています。

- ① セミナーの開催実績と取上げた手法一覧
- ② セミナーの実施概要
- ③ セミナーで用いた発表資料

ワークショップに参加いただけなかった方々にも、その一端に触れていただくことで、SPI活動に対する理解と興味を一層深めていただければ幸いです。

2012年4月
プロセス改善 WG NPT4 一同

CONTENTS

プロセス改善セミナー 事例紹介

～ ベストプラクティスワークショップ編 ～

セミナー開催実績	1
セミナー概要	3
セミナー講演資料	33
終わりに	73

セミナー開催実績

No.	タイトル	手法・ツールに関するキーワード	概要
01	組み合わせ技術を利用したテストケース生成ツールと適用事例の紹介 株式会社東芝 小笠原 秀人 氏 / 中野 隆司 氏	組合せテスト、直交表、All-Pair 法、テストプロセス	3-4 ページ
02	モデル駆動開発に向けたモデルチェック技術の導入 富士ゼロックス株式会社 服部 彰宏 氏 / 山本 訓稔 氏	モデル駆動、モデル検査、検証モデル、Promela、xUML、SPIN、モデルの資産化	5-8 ページ
03	フィールドワークを応用したプロセス改善の継続方法 富士通株式会社 平田 貞代 氏	行動分析、文化人類学、チェンジ・フィールドワーク、非言語、ファシリテーション	9-10 ページ
04	XDDP を使ったプロセス改善 株式会社デンソー技研センター 古畑 慶次 氏	派生開発、ソースコード流用、SW 開発プロセス、要求仕様記述法、XDDP、USDm	11-12 ページ
05	dSPO の手法を学ぶ 株式会社ニルソフトウェア 伊藤 昌夫 氏	SW 開発プロセス、最適化、HAZOP、SPO マニフェスト、Immaterial Labor	13-14 ページ
06	アジャイルインスペクション ソニー株式会社 永田 敦 氏	品質の早期作り込み、欠陥混入防止、コスト抑制、サンプリング、インスペクション、ロギング	15-18 ページ

No.	タイトル	手法・ツールに関する キーワード	概要
07	レビュー・インスペクションの効果的実践と 阻害要因 QI 法の実践事例の紹介 日本アイ・ビー・エム株式会社 細川 宣啓 氏	SW 成果物、品質検査、 QI (Quality Inspection)、 第三者レビュー、品質バ イタルサイン	19 - 22 ページ
08	品質会計を軸としたパートナー QC レビューの 紹介 (PQR : Partner Quality control Review) NEC ソフト株式会社 後藤 徳彦 氏	品質会計、プロセスアセ スメント、全社展開	23 - 24 ページ
09	ソフトウェアレビュー改善の着眼点 ～ 段階的な取り組みにむけた事例分析 ～ 静岡大学 情報学部 / 奈良先端科学技術大学院大学 森崎 修司 氏	コードレビュー、レ ビュー視点、レビューの 効率化	25 - 26 ページ
10	ソフトウェアレビュー改善の着眼点 ～ 段階的な取り組みにむけた事例分析 ～ (その 2) 静岡大学 情報学部 / 奈良先端科学技術大学院大学 森崎 修司 氏	第三者レビュー、プロ ジェクト・レビュー、 レビューアの教育、レ ビューの観点、レビュー の効果、レビューの終了 条件	27 - 30 ページ
11	品質保証活動を形骸化させないコツ ～ プロセス改善に魂を込めるには ～ ヤマハ株式会社 小池 利和 氏	形骸化、意識向上、SQA、 SEPG、プロセス監査、品 質メトリクス、測定と分 析の活用、データの精度	31 - 32 ページ

2009年3月27日

組み合わせ技術を利用した テストケース生成ツールと適用事例の紹介

講演者：小笠原 秀人氏、中野 隆司氏（株式会社東芝）



概要

ソフトウェアの組み合わせテストを網羅的かつ効率的に実施するための手法として、あるシステム開発部門のテストプロセスに、直交表や All-Pair 法といった組み合わせ技術を導入し、ソフトウェアテストのプロセス改善に取り組んだ事例が紹介された。

前半は、この手法が必要とされた背景および手法の概要説明が行われた。手法が必要とされた背景として、次の2点が挙げられた。

- ・テスト漏れによるリリース後の不具合が皆無にならない
- ・テスト経験が長い担当者に頼っている

これらの課題を解決するため、テストプロセスの改善に取り組んだシステム開発部門では、社内で組み合わせ技術を活用したテストケース生成ツールを開発し適用している。このツールは、テスト対象の因子と水準を抽出後、禁則条件（実現できない組み合わせ）を設定すると、3因子間の網羅度を計測するとともに、2因子間の網羅度 100% を満たすテストケースを自動で生成することができる。

これらの手法、ツールを適用することにより、テストケースとして出現する水準に偏りがなくバランスの良い網羅的なテストケースが生成され、効率的なソフトウェアテストが実現できるようになった。

後半は、直交表や All-Pair 法の使用経験者および未経験者それぞれからの質問をうけて、講師と参加者によるディスカッションが展開された。

年々ソフトウェアの複雑度が増すなか、組み合わせ技術を用いた改善に取り組んでいる参加者も多く、実践経験や現場での悩みをふまえた質疑が多く寄せられた。その中から幾つかの発言を取り上げる。





- ・ベテランのノウハウを伝承する仕掛けとして、組み合わせ技術を用いたテストプロセスの改善に取り組み始めた。ベテランのノウハウに加え、網羅性を保証できる体系的な手法が有効である。
- ・テスト対象の因子を多し過ぎると、直交表を使わない場合と負荷が変わらなくなるため、あまり増やさないよう指導している。
- ・直交表は、要件が固まり基本設計が出来たあたりで作成している。早めに作

成することで仕様の漏れに気づくことができ、仕様にフィードバックすることができる。


このように、取り組みの背景や活用に当たってのコツやノウハウもディスカッションを通して披露され、今回のテーマを切り口にテストプロセスの改善に役立つヒントを得ることができた。

参加者の声





セミナーの有意義度

 まあ有意義	10人 (63%)
 有意義であった	4人 (25%)
 普通	2人 (13%)
 有意義でない	0人 (0%)

セミナーの満足度

 まあ満足	12人 (75%)
 満足	2人 (13%)
 普通	2人 (13%)
 不満	0人 (0%)

セミナーの活用度

 普通	7人 (47%)
 まあ活用できる	6人 (40%)
 活用できる	2人 (13%)
 活用できない	0人 (0%)

直交表や AllPair 法に関する知識はないが、興味が沸いた。一度じっくり調査して、良さそうなら活用したいと思う

テストメンバのみなさんが同じ悩みを共有しているのがわかった

ポイントが分かれば適用できるケースがありそう

設計段階で開発の人にもテストを意識してもらうことで、今以上に品質の良いものが出れると思った



企画担当者のご声

本セミナーでは、直交表や All-pair 法といった組み合わせ技術を利用したテストの方法について、講師が開発したツールおよびその適用事例を中心に紹介いただいた。具体的かつ実践的な分かりやすい内容であったため、業務に活用できるとの意見が多かった。セミナー後のアンケートにおいても半数以上の参加者が業務に活用できると回答していることから、具体的な内容であったことがうかがえる。

また、特定のツールや技術を過度に強調するのではなく、テストのための活動を開発プロセス全体に渡って実施していくことの必要性や、組み合わせ以外のさまざまな視点からのテストの必要性が説明されており、事例紹介およびディスカッションと、いずれの内容も現実を踏まえた実践的なものであった。



2009年3月27日

モデル駆動開発に向けた モデルチェッキング技術の導入

講演者：服部 彰宏氏 山本 訓稔氏（富士ゼロックス株式会社）



概要

組込みソフトウェアの開発で、ユーザ操作やデバイスの動作タイミングによって発生するタイミングに依存した不具合は、再現性が低い傾向があり、調査が長期化して対応工数が増大する。

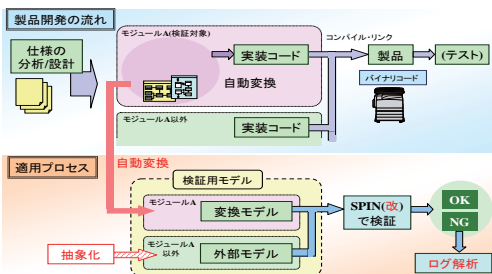
デッドロック、通信交錯、制御設計ミスなどの不具合検出にモデルチェッキング（以降モデル検査）技術を導入し、ソフトウェア開発プロセス改善に取り組んだ事例が紹介された。

富士ゼロックスでは、複合機プリンタの組込みソフトウェア開発において、ユーザ操作やデバイスの動作タイミングによって発生する不具合の再現性が低いため、不具合原因の調査が難航し、開発期間が長期化している課題を克服する必要があった。そこで、テスト期間を短縮することで開発期間の短縮を図ろうと考え、モデル検査技術への漠然とした期待を抱きつつ、小集団活動という自発的な勉強会を立ち上げ、モデル検査による開発プロセス上の課題解決に取り組んだ。

モデル検査の導入にあたり、次のような目的と条件を設定し、実機での検証を試みた。

- ・モデル検査でないと見つかりにくい不具合を1件だけでも抽出する
- ・デッドロック /Assert/ メッセージクロス / インスタンス寿命を検証する
- ・実開発のプロセスとして許容できる検証時間であることを確認する
- ・検証対象モジュール：サービス側ソフトウェアと機構制御側ソフトウェアの調停ソフトウェア（約6万行）
- ・検証対象機能：簡単な印刷

●適用プロセスの概要



モデル検査の試行により、2件の不具合^(*)を数分程度の検証時間で発見した。

^(*) 予期しないタイミングのイベント受信がシステムハングアップを引き起こした。発生頻度は低いが高ストレス状態で発生する可能性がある。

モデル検査導入により、予期しないタイミングによる不具合が短時間で発見できたことから、モデル検査の有効性は確認されたが、実用に向けた更なる課題への対応も必要となった。

- 検証モデルの拡大
 - 外部モデルは何度も仮検証を実施して信頼性を向上
 - 細分化
 - 統合による検証用モデルの作り込みプロセス
- 複数個の不具合を検出
 - 同値不具合^(*)2)を自動選別して報告しないようなフィルタを作成
- 実機テストとモデル検査との住み分け
 - 実機テスト：人間が現実的にオペレーションできるもの / 人間が精神的に耐えられるもの
 - モデル検査：デッドロック / インスタンス残り / Assert で検出 / タイミングに関連

モデル検査によって発見すべき不具合は、実機上でのテストが困難である。例えば、次のようなものが挙げられる。

- ▽ デッドロック / インスタンス残り (メモリ浪費) / タイミングに関する不具合
- ▽ マイクロまたはミリ秒の通信隙間を確認するテスト
- ▽ テスト環境構築が困難なテスト
- ▽ テスタが耐えられないテスト

参加者からの質問は、Promela^(*)3)のモデルに関する技術的な話題、モデルの資産化に関する開発プロセス上の話題、モデル検査を行っているグループのスキルに関する話題、取組みの短所に関する話題など、多岐にわたるものであった。

セミナーでは、以下のような質問があった。

- どのような考えで抽象化するのか？
- xUML^(*)4)のモデルを Promela のモデルに変換する場合、xUML の状態と Promela の状態が 1 対 1 となるのか？
- 開発体制、チームの規模、スキルはどんなレベルなのか？
- SPIN^(*)5)を 2005 年から活用しているのは結構早い方だと思うが、大学の修士レベルで経験者がいたのか？
- K 君がモデル検査を適用しようと考えたきっかけは、不具合が出たときの原因追求を SPIN でやってみようと思ったということか？
- モデルの資産化としては、どう捉えたらよいのか？



参加者の声

セミナーの有意義度		セミナーの満足度		セミナーの活用度	
まあ有意義	4人 (50%)	まあ満足	4人 (50%)	まあ活用できる	3人 (38%)
有意義であった	3人 (38%)	満足	3人 (38%)	活用できる	2人 (25%)
普通	1人 (13%)	普通	1人 (13%)	普通	2人 (25%)
有意義でない	0人 (0%)	不満	0人 (0%)	活用できない	1人 (13%)

モデル駆動開発→モデルチェックの内容が非常に分かりやすく参考になった

実践的で有意義な内容だった。また実践的な視点での同様のワークショップを希望する

ハードモデル検証、検証プログラム生成としては、活用へのハードルが高いと感じた



企画担当者のひとこと

本セミナーは、モデル駆動開発にモデル検査を取り入れた開発プロセスの構築に関して、複合機プリンタの非同期制御検証への適用事例を題材にし、モデル検査を導入するための技術上、プロセス上の課題と施策についての紹介および参加者との積極的なディスカッションが行われた。

◆参考情報

さまざまな形式手法の分類や概要説明の資料としては、国立情報研究所の教授である中島 震氏の資料が参考になる。

- ・ソフトウェア工学の道具としての形式手法
<http://research.nii.ac.jp/TechReports/07-007J.pdf> (As of 2010/03/08)

^{(*)2} 同値不具合：到達経路は異なるが本質的に同じ不具合
^{(*)3} Promela (Protocol / Process Meta Language)：モデル記述言語
^{(*)4} xUML：ExecutableUMLはUMLのひとつの拡張であり、単一の問題領域の振る舞いを実行可能なレベルで十分詳細に定義できるようになるモデル記述言語である
^{(*)5} SPIN (Simple PromelaInterpreter)：開発者はG.J. Holzmann。Promelaで書かれたモデルの検証を行うツール



こちらの講演資料は都合により不掲載

2009年9月14日

フィールドワークを応用した プロセス改善の継続方法

講演者：平田 貞代氏（富士通株式会社）



概要

本セミナーのテーマは、文化人類学で用いられるフィールドワークをもとにして開発された「チェンジ・フィールドワーク」という手法をプロセス改善に活用してみようというものであり、この手法によって「やらされ感」のないプロセス改善活動を推進してきた事例が紹介された。

これからプロセス改善に取り組もうとしている組織、すでにプロセス改善に取り組んでいるが停滞している組織、ある程度プロセス改善の効果が出ている組織のいずれにも、チェンジ・フィールドワークの考え方を取り入れることで、新たな改善のポイントが発見される可能性が高いという内容が紹介された。

講義内容としては、フィールドワークの活用領域、また富士通で開発されたチェンジ・フィールドワークの手法と具体的な実施ステップが説明され、それに基づいた2件の実践事例と効果が紹介された。

セミナーでは、以下のような質問があった。

- ・フィールドワーカーに適した人物像とは？
- ・問題意識や感受性が高い人でないと改善のポイントは発見できないか？
- ・現場の邪魔にならないか？
- ・どのくらいの期間が必要か？
- ・フィールドワーカーの養成セミナーはあるか？
- ・仮説は立てるか否か（インタビューは準備するか）？

セミナーの冒頭に、隣り同士の参加者の会話時間を5分程度設け、質問や発言をしやすい雰囲気作りを行い、和やかに講義が始まった。アイスブレイクが効いたのか、チェンジ・フィールドワークを実践する上での鋭い質問が飛び出すこともあり、穏やかな流れの中にもメリハリの効いたあっという間の2時間であった。

メールリスト (ML) の議論





セミナー開催の1週間前からMLの運用が開始され、セミナーで用いる資料が事前配付された。

資料を見た参加者から「フィールドワークとは何か」、「チェンジ・フィールドワークとは何か」という質問がMLへ投げられたのを皮切りに、「プロセスを形骸化させないための方法」、「やらされ感の払拭」、「メンバの主体性」、「自発的なプロセス改善意識」など、現場で悩んでいる質問が次々と出され、セミナーへの関心の高さがうかがわれた。





MLには、セミナーの開催前後で47件の投稿があった。

参加者の声





セミナーの理解度

 良く理解できた	13人 (54%)
 少し理解できた	11人 (46%)
 あまり理解できなかった	0人 (0%)
 No 理解できなかった	0人 (0%)

セミナーの満足度

 満足	11人 (46%)
 とても満足	9人 (38%)
 やや不満	4人 (17%)
 No 不満	0人 (0%)

ディスカッションの満足度

 満足	13人 (57%)
 とても満足	5人 (22%)
 やや不満	5人 (22%)
 No 不満	0人 (0%)

インタラクティブな進め方で、理解が促進された

普通のやり方とのギャップが大きく、目から鱗の感覚だった。これまでのやり方を否定するものではないと思うので、意図を理解して良いところを取り込んでいきたい

この密度でもう少し時間が欲しかった

インタビューのやり方、非言語のメモなどが非常に参考になった

企画担当のひとこと



本セミナーは、富士通社内実践されている「フィールドワークを応用したプロセス改善の手法」の事例紹介である。2時間の枠の中に講義、Q&A、ディスカッションを盛り込んだため、やや時間が足りなかったように思われる。もう少し講義の内容を絞り込み、ディスカッションに多くの時間を割り当てた方がよかったかも知れない。

インタラクティブな進め方による平田講師のファシリテーションが素晴らしく、受講者の方には楽しく過ぎていただけたと思う。また、今回から始めた講演資料の事前配付によりテーマに対する理解度が深まり、講義開始直後から活発なディスカッションが展開されていたように感じられた。



2010年2月5日



XDDP を使ったプロセス改善

講演者：古畑 慶次氏（株式会社デンソー技研センター）

概要

前半は、デンソーでの XDDP（eXtreme Derivative Development Process）の導入の歴史から始まり、XDDP とこれまでの派生開発方法の違いや、XDDP の導入効果が説明された。講師が、デンソーや関連会社の設計部門と支援部門に対して取り組んだ導入支援における様々な工夫も合わせて紹介した。

後半はディスカッションが行われ、ワークショップ開催前に、メーリングスト上で 180 通のメールが飛び交っていたこともあり、ディスカッション時には、まるで皆が顔見知りであるかのように、ごく自然に深い議論に入ることができた。

ワークショップから、XDDP の生みの親である清水 吉男氏が参加し、議論は更なる盛り上がりを見せた。ワークショップ終了後の懇親会には多くの参加者が出席し、引き続き XDDP やプロセス改善についての議論を深めることができた。

メーリングリスト（ML）の議論

講師の講演資料を事前に公開し、参加者から提出されたポジションペーパーの内容をフィードバックするなど、参加者間での問題意識を喚起させる工夫がなされた。ML のファシリテータとして投稿したメールに対し、参加者から寄せられた意見を小まめに整理し回答を行ったことにより、多くのメールが意見交換に使われた。主な ML での議論は次のとおり。

- ・派生開発の定義
- ・XDDP に適用する見積り
- ・TM（Traceability Matrix）の列項目
- ・XDDP の導入経緯と工夫
- ・組織への XDDP の展開方法
- ・変更要求仕様書／追加要求仕様書／変更設計書のレビュー
- ・適用プロジェクトのプロファイル情報

- XDDP の指導者の育成
- XDDP の継続方法
- XDDP の効果
- 上手くいっているプロジェクトと XDDP のプロセスの類似点
- XDDP の単体テスト
- XDDP で顧客満足度を実現
- 支援部隊の熱き想い (熱意)
- XDDP は銀の弾丸
- XDDP とトヨタ生産方式



参加者の声

セミナーの理解度

良く理解できた	21人 (81%)
少し理解できた	5人 (19%)
あまり理解できなかった	0人 (0%)
理解できなかった	0人 (0%)

セミナーの満足度

とても満足	13人 (50%)
満足	13人 (50%)
やや不満	0人 (0%)
不満	0人 (0%)

ディスカッションの満足度

満足	17人 (68%)
とても満足	6人 (24%)
やや不満	2人 (8%)
不満	0人 (0%)

実際に使われている部門の方の話は参考になる

追加資料で説明していただいて、理解が深まった

もう少し時間があれば、さらにいろいろな疑問点、意見を聞けたかも

TM の使い方や導入など疑問に思っていたことが理解できてよかった



企画担当者のひとこと

本セミナーは、XDDP と銘打ったが、清水 吉男氏が考案した派生開発におけるプロセス改善の事例である。XDDP は、エンタプライズ系のエンジニアには馴染みが薄いようだが、組込み系の現場では実際に数多く使われている手法であり、すでに幾つかの組織で実績も出ているため事例として取り上げた。

派生開発には、XDDP、PFD、USDM の 3 つの技法が内包されているが、今回は XDDP と USDM の紹介にとどめている。セミナー参加者には、初心者と経験者の両方がいたため、初心者が問題を提起し、参加者全員で議論し、解決に至る場面も見受けられた。



2010年7月15日

dSPO の手法を学ぶ

講演者：伊藤 昌夫氏 株式会社ニルソフトウェア



概要

dSPO (dependability Software Process Optimization) の基礎を構成する SPO(Software Process Optimization) という考え方を、伊藤昌夫氏に講義していただいた。

SPO とは、伊藤氏が提唱している開発プロセスを改良する方法であり、自分たちがとり得る最適な仕事のやり方という意味をもつ。下記の2点に該当する場合にプロセスを見直すものであり、ゴール・制約条件・達成基準・必要資源などを明確にし、達成基準に満たないプロセスを変更対象とする。

- ・現在実施されている開発プロセスは、ある制約条件の中で最適化されたプロセスとなっていない
- ・現在のプロセスで、変更したい部分がある

SPO を構成するステップとして、次の4つがある。

- ・全員参加による計画 — Participatory Planning
- ・ミッション定義 — Mission Definition
- ・閉世界モデル / 開世界モデル — Closed-World Model / Open-World Model
- ・解決 — Resolving

そして、すべてのステップに関連する項目として次の2つがある。

- ・モニタリングと評価 — Monitoring/Evaluation
- ・ライブラリ — Library

現在の開発プロセスの変更にあたって、ゴール、制約条件、達成基準などを明確にし、効果を計測しながら、いくつかの段階の手順・調整を繰り返すことで組織やプロジェクトが設定した目的や効果を満たす最適なプロセスに到達できる。

このように、既存の改善手法の適用という発想ではなく、自前の改善の枠組みを創ってプロセスを改良していくという考え方に触れる事ができ、新たな気づきが得られた2時間であった。



メーリングリスト (ML) の議論

ポジションペーパーと講演資料を事前公開し、参加者間での問題意識を喚起させる工夫がなされた。講演資料が英文だったためか、ML への投稿が少なかった。議論された話題は以下の通り。

- dSPO & HAZOP^(*) について
- SPO マニフェスト^(*)
- Immaterial Labor (無形労働)^(*)

HAZOP 以外は新語に近く、勉強になったという回答があった。

^(*) HAZOP (Hazard and Operability Studies) : プロセスパラメータの目標値 (目標状態) からのずれを想定し、そのずれの起こる原因と発生する危険事象を解析し、さらにその原因から危険事象に進展するのを防護する機能を評価し、対策を検討するというもの。

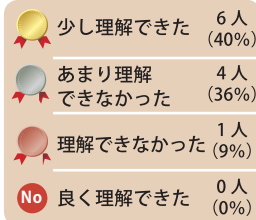
^(*) SPO マニフェスト (ソフトウェア最適化宣言) : 伊藤氏が 2009 年の EURO SPI に参加し、「SPI manifest Event」に刺激を受け制定したものである。ソフトウェアプロセスに関する現実的な話題は取り上げる。ソフトウェアプロセス最適化とはプロセス変更を伴う実践的活動である。今あるプロジェクトが、今いる組織が良くならない限り、全ての議論は不毛と考える。

^(*) Immaterial Labor : Maurizio Lazzarato 氏により、提案された概念である。形の無い生産物は、形のある生産物 (Material Labor) とは、違う形でやるべきではないかと言われている。以下のサイトを参照のこと。 <http://sea.jp/?p=517>



参加者の声

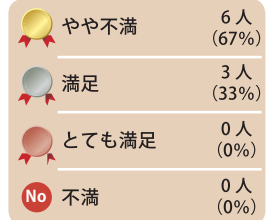
セミナーの理解度



セミナーの満足度



ディスカッションの満足度



かなり前提知識が必要だったと思う

あたらしい視点で新鮮だった

資料が英語だったり、あまり準備がされていない様子だった



企画担当者のひとりごと

本セミナーでは、新しい知見のもと、既成概念にとらわれない改善プロセスを紹介した。企画側の支援不足もあり、参加者の満足度は低かった。どのような改善モデルの形であっても、プロセス改善で実施することは同じであることに気づいてもらえるとありがたい。

こちらの講演資料は都合により不掲載

2010年9月17日



アジャイルインスペクション

講演者：永田 敦氏（ソニー株式会社）

概要

ソフトウェア開発における上流工程での品質を、少ないコストで実現するための方法として期待されている「アジャイルインスペクション」を取り上げた。「アジャイルインスペクション」の手法概要、考え方、効果、位置づけに関する講義に続き、「アジャイルインスペクション」の演習と以下のポイントに関する議論を行った。

- 1) この手法が効果的に使える場面やタイミング
- 2) ドキュメント作成および修正の際の工夫
- 3) プロセスとしてアジャイルインスペクションを導入する場合の工夫

アジャイルインスペクションは、ドキュメントの欠陥を発見することで、欠陥を混入させないようにすることを目的とした手法である。上流工程での発見 / 手戻り / 修正というコストを抑制し、さらに品質向上を狙うもので、そのプロセスの概要は以下のとおりである。

- ・対象ドキュメントから1ページをサンプリングし、短時間（10～30分）で一定の判定基準で内容を精査（インスペクション）する
- ・アジャイルインスペクションでは、この判断基準にチェックリストを用いず、かなり抽象化した「ルール」と呼ばれる観点を3～7に絞り、サンプリングしてインスペクションする
- ・ログをとり、メトリクスを分析して、インスペクションを継続するか次のプロセスへ進むかを決める

これらの活動は、準備、実施、分析・測定という区分で10ステップが定義されている。実際に導入する場合は、アジャイルインスペクションと他のレビュー方法、例えばフォーマルインスペクション^(*)や、デザインレビューと組み合わせると効果的で、当該ドキュメントの作成期間中に、インクリメンタルなレビューを実施し、必要に応じて最後にフォーマルインスペクションを行うという具体例が紹介された。

(*) フォーマルインスペクション：通常は品質管理者によって開催され、モデレータと呼ばれる司会者を置いて進行を管理する。参加者全員でチェックリストとレビュー対象物を付け合わせ欠陥を検出する。

アジャイルインスペクションの特長、効果は、

1. 品質の早期作り込み：ドキュメントが全部揃うまで待つのではなく、できたところからレビューを行い、書き手に欠陥をフィードバックすることにより、インクリメンタルに品質を高めていく。
2. コストの削減：成果物の完成直前に実施する、従来のフォーマルインスペクションと比べ、実施回数は多いが合計の実施工数は少なくて済む。また、発見した欠陥に対する手戻りの合計工数も早期発見なので少なくなる
3. 支援ツール：実施の手順、テンプレート、分析用 EXCEL ツールが揃っているなどが挙げられた。

セミナー時にあげられた質問に、「顧客から要求仕様書自体を成果物として提出を求められているため、ドキュメントの全てをチェックしなければならない。どうしたらよいだろうか」というものがあった。

これについて講師からの回答は、「サンプリングであるので、ドキュメントすべてを網羅しなければならないと考える人においては、不安になるであろう。レビューを全てこのアジャイルインスペクション一本で賄うことができると述べているつもりはない。網羅的にやるためにも、ドキュメントの質をあげておき、本来期待するドキュメントの品質を、パースペクティブレビュー^(*)によって上げていくのが実用的であると考え」というものであった。また、「てにをは」の間違いなどは、ツールで発見できることもあり、そういったツールの導入なども効果的であるとのことであった。

レビュー手法からツールの解説まで、幅広い内容に触れられたセミナーであった。

^(*) パースペクティブレビュー：レビューアに観点（perspective）を割り当てて実施するレビュー。レビュー開始前に、レビューアの立場と読み方の観点を決め、それぞれの観点でレビューを進める。





メールリングリスト (ML) の議論

100 通を超えるやりとりが、事前および事後の ML で行われた。

その中で特に注目されるのは、アジャイルインスペクションの位置づけにかかわる議論であった。

たとえば、アジャイルインスペクションとフォーマルインスペクションとの対比についての議論や、アジャイルインスペクションでどのような欠陥を見つけようとしているのか、「内容に関する確認」をするのか、それとも「表現に関する指摘」をしていくかというテーマである。

これらの議論を通して、アジャイルインスペクションの特徴、目的、スコープなどの共通認識ができ、さらにアジャイルインスペクションだけでレビュー全てを賄うものではなく、他の手法と組み合わせることでより有効になるというイメージを、事前にメールのやり取りを通じて参加者間で共有することができたと考える。またツールに頼るのではなく、どのように使いこなしていくか、などといったツールの位置づけに関することも議論された。



参加者の声

セミナーの理解度

	良く理解できた	11人 (65%)
	少し理解できた	5人 (29%)
	あまり理解できなかった	1人 (6%)
	理解できなかった	0人 (0%)

セミナーの満足度

	満足	8人 (47%)
	やや不満	5人 (29%)
	とても満足	4人 (24%)
	不満	0人 (0%)

ディスカッションの満足度

	満足	10人 (67%)
	やや不満	3人 (20%)
	とても満足	2人 (13%)
	不満	0人 (0%)

やり方の改善だけなので、すぐにでもやってみたいと思う

資料にはなかったが、「インクリメンタルなレビュー」が大変良いと思った

意見交換時間は長いほうが良い

最後にまとめてレビューするというやり方はもう行っていないが、さらにアジャイルインスペクションを取り入れると、どれくらいコストを改善できるか興味がある



今回は、「アジャイルインスペクション」とはどのようなものか、を実感していただくことを狙い、演習の時間を中心に組み立てた。演習で盛り上がり、Q&A や議論の時間が不足気味となったが、事後の ML での議論でフォローできた。

また、永田さんのお人柄のせいか、基本的なものから専門的なものまで幅広い質問が出され、そこから参加者同士の意見交換につながった。

開発現場への効率的な導入に関する議論を通じて、参加者全員がなんらかのアイデアを得ていただけたと思う。



2010年11月19日

レビュー・インスペクションの効果的実践と 阻害要因 ～ QI 法の実践事例の紹介～

講演者：細川 宣啓氏（日本アイ・ビー・エム株式会社）



概要

日本アイ・ビー・エムでは、QI（Quality Inspection）（図1参照）という名前で第三者レビュー、インスペクションを実施しており、そのさまざまなノウハウの一端をQI法として紹介するセミナーである。第三者レビューチームによるレビュー技法の特徴は次の3点である。

- ・レビューチームの視点に対するトレーニングの実施
- ・問題箇所の発見を支援するコード解析ツールの使用
- ・品質バイタルサイン（QVS：Quality Vital Sign）という指標を用いたレビューの範囲や視点の絞り込みの実施

これらは社内外で広く実施しており、次の効果を確認している。

- ・欠陥の効率的、効果的発見による製品品質の向上
- ・指摘内容に基づく開発者の気付きによる開発チームの品質意識の高揚

QIでは、プロジェクトの成果物（要求書・設計書など）に対して、まずは多面的に測定してQVSという指標から、そのプロジェクトの健康状態を診断し、詳しくインスペクションすべき箇所（成果物）、インスペクションする時の視点（紛れ込んでいる可能性の高い不具合）を絞り込む。

その結果をもとに、効率的かつ効果的なインスペクションを実施する。

QVSとは、犯してしまった失敗を浮き彫りにするための指

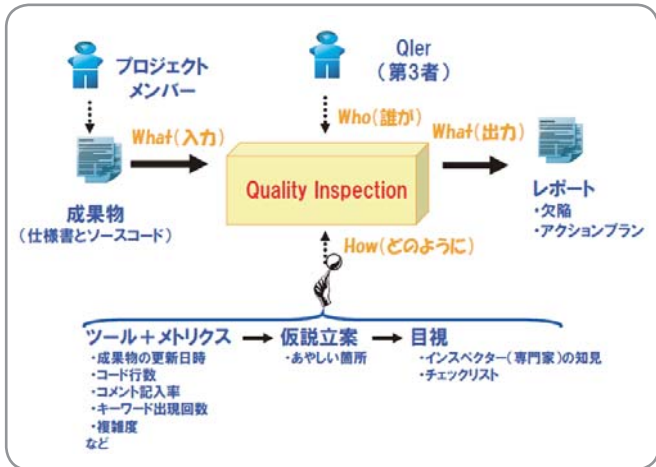


図1. QIの概要

標である。熟視テストすべき箇所を特定するために、成果物の多面的な外的因子を測定してQVSを導き出し、成果物検証作業の入力とする。

例えば、大量の仕様書に対して「最終更新日」「更新時間」を測定し、「夜中の3時にサーバに登録されている仕様書は、他の担当者との不整合が入る危険性が大きい」とか、「要件定義書と外部設計書の更新日がかけ離れているときは、追跡性が悪くなる危険性が大きい」などが想定されるのである。

これらの例が示すように、QVSを使ってインスペクションすべき箇所を洗い出し、視点の仮説を構築する。第三者レビューチームは、現在60を超えるバイタルサインをノウハウとして蓄積しており、それらを最大限に活用してプロジェクトの品質向上に寄与している。

その他にも、インスペクションの目的、レビューのコツ、コードインスペクションのノウハウ、不具合分類の重要性などについて、長年の経験に基づく示唆に富んだ説明があった。その中から印象的なディスカッションを取り上げる。

セミナー参加者からの「レビューイがネガティブにならない環境づくりは、どのようにすればよいのか？」という質問に、講師から次のようなユニークな回答があった。

- ① お菓子をを用意すること（会場でも講演が始まって、すぐにお菓子が配られた）
- ② レビューアの姿勢や態度はレビューイに伝わってしまうため、レビューイに敬意をもつこと
- ③ 「それイイネ！」ということ。不具合を取り除くのであって、人を裁かないこと
- ④ 「今日、楽しかったですか？ 今日、良い不具合が出たよね」といってレビューを締めて、前向きな気持ちで次のレビューにつなげる





メーリングリスト (ML) の議論

提出されたポジションペーパーから参加者の関心事は図 2 のように分類される。まずは、第三者レビューの方法とその効果に関心が集まっている。

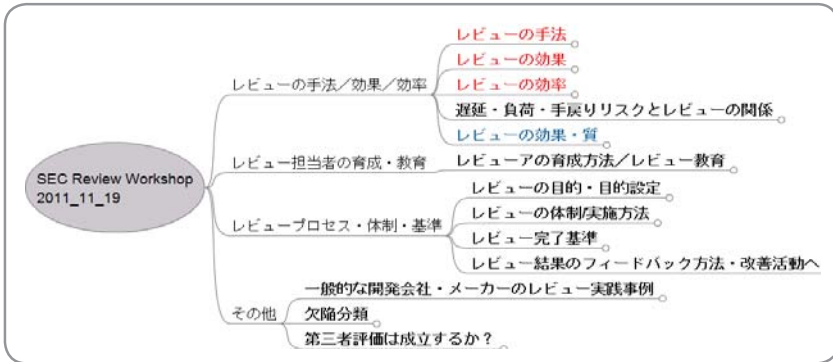


図 2. ポジションペーパーに基づいた関心事の分類

ML には、セミナーの開催前後で 62 件の投稿があった。ML では、次のような事項が取り上げられ、熱い議論が展開された。

- ・第三者レビューの位置づけや使い方は、第三者が検出し易く、開発チームの当事者が気付きにくい欠陥を担当する。QI はフェーズの完了条件として利用するが、品質を保証するものではない。品質を健康に例えると、第三者が健康を管理することはあっても保証することは難しい。
- ・レビューの効率や効果に関して、レビューとテストのトレードオフについて議論がなされた。フェーズ全体でどのように品質を確保するかといった戦略や目的を持ってレビューを計画し実施する。ときには上手に観点を間引くこともある。

◆参考情報

日科技連の 2010 年度 (第 26 年度) ソフトウェア品質管理研究会 (後援: IPA) の第 3 分科会「ソフトウェアレビュー」第 1 グループで、細川氏を主査として類似の研究が行われた。

詳しい報告書とプレゼン資料は以下を参照されたい。

報告書 http://www.juse.or.jp/software/314/attachs/3-1_report.pdf

プレゼン資料 <http://www.juse.or.jp/software/314/attachs/20110225-3-1.pdf>



参加者の声

セミナーの理解度

	良く理解できた	16人 (73%)
	少し理解できた	6人 (27%)
	あまり理解できなかった	0人 (0%)
	理解できなかった	0人 (0%)

セミナーの満足度

	とても満足	18人 (82%)
	満足	4人 (18%)
	やや不満	0人 (0%)
	不満	0人 (0%)

導入・活用について

	導入を検討する	12人 (55%)
	機会があれば導入する	6人 (27%)
	すぐに導入する	4人 (8%)
	不満	0人 (0%)

成果物を提示された形だけでなく、作成される過程に潜む問題にも着目して捉える点が参考になった。色々なアイデアを出してレビューを積極的かつ楽しく取り組む姿勢も参考になった

分析に楽しみを組み合わせることで厳しい仕事をのりきっていることが参考になった

レビュー対象の絞り方が大変参考になった。多面的な計測の必要性やレビューを開催するときの人間関係、空気感の重要性、絶妙な抜き取りチェックの有効性などが興味深かった



企画担当者のひとこと

講師が今後取り組んでいきたいという「欠陥のマスターデータ作成」については、業界横断の活動として取り組む体制作りには IPA/SEC が協力できるとよいと思う。Fault-Prone モジュールの予測のために、特に大きな効果が期待できる。

コードインスペクションのノウハウなど、非常に示唆に富んだ説明がなされた。特に印象に残ったものは、次の2点である。

- ①名前の付け方や改行の仕方をみる。変数にハイフンが入っているか？ アンダースコアが入っているか？ といった観点から、開発言語の癖を知っているかどうかを想像する。例えば、「COBOL 技術者が Java 言語を使っている場合にはここが怪しい！」と想像する。
- ②コードをソートしてみる。ソートすると「コピーペースト」コードが揃う。クセが出る。

セミナーではディスカッションの時間が短くなってしまったが、その後の懇親会で講師と参加者との積極的な議論が継続して行われ、さらに理解を深めていただいた。

なお、QI 手法のレビューイがネガティブにならない環境づくりにならない、講師が開始早々にお菓子を配り、会場は一瞬で和やかな雰囲気になった。さらに、セミナーの終わりは手法のとおり「今日、楽しかったですか？」というフレーズで締めくくった。



2011年1月21日

品質会計を軸としたパートナー QC レビュー の紹介 (PQR : Partner Quality control Review)

講演者：後藤 徳彦氏 (NEC ソフト株式会社)



概要

NEC グループで実践している独自の品質管理手法である「品質会計」と重点パートナーの品質対応力向上をねらいとしたパートナー QC レビュー（以下、PQR）活動の実際について紹介した。

重点パートナー先へ年 1 回、定期的に会社・組織としての品質対応に対する評価結果、改善課題をフィードバックし、継続的な改善活動を通して、パートナーの品質対応力向上を促す活動である。

IT 不況に伴う発注量減、セキュリティ制約や監査対応が役員クラスなど現実的な課題も交えて、PQR プロセスの説明やその効果の他、PQR の核である品質会計について解説した。

ディスカッションでは、パートナー企業側の視点による下記のような鋭い質問から悩ましい現状も披露され、現実感のある盛り上がりを見せた。

- ・優秀なパートナーに対するご褒美はあるか？（パートナーのやる気をどう引き出すのか）
- ・開発中のプロジェクトの監査は？（年に 1 回では、即効性がないのではないか）

メーリングリスト (ML) の議論





ポジションペーパーと講演資料を事前配付し、参加者間での問題意識を喚起させる工夫がなされた。しかし、自社組織での改善ではなく、パートナーへの活動ということもあり、盛り上がりには欠けたようである。

主な、議論は次のとおり。





- ・書籍の「ソフトウェア品質会計」との関連
- ・パートナーが SES 契約の場合
- ・品質会計が得手 / 不得手のプロジェクトタイプ
- ・品質会計の欠点
- ・回帰型バグ予測モデル

参加者の声





セミナーの理解度

 良く理解できた	8人 (62%)
 少し理解できた	4人 (31%)
 あまり理解できなかった	1人 (8%)
 理解できなかった	0人 (0%)

セミナーの満足度

 満足	8人 (62%)
 とても満足	5人 (38%)
 やや不満	0人 (0%)
 不満	0人 (0%)

導入・活用について

 導入を検討する	7人 (64%)
 機会があれば導入する	4人 (36%)
 すぐに導入する	0人 (0%)
 不満	0人 (0%)

やり方の改善だけなので、
すぐにでもやってみたい

内容的に活用できると思うが、
活用の準備をする予算を上司に
もらうのが課題である

ぶっちゃけたレベル
も聞けて良かった

パートナーに対するQC活動
のレビューは実施しているが、
手法、フォローのあり方に改
善が必要かとも考えており、
参考にしたい

企画担当のひとこと



通常、パートナーは勝手に技術力・品質力を向上しなさいという会社が多い中、NECではパートナーが重要な役割を担っている事を数年前から自覚し、積極的に教育・指導を行っていたので、皆さんに紹介した。

質疑の中でも取り上げられていたが、成績の良いパートナーに対するインセンティブについて課題はあるものの、今後の発展に期待したい。



こちらの講演資料は 60 ページへ



2011年6月3日

ソフトウェアレビュー改善の着眼点 ～ 段階的な取り組みにむけた事例分析 ～

講演者：森崎 修司氏（静岡大学 情報学部 / 奈良先端科学技術大学院大学）



概要

前半は、東芝デジタルメディアエンジニアリングのソフトウェア開発プロジェクトで着眼点を設定したレビューの事例を題材に、分析結果からわかった次の2点について説明が行われた。

- ・レビュー観点の設定により、修正コストを低減できる
- ・レビュー観点を絞り込むことにより、レビュー速度を上げることができる

講演の中では、観点を設定したレビューの本質的な部分や、発表された事例に関する詳細について、参加者から様々な質問が出された。

後半は、参加者が事前に提出したポジションペーパーの内容をもとに、ディスカッションが行われた。また、参加者から自社での取り組みの紹介や他社の取り組みに関する質問など、積極的な発言が多くあった。

レビューの効率化、費用対効果、効果測定、技法、観点などの意見交換からはじまり、レビューの終了条件、出荷責任、品質保証などについて次々とディスカッションが行われ、終了予定時刻をオーバーするほどの盛り上がりを見せた。

メールリングリスト（ML）の議論

MLには、セミナーの開催前後で68件の投稿があった。

森崎氏がポジションペーパーから参加者の問題意識を分類し、MLを使ってグループ別に質問をした。レビューの品質、レビューの観点、開発全体から見たレビューの位置付け、欠陥を発見した効果の推定方法、レビューの開始および終了条件、レビュー効率などについて議論が行われた。



参加者の声

セミナーの理解度

	少し理解できた	19人 (66%)
	良く理解できた	10人 (34%)
	あまり理解できなかった	0人 (0%)
	理解できなかった	0人 (0%)

セミナーの満足度

	満足	17人 (59%)
	とても満足	8人 (28%)
	やや不満	4人 (14%)
	不満	0人 (0%)

導入・活用について

	導入を検討する	13人 (48%)
	機会があれば導入する	9人 (33%)
	すぐに導入する	5人 (19%)
	不満	0人 (0%)

ディスカッションがやや発散したような気もするが、いろいろな意見も聞けたので良い

観点を絞ったレビューを行うという考えがなかったので、とても興味深かった

様々な意見が聞けた

皆さん悩みは同じだと感じた

継続（フォローアップセミナー）を希望

QCについてよく考えたい



企画担当のひとこと

本セミナーは、2011年3月18日に開催する予定であったが、3月11日に発生した東日本大震災のため急遽中止となり、6月に再度参加者の募集を行い開催した。震災の影響がいろいろある中、大勢の方に参加いただき、大変印象に残るセミナーと

なった。

講師と参加者によるディスカッションでは、レビューをとりまく様々な問題について議論がなされ、課題を次回に引き継ぐ形でセミナーは終了した。



こちらの講演資料は 64 ページへ



2011年10月7日

ソフトウェアレビュー改善の着眼点 ～ 段階的な取り組みにむけた事例分析 ～ (その2)

講演者：森崎 修司氏（静岡大学 情報学部 / 奈良先端科学技術大学院大学）



概要

前回（セミナー 09）の続編として企画されたセミナーであったが、今回の参加のみでも十分に意義のある内容であった。

最初に、レビューに関する実証実験の事例から、次の3点に重点をおいた解説がなされた。※セミナー 09（p.64-68）の講演資料参照

- ① レビュー観点による欠陥種別の変化
- ② ソフトウェアレビューの効果
- ③ 過去の不具合修正時間に基づくレビュー観点の設定

参加者からは次のような質問が出された。

- ・ レビューしたチーム間の能力のバラツキは？（p.64 ★印参照）
 - バラツキがないとはいえない。また、レビューの観点に個人差がないとはいえない。経験豊富な人ほど数多くの観点でレビューができる。その際には、本来書かれているべき事項という自分のイメージと比較している。
- ・ 修正時間を個々のモジュールで計測するのは難しい（p.66 ★★印参照）
 - 複数件の修正をまとめて計測し、時間を按分していると思う。ひよっとすると休憩時間を含んでいるかもしれない。
- ・ 例えば、リリース間際の作業の場合、本来は多くの工数が掛かるところを短時間で済ませていることもあるのでは？
 - データを分析する場合には、前提を崩さないように実績時間だけでなく、この位は掛かるという想定時間も記録しておく。二重帳簿的な管理も必要となる。

実験環境について、次のような補足説明があった。

- ・ ソースコードの1本目をしっかりとレビューして傾向を把握し、2本目以降のレビューにフィードバックする
- ・ レビュー後に振り返りを行い、誤りの多い観点についてその理由を聞く

次に、参加者のポジションペーパー、MLでの事前討議から関心の高い事項について議論した。

<ソフトウェアレビューの目的設定（森崎氏から）>

- ・特に大規模ソフトは、全てをレビューしきれないので観点を絞る
- ・特定部分のレビューになるが、リーディングテクニックを使うことでテストで抽出しにくいところを検出してはどうか
- ・第三者レビューでは必ずしも中身には踏み込めないが、典型的な欠陥、国際規格 (ISO) の観点、表現方法などの観点からは指摘できる

<レビューの終了条件（参加者との議論から）>

- ・客観的あるいは合意が取りやすい終了条件は？
- 合意、品質会計、全体における位置づけを踏まえて「一人ひとりが合意できるか」と考える。そのために、レビュー対象物に対して点数を付ける。
例えば、要件を満たしていることを確認するために、必須の記載項目に関するチェックリストを用いる。
- ・点数付けて点数の目安を設けたり、指摘の中身を見たりして終了の可否判断をする。また、傾向から逸脱していないかを分析する
 - ・何をもってレビューをしたから大丈夫であるといえるか？
- スケジュールや欠陥数と基準値やプロジェクトの予測値との比較により、総合的に判断する。

<レビュー効果の計測（森崎氏から）>

- ・工数や網羅性などのメトリクスがある
- ・欠陥を分類して件数や比率を分析する
- ・レビューの健全性（目的設定、観点設定があるか）を把握する

<レビューアの教育（森崎氏および参加者との議論から）>

- ・ドメインに依存しない部分をレビュー教育の内容にする
- ・レビューの知識、技能、心構えなどを教育する
- ・理想的なレビューアは、欠陥を見つける前にアーキテクチャを押さえている
- ・育てるスキルは、プロダクト、およびありがちな欠陥を中心にする

<プロジェクト・レビュー（森崎氏および参加者との議論から）>

- ・均等なプロジェクトになっているかを確認する
 - ・中身を細かく見ずに出来栄を判断したい
 - ・プロジェクト・レビューの方が客観性を求められるのはなぜ？
- 顧客への影響が大きいことや外部のステークホルダとの合意が必要なことが考えられる。

<ソースコード・レビュー（森崎氏から）>

- ・ fault-prone モジュール^(*) を特定する

<レビュー技法および形態（森崎氏から）>

- ・組織、文化、品質に合わせたテーラリングを行う



メーリングリスト (ML) の議論

ML には、セミナーの事前事後で 76 件の投稿があった。

ML および提出されたポジションペーパーの内容から、参加者は次のようなテーマに関心を持っていることがうかがえる。

- ・レビューアの育成
- ・客観的あるいは合意が取りやすい終了条件
- ・レビュー効果の測定
- ・コードレビューで fault-prone モジュールの特定
- ・組織や文化に合わせたレビュー技法や形態のテラリング



参加者の声

セミナーの理解度	セミナーの満足度	導入・活用について
少し理解できた 14人 (61%)	満足 10人 (45%)	導入を検討する 11人 (52%)
良く理解できた 8人 (35%)	とても満足 9人 (41%)	すぐに導入する 5人 (24%)
あまり理解できなかった 1人 (4%)	やや不満 3人 (14%)	機会があれば導入する 5人 (24%)
理解できなかった 0人 (0%)	不満 0人 (0%)	不満 0人 (0%)

自社の状況と改善ポイントを思い浮かべて聞けたので大変参考になった

他の会社がどうやっているかが知れてよかった

内容はとても満足。一方で時間が不足していると感じた

ポジションペーパーに沿った内容でとても有意義だった

^(*) fault-prone モジュール：バグを含む確率の高いモジュール



多くの参加者から活発な発言がなされ、参加型のセミナーとなった。MLを活用して事前に関心事についての議論を行っており、当日は重点を絞って数多くの事項に触れたため、参加者の理解度、満足度が極めて高い有意義なセミナーとなった。とくに評価したいのは次の2点である。

- ①セミナーでは各社、各人の経験や教訓を踏まえた積極的な発言が多く、参加者で共有が図れた
- ②セミナー後に、実践したり具体的な行動を起こしたりした参加者が少なからずいたことで、今後の実施効果が期待できる



2011年12月2日

品質保証活動を形骸化させないコツ ～プロセス改善に魂を込めるには～

講演者：小池 利和氏（ヤマハ株式会社）



概要

開発現場において、「品質保証活動が形骸化して・・・」という話を耳にすることが多い。

講師は、「形だけの監査なんてもう止めませんか？」をスローガンに、社内で行くつかの部門を監査し、さらにはさまざまな社外の方との会話から得られた豊富な知識を習得している。これらの経験から、形骸化の典型的な特徴を見極め、プロセス改善に魂を込めるベストプラクティスを実践している。

今までの経験談を踏まえ、形骸化しない活きた品質保証活動やプロセス改善活動を展開するコツを、具体的な実践例を基に紹介し、参加者の品質保証活動の意識向上につなげた。

講義内容としては、次の3点に絞って進め、参加者とディスカッションした。

- ① SQA、SEPGの体制、教育
- ② メトリクスの活用
- ③ 形骸化させないSQA&SEPG活動

ここでは、配布資料にはない、具体的な実践事例と効果が紹介され、またセミナーでは、次のような質問があった。

- ・ プロセス監査は一人でもどのくらいのプロジェクトを担当しているのか？
- ・ SQAの要員は何名いるのか？
- ・ プロジェクトの期間は平均どのくらいか？
- ・ 外注要員の比率はどのくらいか？
- ・ 毎月のプロセス監査はどこまで詳細に見るか？
- ・ プロジェクト成功の定義はなにか？



メーリングリスト (ML) の議論

MLには、38件の投稿があり、その中で特に次の項目についてのディスカッションが多かった。

- ・品質保証活動の継続方法
- ・有効なプロセス監査の実施方法
- ・有効な品質メトリクスやSQA活動のメトリクスの設定と活用

また、現場の意見としては、以下の内容が多く見受けられた。

- ・どのようなメトリクスを設定しているか？
- ・そのメトリクスにより、何を把握しようとしているか？(目的は何か？)
- ・測定、分析、活用が上手くいかず困っている
- ・データがタイムリに収集されない
- ・データの精度があやしい
- ・分析方法が分からない
- ・そもそも適切なメトリクスが分からない



参加者の声

セミナーの理解度

	良く理解できた	9人 (56%)
	少し理解できた	7人 (44%)
	あまり理解できなかった	0人 (0%)
	理解できなかった	0人 (0%)

セミナーの満足度

	満足	8人 (53%)
	とても満足	6人 (40%)
	やや不満	1人 (7%)
	不満	0人 (0%)

導入・活用について

	導入を検討する	6人 (43%)
	すぐに導入する	5人 (36%)
	機会があれば導入する	3人 (21%)
	不満	0人 (0%)

方法論だけでなく具体的な事例として、帳票例を見せていただいていたのが大変よかった

最初「工数をとっている」という話があり、もっとすごいメトリクスかと思ったが、活用するとすごく効果があると思う。期待以上だった

勉強会やグラフなど、意識向上につながる取組みは、早速実施計画を立ててみたい



企画担当者のひとこと

今回は、講師自身が宙に浮いた話があまり好きで無いこともあり、プロセス監査にメトリクスをどう役立てているかなど、出来るだけ具体的な帳票例を見せながらの講演となった。現場の生々しい実践状況を参加者に見せることで、表面的な話ではなく具体的な悩みを解決するためのディスカッションが展開されたと考える。



組み合わせ技術を利用した テストケース生成ツールと適用事例の紹介

講演者：小笠原 秀人氏、中野 隆司氏
(株式会社東芝)

すべてをテストすることはできない

■論理的な問題

- 組み合わせが膨大
- バグがこれで最後と証明することができない

■コスト、時間の問題

■テストすべきものを選択して実施

- 合理的、効率的なテストを実施 (←良いテスト設計)
- 予算に応じてテストを選択
- 全体的なリスクとのコストバランスに応じてテストを選択



TOSHIBA Leading Innovation Copyright 2009, Toshiba Corporation

2

良いテスト設計とは

- 「より少ないテスト実施回数」で
- 「より多くのバグが見つかるテスト」を行い
- 「テスト対象を漏れがないように網羅」する



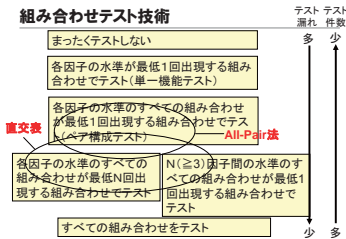
ムダなく効率的に！しかも網羅的に！

一この考えに基づいたテスト設計技法の一つとして、組み合わせテスト技術(直交表を用いたテスト、ペア構成テスト)がある

TOSHIBA Leading Innovation Copyright 2009, Toshiba Corporation

3

組み合わせテスト技術



TOSHIBA Leading Innovation Copyright 2009, Toshiba Corporation

4

組み合わせテスト技術の利用について

- テスト効率を考え、まずは各因子の水準のすべての組み合わせが最低1回出現する組み合わせでテスト(ペア構成テスト)を行うことに着目
- どこまで狙うかは、テスト対象の重要性やかけられるコストにもよる
- 組み合わせテストを行う前にテスト計画、テスト設計が十分行われていることが前提(他のテスト技法と併用する)
- 組み合わせテスト技術を利用するテストプロセスもしっかり構築

TOSHIBA Leading Innovation Copyright 2009, Toshiba Corporation

5

ペア構成テストの実現

- 2因子間のすべての組み合わせを見つける方法として「直交表」と「All-pair法のアルゴリズム」を採用
- 直交表の3因子間の組み合わせも平均的に出る(偏りがない)という特徴を利用しつつテストケースがより少なくなるようにAll-Pair法のアルゴリズムの中に採用
- 直交表のほうが馴染みが深い
- 組み合わせの作成、ありえない組み合わせ(禁則)の回避は、ツールで支援

TOSHIBA Leading Innovation Copyright 2009, Toshiba Corporation

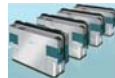
6

TOSHIBA Leading Innovation >>>

ツール紹介

直交表を利用したソフトウェアテスト

- テストの網羅性向上が急務
- テスト漏れによるリリース後の不具合が皆無にならない
- テスト経験が長い担当者に頼っている
- 自動改札機の判定処理
- 取り扱う券
 - 乗車券、定期券、IC
 - 200~300種類
 - 複数枚の券組合せ
 - 駅定常乗降別に従う
- 改札機の処理
 - 乗車経路、運賃の判定
 - 複数枚の経路接続
 - 複数ルートから最安精算
 - 券数出まで2秒で処理
 - 券情報の置換え



自動改札機

TOSHIBA Leading Innovation Copyright 2009, Toshiba Corporation

8

テストケース生成

■判定テストのテスト条件洗い出し

- 川崎駅から出発し東京駅で新幹線に乗車するとき、東京駅までの在来線乗車と東京駅からの新幹線乗車が必要。
- 東京駅改札機判定のテスト条件は、**独立した接続、在来線券、新幹線券の組合せ**

条件1. 券の接続パターン 6種類

接続パターン	条件	種類	回数
1. 在来線乗車後、新幹線乗車	在来線乗車後、新幹線乗車	在来線乗車券	1回
2. 在来線乗車後、在来線乗車	在来線乗車後、在来線乗車	在来線乗車券	1回
3. 在来線乗車後、在来線乗車	在来線乗車後、在来線乗車	在来線乗車券	1回
4. 在来線乗車後、在来線乗車	在来線乗車後、在来線乗車	在来線乗車券	1回
5. 在来線乗車後、在来線乗車	在来線乗車後、在来線乗車	在来線乗車券	1回
6. 在来線乗車後、在来線乗車	在来線乗車後、在来線乗車	在来線乗車券	1回

条件2. 在来線券 5種類

種類	回数
1. 在来線乗車券	1回
2. 在来線乗車券	1回
3. 在来線乗車券	1回
4. 在来線乗車券	1回
5. 在来線乗車券	1回

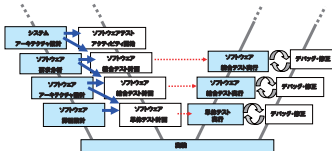
条件3. 新幹線券 5種類

種類	回数
1. 新幹線乗車券	1回
2. 新幹線乗車券	1回
3. 新幹線乗車券	1回
4. 新幹線乗車券	1回
5. 新幹線乗車券	1回

組合せは 6種類 × 5種類 × 5種類 = 150件

テストの効果的実施のためのポイント

■開発工程の上流からテストのための活動をスタートさせる



■テストが後手に回るパターン

- ソフトウェア詳細設計→実装工程からテストのための活動をスタート

■全組合せと直交表の組合せを比較

- 全組合せは150件
- 直交表の組合せは30件、全組合せの1/5
- 直交表はテスト項目数を1/5にできる

■直交表を使ったテスト結果

- 不具合の検出3件
- ①3-2⑤-③2の組合せ
- ①4-2⑤-③3 #
- ①1-2⑤-③4 #
- 特徴
- ②6の在来線長距離券が原因のひとつ

項目	水準	結果
接続パターン	1, 2, 3, 4, 5, 6	
在来線券	1, 2, 3, 4, 5	
新幹線券	1, 2, 3, 4, 5	

□は全組合せ
■は直交表組合せ

テストケース生成支援ツールAPTNavTM

- 「エイビーターナビ」95%
- All-Pair Testing Navigation System

■直交表、All-pair法に基づく組み合わせテスト技術(ペア構成テスト)を利用したソフトウェアテストを支援するためのExcelマクロ

■付属マクロとして「直交表生成マクロ」も提供

- 31種類の直交表テンプレートの生成
- 最適直交表の自動選択可(一部追加法の利用も考慮)

■直交表を利用して組合せを作成

- L25が最適
- ⑤水準まで
- 網羅性を維持
- 25件の組合せ

■条件と種類をL25に割付

- 条件=因子、種類=水準
- L25に割付ると25件生成
- さらに⑥水準目を一部追加法を使いL25展開後1水準追加

項目	水準
接続パターン	1, 2, 3, 4, 5, 6
在来線券	1, 2, 3, 4, 5
新幹線券	1, 2, 3, 4, 5

(参考)直交表 vs All-pair法アルゴリズム

特徴	直交表	All-pair法アルゴリズム
長所	直交表を利用してテストケースを作成 既存の直交表書庫に照える 因子水準がばらばらなので因子の組み合わせも平均的に出現する 因子水準、水準数が大きてもテストケースを作成できる	直交表より生成できるアルゴリズムを利用してテストケースを生成する ソフトウェアシステムに正しく最小の項目数でテストケースを作成できる 因子水準、水準数が大きてもテストケースを作成できる 組み合わせの多い組み合わせ(特定の項目に組み合わせ)がある 組み合わせの多い組み合わせがある テストケース数は直交表のサイズに依存し最小の項目数でない場合もある
短所	扱える因子数、水準数にある程度制限がある 既存の直交表では、大きな水準を扱えるものがほとんど見当たらない(作成可能)	組み合わせの多い組み合わせ(特定の項目に組み合わせ)がある 組み合わせの多い組み合わせがある 組み合わせの多い組み合わせがある 組み合わせの多い組み合わせがある

- APTNavTMでは、双方の考えを取り込んだ
- 直交表もAll-pair法のアルゴリズムの一種だとも言える
- 実験計画法や品質工学で直交表に馴染みが深いので最初の導入としては、直交表を利用したソフトウェアテストから推進

■直交表に展開した組合せ

項目	水準
接続パターン	1, 2, 3, 4, 5, 6
在来線券	1, 2, 3, 4, 5
新幹線券	1, 2, 3, 4, 5

- 1. テスト時、不具合検出
- ① L25を展開し 25件の組合せを生成
- ② 一部追加法により 1水準5件を追加

APTNavTMツール開発の経緯

- 2因子間を網羅する組み合わせを容易に作成したい
- 禁則(ありえない組合せ)に対応するのが人手では、非常に煩雑
- 市販のツールでも禁則に対応しているものは少ない
- 禁則を回避し、2因子間の組み合わせを網羅するテストケースを作成(追加)するのを支援するマクロの開発

コンセプトはシンプル、気軽に実行

APTNavitTMマクロ

- 2機能間の動作を保証するためのテストケースを生成
 - テストケース生成(直交表、All-pair法)
 - 禁則(ありえない組合せ)の回避
 - 網羅度算出(2因子間、3因子間)



- 新しいExcelファイル上で、直交表テンプレートが生成される
- 因子名と水準名を入れるとテストケースのシートにリンクが張られており、テストケースが生成



APTNavitTMマクロの主な機能

- テストケース追加・生成
 - 2因子間網羅度を100%にするのに必要なテストケースを追加・生成(直交表、All-pair法に基づいて作成)
- 2因子間網羅度計算
 - 因子・水準およびテストケース(入力の組合せ)から2因子間網羅度マトリクスを作成し、2因子間網羅度を計算
- 禁則設定
 - 2因子間網羅度マトリクス上で禁則の組み合わせを指定、解除し、網羅度計算、テストケース生成に反映する
 - 多因子間禁則規定シートにて3因子間以上の禁則の組み合わせを設定し、網羅度計算、テストケース生成に反映する
- 禁則確認
 - 設定しておいた禁則に対し、テストケース上で禁則の組合せを表示、除去
- 2因子間組合せ回数計算
 - 作成済みの2因子間網羅度マトリクス上に組合せの出現回数を上書き出力

直交表生成マクロで作成できる直交表の種類

2因子	3因子	4因子	5因子	6因子	7因子	8因子	9因子	10因子	11因子	12因子	13因子	14因子	15因子	16因子	17因子	18因子	19因子	20因子	21因子	22因子	23因子	24因子	25因子	26因子	27因子	28因子	29因子	30因子	31因子
2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

APTNavitTMマクロの主な機能

- テストケースチェック
 - 2因子間網羅度を100%にするのに不要と思われるテストケースを表示、除去
- 3因子間網羅度計算
 - 任意の3因子間ごとの3因子間網羅度、全体の3因子間網羅度を算出する
- 総組み合わせテストケース作成・カウント
 - 総組み合わせのテストケースの出力を行う
 - 総組み合わせのテストケース数のカウントを行う

APTNavitTMを使う利点

- 2因子間の組み合わせを網羅するテストケース作成が容易
 - 効率よいテストの実施
 - 想定していない組み合わせで不具合を発見することも
- 2因子間、3因子間網羅度をすぐに計算
- 禁則がある場合の対応が容易
 - 禁則が多々あるとこれを回避してテストケースを作成するのは人手だと煩雑
- 直交表、All-pair法の知識がなくてもテストケースの作成が可能
 - 一方で危険(単に禁則を回避して2因子網羅の組合せを出しているだけ...)、目的にあった使い方になっているか?
 - 上記手法を理解した上で使用することを推奨

直交表生成マクロ

- 2水準から17水準まで揃える4から1289案での31種類の直交表テンプレート(Excel)を生成
- それぞれの水準の数から最適な直交表テンプレートを自動選択(一部追加法の利用を考慮した直交表テンプレートの選択も可能)



組み合わせテストの留意点

- 直交表やAll-Pair法を用いたテストは単なる組み合わせ
 - 別途、利用シナリオに基づいたテストや過去の不具合に基づくテストが必要
 - 2因子間網羅度100%だけで十分かという話はあるが、利用シナリオや過去の不具合を抑えれば、2因子間網羅度100%で信頼性はかなり高まる
- さだま視点でのテストが必要であり、他のソフトウェアテスト技法と併用することが望ましい
- まずは、開発者個人レベルで気軽にAPTNavitTMを使って入力の組み合わせを考えて動作させてみるという使い方から始めてもらおうと種々の品質に対する意識も高まると考えている

2009年3月27日

モデル駆動開発に向けたモデルチェッキング技術の導入



こちらの講演資料は、講演者の意向により
不掲載とさせていただきます。

フィールドワークを応用した プロセス改善の継続方法

講演者：平田 貞代氏（富士通株式会社）

目次

- 概要
- フィールドワークとは
- フィールドワークを基に開発した方法
：チェンジ・フィールドワーク
- 実績
- 事例
- 効果

All Right Reserved, Copyright FUJITSU LIMITED 2009

2

概要

課題：プロセス改善には **やらされ感** があり、**停滞** しがち。

【従来のプロセス改善】

組織員	問題	原因	対策
組織員			
組織員			

【チェンジ・フィールドワーク】

組織員	現状	良い点・納得	対策	プロセス改善	効果測定
組織員					
組織員					

フィールドワーカーが、**気付きを促す段階**、**納得や協力を高める段階** を明確に設ける。組織員は、プロセス改善に **意中** して取り組むことができる。

All Right Reserved, Copyright FUJITSU LIMITED 2009

3

体験①

やらされ感と停滞のメカニズム

組織員A	組織員B	
	実行ない	実行する
実行ない		
実行する		

All Right Reserved, Copyright FUJITSU LIMITED 2009

4

フィールドワークとは

文化人類学における **現状調査法**。

フィールドワーカー が **異なる文化に直接参加** し、観察対象となる人々の視点から現状を **選解** する方法。

フィールドワークから得た情報から **エスノグラフィ** (観察対象者の視点から整理した物語) を作成し、現状を **生々しく共有** する。

[1]佐藤和也、「フィールドワークの技法」、新曜社、2002
[2]野中郁次郎・柳野亮、「知識創造の方法論」、東洋経済新報社、2003

All Right Reserved, Copyright FUJITSU LIMITED 2009

5

フィールドワーカーの育成

フィールドワークの権威である **パロアルト研究所** と 2004年より共同研究を開始。フィールドワーカーを育成。

育成されたフィールドワーカーは、文化人類学のフィールドワークを応用し、情報システム業界の現場のプロセス改善活動に適した方法として **チェンジ・フィールドワーク** を開発し普及。

All Right Reserved, Copyright FUJITSU LIMITED 2009

6

開発した方法

：チェンジ・フィールドワーク

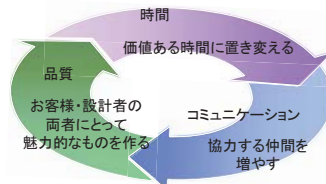
- ・チェンジ・フィールドワークのねらい
- ・実施手順と時間

All Right Reserved, Copyright FUJITSU LIMITED 2009

7

チェンジ・フィールドワークのねらい

プロセス改善の継続、および、改善力の向上



All Right Reserved, Copyright FUJITSU LIMITED 2009

8

チェンジ・フィールドワークのゴール

人とプロセスとITの全体最適化



All Right Reserved, Copyright FUJITSU LIMITED 2009

9

体験② フィールドワークの手軽な実施例

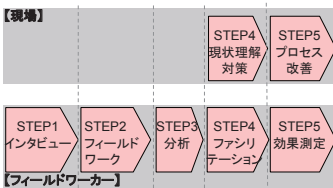
SEが独りで操作し、誤って顧客情報を削除してしまいました。
そのSEはどう尋ねますか？

- 例えは
- × 「なぜ、～したのですか？」
 結局口調に聞こえる。
 - 「普段は何をしていますか？」
 「今日は何をしましたか？」
 「もし～しなかったらどうなりますか？」

All Right Reserved, Copyright FUJITSU LIMITED 2009

13

実施手順と分担

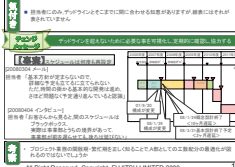


All Right Reserved, Copyright FUJITSU LIMITED 2009

10

Step3:分析

Step1,2から得た事実を示しながら
気付きとプロセス改善のための提言(チェンジメッセージ)
を資料化し
生々しく
報告。



All Right Reserved, Copyright FUJITSU LIMITED 2009

14

Step1:インタビュー

【フィールドワーク流インタビュー】
観察の対象となる方々による語りをありのままに記録する方法

- ・仮説検証や誘導的な質問はさける
- ・自発的に語っていただく
- ・日常のエピソードを聞く
- ・詳細を階層的に聞く

【トランスクリプト】

録音を基に、言語および非言語の全てを書き起こす。

All Right Reserved, Copyright FUJITSU LIMITED 2009

11

Step4:ファシリテーション

フィールドワークを通じて、
現場の様々な立場でとらえた分析結果から
事実認識ではなく**事実**について共有する。

事実を並べることで、現場の**気付き**を促す。

結論を考えさせ、**納得性**を高める。

All Right Reserved, Copyright FUJITSU LIMITED 2009

15

Step2:フィールドワーク

【シャドウイング】
観察の対象となる現場の人々と**行動を共にし**、
記録する方法。



例えば

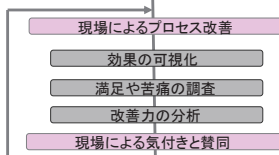
- ・作業中に隣に座る、出張や会議に同行する、
などの方法により日常的な業務に参加
- ・録音・録画を活用し記録

All Right Reserved, Copyright FUJITSU LIMITED 2009

12

Step5:効果測定

フィールドワーカーが
現場で実行されたプロセス改善を観察し、効果を分析



All Right Reserved, Copyright FUJITSU LIMITED 2009

16

体験③ 抵抗を無くす方法

・ラポール：現場の方々の「友好的な人間関係」

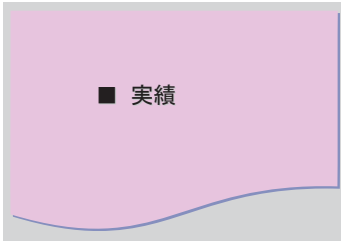
はじめから協力的な関係が得られることはない。段階をふむという心構えで臨む。^[7]

- 懸念段階
- 模索段階
- 協力段階
- 参加段階

[7] Spradley (1979)

All Right Reserved, Copyright FUJITSU LIMITED 2009

17



All Right Reserved, Copyright FUJITSU LIMITED 2009

18

チェンジ・フィールドワークの特長

■比較的短期間で実施可能

- ・フィールドワーカー2名程度
- ・Step1からStep3までは4週間程度

■現場にとっての負担は少ない

- ・現場にとって負担が少ない(インタビューは1人1時間程度)
- ・録音・録画を活用した、事実起点的臨場感あふれる報告

All Right Reserved, Copyright FUJITSU LIMITED 2009

19

実績

No	業種	業務	規模(人)	直近3年間の主な実績		
				2007	2008	2009
1	金融	運用保守	110	■	■	■
2	金融	運用保守	110	■	■	■
3	農材	運用保守	20	■	■	■
4	農材	システム設計	80	■	■	■
5	公共	運用保守	10	■	■	■
6	IT	システム設計	90	■	■	■
7	IT	製品開発	90	■	■	■
8	IT	製品開発	20	■	■	■
9	IT	製品開発	10	■	■	■
10	IT	製品開発	10	■	■	■
11	IT	営業	40	■	■	■
12	IT	製品開発	10未満	■	■	■
13	IT	医療	60	■	■	■

All Right Reserved, Copyright FUJITSU LIMITED 2009

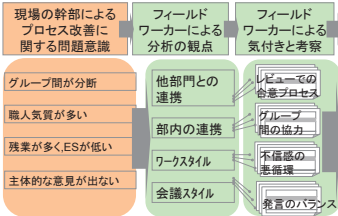
20

■ 事例の紹介

All Right Reserved, Copyright FUJITSU LIMITED 2009

21

チェンジ・フィールドワーク以前の状況

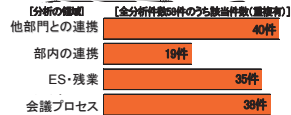


All Right Reserved, Copyright FUJITSU LIMITED 2009

22

チェンジ・フィールドワークによる気付き

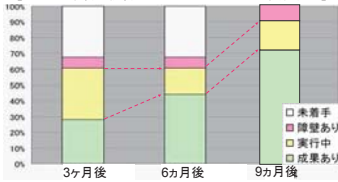
分野の別	気付き	改善の提言(アクション)
情報共有	管理業に記録されない高い声	問題管理だけでなく高い声を記録し調査
役割	担当に対して声となる者の不在	メンバーの役割、指示系統を明示
要求管理	方針を決めずに細かな雑務に固執	雑務・設計方針の見直しを促す



All Right Reserved, Copyright FUJITSU LIMITED 2009

23

チェンジ・フィールドワーク後の経過 【プロセス改善の提言(チェンジ・メッセージ)の実行状況】



現場の声「次のプロジェクトでは、立上時から実行しておく」

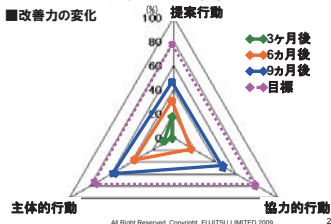
All Right Reserved, Copyright FUJITSU LIMITED 2009

24

■ 効果

All Right Reserved, Copyright FUJITSU LIMITED 2009 25

改善力の増強



チェンジ・フィールドワークの効果

プロセス改善につながる主体的な行動の増加

事例1) 部門間の協力による作業工程の短縮

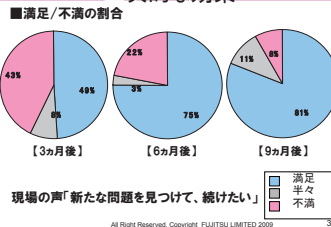
分析していた部門が互いに協力し、リスクの抽出や対策に知恵を出し合い、過剰作業を検出、作業工程を短縮。

事例2) 良い設計を継承する習慣

保守者から好評を得た設計について、他の設計者と共有し継承する習慣。

All Right Reserved, Copyright FUJITSU LIMITED 2009 26

二次的な効果

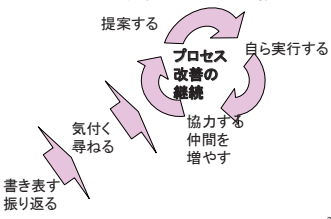


事例1) 2ヶ月間の延伸を回避

<p>【フィールドワーク結果】 部門間が分析し、障害や依頼の連絡の遅延や漏れが多発</p>	<p>【現場の反論】 幹部「他部門の事で手で回す時間が無いんです」 担当「部門間の調整は上級幹部の仕事」</p>
<p>【フィールドワークを介し相互理解】 自部門と各部門との業務の関わりを異なる視点で共有</p>	<p>【現場の変化】 幹部「部門単位の作業と影響が見えたから、他部門に対して、リスクをはっきり説明できました」 担当「自分達も誰がどう困るのかわかった」</p>
<p>部門間で、知恵を出し合い、作業の要否・優先度を見直し</p>	<p>作業期間を短縮</p>
<p>作業時間にゆとりが生まれ、改善に協力する仲間が増えた</p>	

All Right Reserved, Copyright FUJITSU LIMITED 2009 27

プロセス改善の成功の構造

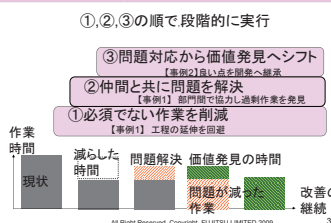


事例2) 良い事を継承する習慣

<p>【フィールドワーク結果】 問題点は管理するが、良い事は残らない</p>	<p>【現場の反論】 幹部「議事録に書けば十分」 担当「良い事の目標は嫌」</p>
<p>【フィールドワークを介し相互理解】 良い事の管理は、相手にとってもしんどい</p>	<p>【現場のリーダーからフィールドワーカーへ嬉しい電話】 「良い事は残らない」というフィールドワーク結果が気になってたけど、忙しくてそのままにしていたんです。でも部下が、他部門から好評を聞いて記録し、仲間へ伝えてくれたんです」</p>
<p>良い事の管理への賛同と共に活気と一体感が増加</p>	
<p>「良い事も伝えよう」が事業本部に認められ、ポスター化</p>	
<p>加点主義のポジティブな設計</p>	<p>「他部門連携が協力的に」</p>

All Right Reserved, Copyright FUJITSU LIMITED 2009 28

プロセス改善を継続するステップ




XDDPを使ったプロセス改善

講演者：古畑 慶次氏
(株式会社デンソー-技研センター)

XDDPとは

- XDDP : eXtreme Derivative Development Process
 - 派生開発の要素に合った合理的な開発プロセス
 - 清水吉男氏(コンサルタント)が提案
- 特徴
 - 「即効性」と「恒久性」
 - 現状のシステムで効果が出る(派生性)
 - 2000年半年ほど経たずに(派生性)
 - 3つの成果物
 - 要求定義仕様書 (派生開発要求仕様書)
 - ドキュメント管理システム (DMS)
 - 変更設計書
- プロセスの考え方
 - 派生開発のプロセスを派生開始から終了まで管理したい
 - 「すべてを管理したい」という思い込みを排除する
 - ソースコードへの格差を対応から解放する



2010/2/5 #KAGAC0817A

自己紹介

古畑 慶次 (こばた けいじ)

<所属> (株)デンソー-技研センター
<業務> 研修講師、技術支援/指導、研修企画
<専門> ソフトウェアエンジニアリング

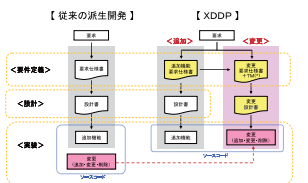
<略歴>

- 1988年: 「(株)デンソー-入社 (当時: 日本電装株式会社)」
- 1990年: 「研究開発部 配属」
- 1994年: 「選抜技術部」
- 2002年: 「IT技術部」
- 2004年: 「デンソー-技研センター」

2010/2/5 #KAGAC0817A

従来方法との比較

【従来の派生開発】 vs 【XDDP】



2010/2/5 #KAGAC0817A

アジェンダ

- 派生開発におけるプロセス改善
 - デンソーでの展開
 - XDDPの概要と効果
- 要求と仕様の明確化
 - USDMの概要
 - XDDPにおける要求仕様書
- 開発部門と支援部門の役割
 - 開発部門と支援部門の関係
 - 展開の考え方

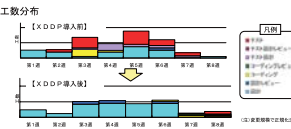
2010/2/5 #KAGAC0817A

XDDPの効果 ~SQiP 2008の発表より~

開発事例

対象	1/2007年以降	結果
開発期間	約 100%	短縮 10%
開発コスト	約 100%	削減 10%
開発品質	約 100%	向上 10%

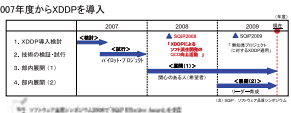
工数分布



2010/2/5 #KAGAC0817A

デンソーにおけるXDDP導入

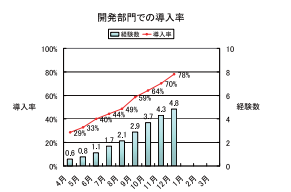
2007年度からXDDPを導入



2010/2/5 #KAGAC0817A

開発部門での展開

開発部門での導入率



2010/2/5 #KAGAC0817A

http://sec.ipa.go.jp/seminar/2010/20100205.html

要求と仕様の明確化 ～ USDM ～

- 要求と仕様
 - 要求とは
 - 機能、性能、契約条件を抽象的に表現したもの
 - 一部の要件は人・機・法・財の4つの属性を有し
 - 適切な「範囲」を示し、必要な仕様を導き出す
 - 要求仕様とは
 - 要求に基づき、具体的な説明や論理的な表現したもの
 - 仕様はユーザーに交換される
 - 設計の様子が一層ついでに「裏取り可能」
 - 仕様が表現されていることを確認するの高低を伝える（検証可能性）
- USDMのフォーマット



2010/2/5

#A06/AC08174

「開発部門」と「支援部門」

開発部門	役割	対象	顧客
開発部門	主役	製品を創出する	ソフトウェア(CO) 納入先
支援部門	脇子	開発のサポートをする	技術者・管理者 開発部門

- 支援スタッフの取り組み
 - PCA: Person-Centered Approach
 - 一人ひとりの個性や、得意分野の活用、強さを尊重するも取り込み
 - コンラチング、社風文化(徳)を中心としたアプローチ
 - 支援する際の着目点
 - 「得意」: 得意分野が得意なプロジェクトへの「得意」領域を支援する
 - 「得意」: 得意分野が得意なプロジェクトへの「得意」領域を支援する
 - 「得意」: 得意分野が得意なプロジェクトへの「得意」領域を支援する
 - 開発部門は個別に対応し、ノウハウは展開する
 - 開発部門は個別に対応し、ノウハウは展開する
 - 開発部門は個別に対応し、ノウハウは展開する
 - 組織構築は大きな数の数を子システムと、積極的に要する
 - 選択とアレンジ(追加)でプロジェクト構築を作成する

2010/2/5

#A06/AC08174

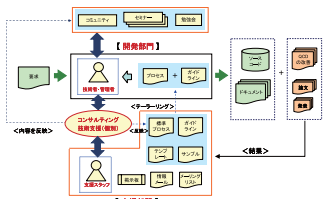
XDDPにおける要求仕様書

- 新規開発と派生開発
 - 新規開発: 要求仕様書
 - 派生開発: 「追加機能要求仕様書」+「変更要求仕様書」
- 追加機能要求仕様書 (=要求仕様書)
 - 「要求するが実現していないこと(Requirements)」について、
 - 「実現可能な(仮)要求を記述(Verify)して記述する文章」
 - 「機能仕様書は実現する文章ではない。開発も特定できない」
 - 実現を期待すること意味のある文章
 - 実現するかどうかによって書き出しの表現を変えてもよい
- 変更要求仕様書
 - 「変更要求するが実現していないこと(Change Requirements)」について、
 - 「変更する開発者が変更内容について認識を特定(Specify)できている文章」
 - 「要求」には「変更要求が可能な範囲を明確に表現する」
 - 旧仕様(Spec)と新仕様(Spec)を明確に示す
 - 変更作業を効果的に誘導する文章

2010/2/5

#A06/AC08174

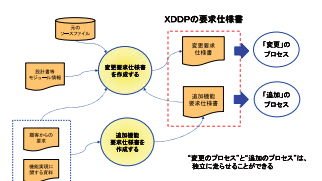
「開発部門」と「支援部門」の関係



2010/2/5

#A06/AC08174

「追加機能要求仕様書」と「変更要求仕様書」

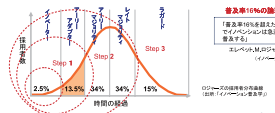


2010/2/5

#A06/AC08174

展開の考え方

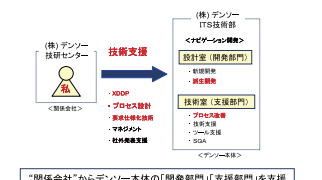
- 方針:
 - 「バージョンセンター」アプローチ (コンラチング、個別支援)
 - 基本は、コミュニケーション
- 進め方:
 - 「選手」は全体の10%の意見と意見を目標
 - 「意見」(10%の意見)
 - Step 1: まずは全体の意見から (インベータ)
 - Step 2: リーダーを育成する (アーリーアダプター)
 - Step 3: 草の根活動へ展開 (マジョリティ)



2010/2/5

#A06/AC08174

私の立場 ～ XDDPの展開 ～



2010/2/5

#A06/AC08174

まとめ

- 派生開発におけるプロセス改善
 - XDDPは派生開発向けの合理的な開発プロセス
 - テンソでも、品質、生産性の大幅な改善を促進
- 要求と仕様の明確化
 - 要求と仕様書の表現方法: USDM
 - XDDPにおける要求仕様書
 - 追加機能要求仕様書
 - 変更要求仕様書
- 開発部門と支援部門の役割
 - 主役と脇子
 - PCA、インベータ理論(意見率10%の論理)をもとに計画
 - コンラチングと技術支援中心に展開中

2010/2/5

#A06/AC08174

参考文献

- 清水吉男: 『派生開発』を成功させるプロセス改善の技術と極意、技術評論社、2007
- 清水吉男: 要求を仕様化する技術・表現する技術、技術評論社、2005
- (株)デンソー研修資料 TB向け(第1版)、派生開発プロセス(XDDP)と要求仕様の書き方(XUSDM)、2008
- (社)日本産業カウンセラー協会: 産業カウンセリング 産業カウンセラー養成講座テキスト、1998
- エレベイト.M.ロジャーズ: イノベーション心理学、産能大学出版部、1990

2010/2/5 #N/A/A20117A

展開の考え方 ~改善への取り組み~

- 方針
 - 基本は3次元モデルによる展開(伴走者の気持ち)
 - 「仕事から」ではなく、「仕事のために」展開(自分からやる)ではなく、「結果を創出する」心での積極的展開を心がけること
- 進め方
 - Step1: まずは関心ある人から
 - 「コアメンバー」から活動の10%を目指す
 - 「重要業務10%」を拡大しポイントインメンションは急進に突き進む(成功は10%から50%)
 - コンサルテンディング/セミナー/発表会、デモンストレーション
 - Step2: 組織の取り組みとして
 - コアメンバーを増やして推進する
 - リーダー育成の制度化 / 若手育成 / 横断育成 / 発表指導 / ガイディング(PI)
 - Step3: そして「草の根活動」へ
 - 誰もが参加し始める展開を作る
 - コミュニティ/勉強会 / ユーザーグループ / 情報メール

2010/2/5 #N/A/A20117A

追加資料

2010/2/5 #N/A/A20117A

導入・展開のフレームワーク

2010/2/5 #N/A/A20117A

トレーサビリティ・マトリックス (TM)

- 特徴
 - 「気づきの誘導」: いかに関与範囲を調整させるか?
 - 変更理由の抽出整理
 - 思い込みによる変更削減
 - 企業文化/社員の意識の違い
 - レビューで変更箇所/担当者/担当が確認できる
 - 気づきの誘導: 担当者 / レビューの参加者
- 列の表現
 - 「機能の単位」でまとめる (気づきを誘導する仕掛け)
 - システムやサブシステムで扱うすべてのタスク/ソースファイル
 - 変更の発生しないモジュールも列に現れる
- 表現の工夫
 - 関係するサブシステムも入れるとわかりやすい
 - エクスセル機能(グループ化)の活用
 - 機能仕様書や設計書などの文庫、ビルドのパラメータ等も入る

2010/2/5 #N/A/A20117A

具体的な展開 (1) ~ 営業を参考に ~

- 取り組みへの誘導 = 「営業活動」と同じ
 - 成果は短期間
 - 営業と異なり、顧客を育てている
 - 営業は顧客は時間ではない
 - 担当者が置かれた状況、気持ちに寄り添う
 - 担当者の状況、気持ちに寄り添う (両得意ではない)
 - 「どうしてできるの?」「どうしてできないの?」 = How are you doing?
 - PMや上司にも聞いてみる
 - 技術的な説明は明確に行う
 - なぜ、その展開が必要か? 理由があるか?
 - 何とどうなるのか?
 - 展開については妥協はない
- 「やらざるを得ない」ところへ進んでいく
 - 顧客の誘導
 - 「そこまでやってほしいからお願い...」
 - 「結果がよければいい」「結果が気に入らなければいい」

2010/2/5 #N/A/A20117A

「現場の反応」と「導入の現実」

- 現場の反応
 - できない理由
 - 「取り組む時間がない」「あのプロジェクトはまだ...」
 - 非協力的な姿勢
 - 代わって打合せできない / どうしてもできるかを考えない
 - 技術への不信感
 - 「本当に役に立つのか?」「本当に必要なの?」
- 導入の現実
 - スケジュールが合わない
 - 関係の理解を深めていない / 表のことは理解 (思っていない)
 - 時間がない
 - 取り組まない / 時間を十分かけていない
 - 上司は無関心
 - 打合せと一緒に進めようとはしない / 相談にも応じていない

2010/2/5 #N/A/A20117A

具体的な展開 (2) ~改善の基本に立ち帰る~

- 改善の基本
 - 自分のあった活動、いつでも、だれでも、どこからでも (改善ではありせん)
 - 0点から100点にするのが目標ではない - 80~90点程度
 - 初期関心であること、続けられる改善をやるのが重要
- 改善はどこからでも取り組める
 - 「プロジェクト単位でも現場単位でも...」
 - プロジェクトの進捗からでも、PM、手前や相手からでも...
 - プロジェクトの進捗からでも、PM、手前や相手からでも...
 - 「プロジェクト単位でも現場単位でも...」
 - 「その人の現場でも...」
 - 「それなら現場でも...」
 - 現場のプロジェクトから取り組んでいく
 - 現場のプロジェクトから取り組んでいく
 - 現場のプロジェクトから取り組んでいく
 - 現場のプロジェクトから取り組んでいく

2010/2/5 #N/A/A20117A

2009年7月15日
dSPOの手法を学ぶ



こちらは、海外での講演資料を参考資料として使用しましたので掲載は控えさせていただきます。

アジャイルインスペクション

講演者：永田 敦氏（ソニー株式会社）

自己紹介

- ソニー株式会社 永田 敦
ソフトウェアテストプロセス改善
- SGP研究会 第三分科会 副主宰
- SGPシンポジウム運営委員
- 京王線内務検査委員会運営委員



情報処理学会 2009 5月号特集
「ソフトウェアエンジニアリングソフトウェアインスペクションと
欠陥予防の視点」

Agile Inspection Workshop

2010/12/3

Copyright © 2010

Agile Inspection Workshop

今までの活動

- ワークショップ
 - 社内外でのべ 214人が受講
 - アジャイルインスペクションの概要と効果を体験してもらう
- 社内でのコミュニティ活動
 - ガイドラインの検討
- 社内のパイロット適応

Agile Inspection Workshop

2010/12/3

Copyright © 2010

Agile Inspection Workshop

2010/12/3

アジェンダ

- インスペクションの目的、位置付け
- アジャイルインスペクションとは
- アジャイルインスペクションプロセス
- ドキュメントの欠陥とは？
- 体験してみよう
- アジャイルインスペクションの効果
 - ケーススタディより
- まとめ
- 質疑と議論

Agile Inspection Workshop

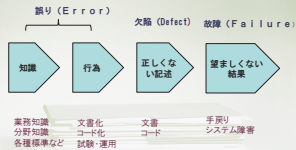
2010/12/3

Copyright © 2010

Agile Inspection Workshop

2010/12/3

誤り、欠陥、故障



http://www.kbc.co.jp/index.html | 山本新一郎 東京工科大学 准教授 | 2010/12/3

Copyright © 2010

Agile Inspection Workshop

5

ドキュメントインスペクションの従来の目的



ドキュメントの欠陥を発見する



レビューのプロセス

チェックリスト

2010/12/3

Copyright © 2010

Agile Inspection Workshop

6

コスト：インスペクション

- 開発予算の 10~15% (Tom Gilb: ソフトウェアインスペクション)

時間

・ ？時間/回

・ ？分/件

- 人：例

・ リーダー 1名

・ モデルター 1名

・ インスペクタ 6名

・ 提案者 2名

- 工数

インスペクション 人数 x 時間 x 費率

事前の仕様の読み

インスペクション 時間とコストはかかる

2010/12/3

Copyright © 2010

Agile Inspection Workshop

7

サンプリング・インスペクション

- 課題：対象ドキュメントの欠陥を網羅的に抽出する
 - 欠陥を完全にゼロにすることはできない
 - ドメイン知識、技術力、経験、文書分析力
 - 観点：開発設計者、テストエンジニア、営業、保守

⇒より上流で

ドキュメントを書く側の質を上げていく

⇒上流で欠陥の流入を防ぎコストを下げる

レビュープロセスのコストを下げる

レビュープロセスのコスト =

レビューアの人数 x レビューアの費率 x 時間

2010/12/3



Copyright © 2010

Agile Inspection Workshop

8

<http://sec.ipa.go.jp/seminar/2010/20100917.html>

AGILE INSPECTIONS:
Reviews by sampling and measuring defects

- JUSE, Tokyo 2008
- Keynote 90 minutes with Consecutive Translation (45 minutes effectively)
- Tom Gilb
- kyoritsu-pub.co.jp Tom@Gilb.com
- Gilb and Graham, Software Inspection (1993)- Japanese Edition

Copyright © 2010 Agile Inspection Workshop 9

アジャイルインスペクションのプロセス

- インスペクタ(チェッカー)を決める
- ルール(インスペクションの観点)を決める
- 欠陥密度の閾値を決める
- 実施時間を決める
- ドキュメントをサンプリングする
- インスペクタにルールなどの説明をする
- サンプルをインスペクションする(約10分から30分)
- ログをとる
- メトリクスを分析する
- 欠陥密度が閾値より低い場合、次のプロセスに進む
- 欠陥密度が閾値より高い場合、ドキュメントの質の改善のために戻す

Copyright © 2010 Agile Inspection Workshop 13

アジャイルインスペクションの目的

**高品質のドキュメントを
はじめから生産すること**

Extreme Inspection

Copyright © 2010 Agile Inspection Workshop 10

ルール

- 3つから7つぐらいのルール

一例

- 曖昧ではないか?
- 明確か?
- 矛盾はないか?
- テスト可能か?
- 設計要素がないか (要求仕様において)

Copyright © 2010 Agile Inspection Workshop 14

Agile Inspection Principle 1

The Prevent - Do not Clean Principle

- リビューの目的を、
欠陥を取り除き修正する

やり直して
“元から”ドキュメントの品質を上げてもらう

Copyright © 2010 Agile Inspection Workshop 11

ルール：曖昧

**絶対ドラゴンズに
勝ってほしい
僕はウナギだ**

Copyright © 2010 Agile Inspection Workshop 15

アジャイルインスペクションプロセス



Copyright © 2010 Agile Inspection Workshop 12

ルール：曖昧

- 子供が好きなおばさんが来た
- 太郎は自転車で逃げた花子を追いかけた
- 父は弟に自分の部屋で勉強させた

Copyright © 2010 Agile Inspection Workshop 14

ルール：曖昧

多義文

2010/12/3 Day Right 株式会社 2010 Agile Inspection Workshop 17

ルール：不明確 7

- テータの輻輳に注意して…
- 応答をいつまで待っても来なかったときは…
- できるだけ早く応答を返す
- 「以上」「以下」「同じ」

2010/12/3 Day Right 株式会社 2010 Agile Inspection Workshop 21

ルール：曖昧

- 全部網羅しなくても合格とする
- テストケースは全部できなかった
- 全部 + 否定語

2010/12/3 Day Right 株式会社 2010 Agile Inspection Workshop 18

ルール：矛盾 8

- できるだけ早く応答を返す

⇒ 2秒以内に…
→ 要件間の衝突(矛盾)

2010/12/3 Day Right 株式会社 2010 Agile Inspection Workshop 22

ルール：曖昧

AまたはBでないとき Cである

2010/12/3 Day Right 株式会社 2010 Agile Inspection Workshop 19

ルール：矛盾

- 設定された閾値(表 3-3)で検出した個数が16個以下ならすべて追加。
- 16個以上なら異常状態として追加しない。

2010/12/3 Day Right 株式会社 2010 Agile Inspection Workshop 23

ルール：曖昧 6

「入力項目は生年月日と 氏名あるいはカタカナです」

2010/12/3 Day Right 株式会社 2010 Agile Inspection Workshop 20

ルール：設計要素

要求仕様書では欠陥となる

- ATM
 - 金の引き出しにおいて、引き出し口開いてから1分以内に貨幣を取らぬと引き出し口は閉じる
 - タイムアウトの1分は内部のタイマーICにより正確に測る
- ログインアカウント
 - ユーザー名とパスワードを正しく入れたらアカウントにログインすることができる。

パラメータの指定だけでは設計問題ではないこともある

2010/12/3 Day Right 株式会社 2010 Agile Inspection Workshop 24

http://sec.ipa.go.jp/seminar/2010/20100917.html

重要 (Major) な欠陥

曖昧性, 不明確性, 矛盾性を持ったドキュメント(仕様書)の表現により
そのあとの設計, コーディングでエラーを起こして埋め込まれた,
お客様に影響のある
故障を発生するリスクをもつ欠陥.

2010/12/3 Copy Right © 森島 幸 2010 Agile Inspection Workshop 25

ケーススタディー：計画

Inspection ID	Display 1	Inspection Leader	本誌 数
Agiliter	X.XX.XXX	Agile Inspector registered	
Product	X.XX.XXX	No. Pages	1 Status
Entry Criteria with Apply			
Current Entry Status	No Exit		
Entry Criteria with Apply		10 unique defects L.P	

Documents	サブドキュメントの種類	ページ数	11ページ
ドキュメント	仕様書	20	2ページ
1st	1時間	8	設計, 設計の仕様
2nd	1時間	13	設計, 設計の仕様, SQA
3rd	1時間	9	設計, 設計の仕様, SQA
4th			2時間

ルール	イテレーションの目的
非侵襲性	ボイ
明確性	チェック
非侵襲性	チェック
明確性	チェック
非侵襲性	チェック
明確性	チェック

2010/12/3 Copy Right © 森島 幸 2010 Agile Inspection Workshop 29

インスペクション

・ Agile Inspection Workshop

やってみましょう

どのくらい、欠陥があるか 体験してみよう

- **ルール 対象 要求仕様書**
- あいまいでないこと
- 明解であること
- 矛盾のないこと
- 設計要素がないこと

・ 1ページ分15分かけてチェックしましょう

2010/12/3 Copy Right © 森島 幸 2010 Agile Inspection Workshop 26

アジャイルインスペクションの結果例

Inspection Data Summary

Date	2010.9.24	ID	10-01	Inspection Leader	川口 龍
Product/Document Reference	agiliter	Project	アゲイル	Page	1 page
Goal Page	1 Page	Start	10:00	End	10:15

Individual Checklist Results

No.	Inspector	No. of errors	Average (Time/Defect)	Major Issues	Value L.P	Checking time (hour/30)
1	A	20	10.0	2	2.0	200
2	B	10	20.0	1	1.0	100
3	A	20	10.0	2	2.0	200
4	B	10	20.0	1	1.0	100
5	A	20	10.0	2	2.0	200
6	B	10	20.0	1	1.0	100
7	A	20	10.0	2	2.0	200
8	B	10	20.0	1	1.0	100
9	A	20	10.0	2	2.0	200
10	B	10	20.0	1	1.0	100

Average of defects: A: 2.0, B: 1.0

Estimated Value/ Major Defects: 21.8

Value/ Major Defects: 21.8

Score: 1.8

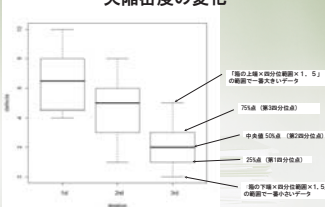
2010/12/3 Copy Right © 森島 幸 2010 Agile Inspection Workshop 30

湯沸かしポット



2010/12/3 Copy Right © 森島 幸 2010 Agile Inspection Workshop 27

欠陥密度の変化



「最の上端×四分位範囲×1.5」の範囲で一番大きいエラー

75%線 (第3四分位)

中央値 50% (第2四分位)

25%線 (第1四分位)

「最の下端×四分位範囲×1.5」の範囲で一番小さいエラー

2010/12/3 Copy Right © 森島 幸 2010 Agile Inspection Workshop 31

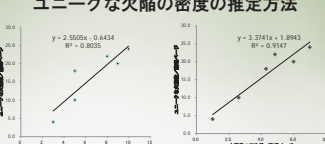
ロギング

- ・ **Respect & Influence**
- 敬意をもち影響を与える
- ・ レビューは合意の技術
- 自分が欠陥とされていることを理解してもらい修正のモチベーションを持ってもらう

欠陥を飲んでもらう

2010/12/3 Copy Right © 森島 幸 2010 Agile Inspection Workshop 28

ユニークな欠陥の密度の推定方法

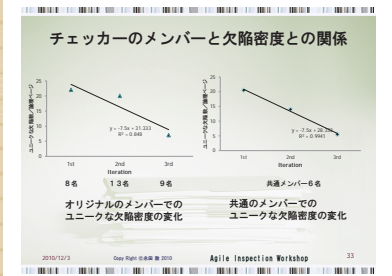


$y = 2.5502x - 0.6434$
 $R^2 = 0.8203$

$y = 3.3741x + 1.0943$
 $R^2 = 0.9197$

ユニークな欠陥の密度は、欠陥密度の平均値から推定することができる。

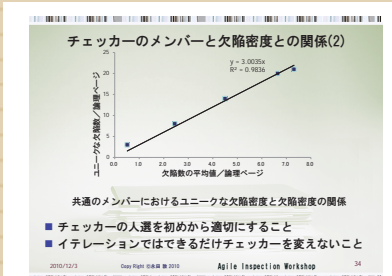
2010/12/3 Copy Right © 森島 幸 2010 Agile Inspection Workshop 32



インクリメンタル・レビュー

- 最初の1ページを書いたところから始める
- ドキュメントの作成スケジュールに従い、定期的、計画的にアジャイルインスペクションを実施する
- イテレーティブな実施によりドキュメントの品質を上げていく
- 書き手ごとに別のイテレーションを回す

2010/12/3 Day Right 発表者 2010 Agile Inspection Workshop 37



インクリメンタル・レビュー

パッチ的レビュー

数百ページのSRS

インクリメンタルなレビュー

初期の1ページからレビューを始めてしまう

2010/12/3 Day Right 発表者 2010 Agile Inspection Workshop 38

どこの範囲まで修正するか

- 指摘されたところ以外も直してくれるか?
 - 指摘されたところの周辺
 - 全てを見直しているわけではない
- 変更されたところ、追加されたところは、直されている
 - 欠陥密度は減っている
- ドキュメントが小さいうちから、インスペクションを行うと効果がある

2010/12/3 Day Right 発表者 2010 Agile Inspection Workshop 35

インクリメンタル・レビュー

複数人でSRSを作る場合

A B

A B

2010/12/3 Day Right 発表者 2010 Agile Inspection Workshop 39

Agile Inspection Policy

・REVIEW EARLY

- リビューは早いうちに行っているところからやる。

全部そろってまで待つ

2010/12/3 Day Right 発表者 2010 Deepen Your Review Process, Tom Ito 2000 Agile Inspection Workshop 36

インクリメンタル・レビュー

A B

作成者ごとにインクリメンタルにレビューをおこなう

パースペクティブなレビュー

2010/12/3 Day Right 発表者 2010 Agile Inspection Workshop 40

<http://sec.ipa.go.jp/seminar/2010/20100917.html>

まとめ

- ・アジャイルインスペクションは、ドキュメントの品質を向上させる効果があります。
- ・判定基準のカギとなるユニークな欠陥の密度は、検出された欠陥密度の平均値で推定できます
- ・チェッカーは途中で変えないようにしましょう
- ・アジャイルインスペクションは早いうちにできているところからやりましょう

2010/12/2

Copyright © 2010

Agile Inspection Workshop

41

アジャイルインスペクション やりませんか？

ご清聴
ありがとうございました

2010/12/2

Copyright © 2010

Agile Inspection Workshop

42



レビュー・インスペクションの 効果的実践と阻害要因 ～ QI 法の実践事例の紹介～

講演者：細川 宣啓氏
(日本アイ・ビー・エム株式会社)

世界で最初のバグ 1947年9月9日(1947-09-09)

The First "Computer Bug" (1947)
The first computer bug was a fly that got stuck in the relay contacts of the Harvard Mark II computer. It was discovered on September 9, 1947, by Grace Hopper and her team. The incident led to the term "computer bug" and the idea of "debugging".

予防/予測/検出の技術 欠陥の発生と除去タイミング

- 上流で埋めこまれるバグは影響度が大きく、除去が困難なため、重大なバグが漏れしないよう防止しておく
- テストスタート時の残存欠陥数を最小にする
- 欠陥が生まれたらできるだけ早く検出する。

検出フェーズ

テスト中心プロセス vs 予防・予測中心プロセス

下記の表のように、修正やテスト中心のプロセスよりも、品質に注力した予防・予測中心プロセスのほうが約80%の欠陥削減が観測される事例が報告されています。

Total Issues	128.7
Defects Prevented	93.3
Defect-Centric	128.7
>> Result	72.5% (-27.5%)

生産性と品質の関係

単純に生産性だけ上げてもダメ（品質を下げると生産性が下がる）

生産性

品質

品質はソフトウェアで品質管理が難しい。品質管理は、品質管理の仕組みを構築し、品質管理の仕組みを運用する。

トラブルプロジェクト発生メカニズム

トラブルプロジェクトは発生するものではない。トラブルという概念（＝警告）、次の4要素の連鎖（P-M-I-M）からトラブルが発生します。

1. Problem (問題) → 2. Misunderstanding (誤解) → 3. Incomplete Information (不完全な情報) → 4. Miscommunication (誤伝)

図1 Heinrichによる事故のモデル

予防/予測/検出の技術 品質管理の3C: 予防+予測+検出サイクル

- 1) 予防
 - 欠陥の再発予防
 - 作業の遅れ予防
 - 同種欠陥の再発予防
- 2) 予測
 - メトリクスデータ・モデル
 - メトリクスデータ報告書
 - データ収集（＝測定）
 - 数値で兆候を捉える
- 3) 検出
 - 通常のインスペクション
 - 上流重視のプロセス
 - 目視で確実に検出

Theory of Inspection

Copyright©2010 Nobuhiko Hasekawa

<http://sec.ipa.go.jp/seminar/2010/20101119.html>

様々な品質レビュー活動

コストはかかるが、最も欠陥検出・欠陥予防効果の高い手法をQIプロセスのベースに採用

Copyright © 2010 Nishimura Hosokawa

フォーマルインスペクションとは4つのロール

インスペクションの目的は「致命的欠陥 (Operation Defects)」を検出すること

Copyright © 2010 Nishimura Hosokawa

フォーマルインスペクションとはプロセス

Copyright © 2010 Nishimura Hosokawa

レビューの大敵:

- プロジェクト制約:
 - 予算制約・時間制約 → 丁寧に実施すると時間がかかる
 - 安易な商談 → 「早くリリースせよ」というプレッシャー
- 慢心・おこり・心理的思い込み
 - 「自分だけは品質がよい、大丈夫だ」という感覚
 - レビューすれば必ず品質があがる
- 過去の悪習・ベテランのエゴ
 - 「いつもこうやってきたから」

Copyright © 2010 Nishimura Hosokawa

The Art of Quality Inspection

Copyright © 2010 Nishimura Hosokawa

目視品質検査の重要性とIBM-Quality Inspectionの導入

一般的にレビューやインスペクション等による品質管理作業の重要性は広く認知されています。しかし、それらの多くは単手取字のみを継承する形式プロセスに過ぎ、労力に見合った効果が見られず、検査そのものの導入も、顧客すべての改善活動と見做れないなど多くの課題を内包します。

IBM Quality Inspection (品質インスペクション) とは?

- 2000年に日本アイ・ビーエムのQA組織で開発された、プロジェクト外側の第三者による目視の品質検証サービスです。
- 弊社が独自に開発した品質検証活動 (QI: Quality Inspection) では、欠陥検出効率は従来従来約5%、品質検査工数全体の約10%削減にプロジェクト全体フェーズ中での顕著な効果が期待できます。
- QI手法によってプロジェクト生産物 (仕様書・コード・テスト計画書等) の対象を見直しによる効果検証を実施することで
 - 早期欠陥発 品質収収可視化
 - 品質のばらつき抑制

を実現し安定したプロジェクト進捗とコスト超過予防を実現します。

Copyright © 2010 Nishimura Hosokawa

1) お客様向け品質エンジニアリング活動

グループ 戦略的競争優位確保 1-1) プロジェクト計画支援 (Project Plan) 1-1-1) 実行システム構築サービス 1-1-2) 要件定義支援サービス 1-1-3) プロジェクト新業務支援サービス	1-2) プロジェクト実施支援 (Project Execution) 1-2-1) 第三者品質検証サービス 1-2-2) 品質保証サービス 1-2-3) PMO品質管理支援サービス 1-2-4) 仕様管理支援サービス 1-2-5) 品質管理支援サービス
グループ 中期競争力強化 2-1) 開発/保守品質支援 (Development/Operation) 2-1-1) サンプル診断サービス 2-1-2) 品質改善支援サービス 2-1-3) 検出率向上支援サービス	2-2) 運用/保守品質支援 (Maintenance/Operation) 2-2-1) システム信頼性向上サービス 2-2-2) 信頼性向上支援サービス 2-2-3) 検出率向上支援サービス
グループ 長期的競争優位確保 3-1) 検出率向上支援サービス 3-2) 品質管理支援サービス	

Copyright © 2010 Nishimura Hosokawa

Ref) Effectiveness: Competitiveness in Japan Market

Workload (effort-days)

A) Workloads needed
 - Other Competitor: 150 effort-days
 - IBM-Japan QI team: 9 effort-days

B) # of defect type detected.
 - Formal Inspection: 64 type of defects detected
 - Quality Inspection: 62 type of defects detected

Copyright © 2010 Nishimura Hosokawa

Fault-Proneモジュール予測とは？

「**変更**」
ソフトウェアを構成する要素集合(モジュール集合)の中で、**フォールト(障害/欠陥)を含有している可能性の高いモジュールを特定する技法**

- Fault-Proneの捉え方
 - 欠陥の有無
 - 欠陥の発生率
 - 欠陥分類を考慮
 - 修正工数を考慮
- "Fault"の識別タイミング
 - 出荷前の欠陥
 - 出荷後の欠陥

特定のモジュール内に欠陥がある/ない(最も一般的)
特定のモジュールにいくつかの欠陥がある(絶対一般的ではない)
特定のモジュールに高い欠陥発生率(例:Chen, 2008)
例:欠陥修正で追加/変更されたコード行(Kitaguchi, 1997)

結合子など、機能テストで検出され欠陥の95%を予測
この結果は100%ではない

Fault-Proneモジュール予測の判明技法
ソースコードから判定できる情報
設計ドキュメントから判定できる情報
プロセスに関する情報

●LOC(Cyclomatic Complexity) Metrics
●設計要素のグラフ理論(Onison, 1990)
●Hamlet(1997)優美点と劣質

コーディングの技術

早く読むための技術

レビューを実施する際、なるべく早く(コード作成者の意図を汲み取り、設計基準を把握した方が、時間効率/欠陥検出効果が高まります)

コードの目視レビュー時には、ソースコードを「文字」「文字配列」「数値処理」「プログラムの塊」「日本語(コメント)のみ」の集約(「着目点」)として捕らえる等様々な視点を変えて全体をすばやく把握することが重要です。

最も欠陥検出率の高い目視レビューが実施できる限られた時間の中で、いかに「目視自反プログラミング」を高めるかが全体品質を高める鍵を握ります。

次ページ以降に示した技術はコーディングの効率化と、より大域的な「コード断片化」の技術です。どのような場面でもコードに関わるあらゆる担当者にとって有用な技術を中心に伝えます。

Let's Practice!

Copyright ©2010 Nobuhito Hoshinaka

2010.11.19 BECセミナー「Javaセオリーとベストプラクティスワークショップ」

早く読むための技術:コードはお尻から読む

問題	コードは読者が読む
原因	コードが修正される際、行数やメソッドの追加はファイルの末尾に追加される場合が多いため、実現に追加する領域でだけ光の当たりにくい箇所が存在する可能性がある
対策	● 最高行数の行、元のコードの行数を十分に理解しない場合はコード修正が入り込みやすい ● 行数が多い箇所はリスト容易性(=試験の一角)が低下する

```

public class Animal implements IBird {
    private String name;
    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
    public void fly() { System.out.println("Fly!"); }
}

public class Duck implements IBird {
    private String name;
    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
    public void fly() { System.out.println("Fly!"); }
}
    
```

●publicメソッド?
getXXXメソッド命名違反
-1,7という定数書き

インスプレションの飛と穴:「欠陥は仕様書と仕様書の間に落ちる」

インプレションは改善できるはず
「**全てのインプレションが全体像から**」
という原則に従わないと、各ページを独立完了しても、間に落ちる欠陥(=不整合)を見落とすことになる

インプレションで輸出しきれない不統一「互換性欠陥(不整合)」といった欠陥は比較的初期で存在インプレションでも検出しやすいです。しかし全体像を把握した上で仕様書間に発生する整合性欠陥(=不整合)欠陥が最も検出が難しく、後フェーズで検出されることが多い。インプレション作業中の注意ポイントの一つです。

早く読むための技術:ファイル名・行数から着目

問題	ファイル名、変数名が、どれか1つの部分のみが利用される
原因	バージョンアップの際に追加する、汎用標準語、言語、別の言語、VBETC関係者の開発者が存在する。利用が目的が異なる。
対策	● 適法な標準、元のコードの行数を十分に理解しない場合はコード修正が入り込みやすい ● 行数が多い箇所はリスト容易性(=試験の一角)が低下する可能性がある

```

public class SampleJava {
    public void main() {
        try {
            // ...
        } catch (Exception e) {
            // ...
        }
    }
}
    
```

● 関係長、長いメソッド、メモリー処理、例外処理
● new/Singletonized
キャスト

ばらつき検査:「最初と最後のページ」

全ての仕様書(要件定義書、設計書)は、その**記述詳細度が適切か**を判定するため、同一種類の仕様書の「**最初ページ**」と「**最後ページ**」に記述された内容を比較します

- 多数の仕様書検査(例:機能記述)を行う場合、全てを検査する時間がない場合は、**最初と最後のページ**と比較し、**仕様書の記述精度に差異がないこと**を確認する
- もし、**数度差異、詳細度差異が検出された場合**には、プロジェクト全体の制約や開発者の業務量、クォリティ/コスト/不整合/仕様抜け等も同時に確認する

全体を俯瞰するための技術:コードズームアウトする

Microsoft Word等の「ズーム」機能が付いた「ズームアウト」表示すると全体の制約構造が把握しやすくなり、どこにコードを見るべきか把握しやすい

SampleJavaのWord表示(標準的なズームアウト)

検査結果:

1. 黒いところ→処理が複雑で重い可能性が高い
2. 右にこんなところ→ネストが深い構造、複雑な分岐やループの可能性
3. メソッドの塊は見えるか? →メソッドコメントの存在とその統一を観察

http://sec.ipa.go.jp/seminar/2010/20101119.html

Quality Engineering, Services Quality IBM

全体を俯瞰するための技術:コードをソートする

- コードを各行の文字数で降順ソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある

例1) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある

```

1) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある
2) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある
3) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある
4) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある
5) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある
6) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある
7) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある
8) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある
9) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある
10) 異なる行数のコードをソートすると、コピー＆ペースト状況や処理の抜けを検出できる場合がある

```

ソートされたコードからは、①実行に際してエラー発生、②コンソール出力条件が満たれていない箇所、③バグやエラー箇所が一目で定量的に明らかになるというメリットが確認できる

Copyright © 2010 Nobuhiko Hoshikawa

Quality Engineering, Services Quality IBM

マトリクス測定の動機

- 品質に関する証拠・確認がほしい
- 製品リリースに拠り所や基準がほしい
- より早い段階で欠陥(とその兆候)を捕らえたい
- 計測データなしではプログラムを納品できないから
- 理由はないが決まりだから・仕方なく
- 数字を測るとなんとなく安心する
- 定量データ好きの管理者がいる・測定マニアがいる
- 経営指標として測定が義務付けられている

Copyright © 2010 Nobuhiko Hoshikawa

Quality Engineering, Services Quality IBM

全体を俯瞰するための技術:コメントの記載を眺む

- コメント文中の「**」**や「**」**の入りに着目すると、複数人数で作成・修正を行っている箇所が分かる

例1) 書名と書籍説明内容が、「**」**が同じか?

- 例1) 書名と書籍説明内容が、「**」**が同じか?
- 例2) 書名と書籍説明内容が、「**」**が同じか?
- 例3) 書名と書籍説明内容が、「**」**が同じか?
- 例4) 書名と書籍説明内容が、「**」**が同じか?
- 例5) 書名と書籍説明内容が、「**」**が同じか?
- 例6) 書名と書籍説明内容が、「**」**が同じか?
- 例7) 書名と書籍説明内容が、「**」**が同じか?
- 例8) 書名と書籍説明内容が、「**」**が同じか?
- 例9) 書名と書籍説明内容が、「**」**が同じか?
- 例10) 書名と書籍説明内容が、「**」**が同じか?

合計6名の開発、コード行数(4004行)にたいして、「**」**が同じか?という質問が投げかけられる

Copyright © 2010 Nobuhiko Hoshikawa

Quality Engineering, Services Quality IBM

医学の場合

- 医療行為のサイクル

Copyright © 2010 Nobuhiko Hoshikawa

Quality Engineering, Services Quality IBM

Mystical Metrics Measurement

Copyright © 2010 Nobuhiko Hoshikawa 2010.11.19 SECセミナー プロセス改善&ストラテジクスワークショップ

Quality Engineering, Services Quality IBM

Pump up! Your Quality

Copyright © 2010 Nobuhiko Hoshikawa 2010.11.19 SECセミナー プロセス改善&ストラテジクスワークショップ

Quality Engineering, Services Quality IBM

マトリクス測定の動機:「なぜプロジェクトを測定するのか?」

- 経営者、プロジェクト管理者、リーダーズといった60%以上は測定は必要ですが、特になぜ指標測定(マトリクス測定)が必要か?という質問に対して代表的な「マトリクス測定の動機」を以下に列挙します。
- 重要な点は「測定の目的」を明らかにすることで、目的に合致した、様々なレベルの範囲に答えることが可能な「的確な」測定でなければ時間とコストを費やす意味がありません

様々な意思決定・判断を正偏化する

プロジェクト管理者の目標を透明化する

重要なトレードオフの判断を下す

問題を早急に特定し決定する(強制的な意思決定)

マトリクス測定

Copyright © 2010 Nobuhiko Hoshikawa

Quality Engineering, Services Quality IBM

マトリクス測定データの利用:「組み合わせる使う」

品質には絶対値がないため、品質マトリクスを単独で使っても無意味

組み合わせる利用する

描画して「品質傾向を掴む」

- 感情的偏入性はあるか?
- 管理・問題はあるか?
- 予防/抑制できるか?
- 気鋭性は確保されているか?
- 確信が念性か?
- スキルか経験か?
- 真の要因は何か?

Copyright © 2010 Nobuhiko Hoshikawa

Quality Engineering, Services Quality

欠陥を予測するための測定：品質バイタルサイン (QVS)

品質バイタルサイン (QVS) とは?
 - 非果物の中身や意味を熟練テスト (Stare Test) 実施前に、成果物のあらゆる外的因子を測定し、加工して成果物検証作業の入力として利用する。この時に測定する外的因子を「品質バイタルサイン」(以下QVS: Quality Vital Signs) と呼ぶ。

- プログラムによる継続的な輸出 - グラフ等による視覚化
- パターン分析による推奨事項/リスクアセス

【効果】

- 品質検証時間/TATの短縮
- 属人性の排除+品質検証コストの削減
- 欠陥検出活動の精度向上

Copyright © 2010 Nishino Hoshikawa

Quality Engineering, Services Quality

COBOLコードの内部構造図

Copyright © 2010 Nishino Hoshikawa

Quality Engineering, Services Quality

AMH (意味的分散度) 検査の場合

AMH (Meaningful Mutation) Inspection

第1期	急性期	1) 実行 / 検査結果受取	2) 初期診断	3) 初期診断	4) 初期診断
	慢性期	3) 初期診断	4) 初期診断	5) 実行	6) 検査結果受取
第2期	回復期	7) 治療計画立案 / 検査結果受取	8) 検査結果受取	9) 検査結果受取	10) 検査結果受取
	終末期	11) 検査結果受取 / 検査結果受取	12) 検査結果受取	13) 検査結果受取	14) 検査結果受取

Copyright © 2010 Nishino Hoshikawa

Quality Engineering, Services Quality

3D-Plot : Visualizing tool for project quality status

- Plot metrics to visualize quality
 - X-axis: "Risk Factor" / LOG
 - Y-axis: "Risk Exposure" / Comment Rate
 - Z-axis: "Complexity" / # of F
 - Color: Sub-system groups
- Findings
 - Found 1 outlier of size metrics
 - Comment rate is wide distributed
 - "Green" is size controlled well.
 - "Yellow" has less comment and complex
 - > Need careful test management.
- Recommended Action:
 - Use outliers on 3 aspects and monitor them
 - Re-design target size program
 - Add comment to small size scenarios.

Copyright © 2010 Nishino Hoshikawa

Quality Engineering, Services Quality

可視化表現の例1) Tree表現

通称: ネバナハ納豆モデル

Copyright © 2010 Nishino Hoshikawa

Quality Engineering, Services Quality

Indispensable "Defect Engineering"

Copyright © 2010 Nishino Hoshikawa

Quality Engineering, Services Quality

可視化表現の例2) Graph表現

"Rational Software Analyzer" ツール出力
 「Architectural Discovery」機能による
 クラス間依存関係 (Affect/Effect) の可視化

Copyright © 2010 Nishino Hoshikawa

Quality Engineering, Services Quality

Problem need to solve : 問題定義

- バグ記録してますか?
- バグ記録を活用してますか?
- プロジェクト内でバグ情報を記録してますか?
- 病気標本と病理学研究みたいものが、ITの世界には?
- プロジェクトと欠陥の関係は?


Copyright © 2010 Nishino Hoshikawa

Quality Engineering, Service Quality

インスペクターの思考1) = 洞察+記憶+俯瞰

洞察方法: まず全体量を把握し単位に分割

- 「それは一言で表現するとどんな処理か?」
- 「上位概念に抽象化すると何の処理か?」
- 「解決すべき問題は換言可能か?」
- 「一意に解釈できるか? 一般的表現か?」
- 「類似の改善-処理は存在するか?」




Copyright © 2010 Nishimura Hirokazu

Quality Engineering, Service Quality

インスペクターの思考5) 試験性: Testability

試験性検証: テストできなければ仕様確定とは言えない

- 実行可能性 (Operability)
- 観測可能性 (Observability)
- 制御可能性 (Controllability)
- 分解可能性 (Decomposability)
- 単純性 (Simplicity)
 - 機能単純性: Functional Simplicity
 - 構造的単純性: Structural Simplicity
 - コード単純性: Code Simplicity
- 安定性 (Stability)
- 理解可能性 (Understandability)




Copyright © 2010 Nishimura Hirokazu

Quality Engineering, Service Quality

インスペクターの思考2) = 洞察+記憶+俯瞰

記憶方法: 資料から離れる

- 金ワコーロー10%〜20%を「俯瞰」に乗せる
- 俯瞰後、記憶に落とし単語を書き出す
- 何が書かれていないか口頭で質問する
- 書かれていないものを尋ねる
- 別の言葉を呼び添えてみる




Copyright © 2010 Nishimura Hirokazu

Quality Engineering, Service Quality

インスペクターの思考6) 内省的観点

検証者の検証: 自らの検証観点を検証せよ

- 自らが検証する範囲/しない範囲
- 集中力の持続時間
- 取捨選択/バイパス
- 先入観/過去の経験/情報集約/お約束
- なぜそれが欠陥だと思ったのか?
- なぜその観点に辿り、気づいてしまったのか?
- その欠陥に改善はあるか?




Copyright © 2010 Nishimura Hirokazu

Quality Engineering, Service Quality

インスペクターの思考3) = 洞察+記憶+俯瞰

俯瞰方法: 読み上げている対象の前後に着目

- 書かれていない「前提」「後続」「指定」「条件」はないか?
- その仕様でテストはできるか?
- 作者は、何のために仕様書を書いたのか?
- 何が足りていないのか? 何が書かれているのか?
- なぜこの位置で完成し書けるのか?
- 統一性/適合性/可読性等「商品」として適切か?
- 全体としてJSはあるか?



Copyright © 2010 Nishimura Hirokazu

Quality Engineering, Service Quality

インスペクション実施の観念-欠陥の見つけ方

インスペクションの思考プロセス

```

    graph TD
      A[1) 資料の全体像把握] --> B{2) プロセス上の欠陥はあるか?}
      B -- Yes --> C[3) プロセス上での欠陥はあるか?]
      B -- No --> D[4) 検取サンプリング検証]
      C --> D
      D --> E[5) 詳細検証]
      E --> F[6) 総評まとめ]
      F --> G{7) 対策立案}
      G -- 否 --> H[8) 再発防止策立案]
      G -- 是 --> I{9) 対策状況チェック}
      I --> G
  
```


Copyright © 2010 Nishimura Hirokazu

Quality Engineering, Service Quality

インスペクターの思考4) 初めて出会う技術に対して

新技術対応: 技術が生まれた背景を調査

- その技術が解決すべき問題はなにか?
- その技術が解決できないことは何か?
- 個人性/リスクの軽減/回避はあるか?
- 技術の寿命はどのくらいか? 続いた技術か?
- 既存技術の組み合わせで代替可能か?



Copyright © 2010 Nishimura Hirokazu

Quality Engineering, Service Quality

1) レビューに対する期待と現実

「設計ミスを防ぐためにレビューを行う」という目標を掲げること

- 「早期の欠陥検出効果」
- 「欠陥予防効果」

レビューを行う利点は一般的には以下のように書かれている

- 上流欠陥の早期検出
- 下流での欠陥混入予防策
- 初期導入コストの低い品質活動
- 下流でのテスト容易性確保
- 教育・スキル移転効果

→ 上記利点を得られるレビューを開発の現場で実践できているか?

Copyright © 2010 Nishimura Hirokazu

http://sec.ipa.go.jp/seminar/2010/20101119.html

Quality Engineering Service Quality

2) 実際のシステム開発の現場では...

- 現実のレビュー会議
 - 議論が発散する
 - 時間がかかり重大な欠陥が検出できない
 - 雰囲気が悪くなる
 - 「若手演」
 - なじみあいウケカ
- 一般的なレビューの課題
 1. レビュー目的が曖昧なまま会議を招集する
 2. レビュー実施時期と回数: フェーズ終盤で1回だけレビューする
 3. 実施時間・工数不足に起因するプレッシャーから雰囲気が悪くなる
 4. 統一した観点が存在しないため議論がブレる

Quality Engineering Service Quality

2-3) 実施時間・工数不足に起因するプレッシャーから雰囲気が悪くなる

- レビューが失敗する一因大きな原因は「参加者同士のウケカ」「参加者同士のなじみあい」

 - 時間的なプレッシャー
 - 1度聞けるあるために生じる必死・偏屈意識
 - 開発するシステムに対する各設計者の「思い入れ」の差
 - 技術者の経験値・経験の差

- 参加者同士の「ウケカ」が発生する主な理由と心理は大きく分けて2つあり、

 - 参加者が結果としてこの目的レビュー「今」に集中してしまいたいという心理が
 - 何度もしようするはややしい、かりすぎる工数・時間を嫌う心理
 - 一人の設計者も両面から、自分の設計も人は見せたくないという心理
 - 自分設計を過信するあまりレビューは不要と思いつく心理
 - (レビューは時間がかりの面なので早く終わらせようという心理から)バグを数多く叩きだして早くレビューを終わりたい

- 統一した観点が発生しない一因の理由は「技術者同士を尊重しない」気持ちと「時間プレッシャー」による場合が多い。

Quality Engineering Service Quality

2-1) レビュー目的が曖昧なまま会議を招集する

- (次のページに一般的なインスペクションのプロセス:流れを示す)
- 図中、②の集合会議と呼ばれるミーティングにおける「レビュー目的」が不明確なまま招集される。
- 本来は①のレビュー計画段階にて目的や観点を共有し、事前配布された資料に対して個人単位で最高責任を行い、各参加者が観点を洗い出し、意見を述べ、
- 今回のレビュー目的が資料配布時以前に共有されないため、どのような観点を着目しどのような欠陥を重点的に検出するか判断が参加者によりかわられてしまう。
- 参加者が「思いついたこと」を指摘する場となり、議論が発散してしまふ。
- このような状況では、技術者として経験の長い人、年長者、声の大きい人だけの意見が優先され、会議の場が「意図するためのレビュー」にしてしまう事象が少なくない。(「二度と参加したくない」参加者の発生)
- 参加者の目的意識が統一されておらず議論が混乱・発散するのは当たり前。

Quality Engineering Service Quality

2-4) 統一した観点が存在しないため議論がブレる

- 統一したレビュー書籍や論文には
 - 手順
 - 開催形式
 - ルール
 - 役割
- 等しいレビュー書籍の「方法論」は多く記載されている
- しかし、レビュー会議を行うに当たってどのような観点をレビューを行うかは一冊の書籍にはない。
- 多くのレビュー書籍ではレビュー観点が一定で決められているという内容が多い。
- レビュー観点は様々なバリエーションを必要とすべきであり、統一した観点を押し付けることは、
- 一冊の書籍や論文に頼りてはならない。自分の経験や知識から、設計上のシステム構造や機能・インターフェースの観点などから観点を洗い出し、共有する必要がある。そのためには「重大欠陥に集中するべき」観点を事前に共有し、議論の場では観点を共有し、議論を進める必要がある。
- 結果として参加者個人間の経験や知識の差によって異なる観点が多く出てくるレビュー会議での観点が異なることになる。(＝議論しやすい体制)

Quality Engineering Service Quality

参考) 一般的なフォーマルインスペクションのプロセス

Quality Engineering Service Quality

3-1) レビュー計画段階

- レビュー計画は、プロジェクト開始前に行う。プロジェクト管理計画に基づき、品質管理計画が立案された段階で、計画の下流に下流のレビュー計画を策定する必要がある。特に工数・時間・コスト、当初、中期、最終の3回を計画すべきである。かつ、そのうち上流の時期は必須で、それはより早く実施されるべきである。同時に決定された後、観点を洗い出し、共有する必要がある。観点を共有する必要がある。
- 以下に詳細な内容を示す。
 - ① 目的: 最初の日時は短く・短時間で全体品質傾向の把握のためとする。
 - 重点目標: 大人数が参加してレビューの雰囲気や経過を把握してもらうこと
 - ② 二重の開催
 - 初期: 現場が最初の仕様書や仕様の正確性を確認する
 - 中期: 各人が担当する部分の正確性を確認する。個人的欠陥の存在を共有し、個人レベルで認識させる。この段階での欠陥は、修正が容易である。修正が容易である。修正が容易である。修正が容易である。
 - 最終: 最終的な仕様書や仕様の正確性を確認する。修正が容易である。修正が容易である。修正が容易である。修正が容易である。
 - ③ 開催の頻度
 - 頻度: 工数終了後から修正し、修正可能な日数を確保する
 - 目的: 設計士様の完了に向けて、未決事項・修正・欠陥検出の検出
 - 効果: 試験性欠陥の特定が容易になる
 - 重点目標: 例外欠陥の統一、試験性確保、要件適合性、インターフェース確認

Quality Engineering Service Quality

2-2) レビュー実施時期と回数: フェーズ終盤で1回だけレビューする

- レビューは最終に固行実施するべきではない
- 成果物作成直後で「こまめ」にレビューする

Quality Engineering Service Quality

3-1) レビュー開催計画のコツ (続き)

- 短時間複数回: 複数回を細かく頻りに実施するほうが1回長時間大工数をかけて実施するよりも効果的である。後者がいい理由も、後者よりも修正時間がないという課題を克服できる。例: レビュー一回を30分として実施計画した場合、週に2時間30分の設計レビュー開催(1時間を2回以上)に相当する。
- レビュー=4人以下になるように計画する。参加者が人により異なる観点が検出される傾向があるという統計がある。
- レビュー会議には設計担当者=仕様作成者を含めること

Qualify Engineering, Service Quality

3-2) レビュー準備段階のコツ

レビュー会議前の準備段階でのコツを以下に示す。

- 場あたりのなレビュー開催では前述の「単に雰囲気・気分の悪い、怒られる場所」としての会議にしかならない。入念な準備が必要。
- レビューの目的や重点観点などは、各人が事前準備を行う以前の資料配布前に確定しておく必要がある。
- レビュー会議前に必ず参加者個人が準備を行うこと。軽微欠陥は個人準備の段階で検出・列挙する
- 参加者は、今回のレビューが何回目のレビューかを意識すること。前回までに検出した欠陥、および過去の類似プロジェクトで検出したような欠陥が存在しないか十分に調べてレビュー会議に挑むこと。

Copyright © 2010 Nobuhito Hironaka

Qualify Engineering, Service Quality

第三者品質検査導入のメリット

下流プロセスで検出する製品・サービスの欠陥は、前道の4種類の欠陥が原因です。これら4欠陥は「開発者本人が気づきにくく、再発も学習しにくい」という特徴があります。この欠陥を upstream で集中的にレビューし、事前に検出することで downstream の欠陥を削減します。

開発段階で検出する欠陥＝軽微 検出するプロセスでの選別＝欠陥

Copyright © 2010 Nobuhito Hironaka

Qualify Engineering, Service Quality

The world of IV&V

Copyright © 2010 Nobuhito Hironaka

2010.11.19 SECセミナー 2Dセビアス・システム・ソフトウェア・ワークショップ

Qualify Engineering, Service Quality

調達管理上の品質管理:「点検V字モデル」

- 近年の調達管理構造の複雑化によって従来の「点検」と呼ばれた開発プロセスが新形態を醸成しています。新形態が従来の工程のつなぎ目で仕様伝達遅延・確認不足等の問題を発生しています。
- 従来の「点検」は開発者と納品業者間の品質管理が、品質管理で必要な品質管理活動を確保できなかった。多くの場合「品質管理」が従来、行われていない品質管理の専門家による管理であった。
- 弊社は、各社内では「品質管理の役割」を担う品質管理の専門家・検査員による一貫した品質管理の導入が最も有効な手段でもあり得ます。第三者検査も多くの場合有効です。

Copyright © 2010 Nobuhito Hironaka



http://sec.ipa.go.jp/seminar/2011/20110121n.html

品質会計を軸とした パートナー QC レビューの紹介 (PQR : Partner Quality control Review)

講演者：後藤 徳彦氏 (NEC ソフト株式会社)

人と地球にやさしい情報社会を
イノベーションで実現する
グローバルリーディングカンパニー

NECグループビジョン2017

NECソフト(株)

- 本店所在地 : 東京都江東区新木場
- 従業員数 : 4,947名(2010年3月末現在)
- 設立 : 1975年9月9日
- 資本金 : 8,668百万円
- 売上高 : 108,441百万円(2010年3月期)

■ 事業内容
官公庁や民間企業のお客さまに向けた、システム構築や保守・運用、アウトソーシングなどのITサービスの提供、およびアプリケーションソフトウェア、ミドルウェア、組み込みソフトウェアなどの各種ソフトウェアの開発を行っています。

パートナー-QCレビューとは

NECグループ重要パートナーにおける**請負開発案件での品質対応力向上を図る取組み**

◆目的

- ①パートナー会社に、**品質会計プロセスを正しく理解**してもらおう。
- ②評価結果をパートナーへフィードバックし、**改善取り組み**を促す。
- ③社内発注部門へもフィードバックし、**発注の参考情報**とする。

品質会計(SI管理会計/品質会計票)とは？

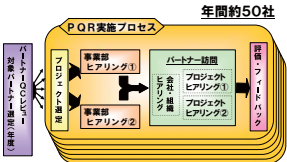
品質会計とは、高品質なソフトウェアを開発するためのNECグループ独自の品質管理手法

品質会計とは？

2011年3月
NECソフト
品質管理課
後藤 徳彦

※発注元/発注先間では、「品質会計発注制度」で品質会計の活用促進を促進

パートナー-QCレビュー 実施プロセス



パートナー-QCレビューこれまでの活動

2006年 不採算プロジェクト削減を目的に、レビュー徹底指導としてスタート
NEC連携によりOneNECの活動を展開拡大すると共に、内容も品質会計に則しパワーアップ



品質活動確認シート(サンプル)

項目	内容	確認
1. 品質会計票の提出状況	品質会計票を毎月提出しているか	○
2. プロジェクトヒアリングの実施状況	プロジェクトヒアリングを毎月実施しているか	○
3. 品質改善の取り組み	品質改善の取り組みを実施しているか	○
4. フィードバックの活用	品質フィードバックを活用しているか	○
5. 品質管理の徹底	品質管理の徹底を図っているか	○

人と地球にやさしい情報社会を
イノベーションで実現する
グローバルリーディングカンパニー

NECグループビジョン2017

書籍のご紹介「ソフトウェア品質会計」

ソフトウェア品質会計
～NECの高品質ソフトウェア開発を
変える品質保証技術～
著者 高橋孝
2010年8月22日発行
技術者向け
定価 2150円(税込)



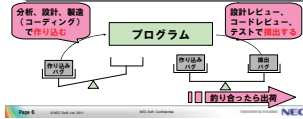
品質会計は、NEC独自のソフトウェア品質管理手法であり、NECグループで広く活用されています。
ケーススタディを含め、その使い方を具体的に解説しました。
現場・現場・現場を重視している点が、大きな特徴です。

日経品質管理文庫受賞!

品質会計の基本的な考え方

品質会計とは、高品質なソフトウェアを開発するための
NECグループ独自の品質管理手法

品質会計とは、SW製造工程 (SD) までに作り込んだバグを「負債」とし、レビューやテストにより、負債を「蓄積」(蓄出して、負債(残)バグ)となった時点で出荷するという管理手法



Page 6 品質保証課「ソフトウェア品質保証部」, 日経品質管理文庫2010

品質会計の原則

- <品質会計の原則>
バグは作り込まない。作りこんだバグはすばやく抽出する。
- <上工程品質会計の原則>
作り込んだバグは次工程までに抽出する
●作り込み工程で80%抽出
●次工程で残り20%抽出
- <テスト工程品質会計の原則>
作り込んだバグは、すべて抽出してから出荷する
- <目標>
上工程バグ抽出率 80%

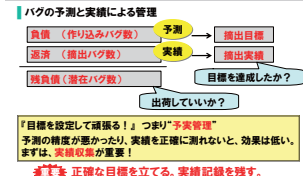
Page 7 品質保証課「ソフトウェア品質保証部」, 日経品質管理文庫2010

品質会計: Quality Accounting System とは

- 「品質」が作り込まれたことを、確かな根拠をもとて説明するソフトウェア品質管理手法
- 「account」は、元々「理由・根拠を説明する」という意味を持つ
- 品質会計を特徴づける技法
 - 上工程品質会計
 - テスト種類の順序課題の分析
 - バグ傾向分析、バグ分析と1+n監査、バグ収束判定の3つの組み合わせ
- 1982年頃、NECの開発現場で考案された、NEC独自の品質管理手法
 - NECソフトでは、2002年より、ソフトウェア開発案件に対して、全面的に品質会計(算)を導入
 - 現在は、標準帳簿「SI管理会計/品質会計算」を採用

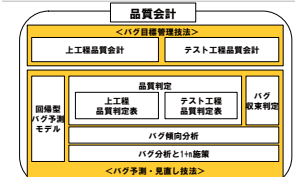
Page 8 品質保証課「ソフトウェア品質保証部」, 日経品質管理文庫2010

品質会計のツボ(1)



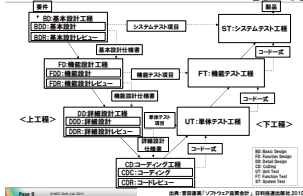
Page 9 品質保証課「ソフトウェア品質保証部」, 日経品質管理文庫2010

品質会計の技術体系



Page 10 品質保証課「ソフトウェア品質保証部」, 日経品質管理文庫2010

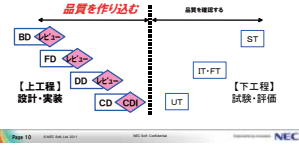
開発プロセス



Page 11 品質保証課「ソフトウェア品質保証部」, 日経品質管理文庫2010

2011/1/19/1027720288-0279942-0-0567
上工程(かみこうてい)・・・上流工程→設計/製造工程

上工程の品質は、レビュー(チェック・ゲートなど)で作り返す。この際、レビューごとに適切なチェック項目を設ける(プロセス)必要がある。

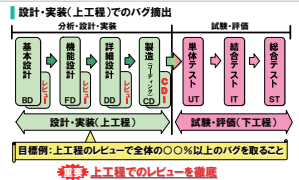


2011/1/19/1027720288-0279942-0-0567
品質会計のツボ

- 数値だけを言うのではなく、プロジェクト管理者がプロジェクトの実態に引き合う
 - バグ検出目標達成のため、数字だけあわせようとする意識が強く、本質的な改善が見えないので注意
- バグを出すことが悪いのではなく、バグを取り除くことを目的とする
 - 多くのバグ検出が記録として残ると、担当した開発グループを低評価(減点主義)で見る傾向があるが、本来の目的である「バグを取り除くこと」の意識付けが重要(質文化管理・バグフリー意識での推進)
- 工程終了の妥当性判断、次工程へ進んで問題無いかを確認する
 - 本来、記録しておかなければならないもの(レビュー記録票・バグ一覧など)を別途おく。
 - 工程の目標を達成したかどうかの判断や、工程を終了して問題無いかの判断を行う。問題があるとするは、その原因分析や対策が必要になる。

数字だけの品質管理や制度に對するだけの品質管理にならないように、本来の目的である「検証活動の証明」を認識すること

2011/1/19/1027720288-0279942-0-0567
品質会計のツボ(II)



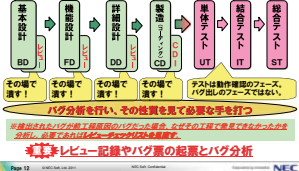
Empowered by Innovation



【参考文献】
 ソフトウェア品質会計
 ~NECの高品質ソフトウェア開発を支える品質保証技術~
 吉田 眞典著

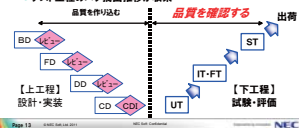
2011/1/19/1027720288-0279942-0-0567
品質会計のツボ(III)

作ったバグは後工程へ持ち越さない



2011/1/19/1027720288-0279942-0-0567
下工程(しもこうてい)・・・下流工程→テスト工程

下工程は、上工程で作り返した品質を確認する工程
 「作り込んだバグは、すべて顕出してから出荷する」ためには、以下を確認
 ● 独特バグ数がゼロ(負債がゼロ)
 ● テスト工程のバグ検出推移が収束



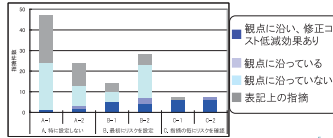
http://sec.ipa.go.jp/seminar/2011/20110603.html

ソフトウェアレビュー改善の着眼点 ～段階的な取り組みにむけた事例分析～

講演者：森崎 修司氏
(静岡大学 情報学部 / 奈良先端科学技術大学院大学)

レビュー観点数減みによる評価結果

- 指標件数: グループA > グループB > グループC
- 観点に沿った指標件数: グループC > グループB > グループA
- 観点に沿っており、修正コスト低減効果のある指標: グループC > グループB > グループA



アジェンダ

- **着眼点1 レビューの質** レビュー指摘の分類
 - 事例1 観点を設定しない場合の「evaluability」「functionality」の比率
 - 事例2 観点設定による検出欠陥の変化
- **着眼点2 レビューの効率** 観点設定(リスク設定)による効率
 - 観点設定によるレビュー効率の変化
 - 事例3「セキュリティに限定したレビュー」
- **着眼点3 開発コンテキストに応じた観点設定**
 - 開発対象の構造にもとづいた観点設定
 - 事例4「回帰テスト規模による優先順位付け」
 - 不具合修正時間にもとづいた観点設定
 - 事例5「過去のバージョンの不具合情報から観点設定」
- レビューのインフラストラクチャー
- ディスカッション

観点の有無による欠陥指摘の例

A. 観点設定なし

ID	指摘内容	観点	修正コスト低減
1
3	ページ番号がページ目だけ付いている	沿っていない	なし
4	線の番号が違う(2, 3, 4のようロケットがないものがある)	沿っていない	なし
...

B, C. 観点設定あり(観点: セキュリティ)

ID	指摘内容	観点	修正コスト低減
5	本人確認用IDが不正に使用される可能性	沿っている	あり
6	個人情報登録前にパスワード公開するか同意を得る画面が必要	沿っている	あり
...

コードレビューでの検出欠陥を分類した論文

- M. Mantyla and C. Lassenius "What types of defects are really discovered in code reviews?", IEEE Transactions on Software Engineering, Vol. 35, No. 3, p. 430-448(2009)
- 対象データ
 - 商用開発での388件の指摘
 - 8~15年保守されている100万行以上のC/C++(複数プロジェクトから)
- コードレビューでの指摘欠陥の分類
 - Functionality欠陥 (21.4%)
 - 利用者観点が機能期待と対応されていないもの(テストで発見可能)
 - Evaluability欠陥 (71.1%)
 - 保守上の問題、プログラムの構造、変数名等(テストでの発見は難しい)

アジェンダ

- **着眼点1 レビューの質** レビュー指摘の分類
 - 事例1 観点を設定しない場合の「evaluability」「functionality」の比率
 - 事例2 観点設定による検出欠陥の変化
- **着眼点2 レビューの効率** 観点設定(リスク設定)による効率
 - 観点設定によるレビュー効率の変化
 - 事例3「セキュリティに限定したレビュー」
- **着眼点3 開発コンテキストに応じた観点設定**
 - 開発対象の構造にもとづいた観点設定
 - 事例4「回帰テスト規模による優先順位付け」
 - 不具合修正時間にもとづいた観点設定
 - 事例5「過去のバージョンの不具合情報から観点設定」
- レビューのインフラストラクチャー
- ディスカッション

レビュー観点による欠陥種別の変化

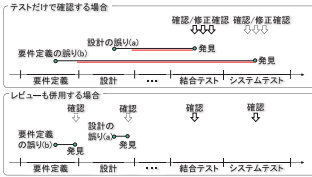
- 目的
 - レビュー観点の設定による指摘内容の変化を明らかにする。
 - (A) 観点設定なし
 - (B) 設定する
 - (C) 設定し指標の粒度確認
- 対象ドキュメント
 - Webアプリケーションの外部仕様書(実際に発注に使ったものを流用)
 - 日本語で記述
- レビューア
 - 各段階ソフトウェア開発に従事する実務者
 - 4~5名で6グループ
- 演習として実施
 - 実施時間: 40分

ソフトウェアレビュー

- プログラムが動作する前に(中間)成果物に問題がないかを目標で確認する静的解析技法
- 目的
 - 不具合の修正コストの低減
 - 状況把握と承認
 - 保守性向上
- インプット
 - 各種ドキュメント(ソースコードを含む)
- アウトプット
 - インプットのドキュメントに含まれる問題点のリスト
- 課題
 - 実施の自由度が高く、エビデンスが残りにくい。

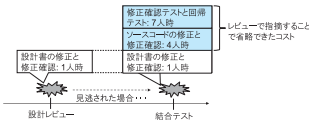
ソフトウェアレビューの効果 (ウオータフォールの場合)

- テストでの修正確認コストの低減
- 誤りに基づいた設計や実装コストの低減



ソフトウェアレビューの効果が得られる例

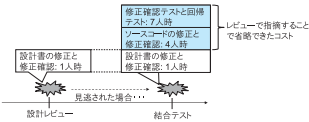
- レビューで見逃してテストで発見された場合に、修正コストが大きな欠陥を発見できると、効果が得られる。(重大欠陥)



出典: 自IT「レビュー」を「数十分」で管理するからコストが劇的に
<http://www.atmark.co.jp/irc/csr/serial/review/05/01.html>

ソフトウェアレビューの効果計測

- レビューの効果 = レビューで指摘することにより省路できたコスト - 検出コスト
- 例) 4人2時間 で実施し5件のコスト削減効果のある欠陥指摘
 - 省路できたコスト: 5 × 1人時 = 55人時
 - 実施コスト: 4 × 2人時 = 8人時 + 調整コスト
 - 55 - 8 = 47人時



ソフトウェアレビューの効果計測

- 見逃した場合の修正コストを指摘の度に見増るのはたいへんなので、深刻度で近似する。
- レビュー指摘の深刻度の定義はまちまち
 - ユーザや顧客に対するダメージ
 - 修正規模の大きさ(根本的な改修が必要、軽微な修正も可能等)
 - 影響を与えるチームの多さ(共通ラブリ、めったに実行されない等)
- 欠陥分類の例
 - 重大
 - 中程度
 - 軽微
- 効果計測をするならば、修正規模によって分類されるべき

アジェンダ

- **着眼点1 レビューの質** レビュー指摘の分類
 - 事例1 観点を設定しない場合の「evolvability」「functionality」の比率
 - 事例2 観点設定による検出欠陥の変化
- ソフトウェアレビューの効果とその計測
- **着眼点2 レビューの効率** 観点設定(リスク設定)による効率
 - 観点設定によるレビュー効率の変化
 - 事例3「セキュリティに限定したレビュー」
- **着眼点3 開発コンテキストに応じた観点設定**
 - 開発対象の構造にともなう観点設定
 - 事例4「固着テスト指標による複数観点付け」
 - 不具合修正時間にもとづいた観点設定
 - 事例5「過去のバージョンの不具合情報から観点設定」
- レビューのインフラストラクチャー
- ディスカッション

セキュリティ観点に限定したレビューの制御型実験

- 対象
 - 書籍の貸し出し、販売Web上で実施するサーバサイドJava
 - 要件定義書(A4 自然言語 20ページ)
 - 設計書
 - ソースコード(Java 4,500 LOC)
 - ユーディング支援
- レビュー観点
 - ユーザ認証の一貫性
 - セキュアなユーザ登録
 - パスワード再設定

ソフトウェアインスペクションワークショップ2009

- 主催: Working Group of International Research Cooperation on Software Inspections
- 共催
 - 奈良先端科学技術大学院大学
 - Fraunhofer IESE
 - 日本IBM
- コードレビュー実験参加を呼びかけ、100名程度が参加

観点絞り込みによるレビュー速度の比較

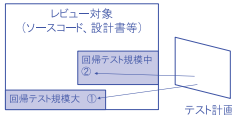
全指摘数(参加人数)	370件(93名:1名あたり93.98件)
正答率	76.1%
観点以外の指摘	91件(24.6%)
見かけ上のレビュー速度	3,860行 (Technical review, 問題発見, Java, コメント込み)

既存文献でのレビュー速度

文献名	公開年	速度(行/時)
Fagan M. "Design and code inspection to reduce errors in program development", IBM System Journal vol. 15, no. 3.	1976	700~900行(問題発見), 500~700行(問題指摘), (インスペクション COBOL_コンタ行含まず)
Introduction to the Team Software Process TSP)	1999	200行以下 (インスペクション)
IEEE 1028 Software Reviews and Audits	2008	100~200行 (Technical reviews)

帰還テスト規模による優先度つきレビュー

- テスト計画から大きなテスト規模を推測する。
- テスト規模が大きくなる部分の問題から優先して指摘する。



田村 賢一、亀井 雅典、上野 秀樹、森崎 啓司、松本 健一、「検証課題とテスト規模の成長を目的としたコーディング手法」『情報処理学会論文誌』, Vol.56, No.12, pp.3034-3043 (2009)

詳細は <http://www.jp.html/rev/returntest.html>, <http://book.gijac.jp/996/> 17

© 2011 Copyright National Institute of Advanced Industrial Science and Technology. All Rights Reserved.

修正によるテスト規模が小さい欠陥の例

```

if print specified file {
  int open(String filename) {
    (f=openPrinterDEF_PRINTER)
    print(OUTDEF_PRINTER, filename);
    else
      error("Printer not connected");
  }
}

if check if printer is online {
  bool exist(String name) {
    Device dev = null;
    dev = getDevice("Printer", name);
    if(dev == null)
      return true;
    else
      return true;
  }
}
    
```

T1: プリンタが接続されているとき、指定したファイルの内容が印刷されるが確認する。
T2: プリンタが接続されていないとき、プリンタが接続されていないことをエラーメッセージで報告されることを確認する。

詳細は <http://www.jp.html/rev/returntest.html> 17

ファイルを読む、保存するソースコードの例

```

// Opening specified file
int open(String filename) {
  f=open(filename);
  read(filename);
  else
    error("No file exist!");
}

// Saving new file
int save(String filename, Data data) {
  f=open(filename, "a");
  write(filename, data);
  else
    error("Specified file already exists.");
}

// return true if file exists
bool exist(String filename) {
  File file = null;
  file = fileOpen(filename, "read");
  return true;
  else
    return true;
}

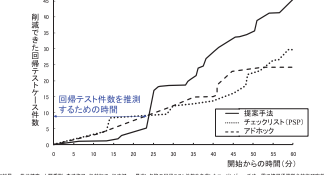
// open existing file
void read(String filename) {
  file = fileOpen(filename, "read");
  read(file);
}

// save new file (only new not existing)
void write(String filename, Data data) {
  file = fileOpen(filename, "write");
  write(file, data);
}
    
```

詳細は <http://www.jp.html/rev/returntest.html> 17

実験結果

- Webベースのカンファレンス申込みシステム(C言語, 1,500行 CGIプログラム)
- 6名の実務経験者と12名の情報系修士、博士課程の学生



田村 賢一、亀井 雅典、上野 秀樹、森崎 啓司、松本 健一、「検証課題とテスト規模の成長を目的としたコーディング手法」『情報処理学会論文誌』, Vol.56, No.12, pp.3034-3043 (2009)

© 2011 Copyright National Institute of Advanced Industrial Science and Technology. All Rights Reserved.

修正によるテスト規模が大きい欠陥の例

```

// Opening specified file
int open(String filename) {
  f=open(filename);
  read(filename);
  else
    error("No file exist!");
}

// Saving new file
int save(String filename, Data data) {
  f=open(filename, "a");
  write(filename, data);
  else
    error("Specified file already exists.");
}

// return true if file exists
bool exist(String filename) {
  File file = null;
  file = fileOpen(filename, "read");
  return true;
  else
    return true;
}

// open existing file
void read(String filename) {
  file = fileOpen(filename, "read");
  read(file);
}

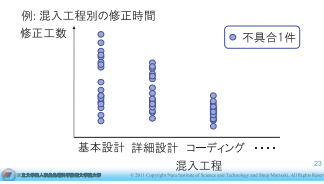
// save new file (only new not existing)
void write(String filename, Data data) {
  file = fileOpen(filename, "write");
  write(file, data);
}
    
```

T1: 既読のファイルを読み出し、ファイルオープンで新しいファイルを開いてファイルオープンエラーメッセージが来ることを確認
T2: 書き込みするファイルを開いて書き出し、既読ファイルの書き込みでエラーメッセージが来ることを確認
T3: 書き込みしないファイルを開いて書き出し、閉じたファイルが読み出されたことを確認

詳細は <http://www.jp.html/rev/returntest.html> 17

過去の不具合修正時間からのレビュー観点設定

- 単一要因による分析
不具合の修正工数をプロットし、現状把握や効率化の施策を検討している。



例: 混入工程別の修正時間

修正工数

ファイルの内容を印刷するソースコードの例

```

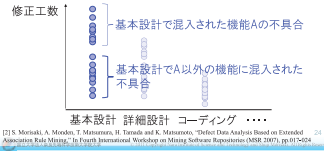
if print specified file {
  int print(filename) {
    (f=openPrinterDEF_PRINTER)
    print(OUTDEF_PRINTER, filename);
    else
      error("Printer not connected");
  }
}

if check if printer is online {
  bool exist(String name) {
    Device dev = null;
    dev = getDevice("Printer", name);
    if(dev == null)
      return true;
    else
      return true;
  }
}
    
```

詳細は <http://www.jp.html/rev/returntest.html> 17

過去の不具合修正時間からのレビュー観点設定

- 条件の複合による分類ができれば、さらに詳細な現状把握や対策を検討できる場合がある。
- 「(混入工程 = 基本設計)かつ(機能 = A) → 修正時間(平均, 16.5時間)」のようなルール形式で不具合修正時間の傾向を分析している。[2]



基本設計 詳細設計 コーディング ...

[2] S. Morita, A. Mizota, T. Matsuda, H. Tamaki and S. Yamamoto, "Defect Data Analysis Based on Extended Association Rule Mining," In Fourth International Workshop on Mining Software Repositories (MSR 2007), pp.017-024.

対象プロジェクトと観点設定のための不具合情報

- プロジェクト
 - 東芝デジタルメディアエンジニアングの開発プロジェクト
 - デジタルメディア処理ソフトウェア開発 (デジタルテレビの機能を模したメディア処理等を行う、組み込み用ソフトウェア開発)
 - C/C++で650KLOC(流用込み)、約3.5年
 - 派生開発
- 不具合
 - 発見フェーズ 単体～総合試験
 - 分析対象不具合件数 約150件
 - バグ以外の要求事項等は含まれない。

25

対象とした不具合管理票の項目

項目名	値
修正期間	修正に必要となった期間(手動で入力)
発生時間	不具合の原因箇所特定までの時間
担当者	担当者名(18名)
サブシステムカテゴリ	5種類(ハードウェア、アプリケーション等)
サブシステム名	53種類(ハードウェア、機能名等)
原因種別	設計ミス、設計漏れ、調査漏れ、調査ミス、外部設計漏れ、外部設計ミス、内部設計漏れ、内部設計ミス、コーディング漏れ、コーディングミス
修正期間	解決日と発生日の差(休日含む)
検査種別	正常系、異常系、境界系、その他
発生種別	新規バグ、内在バグ、既知バグ、その他

26

注目した要因(ルール)

- 出現頻度、修正期間の平均、標準偏差が大きいものに着目

ルール	出現頻度	修正期間平均 (対全体平均)	修正期間標準偏差 (対全体標準偏差)
(サブシステムカテゴリ=MW(P)かつ(原因種別=内部設計ミス)	3.8%	6.62 (1.26)	1.26 (0.73)
(担当者=担当-015)かつ(検査種別=異常系)	3.5%	6.33 (1.32)	1.50 (0.73)
(原因種別=内部設計漏れ)	10.2%	3.68 (1.18)	2.09 (0.62)
(サブシステムカテゴリ=MW(P)かつ(検査種別=異常系)	4.4%	5.67 (1.18)	2.29 (1.02)
(サブシステム名=FW-00 MW(P)-001)	14.0%	5.62 (1.17)	2.11 (1.04)

27

注目した要因からレビュー観点の設定

- 注目したルールに当てはまるバグの現象説明欄等からレビューの観点として設定できる項目を導出

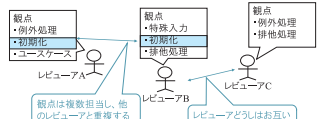
↓
設計レビューの観点として適当なものを選択(表中色塗りのセル)

分類	件数
例外処理	20
パラメータ初期化/クリア	8
特殊入力/データ/パラメータ	8
排他処理	5
内部シーケンス	5
ユーザケース	4
アルゴリズム	3
パフォーマンスオーバーフロー	2
データ領域管理	2
状態遷移	2
条件分岐	2
IF(戻り値)	2
操作ミス	1

28

試行: レビューアへの観点割当て

- レビューア毎に異なる観点を割り当てる。観点は毎回同じ人にも同じものを割り当てる。
 - 1人2~3観点(欠員の考慮)
 - 観点の割り当てはランダム
- 他のレビューアの観点は伝えない。観点の設定の有無による違いを見るため。



29

試行: レビューアへの指示

- 割り当てられた「観点」を中心にレビュー対象をチェックする。
- 割り当てられた観点以外の指摘も可
- 他のレビューアの指摘時に「同様の指摘をしようと思った」場合、その旨を報告する。
- 観点が適用されなくても、発見できた指摘が否かを判断するため



30

試行概要

- 試行1
 - 設計書作成者が内容を説明しながら、レビューアが気づいた点を指摘
 - 事前準備はなく、会議での読み合わせのみ
 - 対象
 - 外部設計書(サブシステムA)
 - 内部設計書(サブシステムB)
 - レビューア
 - プロジェクトメンバー4名 + 有識者1名
 - 外部設計書、内部設計書と同一メンバー
- 試行2
 - レビューア: 3名(試行と同一メンバー、特に対象に関わりが深いメンバー)
 - 対象: 外部設計書(サブシステムA)
 - 指摘会議のみ

31

試行結果1

- レビューア対象
 - 外部設計書(サブシステムA)
 - 内部設計書(サブシステムB)
- レビュー観点設定による効果は見られなかった。
 - 各レビューアに設定した観点とは関係性の指摘がほとんど
 - 仕様に関する一部のレビューアからの指摘が中心

レビュー対象	外部設計書 (サブシステムA)	内部設計書 (サブシステムB)
レビューア人数	6人時	7.5人時
指摘件数	11件	6件
欠陥レベル中～高の指摘	6件	4件
観点を適用したと思われる指摘	1件	1件
予測修正工数	7.5日	9.8日

32

試行結果2 - 外部設計書(サブシステムA)

- レビュー方法
 - レビュー対象を事前配布し、各レビューアがチェック
 - レビュー時間は試行1の外部設計書と同様
- レビュー結果
 - 観点からの指摘が多く見られた。
 - 欠陥レベル中以上の指摘率が高い。
 - ⇒ 欠陥レベルの高い問題が早く見つかると、観点適用の効果も高い。

	試行結果1	試行結果2
レビュー工数	6人時	3人時
人数	5名	3名
全指摘件数	11件	9件
欠陥レベル中～高の指摘	6件	8件
観点を適用したと思われる指摘	1件	6件
予測修正工数	7人日	16人日

© 2011 Copyright (Japan Institute of Science and Technology and Sharp Network). All Rights Reserved.

リソースの問題

- 問題
 - レビュー対象の分量が多すぎる。
 - 進め方がわかる人材、専門・業務知識を持つ人材が少ない。
 - いつも「今更言われても修正期間がない」という状況にある。
- 解決にむけて
 - レビュー対象を減らす、観点を限定する、自動化する。
 - 人材確保を目指すつつ、委任者の担当分量を減らす。
 - 工を終了段階ではなく、完成している部分だけを対象に早めに1回やってみる。
 - 複数の項目に共通の部分(例外の書き方等)の書き直しを減らすことができる。致命的な部分が発見できる可能性がある。

© 2011 Copyright (Japan Institute of Science and Technology and Sharp Network). All Rights Reserved.

アジェンダ

- 着眼点1 レビューの質 レビュー指摘の分類
 - 事例1 観点を設定しない場合の「evaluability」「functionality」の比率
 - 事例2 観点設定による検出欠陥の変化
- ソフトウェアレビューの効果とその計測
- 着眼点2 レビューの効率 観点設定(リスク設定)による効率
 - 観点設定によるレビュー効率の変化
 - 事例3「セキュリティ」に限定したレビュー
- 着眼点3 開発コンテキストに応じた観点設定
 - 開発対象の構造にもとづいた観点設定
 - 事例4「回帰テスト環境による優先順位付け」
 - 不具合修正時間にもとづいた観点設定
 - 事例5「過去のバージョンの不具合情報から観点設定」
- レビューのインフラストラクチャー
 - ディスカッション

© 2011 Copyright (Japan Institute of Science and Technology and Sharp Network). All Rights Reserved.

コミュニケーションの問題

- 問題
 - 日ごろの恨みを晴らすための攻撃手段になっている。
 - 「お互いに余計なことば言わない」文化がレビューにも適用されている。
- 解決にむけて(解決が難しいが。。。)
 - 恨みを晴らす場にしても残業が減ったり、リスクにさらされる頻度が下がらなければいけないことを理解してもらう
 - レビューの信頼度を守った指摘が関係者全員にとって利益をもたらすことを確認する。(まずは特定観点のみに限定したり、これまでのもの1つだけ観点を加える)

© 2011 Copyright (Japan Institute of Science and Technology and Sharp Network). All Rights Reserved.

事前合意、意識合わせの不備

- 問題
 - 一律、何が指摘されるのかわからず、びくびくしてしまう。
 - 仕様変更の場である。
 - レビューは儀式という共通認識
- 解決にむけて
 - レビューで指摘すべき欠陥を事前合意する。
 - プロセスを定義し関係者で合意する。(複数回レビューの機会を設けるのも1つの手)
 - レビューの効果を再度認識する。(できればデータ計測で)

© 2011 Copyright (Japan Institute of Science and Technology and Sharp Network). All Rights Reserved.

ソフトウェアレビュー改善の着眼点 ～段階的な取り組みに向けた事例分析～ (その2)

講演者：森崎 修司氏
(静岡大学 情報学部 / 奈良先端科学技術大学院大学)

レビューの終了基準

- ・ 顧客を含める。
 - 含めて終了を合意する。
- ・ 顧客を含めない。
 - 品質保証部門が判断する。
 - ・ 品質保証部門が開発者の感触をさぐる。
 - ・ 品質保証部門が開発者の感触をさぐらない。
 - 品質保証部門が判断しないとするならば・・・
 - ・ 数点がせまくなってしまう可能性がある。
 - ・ 納期がさし進められたときに品質保証部門の手を廻りれない。

ディスカッション

- ・ 効率化、コスト対効果
 - 効率化、コスト対効果が向上できないのもっとも大きな原因は何でしょうか？
 - それは他の参加者の方々にもあてはまると思われますか？
 - 原因を軽減したりなくしたりするためには何ができそうでしょうか？
- ・ 効果測定
 - 指摘件数や指摘密度の計測だけではうまくいかない原因は何でしょうか？
 - (たぶん) 打ち合わせ段階的にはこれを測るとうまくいくものがあるのでしょうか？
- ・ 技法・観点
 - (技法や観点に唯一の正解はないと私は考えているのですが、) もしプロジェクツとソフトウェアに合わせて選択するならば、どのように選択すればよいでしょうか？

レビューの終了基準は？

- どこまでやったらレビューと考えるのか？
- レビューで指摘すべき文量が定まり切れたというのはどう判断するか？
 - いい方法はない。(レビュー指摘がないレビューが判断すれば)
 - (人事評価に影響が出るので、影響が大きいモノから検出している)
- 効率が悪くなるので、最終検査ではないテストとレビューのバランスをとっている。検査でみつかた場合には、原因分析をする。終了基準も原因分析の中で検討される。
 - 「レビューで指摘されることはよくないことだ」という文化があったが、早めに検出しようという文化にかわりつつある。プライマリコントラクトをまじえて終了基準が妥当か判断する。
 - 終了基準は決められない。KSあたりの指摘密度を決めている。同一開発物のテストからフォード/バックしている
 - 終了基準を決めることは難しい。テストで見逃し欠陥が見つかる。品質保証部が第三者として介入している。
 - 第三者としてみるので、自身には踏み込まない。「どうでしたか？」と開発メンバーに聞く。

今後にむけて

- ・ 指摘密度だけで判断できるのか？
- ・ 観点をどうやってみつめていくか？ → バグ票からみついている
- ・ レビューア育成

http://sec.ipa.go.jp/seminar/2011/20111202.html

品質保証活動を形骸化させないコツ ～プロセス改善に魂を込めるには～

講演者：小池 利和氏（ヤマハ株式会社）

自己紹介(略歴)

- ～1997年
入社以来、**スポーツ**事業部でテニスシューズ開発
- 1997年～1998年
スポーツ事業撤退により、**品質保証部**に拾われる
製品試験や**品質管理教育**を担当
- 1998年～2008年
ここから、SEPG人生のスタート、**全社スタッフ**として、
プロセス改善の**全社普及啓蒙**や**個別事業部支援**など
- 2008年～
電子素子ソフト開発部門に移り、**現場のSEPG&SQA**
この頃から**社外活動にも活発**に取り組みよう

開発

全社スタッフ

現場スタッフ

**全社スタッフ時代は支援先部門から費用を撥替
開発部門はお客様という意識が自然と芽生えていた**

自己紹介(社外活動)

SQIP関連

- SQIPステアリング委員、普及推進小委員会委員長
- SQIP研究会副委員長
- 同研究会特別コース「ソフトウェア品質保証の基礎」主宰
- SQIPワークショップWRリーダー

TEF関連

- JaSST東海実行委員
- メトリクス勉強会 (<https://sites.google.com/site/teftokaimetrics/>) 主催

発表など

- Web記事「即活用！レビューの賢者チェック風」
※内容はSQIP研究会の論文(SQIPシンポジウム2008 Future Award受賞)
「レビューの質と価値の定量化の提案」
<http://think.co.jp/article/055/1>
- 日経エレクトロニクス
「ソースコード納め検証によるソフトウェア品質評価の意義」
<http://tefton.kohkeibooks.com/article/KJN0904/20110320/17846/>
- 個人ブログ「実践！ソフトウェアメトリクス」※休載中・・・
<http://metrics.seesaa.net/> Twitterアカウント koike0125

本日の進め方

- 前半は私からプレゼン資料をベースに、随所で実際に現場で使用している帳票等をご紹介しながら、話を進めます。
(申し訳ありませんが、帳票等は配布資料に含めています)
- 資料は予め頂いている質問に答える形で構成します。
私の経験の中での回答と、それではあまり参考にならないようなものは一般的な情報も提供したいと思います。
- 後半のディスカッションでは、上記の回答が不足している部分などの質問でも結構ですし、「うちでは、こうしている！」みたいな事例紹介をして頂けるとうれしいです。

アジェンダ

・SQA、SEPGの体制、教育に関して

- そもそもSQAとSEPGの違いとは？
- 組織における人数構成
- SQA、SEPGの教育、普及啓蒙

・メトリクス活用に関して

- プロセス改善効果の見える化
- プロジェクト実行中に有効なメトリクス
- ソースコードのメトリクス
- SQA活動のメトリクス

・形骸化させないSQA&SEPG活動の実態

- 形骸化の典型例
- 魂を込めたSQA&SEPG活動の実態事例紹介

SQA、SEPGの 体制、教育に関して

SQAとSEPGの違い



「ソフトウェアプロセス成熟度の改善」
CMM、CMMIのベースとなるこちらの書籍がCMMの
心を理解する上で、非常に参考になります。SEPG、
SQA必読の書と考えています。

SQAの役割(要約)

- ソフトウェアとその生産プロセスの監視
- 標準と手順に沿っていることの保証
- 生産物、プロセス、標準の不備を管理者に提言

SEPGの役割(要約)

- プロセス標準とDBの確立
- 技術導入の支援、教育、コンサルテーション
- アセスメントの実施

SQAとSEPGの連携が重要！

SQA、SEPG人数構成

- 弊社の状況を口頭で説明します(資料提示出来ません)
- 某H社の話
- 某T社の話
- 先ほどの書籍によると
 - SEPGは開発者に対して**典型的には1～3%、2%が目安**
 - SQAに関しては特に記述無し
- 以前に何かの文献で読んだり、話を聞いた記憶では、スタッフとして**5%いると良いらしい**(多分SQAとSEPG合わせて)

SQA、SEPGの教育、普及啓蒙

- ・私が全社スタッフ時代に行っていた教育や普及啓蒙活動を実際の資料をお見せしながら、ご紹介します。
- ・PG連絡会
 - ・全社のSEPGのコミュニティ活動
 - ・定期的な情報交換会（以前は月1、最近はQ1）とML
 - ・近況報告と事例紹介で、トータル2H
- ・SPワークショップ
 - ・上記メンバー向け勉強会
 - ・隔週、半日×7～8回
 - ・テーマはCMM、PM、メトリクス
- ・プロセス改善講演会
 - ・年に1回の講演会、基本的に全社のソフト開発者全員参加
 - ・仕掛けるテーマに沿った講演と社内事例発表の2本立て

ソースコードのメトリクス

- ・私が齋崎先生（早稲田）、オージス総研さんと共同研究にて開発した「ソースコード品質診断ツールAdqua」について紹介
- ・ただし、本題では無いので、最後に時間が余ったらしましますきちんと説明するとこれだけで終わってしまうので、本当に触れる程度
- ・オージス総研 Adquaの紹介ページ
<http://www.ojsr.co.jp/product/?p=08-000001A6.html>

メトリクス活用に関して

SQA、SEPG活動のメトリクス

- ・実は、あまりきちんとは取れていないのですが、参考程度に私のSQA活動とSEPG活動の工数比率などをご紹介します。
- ・これも後半の事例紹介の後に時間が有れば紹介します。

収集メトリクスと算出指標

- ・基本メトリクスとして収集しているもの
 - ・工数、行数、バグ件数（テスト工程のみ）、マイルストーン
 - ・上記の計画値と実績値
 - ・その他、リリース実績、重要品質問題件数（クレーム等）
- ・QCDの指標として定期的に見ているもの
 - ・Qの指標：重要品質問題件数、バグ密度、バグ指摘率
 - ・Cの指標：生産性、工数予実差
 - ・Dの指標：納期遅れ日数
- ・その他の指標、メトリクス
 - ・遅れ期間工数比、増減前後工数比、手戻り工数比
 - ・アクティビティ別工数比、レビュー工数比、
 - ・コーディング効率
 - ・行数カウント分類（新規、修正、未修正、再利用）

形骸化の典型例

メトリクスの活用例

- ・改善効果の見える化
 具体的にどのような形で改善効果を示しているか、実例をご紹介します（資料配布出来ず）
- ・開発計画書作成時の見積参考値、標準値
 活用例は、後半の事例の中でまとめて紹介します。
- ・プロセス監査で活用
 実はプロジェクト実施中に収集出来ているデータは、工数とバグ件数だけです。こちらの活用例も後半に。
- ・組織のコストデータの元
 工数を開発コスト算出のための元データ、税務申告用データとして、活用しています（今回テーマに関連薄いので紹介は割愛）

今まで見聞きした「アレ？」と思うこと

- ・冒頭の自己紹介の通り、かれこれ十数年SQA&SEPGをやって参りました。
- ・当初、品質保証部でISO推進活動（これもSQA活動の一種）を傍目で見ていたり、全社SEPGとして色々な事業部を支援したり、近年では社外の方とも色々とお話をする機会に恵まれています。
- ・その中で、私が「アレ？何か違うんじゃない？」と感じたことについていくつか例を挙げます。
- ・実は形骸化の根本原因は1つではないかと考えています。そうならないための考え方、行動について実践例を元に紹介をしていきたいと思います。

<http://sec.ipa.go.jp/seminar/2011/20111202.html>

「～に書いてあるから」

- こちらの業務内容を全く理解していない人から「～に書いてありますけど、実施していないようですね。不適合です。」とか言われても腹が立つだけですよね。
- 私は、プロセスは「ルール」ではなく「道具」だと考えています。なので、うまく使っているか/いないか、という話は有っても、従っていないからNGというのは、あまりに短絡的過ぎます。
- 書いてあるから実施するのではないのです！**それが、プロジェクトの成功(失敗の予防)に役に立つから実施するのです。考えてみれば当たり前のことですが、**常日頃からSQA、SEPGはその姿勢を貫き通す**情報が重要です。

**プロセスはルールではなく道具
必要なければ使わないし、使うならばうまく使うべき**

「開発側が～してくれなくて・・・」

- SQA、SEPGの方から出る典型的な愚痴の一つに以下のようなものがあります。
 - A.「データ、資料を提出してくれないですよ・・・」
 - B.「手順書や帳票を使ってくれないですよ・・・」
- Aに関して、データを収集したり、資料を整理して提出するなどという作業は**プロジェクトに直接的に貢献をもちません**。このようなことで開発側に負担を掛けてはいけません。
- Bに関して、これはお店屋さん为例れば「うちの商品誰も買ってくれないんですよ」という愚痴と同じです。これをお客さんにせいでいしますか? そんなお店はすぐ1に潰れます。**買ってもらう努力をしなければ!**

スタッフはサービス精神が命、帳票愚けやどんどん直す

「データが悪いと反省させられてしまう」

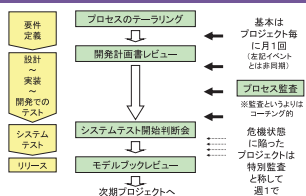
- (後述する)プロジェクトのメトリクスをまとめたレポートを作成し、それを材料にしながら反省会をするというお話を紹介した時「えー、データが悪いと**反省させられてしまう**ですよな。それじゃあ、正直なデータなんて集まる訳無いですよ。」
 - 今でも忘れられない衝撃的な一言でした。
 - 組織内で日頃から適切なマネジメントがされていないのだな— と思い、更にお話を伺うと、以下の根深い問題が
 - プロジェクトが厳しい状況でも**上位マネージャの支援無し**
 - 改善しようにも全く余裕が無いし、してもさほど評価されない
 - ので、**余計なことはせず与えられた仕事をこなすのみ**
- 適切なマネジメントの上で始めてSQAが活躍する
目標はQOQ達成だけではなく、チャレンジも盛り込む**

全てはプロジェクトの成功のために

- 3つのエピソードは、「プロセス」「スタッフ」「マネジメント」の3要素の在り方の問題です。
- 私は、これらは全てプロジェクトを**成功させるためのサポート的なもの**だと考えています。
- なので、プロジェクトの**成功に寄与しないものは不要**ですし、ましてや足を引っ張るようなことは断じてするべきではないと思っています。
- 形骸化が進む一番の理由は、開発者がその活動がプロジェクトの成功に寄与しないと考えているから**です。
- 我々、スタッフは常にプロジェクトの成功に**貢献出来るのかを自問自答**しながら活動していく必要が有ると 생각합니다。
- ただし、単一プロジェクトの成功ではなく**組織の成長に寄与する活動**も有ります。でも、それは上記が出来ている前提の話です。

形骸化させない SQA & SEPG活動 の実際

SQA～SEPG活動の流れ



このように定期的にプロジェクトと接しながら、問題を指摘するばかりではなく、プロセスの相関に乗ったり、改善点を収集する(これがSEPG活動へ繋がる)

紹介するアイテム

- 時間の許す範囲で以下の視物を前頁の流れの順で紹介をしていきたいと思えます
 - テラリング
 - プロセス監査記録
 - 工数推移レポート
 - ソフト開発計画書
 - システムテスト開始判断会
 - 特別監査レポート
 - モデルブック

プロセス改善に魂を込めるには

- あくまでも自論ですが、私が心掛けていること
 - プロセスは「ルール」では無いとは言っても、最初は何とかが**使ってもらう努力は不可欠**(そのために色んな手を使います)
 - 良いプロセスは、そのうちに放っておいても使われます。**淘汰されたものは要く捨てる**(そうしないと増える一方に)**淘汰したものはマネージャで示す**(メンテナンスされてこそ良い道具に)
 - 新しいプロセスを試しても導入しない、まずは**パイロットで実施し**、感触を確かめながら、合意できたら徐々に。
 - 常にプロジェクトの成功を考えて発言、行動します。(～にこう書いてあるから、は厳禁)時にはマネージャにも働きかけます。
- でも、プロジェクトとは友達では有りません。**客観的な姿勢**で扱います。(なので、メトリクスを活用)

プロセス改善に関するセミナーの概要をご紹介するガイドをまとめるにあたり、本ワーキンググループが重要視してきた「参加者が主体となるセミナー」であることをより強く感じていただくため、以下のような内容を盛り込みました。

- ・事前 / 事後の「メーリングリストによる議論」の紹介
- ・「参加者からの声」の紹介

毎回セミナーを開催する際には、より多くの参加者のみなさまに「明日の業務へのヒント」を得ていただけるよう心がけています。そのための工夫として、事前に参加者のみなさまからポジションペーパーを提出いただき、日々どのような問題を抱えており、セミナーへどのような期待をしているのかを、講師に伝えています。

加えて、講師を含めた参加者全員によるメーリングリストを作成し、事前にセミナーのテーマについて討議をすることで、セミナーの内容をより濃いものとし、参加者のみなさまの業務に繋がる内容にできるようつとめています。

おわりに

本ガイドをお読みいただき、本ワーキンググループが主催するセミナーが他とは一味違うものだということが、よくわかっていただけたのではないかと思います。文献や聞くだけの講義ではなく、多くの議論を出し合って「知識を結果に繋げる」ことこそ、我々が目指すセミナーの意義であると考えています。

これまで数多くのセミナーを開催してまいりましたが、まだ改善の現場における多くの課題は解決されないままとなっています。いえ、“そのまま”なのではなく、社会や企業の変化につれて、新しい課題が日々生まれているというのが正しいのでしょうか。ただ過去の実績を収集・分析した知識だけでは、現場の「今」を改善することはできません。企業を越えた現場の「今」を共有し、新たな課題に向かっていく「知恵」を得ることが、現場の改善を着実に進めていく1つの方法ではないでしょうか。

セミナーという「場」を通して、多くの方が得てきた「知恵」の軌跡を、このガイドからご紹介することで、またこれから各所で展開される改善の現場に、ひとつのヒントをもたらすことができればと、一同願っております。

編著者 (敬称略)

プロセス改善 WG NPT4

リーダー	和田 典子	ソニー株式会社
	穴田 直也	株式会社大和コンピューター
	臼杵 誠	富士通株式会社
	小川 清	名古屋市工業研究所
	串田 幸江	株式会社アジルコア
	河野 文昭	株式会社アドヴィックス
	込山 俊博	日本電気株式会社
	砂塚 利彦	砂塚コンサルティングサービス株式会社
	田中 一夫	アイエックス・ナレッジ株式会社
	徳永 享	富士ゼロックス株式会社
	新井本 武士	株式会社東芝
	服部 祐二	ブラザー工業株式会社

セミナー講師

小笠原 秀人	株式会社東芝
中野 隆司	株式会社東芝
服部 彰宏	富士ゼロックス株式会社
山本 訓稔	富士ゼロックス株式会社
平田 貞代	富士通株式会社
古畑 慶次	株式会社デンソー技研センター
伊藤 昌夫	株式会社ニルソフトウェア
永田 敦	ソニー株式会社
細川 宣啓	日本アイ・ビー・エム株式会社
後藤 徳彦	NEC ソフト株式会社
森崎 修司	静岡大学 / 奈良先端科学技術大学院大学
小池 利和	ヤマハ株式会社

IPA/SEC 事務局

新谷 勝利
倉持 俊之
室谷 隆
井上 麻希

プロセス改善 WG メンバ

主査	菊島 靖弘	東京海上日動システムズ株式会社 / 株式会社アイネス
	赤坂 幸彦	エヌ・ティ・ティ・データ・システム技術株式会社
	安達 賢二	株式会社 HBA
	足立 久美	株式会社デンソー
	穴田 直也	株式会社大和コンピューター
	安倍 秀二	パナソニックエレクトロニックデバイス株式会社
	飯田 元	国立大学法人奈良先端科学技術大学院大学
	板橋 吉徳	パナソニック株式会社
	臼杵 誠	富士通株式会社
	江崎 美保	株式会社日新システムズ
	岡崎 靖子	日本アイ・ビー・エム株式会社
	尾形 俊彦	みずほ情報総研株式会社
	小川 清	名古屋市工業研究所
	片平 真史	独立行政法人宇宙航空研究開発機構
	北島 義弘	株式会社 PM Academy
	串田 幸江	株式会社アジルコア
	梶野 友靖	楽天株式会社
	小泉 浩	マイクロソフト株式会社
	河野 文昭	株式会社アドヴィックス
	込山 俊博	日本電気株式会社
	近藤 聖久	三菱電機株式会社
	斎藤 吉正	新日鉄ソリューションズ株式会社
	阪本 太志	東芝デジタルメディアエンジニアリング株式会社
	砂塚 利彦	砂塚コンサルティングサービス株式会社
	田中 一夫	アイエックス・ナレッジ株式会社
	谷川 浩	トヨタ自動車株式会社
	徳永 享	富士ゼロックス株式会社
	新井本 武士	株式会社東芝
	丹羽 武志	株式会社インテック
	野村 典文	伊藤忠テクノソリューションズ株式会社
	服部 祐二	ブラザー工業株式会社
	伏見 諭	合同会社ソフデラ
	松原 友夫	松原コンサルティング
	宮本 祐子	独立行政法人宇宙航空研究開発機構
	和田 典子	ソニー株式会社

IPA 独立行政法人情報処理推進機構
技術本部 ソフトウェア・エンジニアリング・センター

〒 113-6591 東京都文京区本駒込 2-28-8
文京グリーンコートセンターオフィス
TEL : 03-5978-7543 FAX : 03-5978-7517
<http://sec.ipa.go.jp>

リサイクル適性(A)

この印刷物は、印刷用の紙へ
リサイクルできます



R100
各社ハルフ配合率100%再生紙を使用