第７分科会（B グループ）

単体テストにおける CMM 的アプローチ

A CMM-like approach for improvement of the unit testing process

主　査	**保田 勝通**	**つくば国際大学**
副主査	**西 康晴**	**電気通信大学**
	高橋 寿一	**ソニー株式会社**
研究員	**尾崎 智晴**	**株式会社アルゴ 21**
	前田 直毅	**株式会社インテック**
	竹谷　誠	**富士フイルムソフトウエア株式会社**
	村中 克己	**富士電機システムズ株式会社**
	渡邊 茂	**株式会社アスプコミュニケーションズ**
	三善 克彦	**株式会社セゾン情報システムズ**

（敬称略　順不同）

研究概要

テストプロセスの改善といえば、TMM（Testing Maturity Model）や TPI（Test Process Improvement）が有名であり理論的にもすぐれているが、我々分科会メンバにとっては、身の丈にあっておらず、改善活動の実践にはつながっていない。

そこで、今後、単体テストに関して改善活動を行っていくために、「単体テストのプロセス品質を向上させるためのベンチマークを作成」し、「明日から使える単体テストの改善方法の考案」することが本論文の主要なテーマである。今回示した改善案を利用し、単体テストプロセスを段階的に改善し、品質向上のための第一歩としていきたいと考えている。

Abstract

If it is called an improvement of a test process, although TMM (Testing Maturity Model) and TPI (Test Process Improvement) are theoretically excellent, for the members, it is not in height and does not connect with practice of improvement activities.

Then, in order to perform improvement activities about the unit testing from now on, it is the main themes for a main subject to create " carry out and to carry out "a design of the improvement method of the unit testing which can be used from tomorrow" of the benchmark for raising the process quality of the unit testing.

The improvement proposal shown this time is used, a unit testing process is improved gradually, and it is thought that he wants to consider as the first step for the improvement in quality.

1 はじめに

マイヤーズは、「ソフトウェア・テストの技法」でソフトウェアのライフサイクルを V 字型モデルで表現した。V 字型モデルを右下に下がる品質を作りこむベクトルと、右上に上がる品質を保証するベクトルに分けて考えると、前者の品質を作りこむベクトルに関しては比較的、開発手法やプロジェクトマネジメントで議論され尽くしている感がある一方、後者の品質を保証するベクトル＝テストプロセスについては、十分な手法が確立されていない。特に、現場におけるテストプロセス、テストの管理、テスト完了基準はメンバの経験に依存しており、曖昧にしか実施されていないように思える。

信頼度成長曲線やカバレッジ計測など、ソフトウェア工学の理論は整備されているように見えるが、現実には、上流工程の失敗によるしわ寄せ、プロジェクトマネジメントの失敗のしわ寄せを全てこれらのテスト工程が吸収しており、納期がテスト工程の完了を決めているのではないかという想いがあった。本論文では、こうした現状から抜け出すための方法論について、テストプロセスを中心に探っていく。

1.1 テーマ選定理由

我々はテスト工程を、モジュールの動作を確認する「単体テスト」、機能単位で動作を確認する「結合テスト」、システム全体の動作を確認する「システム・テスト」、ユーザによる「受け入れ／運用テスト」と分けることにした。その上で、まずは、全員が共通して実施しており、各社の実施内容に大きな差がない「単体テスト」を対象に、ソフトウェアプロセスの改善案を考えることとした。

テーマを単体テストにした理由は２つある。一つは、単体テストがテスト工程の中で最初に出てくるので、後ろの工程に行く前に改善にとりくむことができ、後工程から改善するよりも取り組みやすいこと。もう一つは、単体テストが他のテスト工程と比べても属人的で可視化できていないことから、このプロセス改善が最も効果的で必然性があることによる。

1.2 本論文における到達点

テストプロセスの改善としては、既に理論的にも優れた TMM や TPI が存在するが、我々分科会メンバにとっては、身の丈にあっておらず、改善活動の実践にはつながってゆきにくいと考えた。実際問題として、プロセス改善をどこから着手し、どうやってゆけば息切れせずに推進力を維持してゆけるだろうか。

そこで、今後、単体テストに関して改善活動を行っていくために、「単体テストのプロセス品質を向上させるためにベストプラクティスを定義」し、「明日から使える単体テストの改善方法の考案」を本論文の主要テーマとする。今回示した改善案を利用し、単体テストプロセスを段階的に改善し、品質向上のための第一歩としていきたいと考えている。

2 単体テストの課題

ソフトウェア開発において、「単体テストの重要性」は理解されていても進捗の遅れなどでつい疎かにしたり、正しく実施しているつもりが後続プロセスで問題が生じたりする事がしばしば見受けられる。では、なぜ疎かにされやすいのか、また実施しても品質向上に繋がらないのだろうか。このような疑問を各メンバがそれぞれ感じていたため、単体テストの課題として、どのようなものがあるかを洗い出すことにした。

2.1 単体テストの課題整理

まず、単体テストの現状分析のため、本分科会メンバ各社で抱えている現課題を持ち寄り、整理していくことにした。課題を整理していくうえで、「プロセス」と、「関係者の立場」の二つの観点で纏めた。

2.1.1 プロセス

課題を、プロセスの観点から「テスト計画」、「テスト設計」、「テスト実施」、「テスト管理」の４つに分類した。

表１ 「プロセス」の観点による単体テストの課題

プロセス	課題
テスト計画 （計画）	・スケジュールが無計画、行き当たりばったり ・テスト環境の作成時間が確保されていない
テスト設計 （設計）	・テストケースが属人的 ・単体テストの範囲が不明確 ・単体テスト項目の洗い出し基準が不明瞭
テスト実施 （実施）	・テスト基準が統一されていない ・流用品、既存のモジュールのテストの仕方が不明 ・成果物が存在しない ・回帰テストにテストケースを再利用できない
テスト管理 （コントロール）	・途中の進捗を正確に計測できていない ・完了時の品質を正確に評価できない ・50 項目/kstep の指標があるが根拠が曖昧 ・実施結果に対する評価未実施 ・テスト仕様レビューが体系化されていない （実施されていない）

2.1.2 関係者の立場

課題を、関係者の観点から「テスター：現場のテスト実施者」、「プロジェクトマネージャ：プロジェクト管理者」、「品質保証部：プロセス改善活動者」の３つの立場で分類した。

表２ 「関係者の立場」の観点による単体テストの課題

関係者の立場	課題
テスター	・何をテストすればよいか不明瞭 ・テスト方法が不明瞭 ・バグ報告内容が不明瞭
プロジェクトマネージャ	・バグ発生状況が把握できない ・進捗、品質、課題・問題の監視ができない
品質保証部	・品質状況の定期的な測定と分析ができていない ・品質状況の可視化ができていない

2.2 テストプロセスの必要性

課題整理の結果から、全体に満遍なく課題が分散している事、及び体系づけたプロセスと基準が明確になっていない事がわかる。このような状態であれば、単体テストを疎かにしやすく、また品質向上に繋がらないのは自明である。よって、誰がどの単体テストを実施しても、一定基準以上の結果を得られるようなプロセスや基準を明確にする事が必要となる。更に、実施結果の問題分析をフィードバックし、計画、設計、実施、管理の改善活動を常に繰り返す事も重要となる。それでは、プロセスや基準を明確にするにはどうしたら良いだろうか。

我々は今回の課題が生じた根底的な理由は、単体テストが属人性と困難な可視化という特質を持っていることを理由に、それを克服するようなプロセスや基準を明確にしてこなかったためだと考えた。そこで次の３ では単体テストのベストプラクティスを考察し、これをプロセス改善の到達点として明示する。

3 ベストプラクティス

単体テストが、それ以降の結合テスト、総合テストといったフェーズと比較して特異な点は、それが極めて属人的であり、適正なプロセスとして計画・管理されていないことである。適正な品質は適正なプロセスにおいてのみ実現されることから、この単体テストフェーズのプロセス改善こそが、品質向上に最も効果的であり即効性が高いと考えた。一般的に単体テストは個人ワークであり、チーム作業ではないことから事前の計画、記録といったことが疎かにされやすい。逆に言うと、単体テストにおけるプロセス改善は、その実施状況をプロジェクト全体として可視化する努力だと捉えることができる。

以下ではテストプロセスを、テスト計画、テスト設計、テスト実施、テスト管理の 4 つに分けてそれぞれを考察するが、そこに共通する課題は「属人性の排除」と「高い再利用性」である。

3.1 テスト計画

テスト戦略がなくても、特に単体テストにおいてはテスト実施が不可能というわけではない。しかしながら、テスト計画として明文化することは、個々人のスキル、経験、モチベーションに依存していたテストに対してチームとしてのベクトルが生じ、組織的な取り組みを可能にすることができる。テスト計画の要諦は計画としての成果物にあるのではなく、プロジェクトメンバへの Principle としての浸透度と、実施状況からのフェードバックによって調整され続けることにある。

3.1.1 参照され、活用されるテスト計画

テスト計画書としては IEEE std829-1998 Standard for Software Test Documentation がテストドキュメントの標準規格とされている。しかしながらこれはあまりに重厚長大すぎて、そのまま適用することは不可能になっている。実際の現場では多くの場合、ベンダーのプロジェクトマネージャがテスト計画を作成するが、対ユーザ向けという性格が強く、プロジェクトメンバに向けてのドキュメントという視点が欠落している。このため、プロジェクトメンバがこれを参照しながらテストを進めていることはなく、結局は個人のスキル、経験、モチベーションに依存したテストになっている。

プロジェクトメンバが参照し、活用するようなテスト計画とはどのようなものか。それは記載されている内容によるのではなく、むしろその作成過程にあるといえる。つまり、プロジェクトマネージャ、リーダ等から押し付けられた、守るべき規定としてではなく、テスト実施者がお互いの経験とノウハウをベースに討議して作り上げたガイドであるべきである。テストに対するチームとしての組織的な取り組みは、このテスト計画の作成過程での討論、協議によって生まれてくる。

3.1.2 更新され続けるテスト計画

前述のとおり一般的に、テスト計画は守るべきベンダー側が作成した対ユーザ向けのドキュメントであることから、スケジュールや完了基準といった最低限の記載がなされているのみである。また、これを更新するには所定の変更手順を経る必要があり、そもそも頻繁に変更すべきではないと考えられている。しかしながら、テスト自体が存在の知られていないバグの摘出行為であることから、テスト計画は初めに策定されたままを墨守するのではなく、進捗につれて状況に対応して更新され続けるべきである。

この実施状況からのフィードバックが保証されることによって、当初策定するテスト計画に要求される厳密さ、緻密さが緩和される。更に、こうしてプロジェクトの間中に完成度を高めたテスト計画は、次のプロジェクトでのテスト計画に再利用することができる。次のプロジェクトにおいても、期間中にテスト計画が見直され続けるならば、ことさら立ち上がり時のテスト計画に厳密さ、緻密さは必要ないからである。こうして蓄積された過去のプロジェクトのテスト計画は、個人知から組織知に転換したテンプレートである。

3.2 テスト設計

前述のとおり、単体テストにおいてはプログラム作成者とテスト実施者が同一であることから、テスト設計が疎かになりやすい。作ったものを一刻も早く動かしてみて確認をとりたいというのは、人間としてごく自然な感情である。しかしながら、全てのケースをテストするという絨毯爆撃は時間的に不可能であり、闇雲なモンキーテストではテスト項目に抜けがでる。このため、バグを狙い打つようなテスト設計がどうしても必要となる。

3.2.1 バグの生態

バグを狙い打つようなテストを実施するためには、まずは敵（＝バグ）の生態を知る必要がある。一般にバグには以下のような特質があるとされている。

・特定モジュールに偏在
・デバッグ時に、より多くのデグレードが発生
・多くは境界値に生息

各バグへの対処方法は、それぞれ「選択と集中」「リグレッションテストの徹底」「境界値テストを重視」という対策が有効であると考えている。本論文ではこうしたバグをつぶすための方法論を解説することは行わない。逆に、どのようなプロセスにすればこれらのバグを取り除けるかを考える。

3.2.2 テスト設計から始める

結合テスト以降では、比較的チーム的な取り組みがなされており、このため事前にテスト設計がなされている。しかし単体テストにおいては、プログラム作成者＝単体テスト実施者が自身でテストケースも作成し、しかもその時期はコーディング後の単体テスト直前というのが大半である。これではテスト設計として考えられたケースが設計・実装のロジックに引きずられがちになる。逆に言うと設計・実装時に埋め込まれた不良の摘出に対し、テスト自体が十分な独立性を確保することができない。テスト設計を前倒しにすることで、テストの持つ独立検証性を高め、更にはテスト設計からのフィードバックを実装時、設計時へと反映させるような構造転換、これがテスト駆動開発（TDD）もしくはテストファーストの根本思想だと考える。

テスト駆動開発（TDD）もしくはテストファーストは、テストの独立検証性を向上させるとともに、あらかじめテスト設計を行うことでテストし易い構造でプログラムを実装するという、テスト設計からコーディングへのフィードバック、もしくはプロアクティブなテストで開発することを目標としている。

3.3 テスト実施

単体テストは、チームではなく個人ベースの作業である。またそこで摘出すべきバグの原因は、テスト実施者自身に起因することからも、正確な報告があがってきにくい。このように他のテストフェーズと比べても、単体テストは進捗状況、品質状況を可視化することが難しい工程だと考えられる。一方で単体テストには、全てのテスト工程で最も再利用性が求められる。それは単体テストが初めてのテスト工程であるためで、以降の変更に対する回帰テストとして再利用される可能性が最も高いことに起因する。

3.3.1 実施状況の可視化

テスト実施中の記録は、全てのベースラインである。正確な状況の記録があって初めて、原因と対策を分析することが可能となり、コントロールが機能することができる。しかし単体テストが自身による自己チェックである限り、そこに厳密性を求めることには限界がある。これに対しては、やはりXP（eXtreme Programming）のプラクティスとしてペアプログラミングが提起されている。これには、コーディング中から２者によってレビューを並行的に進めるとい

う意味で画期的だが、導入に際しての抵抗感も強く、なかなか実践として浸透していない。
過渡的な手法としては、クロスチェックという方法も存在する。これは、ペア間でコーディングと単体テストを入れ替えるものだが、コーディング中からのチェックこそ入らないが、作成者の潜在意識、思い込みに対しては単体テスト時にチェックすることができる。また、作成者以外の眼で単体テストを実施するということは、それ以前にテスト仕様を整理することを必須としやすく、進捗状況、品質状況についても厳密な記録、報告が可能となる。

3.3.2 テストの再利用性確保

単体テストは、通常その場限りのテストケースが検討され、一過性のテストとして実施後は顧みられることが少ない。しかし、それ以降のテスト工程でのデバッグもしくは仕様変更による修正に対して、再度以前の単体テストを回帰テストとして実施する価値は非常に高いといえる。単体テスト結果を成果物、納品物として記録するだけではなく、そのテストケースを保存し、必要に応じてこれを再利用できるようにする。バグの生態として修正にともなうデグレードの発生確率が高いことを指摘したが、これを回避するためには、単体テストの再利用による再帰テストの徹底が有効である。更にその先には、xUnit のようなテスティングフレームワークによる自動テストツールの活用が、到達点としてある。

3.3.3 テスト設計へのフィードバック

テスト実施時には、主要な関心はバグに集中する。この時、バグの追跡調査を動的な状況として記録することは必須となるが、時に単体テストにおいては、更にテスト設計へのフィードバックが重要になる。それは単体テストが、それ以降の他のテスト工程と比較して対象モジュールが多く、それぞれの実施期間がばらけているからである。このため、初期に実施されたテストの教訓をそれ以降の単体テスト設計に反映することが可能になる。このようなテスト実施からテスト設計へのフィードバックループが完成することで、単体テストの品質が高まることが想定される。更に単体テスト実施時の教訓は、次のテスト工程である結合テストへのインプットとなるべきだし、そういった記録は、組織知としてナレッジマネジメントに蓄積されて次のプロジェクトに活用されるべきである。

3.4 テスト管理

テスト計画は決して完璧なものではなく、その実行にあたっては状態を把握しながら、絶えず計画の見直しが必要となる。テスト管理においては、その状態を進捗管理、バグ管理、品質水準管理の３つの側面からとらえてゆく。最も重要なことは、テストの状態をモニタリングしてそれらを把握するだけではなく、そのデータ分析を通じてどのようにテスト計画を補正し、最終的な品質に繋げてゆくかというアクションである。

3.4.1 テスト実施の計測による発動

コントロールの発動には計測が必須となる。しかし、リカバリ可能なくらいの遅延や些細なケアレスミスは、なかなか報告に上がってきにくい。このような属人性と自己検証の弊害を排除するためには、報告された計測値によって個人評価されることがないようにし、そのことについてメンバが確信していなければならない。

3.4.2 テスト設計へのフィードバック

通常、単体テストはコーディング者自身が実施している。このため単体テスト中に発見されたバグは報告にあがってきにくい。単純なスペルミスの類は、開発中のプロダクトにとっては些細な一過性の間違いであっても、将来の同様のプロダクトにおける再発防止という観点からは、これを記録し参照可能なナレッジとして蓄積しておくべきである。このような目的をプロジェクトメンバに周知し、「失敗」の可視化に取り組む必要がある。
バグ管理の目的は、前述のような将来のプロダクトだけではない。バグは偏在するものであ

り、その傾向をデータから分析することで、元から絶つという効果的な対策を実施することが可能となる。特に単体テストはコーディング完のものから逐次実施されることから、実績に応じてリスクの高い箇所を狙い打つようにテスト設計を調整することは、それ以降に予定されている単体テストにとって、極めて効果的である。

3.4.3 テスト計画へのフィードバック

一旦テストが始まると、テスト計画書は顧みられなくなく。これはテスト計画が更新されないためである。テスト管理で収集されたデータは、次のプロジェクトへの教訓として残されるのは勿論だが、当該プロジェクトにおいても、積極的にテスト計画へ反映してゆくことが必要である。そうして微調整され続けるテスト計画書は、プロジェクトメンバにとっても有意義なドキュメントとして参照され、更新されて精度が高まってゆく。

4 明日から使える単体テストの改善方法

３ では、単体テストのベストプラクティスについて述べたが、我々分科会メンバにとっても、ちょっとした改善活動でこれらのベストプラクティスを実践できるようにも思えなかった。そこで当分科会の成果として、単体テストのベストプラクティスを簡単に実現できるように、単体テストのプロセス品質を向上させるためのベンチマークを作成し、明日から使える単体テストの改善方法の考案をすることにした。本章では、ベストプラクティスを実現するまでのロードマップを提示する。

4.1 単体テストの CMM 的レベル分類

我々分科会メンバは、現状からベストプラクティスまで到達するためのツールとして、次の「単体テストCMM的レベル分類表」（［添付表１　単体テストのCMM的レベル分類表］参照）を作成した。

この表は、前述したテストプロセス（計画、設計、実施、管理）を SW-CMM の考え方を元にレベル１～５の５段階で表現したものである。単体テストの現状を分析するためのツールとして利用でき、それぞれのレベルに応じて改善活動を実施しやすくしたものである。それぞれのレベルには次のような特徴がある。

（1）レベル１：属人的実施

プロジェクトとしての手順が不明確であり、場当たり的、アドホックな状態で単体テストを実施している状態である。単体モジュールの品質レベルは、ある個人の能力や努力に委ねられている。

この状態では、単体テストに計画、設計、管理の各プロセスは存在せず、単体テストの実施記録が無いことが多い。

（2）レベル２：経験的実施

プロジェクトから示された必要最低限の手順を実施している状態である。しかし、これらの実施レベルは個人に委ねられており、プロジェクトとして管理やレビュー体制がとれておらず、品質の安定は依然として個人に委ねられている状態である。

この状態では、テスト項目書作成し、結果についての記録（バグ記録）を作成しており、作成したコードに対してのセルフレビューや文法チェックなどが行われている。

（3）レベル３：安定的実施

プロジェクトで共通手順を作成し、単体テストが特定の責任者により管理されている状態である。プロジェクト管理がある程度できており、単体テスト実施完了時には、一定レベルの品質を確保できている。

この状態は、単体テスト工程として、独立したスケジュール管理やバグ記録のデータ収集が実施されているといえる。テスト項目が仕様から設計されており、単体テストが設計検証の役割を果たしている。共通チェックリストや共通のテスト環境を利用することで、テスト漏れ・抜けを防ぐ工夫ができている。実施したテスト項目の DB 化を図ることが望ましい。

（4）レベル４：定量的管理

プロジェクト内で定量的な品質目標が設定されており、プロセスや成果物に対する計測が実施されている状態である。単体テストの結果を計測することにより現状を正しく把握し、以後の工程への課題材料として使用している。品質は安定している。

この状態では、定量的管理を実現するために、テスト技法を利用してテスト項目の設計レベルを一定に保つ、テストツールを利用してテストの自動化や計測を実施する、などの工夫を行っている。その結果、単体テストの完了基準の設定などが可能になっている。

（5）レベル５：最適

３ で示したベストプラクティスを実施し、定期的に単体テストの手法の改善が行われている状態である。

この状態では、計測結果により単体テストの実施方法が定期的に見直され、常に最善のものとなるように改善がされている。具体的には、設計、実施、管理の各プロセスの結果が単体テスト計画に反映されるようにフィードバックが行われている。（［添付表２　単体テストのCMM的レベル分類表］参照）

4.2 単体テスト自己診断用チェックリスト

また、我々分科会メンバは、単体テストの CMM 的レベル分類表に従って自身のプロジェクトを診断するためのツールとして、単体テストのレベル別要件を洗い出した単体テスト自己診断用チェックシートを用意した（表３）。このチェックシートの項目を満たしていなければ、該当レベル以下であることがチェックできる。

表３　単体テスト自己診断用チェックリスト

要件 No	必須要件の内容	該当レベル				
		1	2	3	4	5
1	単体テストを実施している	●	●	●	●	●
2	単体テスト項目書が存在する		●	●	●	●
3	障害記録を残している		●	●	●	●
4	コードのセルフレビューを行っている		●	●	●	●
5	プロジェクト共通手順が存在する			●	●	●
6	単体テスト工程として独立してスケジューリングやバグ収集の計画が作成され、計画通り実施、管理されている			●	●	●
7	コードからではなく仕様からテスト項目を設計している			●	●	●
8	定量的な品質目標、完了基準が設定されている				●	●
9	収集データの分析結果を以後の工程に連携している				●	●
10	テスト技法やテストツールを利用している				●	●
11	組織的に定期的に単体テスト手法の改善を行っている					●

● … 必須

4.3 改善活動

改善活動は一般的に、単体テストの現状分析→改善するプロセスの特定→改善項目の決定→改善活動の実施　という手順で実施する。

4.3.1 単体テストの現状分析

表３として示したチェックシートを元に、現在の単体テストのレベルを判定する。１１個のチェック項目を１番から順番に照らし合わせていき、該当しない場合、レベルを満たしていないものと考える。

4.3.2 改善するプロセスの特定

判定したレベルをもとに、改善するプロセスを特定する。改善するプロセスはレベルに応じて次の表４のように特定することができる。

表４　改善するプロセスの特定

現在のレベル	目指すレベル	改善するプロセス
レベル１	レベル２	設計
レベル２	レベル３	管理、計画　と　設計
レベル３	レベル４	計画、管理　と　設計、実施
レベル４	レベル５	管理、設計、実施の改善　⇒　計画に反映

改善活動というと、しっかりと計画を立てて実施するというイメージがあるようだが、我々分科会メンバでは計画を作っても計画倒れになっているという状況が見られており、それに対処するために、以下の図１のように「設計」→「実施」→「管理」→「計画」の順に現場から徐々に対象領域を拡大しながらフィードバックループを形成する、スパイラルアップな改善が望ましいという結論になった。
計測のないところは改善することはできない。その意味からも、まずはテスト実施における厳密な記録、報告が初めの一歩であるといえる。テスト実施状況の可視化は、直接的にはテスト管理への改善へと繋がるが、示された傾向はテスト設計への教訓とされてフィードバックループを形成することにも繋がる。

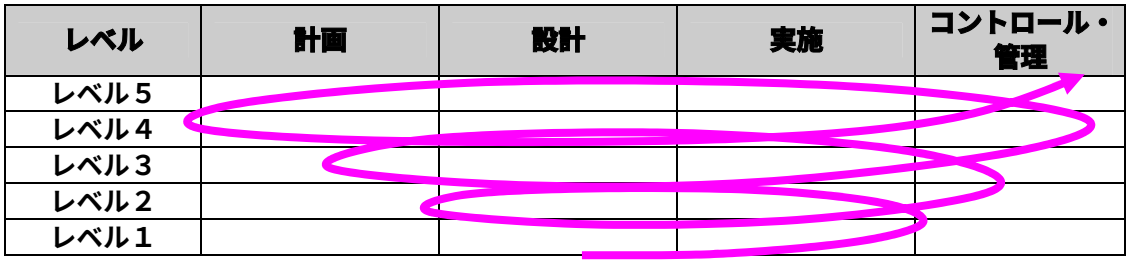

図１　スパイラルアップによる改善

また、立場によっても取り組むべき改善領域は異なる。プロジェクトマネージャや品質保証部が主体的に改善すべき領域はテスト計画、テスト管理であるが、テスト設計、テスト実施は単体テストを実施するテスターが主体的に取り組むべき領域である。この棲み分けを認識した上で、両者の協業を引き出すには、成功体験を積み重ねながら徐々にシンパを増やしてゆくようなアプローチが望ましい。

4.3.3 改善項目の決定

特定した改善すべきプロセス内には、４．１項で示したレベル分類表の通り、いくつかの項目が含まれている。3 項で示したベストプラクティスの状態を目指しながら、足りない点を改善項目として決定すると良い。各企業やプロジェクトによりレベルや実施内容が異なると考えられるので、各企業の実情に合わせて改善項目を決定することになると考えている。実施できるものから改善していくことになるだろう。

4.3.4 改善活動の実施

ソフトウェアの製造プロセスは複数のプロセスから成り立っており、何か１つの技法や手法、言語などを採用するだけで全ての問題が解決するような銀の弾丸は存在しない。単体テストにおいても同様であり、単体テストは 2 項で示したとおり、４つのプロセスから成り立っており、どれか１つを改善するだけでは不十分である。

多くの場合、プロセス改善は計画から取り組まれているが、それがために改善途中で息切れをおこしている。このような失速を防ぐために、初めの一歩として現状の計測ということを提案する。つまり単体テストの可視化からとりかかり、属人性を排除することができれば、少なくとも単体テストが他のテストフェーズと比較しての特異性が解消できると考える。

5 おわりに

今回、我々分科会メンバは単体テストに絞って改善方法を研究したが、レベル分類、チェックシートを実際に利用した検証するところまでは実施できなかった。実際に利用して改善活動を行うのが今後の課題であろう。また、単体テスト以外のテストの改善活動や設計とテストの関係などは今回の研究対象範囲外として取り扱っていない。今後、さらに研究を深めていく必要があると考えている。

6 参考文献

[1] 「ソフトウェア・テストの技法」Glenford J.Myers 著 近代科学社 1980 年出版
[2] 「eXtreme Programming テスト技法」 日本 XP ユーザグループ著 翔泳社 2001 年出版
[3] 「テストプロセス改善 –CMM 流実務モデル」 Tim Koomen, Martin Pol 著 構造計画研究所 2002 年出版
[4] 「基本から学ぶテストプロセス管理」 Rex Black 著 日経 BP 社 2004 年出版
[5] 「テスト技術の A to Z」日経 IT プロフェッショナル 2004 年 6 月号 日経 BP 社
[6] 「ステップアップのための ソフトウェアテスト実践ガイド」 大西健児著 日経 BP 社 2004 年出版
[7] 「体系的ソフトウェアテスト入門」 Rick D Craig, Stefan P Jaskiel 著 日経 BP 社 2004 年出版