管理図を用いた欠陥管理システムによる品質改善とその効果*

中村　伸裕[†,††a)]　高橋　覚[†††]　楠本　真二[††]

Software Quality Improvement with a Defect Management System Using Control Charts and Its Evaluation*

Nobuhiro NAKAMURA[†,††a)], Satoru TAKAHASHI[†††], and Shinji KUSUMOTO[††]

あらまし　ソフトウェア開発現場において，ソフトウェア品質改善の要求は高まっているものの改善に伴うコスト増加は容易に受け入れてもらえない状況にある．住友電気工業のシステム開発部門では，2006 年から開発コストの増加を抑制しながら品質を向上させる取り組みを進めてきている．具体的には，欠陥管理システムを構築し，管理図等のグラフを自動出力することでプログラム単位の欠陥作込と検出の状況を常時把握できるようにした．本システムを用いることで，再レビュー，再テスト必要なプログラムを特定することで効率的に欠陥検出可能となった．また，発生した欠陥の再発防止策が開発進行中に実施できるようになり，作込欠陥数の削減も実現できた．結果として，本番稼働後 3ヶ月間に発見される不具合は従来の 1/2 以下となった．一方，工数面では品質管理工数は増加したものの欠陥検出の効率化と欠陥の再発防止による修正コスト削減で相殺することができ，従来と同じ工数で高品質なソフトウェア開発が実現できた．

キーワード　欠陥管理システム，品質管理，管理図，統計的品質管理，CMMI，構成管理

1. まえがき

近年，ソフトウェアの利用は社会インフラとなる金融・交通等のシステムや企業内システム等に広範囲に広がっており，システム障害が国民生活や企業活動に支障をきたすこともある．日本では証券取引所のシステム障害等により経済産業省が『情報システムの信頼性向上に関するガイドライン』[1] を発行し，情報システムの品質改善の取り組みを後押している．文献 [2] によればシステムの品質は毎年向上し，納品後に顧客よって発見された欠陥の密度は中央値で 0.20 件/開発工数 (人月) まで改善されている．また，文献 [3] によればインド，米国，欧州にくらべ日本の欠陥密度が 1/10 以下と極めて高い品質を実現していることがわかる．一方で，テスト強化により高品質を実現している企業はそのためのコストを負担しており，全体としてのコスト削減が課題となっている．

住友電気工業 (株) ではシステムの品質向上と投資効率の両立を目指し，コスト増加を抑制しながら品質改善を目指してきている．品質改善でよく利用される方法はテストを重視し（例えば，テスト項目を増やすことで網羅性を上げ），より多くの欠陥を納品前に検出する方法である．しかし，この方法ではテスト設計，テスト実施のコスト増加が避けられない．そこで以下の二つの施策を実施することにした．

(a) 作込む欠陥数を削減し，修正工数の削減を図る．
(b) 作込まれた欠陥をできるだけ早期に効率良く検出する仕組みを確立し，修正工数の増加を押さえる（例えば，設計の欠陥がソースコードに残された場合，欠陥場所の特定コストが増加するため）．

これらを実現するためにはプロジェクト進行中に欠陥の作込と検出の状況を把握し，迅速に対策が実施できるようにする必要がある．組織全体でこれらの施策を実施するには欠陥管理のシステム化が不可欠であると考え，レビューやテストの結果を登録することで，

† 住友電気工業株式会社，大阪市
Information System Division, Sumitomo Electoric Ind. Ltd., 4–5–33 Kitahama, Osaka-shi, 541–0041 Japan
†† 大阪大学大学院情報科学研究科，吹田市
Graduate School of Information Science and Technology, Osaka University, Suita-shi, 565–0871 Japan
††† 住友電工情報システム株式会社，大阪市
Systems Solution Division, SUMITOMO ELECTRIC INFORMATION SYSTEMS CO., LTD. 2–1–3 Nishimiyahara, Yodogawa-ku, Osaka-shi, 532–0004 Japan
a) E-mail: nakamura-nobuhiro@sei.co.jp
* 本論文はシステム開発論文である．

欠陥の作込や検出の状況がリアルタイムに把握できるシステムを構築した．プロジェクトマネージャが，毎週，欠陥作込・検出状況を監視し，問題発生後すぐに対策を実施することで欠陥の再発防止ができるようになることが期待される．欠陥管理システムの導入後，導入前に比べて本番稼働後 3ヶ月間に検出される不具合は 1/2 以下となった．本論文では開発した欠陥管理システムの機能とその適用効果について報告する．

以降，**2.** では，本取り組みの背景について説明する．**3.** では，開発した欠陥管理システムについて述べ，**4.** では組織内での展開について紹介する．**5.** で適用結果について考察し，**6.** でまとめと今後の課題について述べる．

2. 取り組みの背景と方針

2.1 組織の状況

今回対象とする組織は住友電気工業のグループ企業を対象とした事務処理システムの開発を主な業務とする組織である．システム規模は 1000～3000FP [4] のものが多い．開発者の人数は約 400 名である（ただし，システム開発の負荷により協力会社の人数が増減する）．当組織では新規開発が多く，既存システムに対する保守開発に比べ約 1.5 倍となっている．技術的な特徴を次の (1)～(5) にまとめる．

(1) 標準化の徹底: システム開発で使用する OS，ミドルウェア，フレームワーク，RDBMS 等の開発環境をはじめ，各種仕様書やソースコードの書式を統一している．開発者はどのプロジェクトでも同じスキルで開発できるため，プロジェクトの需要にあわせて開発者を割り当てることができる．一方，プロジェクトマネージャから見れば，チームメンバが毎回入れ替わるため，事前に開発者の能力を把握できないケースもある．

(2) データ中心設計手法の導入: 1995 年からデータ中心設計を導入し，全プロジェクトで ER 図 (Entity Relationship Diagram) を作成している．プログラムは受注，出荷指示，出荷といった業務ごとに開発し，データベースをインタフェースとして独立して動作する．そのため，プログラムは独立して動作する単位を 1 本として管理している．一部の共通部品を除いてプログラム間の呼び出しは行わない．

(3) 組立型システム開発の実施: 数百種類のソフトウェア部品を組み合わせてシステム開発を行うことにより文献 [5] のデータと比較して 2～3 倍の高い生産性を実現している．再利用率が高いためプログラム 1 本あたりのソースコード (Java) の論理行数は中央値 77，平均 164 ステップと小規模である．部品を利用しない場合，2KStep 程度と想定される．

(4) プロセス改善: 2003 年に CMM 成熟度レベル 3 を達成し，更に上位のレベルを目指していた．しかし，CMMI [6] 成熟度レベル 4 で求められている管理図等を利用した統計的品質管理の経験やノウハウをもった開発者は存在しない．

(5) 開発工程: 主な開発工程は，要件定義，外部設計，データベース設計，プログラム設計，プログラム開発，統合テスト，システムテスト，本番稼働である．

2.2 欠陥管理システム構築の背景

当組織では比較的高い生産性を実現していたが，品質のばらつきが大きく，利用部門から低い評価を受けるシステムもあった．そこで 2006 年から品質改善の取り組みを開始した．品質改善の代表的な方法はテストの網羅性を高め欠陥の検出率を高めるものであるが，テスト設計及びテスト実施の工数増加による開発費の増加が避けられない．一方，一般に，品質を上げれば修正コストが減り，品質向上とコスト削減は両立することはよく知られている [7]．そこで，コスト増加を抑制しながら品質改善を実現することを目指し，CMMI モデルを教科書として，他社の取り組みも参考にしながら具体的な施策を検討した．

2.3 欠陥管理に関する先行事例

日本電気 (株) は「品質会計」[8] と呼ばれる技法により 1985 年から 1990 年代にかけ 1 年間で発生する出荷後バグ件数を 1/20 に削減している．この手法は「バグを作り込まない．作り込んだバグは素早く摘出する」，テスト工程では「作り込んだバグは全て摘出してから出荷する」という二つの考え方に基づいている．具体的には，「回帰型バグ予測モデル」と呼ばれるモデルを使って各工程で摘出するバグの件数を予想し，実績を管理している．品質改善のポイントは検出した欠陥を作り込み視点での分析を行うことで，再発防止を促していることである．この手法では，レビューによる早期品質確保によりテスト段階でのバグ摘出が減少するため全体工数が増加することはないと報告されている．

日本アイビーエム (株) は「混入欠陥予測モデル」と呼ばれるモデルを利用し，品質管理を行っている [9]．

ソフトウェアの欠陥は混入された工程で検出されるとは限らず，欠陥の種類によって検出されている工程が異なることに着目し，欠陥の混入，検出の関係をモデル化している．各プロジェクトはプロジェクトの途中で既に検出されている欠陥をモデルの入力とし，次工程の欠陥検出目標を求める．

上記二つの事例では欠陥の検出だけではなく，欠陥の作込工程に着目しており，作込工程の対策を実施することが品質改善とコスト増加抑制の両立のポイントであると考えられる．また，二つの事例とも過去の実績データからモデルを構築し，作込欠陥数の予測を実施している．しかし，我々の組織では欠陥を作り込む要因に関するデータが収集できていないため，これらの手法にすぐに取り組める状況ではなかった．

CMMI 成熟度レベル 4 では品質予測モデルとともに統計的なプロセスの監視を求めている．レベル 4 を達成した組織では QC 七つ道具の一つである管理図 (JIS Z 9021) [10] を利用しているケースが多い [11], [12]．管理図は管理指標の種類（計数値，計量値）や管理対象のサイズの種類（一定，可変）等により幾つかの種類が用意されている．ソフトウェアの欠陥の管理には基本的に u 管理図を利用する（図 1）．しかし，u 管理図は実績値のばらつきに関係なく，対象物の規模により管理限界が固定されるという性質があるため別の種類の管理図を利用している例もある．文献 [13] では，成果物単位ではなく，1 日ごとにテスト結果を集計して欠陥密度の X-R 管理図を利用している．取り組みの効果としてシステムテストでの修正工数 52%削減が報告されている．

また，文献 [14] では，大量のソースコードの品質や進捗を短期間で俯瞰的に評価する技法が示されている．欠陥が混入されやすい箇所に関するノウハウや経験則を蓄積し，欠陥混入の疑いが強い箇所を検出するツール開発している．このツールにより大量のソースコードの中から目視調査する対象を絞り込むことにより，短期間で品質評価が行えるようにしている．欠陥混入の疑いが強い箇所を重点的にレビュー，テストするという考え方は，欠陥検出の効率化に役立つものである．しかし，実践するためにはノウハウや経験則の蓄積が課題となる．

2.4 管理図について

管理図は品質や工程が統計的に安定しているかどうか判断するために利用するグラフである．図 2 に長さ，重量，時間といった計量値に利用される X 管理図の例を示す．縦軸が管理指標を表す．過去のデータから平均値を求めて，中心線 (CL) を設定する．また，標準偏差（σ）を計算し，CL＋3σ を上方管理限界 (UCL)，CL − 3σ を下方管理限界 (LCL) とする．更に CL と UCL，LCL の間を 3 分割する．横軸は管理対象のデータを発生順に並べる．JIS 規格 JIS Z 9021 では異常判定の八つの判定基準が示されているが，そのうち最もわかりやすい判断基準は (a) プロットした点が管理限界を超えるもので，他に (b) 連続する 3 点中 2 点が 3 分割した領域 A にあるといったものが示されている．

一方，文献 [15] ではソフトウェアへの管理図の適用は正規の統計的プロセス管理 (SPC) やプロセス能力に使うのではなく，むしろ一貫性と安定性を改善するためのツールとして用いることが有効であることが示されている．

管理図は管理対象の種類により幾つかの種類が用意されているが，欠陥数の管理は計数値を扱う u 管理図の利用が基本となる．u 管理図の例を図 1 に示す．縦軸はプログラムの欠陥密度（規模あたりの欠陥数）を

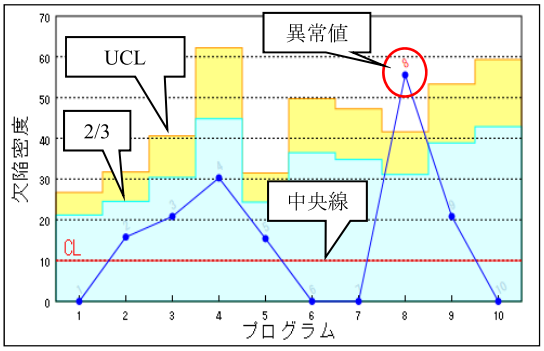

図 1　検出欠陥密度の u 管理図
Fig. 1　u Control Chart of detected defect density.

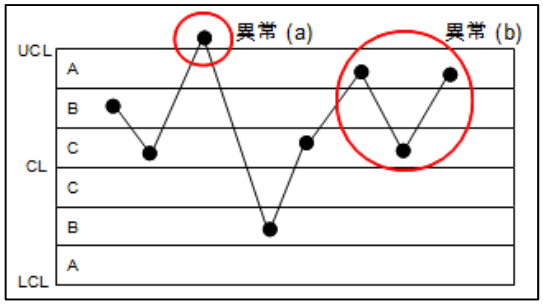

図 2　X 管理図のサンプル
Fig. 2　Sample of X control chart.

示している．横軸はプログラムを表す（図 1 では，10 個のプログラム 1〜10，テスト実施順に並べている）．各プログラムは登録，照会，変更，削除の機能をもち，複数のソースファイルで構成されているため，各点は総ライン数に対する欠陥の加重平均となっている．中央線 (CL) は組織全体の実績から求めた基準値である．u 管理図の管理限界は中心線 (CL) と対象物の規模を使って式 (1) で求めることができる [10], [15].

$$\mathrm{UCL} = \mathrm{CL} + 3 * \mathrm{SQR}(\mathrm{CL}/\text{規模}) \qquad (1)$$

規模に応じて階段状の管理限界になっている．規模の大きいものほど管理限界の幅が狭くなる．更に，中央線と管理限界の 2/3 の位置に補助線を描いている．この例では下方管理限界 (LCL) は計算上 0 以下となるが，欠陥数がマイナスになることはないので LCL は描画していない．

2.5 品質改善の基本方針

組織の実力と他社での取り組みを参考にし，品質改善とコスト増加抑制を両立させるために以下の方針で品質改善を進めた．

(1) 作込欠陥の削減と安定化

欠陥の作込量を削減すれば従来発生していた修正工数の削減が実現できる．また，検出プロセスから見れば欠陥密度が安定し，一定範囲に収まっていることが望ましい．管理図を使って作込欠陥密度を監視し，欠陥の作り込みが多い場合に再発防止策を実施できるようにする．

(2) 欠陥検出効率の改善

管理図やその他の統計情報を使って欠陥の検出状況を監視できるようにし，検出欠陥密度が少ないものを対象に追加レビュー，追加テストが効率的に実施できるようにする．

(3) システム化による効率化

管理図の作成や教育コストを削減するためできる限りシステム化を行い，簡単に欠陥管理が実施できるようにする．

2.6 作込欠陥と検出欠陥の関係

作込欠陥と検出欠陥の関係を表 1 で説明する．この表は縦軸に成果物，横軸に欠陥の検出工程を示している．実際のプロジェクトではより多くの種類のレビューやテストの工程が実施されるが単純化している．この表の PG 設計の列はプログラム設計レビューで検出した欠陥を示しており，外部仕様書の欠陥 1 件，プログラム仕様書（以下，PG 仕様書）の欠陥 3 件を検出している．検出欠陥数は合計 4 件である．一方，PG 仕様書に作り込まれた欠陥は PG 設計，単体テストの工程でそれぞれ 3 件，2 件検出している．この時点で検出されている作込件数は 5 件となる．工程が進むにつれて未検出の欠陥が検出されるため，作込欠陥数は時間とともに増加する．当組織では，これまでの実績データに基づいて，本番稼働後 3ヶ月の時点でほぼ全ての欠陥が摘出されたとみなし，作込欠陥数を確定するルールとした．

表 1 検出欠陥数と作込欠陥数
Table 1 Number of detected and introduced defects.

成果物＼検出工程	外部設計	ＰＧ設計	単体テスト	作込欠陥数
外部仕様書	3	**1**		4
ＰＧ仕様書		**3**	**2**	**5**
ソースコード			6	6
検出欠陥数	3	**4**	8	

2.7 ツールの選定

管理図を作成するためのツールは多数存在し，表計算ソフトでも作成することができる．しかし，以下の理由により自社開発することにした．

(1) 組織全体の品質実績管理

表計算ソフト等のパソコン単体で動作するソフトウェアを利用して管理図を作成する場合，管理図及び管理図作成の元になっている品質データが個人のパソコンに保存される可能性が高く，全社の基準値を作成する際，各プロジェクトからデータを収集するのに手間がかかる．また，管理図の利用状況が品質管理部門から見えにくくなり，プロジェクト管理上の問題を検出できない．

(2) 協力会社への対応

管理図による品質管理は協力会社の開発者も実施する必要があり，有償のソフトウェアを利用する場合，費用負担の問題が発生する．

(3) 作込欠陥の管理

検出欠陥の管理はレビューやテストの結果から容易に作成できる．しかし，作込欠陥の管理は表 1 に示したように複数の工程にまたがって集計する必要があり，下流工程で作成される仕様書，ソースコードとの構成を管理する必要がある．作図の前のデータ集計の効率化が必要となる．

3. 欠陥管理システムの構築

3.1 システム概要

システム開発で作成される仕様書等のドキュメントは全て自社開発の文書管理システムで管理している．欠陥管理システムではレビュー対象となる文書のデータ（規模等）が必要となるため，本文書管理システムに欠陥管理機能を追加する形で構築することにした．

本文書管理システムは Linux サーバーを利用して構築されており，文書閲覧者は一般的なブラウザを使って文書を閲覧することができる．文書作成者の PC には専用ツールがインストールされており，簡単に文書が作成できるようになっている．本文書管理システムの主な機能は以下のとおりである．

(1) 文書作成管理: 外部仕様書，プログラム仕様書等の文書の作成計画（設計計画）が立案でき，計画に従って文書を作成することができる．また，文書の作成状況を確認する進捗管理表が出力できる．

(2) Web 文書生成: HTML タグを記述しなくもブラウザで閲覧できる HTML 形式の文書を簡単に作成できる．また，画像やスプレッドシート等のバイナリファイル，ソースコード等が添付できる．

(3) 文書の履歴管理: 各文書はリビジョン管理されており，過去の文書も閲覧することができる．例えば，外部仕様書の場合，初版 (Rev.001) を作成して利用者に提案する．通常，新たな要求が追加されるため第 2 版 (Rev.002) を作成し，再度利用者に提案する．利用者と合意できれば Rev.002 は正式文書として発行される．更に利用者から新たな要求が発生した場合，Rev.003 を作成し，利用者に提案する．この段階では Rev.003 は作成中のドラフトであり，正式文書は Rev.002 であるため，一般閲覧者には Rev.002 が表示される．改訂履歴の一覧画面からは全てのリビジョンを閲覧することができる．

(4) 関連文書管理: 文書作成（設計）時に入力として使用した文書（以降，先行文書と呼ぶ）を登録することができる．例えば，プログラム仕様書の先行文書は外部仕様書となる．ブラウザによる閲覧ではプログラム仕様書と外部仕様書の双方にリンクが設定され，簡単に閲覧できる．

(5) 文書属性管理: 文書 ID，文書タイトル，作成者，作成日，作成部署，検索用カテゴリ，備考等の文書属性が登録できる．

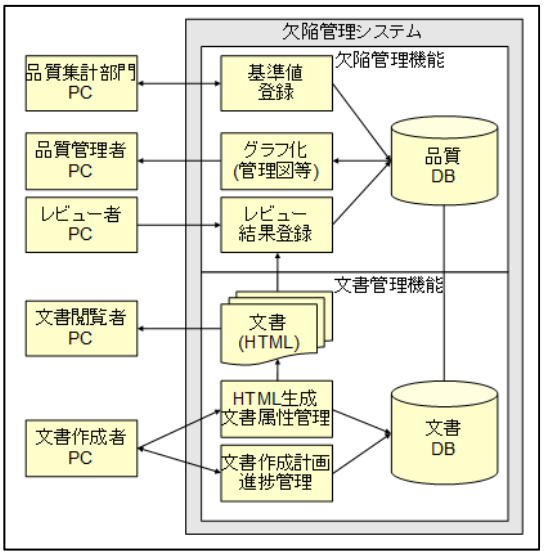

図 3 欠陥管理システム概要
Fig. 3 Overview of the defect mangement system.

(6) 自動測定: 作成した文書の文字数や添付されているスプレッドシートの行数，添付されているソースコードの文字数，行数，コメント文の文字数，IF 文の数等を自動測定し，文書の属性として保管できる．

今回開発した欠陥管理システムの機能概要を図 3 に示す．欠陥管理機能の基本的な流れはレビュー者がレビューの結果を品質 DB (Database) に登録し，登録されたデータをニーズに合わせてさまざまなグラフを表示することである．また，開発プロセスの異常を判断するためには基準値が必要となるが品質集計部門が毎年 1 年間の実績データをもとに基準値を登録している．本システムは Microsoft 社の Visual Basic で開発しており，文書管理機能と欠陥管理機能を含めた総ライン数は約 100KStep である．以下，グラフ化機能について詳細に述べる．

3.2 欠陥検出プロセスの管理図

欠陥検出プロセスはレビューとテストに大別できる．どちらも投入された工数と検出した欠陥数を監視する．

(1) レビュー時間密度（テスト時間密度）

レビュー時間の監視を行う為に成果物規模に対するレビュー時間密度をランチャートで表示する．ランチャートはデータを発生順にプロットしたものであり，管理図との違いは中央線と管理限界が設定されていないことである．将来的には最適値を求め，管理図

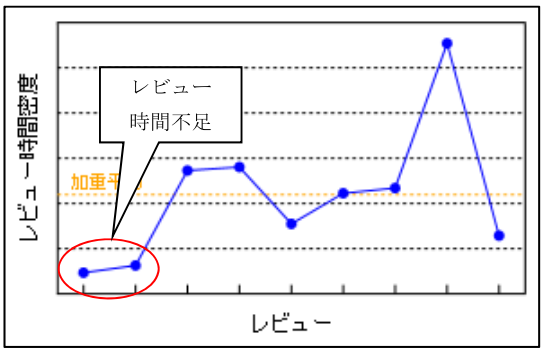

図 4　レビュー時間密度のランチャート
Fig. 4　Run chart of review time density.

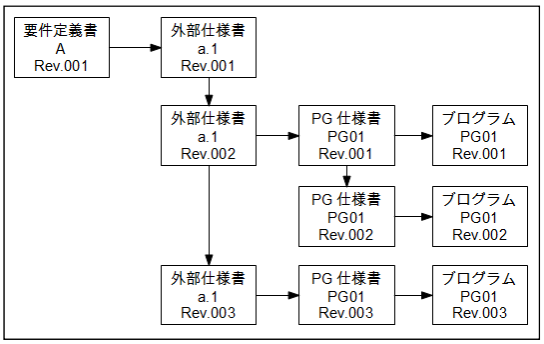

図 5　文書の構成管理
Fig. 5　Configration of documents.

で管理することが望ましい．当組織では，検出された欠陥が少ない場合にレビュー時間密度を確認して再レビューするかどうかの判断に利用している．図 4 は縦軸に成果物規模当たりのレビュー時間をとり，横軸にレビュー対象をレビュー順にプロットしたものである．1 件目と 2 件目はレビュー時間密度が小さく，レビュー時間が不足している可能性が高いと判断できる．

(2) 検出欠陥密度

検出した欠陥は図 1 に示した u 管理図を使って管理する．しかし，試行プロジェクトに適用したところ改善点が二つあることが判明した．一つはレビュー不足により欠陥の検出量が少ないと判断した場合，再レビューを実施することになるが，再レビューの結果，十分欠陥が検出されたかどうか判断できないことである．もう一つは，ソースコードに対してコードレビューと単体テストの 2 種類の検出プロセスが実行された場合，コードレビューでより多くの欠陥を検出すると単体テストでは通常より少ない欠陥しか検出できないという問題である．この場合，再テストを実施しても更に欠陥が検出できる可能性が少ない．

前者の問題は，成果物単位で検出された欠陥を集計して管理図を出力することで解決する．レビューの結果の合計値が管理限界の範囲に収まれば問題なしと判断できる．

後者の問題は，成果物単位で全てのレビュー，テストで検出された欠陥の合計値で管理図を出力することで解決する．例えば，単体テストの欠陥密度が低くても，コードレビューの欠陥密度と合計し，その値が管理限界の範囲に入れば問題ないと判断できる．

文献 [12] では u 管理図の代わりに X 管理図等を利用していたが，試行の結果，当組織ではソースコードの規模が中央値 77 ステップと小さいこともあり規模に応じて管理限界が変化する u 管理図の方が適切に異常値を確認できることがわかった．例えば，中心線が 10 件/KStep である場合，77 ステップのプログラムは欠陥が 1 件あるだけで 13.0 件/KStep となり，中心線を超える．更に小さいプログラムでは X 管理図の管理限界に収まらなくなる．

3.3　欠陥作込プロセスの管理図

表 1 に示したように検出欠陥にはさまざまな工程で作り込まれた欠陥が含まれているため，作込欠陥を把握するには原因となった成果物ごとに欠陥を振り分ける必要がある．一つの方法は欠陥発生の原因となっている成果物及びリビジョンを直接指定して入力する方法であるが，入力ミス防止と入力の負荷を軽減するため文書管理システムの構成管理機能を活用することにした．図 5 に構成管理の仕組みを示す．図中の四角形は一つの文書を示している．例えば 10 種類の業務を支援するシステムでは，外部仕様書，プログラム仕様書，プログラムがそれぞれ 10 種類作成されるが，図 5 では一つの業務に対する文書のみを表示している．外部設計では要件定義書 A を入力に対象業務ごとに外部仕様書 a.1 のリビジョン Rev.001 を作成する．外部仕様書のレビューを実施すると Rev.001 の文書は変更不可の状態となる．レビュー指摘に対応して外部仕様書を修正すると Rev.002 が作成される．プログラム設計では外部仕様書 a.1 の Rev.002 を入力としてプログラム仕様書 PG01 Rev.001 を作成する．更にプログラム開発の工程ではプログラム仕様書を入力としてプログラム PG01 Rev.001 を作成する．このプログラムの単体テストを実施し，プログラム仕様書の欠陥が見つかればプログラム仕様書を修正し，Rev.002 を作成

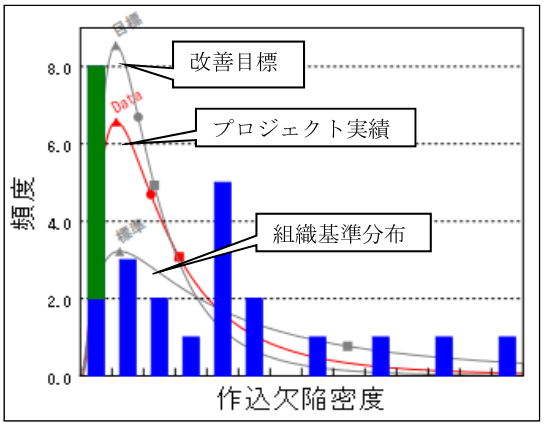

図 6　作込欠陥密度の分布
Fig. 6　Distribution of introduced defect density.

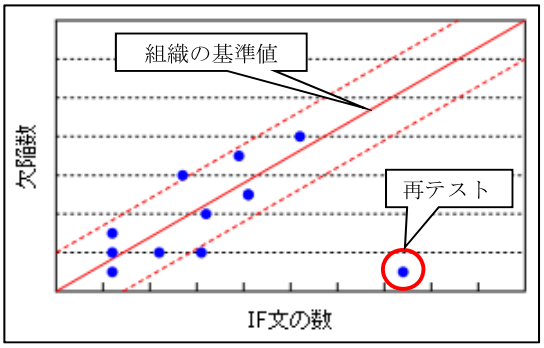

図 7　IF 文の数と欠陥の相関図
Fig. 7　Correlation diagram between the number of defects and IF statements.

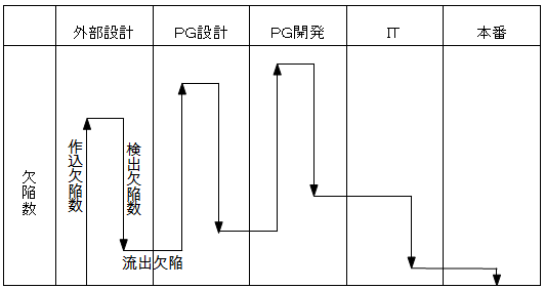

図 8　欠陥フロー図
Fig. 8　Example of a defect flow diagram.

し，プログラムも修正し，Rev.002 を作成する．プログラム PG01 Rev.002 をレビューし，問題が外部仕様書の欠陥であることがわかれば，作込成果物の種類として “外部仕様書” を選択するだけで自動的に対象文書が “a.1 Rev.002” であることが判断できる．このような仕組みにより成果物ごとの欠陥数を自動計算し，作込管理図を自動出力するようにした．

また，改善を促進するために図 6 に示すような作込欠陥密度の分布図を出力するようにした．この図は横軸が作込欠陥密度，縦軸が件数を示す．棒グラフに加えて (a) 組織基準分布：毎年，過去 1 年間の実績値を対数正規分布の近似曲線で示したもの，(b) 改善目標分布：毎年改訂される組織目標を達成するために工程ごとに意図的に定めた目標分布，(c) プロジェクト実績：該当プロジェクトの実績値から求めた対数正規分布の近似曲線を示している．棒グラフだけでは該当プロジェクトの品質評価が難しいが，近似曲線を示すことで評価できるようになる．分布のピークが高く，幅が狭いほど設計・製造プロセスの品質が安定していることを示す．

3.4　IF 文と欠陥数の相関図

文献 [14] に示されているように欠陥が残っている可能性の高い部分を収集したデータで示すことができれば低コストで効率良く欠陥を除去できる可能性が高い．これまでに示した管理図は，規模と欠陥が比例することを前提としているが，一般的にはプログラムの複雑さと作込欠陥数の相関も高いと考えられる．当社で過去のデータを分析した結果，欠陥がよく制御されたプロジェクトではソースコード中に含まれる IF 文の数と欠陥数との間に相関係数 0.7 程度の相関があることがわかった．この結果から，各プロジェクトで作成した IF 文の数と検出欠陥数の相関図を自動出力することにした．図 7 にサンプルを示す．この図は文献 [16] に示されている回帰線の周囲に管理限界を設ける方法に基づき作成している．横軸は IF 文の数を，縦軸は欠陥数である．対象プログラムについてプロットされている．補助線は組織の基準値を示している．この範囲から下側に外れた点は複雑なプログラムにもかかわらず欠陥数が少ないことを示しており，再テストを検討する必要がある．また，上側に外れた場合，簡単なプログラムにもかかわらず欠陥が多いため，プログラム開発の指導を検討する必要がある．

3.5　欠陥フロー図

これまでに示した品質に関するグラフは開発進行中に問題を発見し，是正するためのものである．これらに加え，次回プロジェクトに向けた改善計画が立案できるよう図 8 に示す欠陥フロー図と名付けたグラフを出力する．横軸は工程を示している．実際にはより多くの工程があるが説明のため簡略化している．縦軸は

欠陥数を示す．上向きの矢印は該当工程で作り込まれた欠陥の数を示し，下向き矢印は検出された欠陥の数を示す（数値は省略している）．作込欠陥数が多い工程は次プロジェクトでの改善対象となる．工程をまたぐ横線は次工程への流出欠陥を示す．自工程保証の観点から少ないほど良い．

4. 欠陥管理システムの組織展開

4.1 システム導入教育

開発した欠陥管理システムを組織全体で活用できるよう，2007 年末から 2009 年 3 月にかけて 3 時間の演習付きセミナーを合計 20 回開催した．各セミナーの受講者は 5〜15 名で，合計約 200 名が受講した．

4.2 展開の支援と監視

多忙な開発者が使い方がわからない等の理由で欠陥管理システムを活用しない事態を防ぐため，問合せ窓口を設置し，すぐに不明点を解消できる体制を整えた．また，管理図が作成された際，自動的に窓口の担当者にメールで通知する機能をシステムに追加し，品質データの入力方法等に間違いがないか確認するようにした．使用方法に問題があれば窓口から開発者に連絡し，積極的に活用方法の指導を行った．

4.3 プロジェクト完了報告会の改善

当組織では，従来から本番稼働後 3ヶ月のタイミングでプロジェクト完了報告を行っている．ツールの展開に合わせて，欠陥フロー図及び管理図を使って品質管理の報告を行うように変更した．

5. 成　　果

5.1 品質改善の実績

図 9 は年度別に本番稼働後 3ヶ月で検出された各プロジェクトの欠陥密度 (件/KFP) を箱ひげ図で表したものである．年度別の平均値も点で示している．縦軸は 2005 年の中央値を 100 とした相対値であるが，箱部分と中央値の変化がわかりやすいように縦軸の最大値を 800 にしている．800 を超える外れ値は 2005 年度 2 件，2006 年度 1 件，2008 年度 1 件発生している．なお，2012 年度のデータは 2012 年 11 月まで 7ヶ月分の途中集計となっている．

2007 年に欠陥管理システムを開発し，二つのプロジェクトで適用評価し，運用に問題がないことを確認した．2008 年度は教育を組織全体に広めた年である．実際の適用は，教育を受けた後，新たに開始する新規開発プロジェクトとなる．結果は稼働日を基準に年度

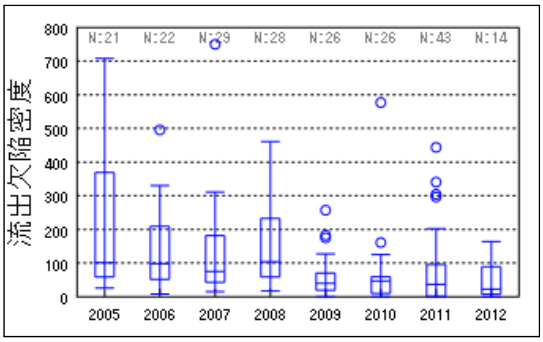

図 9　年度別流出欠陥密度の箱ひげ図
Fig. 9　Box whisker plot of remaining defect density by year.

を判断しているため，欠陥管理システムを利用して 2008 年度中に稼働したシステムは少なく効果が出ていない．2009 年度のプロジェクトはほぼ全てのシステム開発で欠陥管理システムが利用されている．結果として 2009 年度から 2011 年度の中央値は 2005 年度の中央値対比で 22〜46%に改善されている．また，箱の幅は 16〜31%となり，ばらつきも改善されている．欠陥管理システムの効果を判定するために 2005〜2008 年の欠陥密度と 2009 年以降の欠陥密度に対して平均値に有意な差があるか有意水準 5%で検定(注1)を行った．その結果，2009 年以降の平均値が改善していることが確認できた．

5.2 欠陥検出プロセスの実績

(1) プログラム設計 ピアレビュー

図 10 は 2007〜2012 年に欠陥管理システムを利用したプログラム設計ピアレビューの結果を示したものである．縦軸は，品質改善の効果が現れた 2009 年のプログラム仕様書ごとの初回のレビューの検出欠陥密度の平均値を 100 とした相対値で，初回レビューのみの検出欠陥密度（加重平均）と再レビューの結果を合計した平均検出欠陥密度をプロットしている．更に再レビューを実施したプログラム仕様書の割合を再レビュー率 (%) として示している．全体的に再レビューにより初回の 2 割程度の欠陥が検出できている．2010 年は再レビュー率が低下しており，再レビューによる欠陥検出数も減少している．残った欠陥は後述する単体テストで検出されたと考えられる．また，2008 年

(注1)：分布が対数正規分布であることが予想できたため，まずコルモゴロフ・スミノフ検定で二つの集合の log 値が正規分布であることを確認した．次に F 検定で等分散であることを確認し，両側検定の t 検定で有意な差があることを確認した．

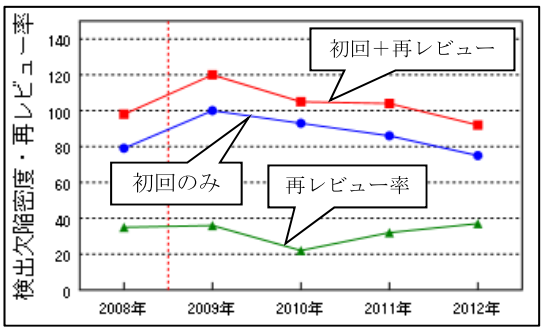

図 10　PG 設計レビューの検出欠陥密度
Fig. 10　Defect density in peer reviews.

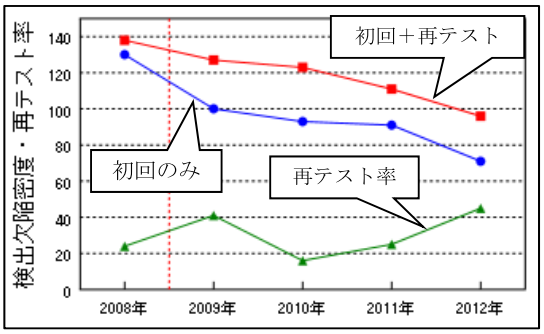

図 11　単体テストの検出欠陥密度
Fig. 11　Defect density in unit test.

表 2　ピアレビューの効率 (2012 年)
Table 2　Efficiency of peer review.

回数	実施回数	実施率	検出数	検出有率	検出効率	工数比
1	1571	100%	3738	62%	100	100
2	650	41%	805	43%	126	23
3	283	18%	296	35%	136	7
4	119	8%	126	41%	168	3
5	52	3%	60	46%	251	1

表 3　単体テストの効率 (2012 年)
Table 3　Efficiency of unit test.

回数	実施回数	実施率	検出数	検出有率	検出効率	工数比
1	1497	100%	2912	62%	100	100
2	666	45%	680	47%	147	24
3	196	13%	185	47%	224	4
4	69	5%	78	58%	206	2
5	32	2%	46	69%	247	1

の初回及び再レビュー後の検出欠陥密度が低いのはレビュー能力の低さによるものであると考えられる．

表 2 はピアレビューの効率を示したものである．回数は，同一仕様書に対する何回目のレビューかを示している．2 回目のデータに着目すると 41%の仕様書が 2 回目のレビューを実施しており，合計 805 件の欠陥を検出している．検出有率はレビューで 1 件以上の欠陥を検出できた割合であり，43%となっている．検出効率は 1 回目の検出効率 (件/時間) を 100 とした相対値で示しており，126 となっている．1 回目に比べて 26%高い効率で欠陥が検出できている．工数比は 1 回目の総レビュー時間を 100 とした相対値で 23%の工数を使っている．2〜5 回目の検出効率は 1 回目に比べて高く，欠陥が残っている可能性の高い仕様書を選択的にレビューできていると考えられる．また，2〜5 回目の工数合計は 1 回目の 34%であった．

(2) プログラム開発 単体テスト

図 11 は図 10 と同じ書式で 2008〜2012 年に実施した単体テストの結果を示している．2009，2012 年の初回テストの検出密度は前年よりも 2 割以上少なく，基準量の欠陥を摘出するために再テスト率が増加したと考えられ，欠陥管理システムの仕組みが有効に働いたことがわかる．表 3 は表 2 と同じ書式で 2012 年の再テストの効率を示したものである．プログラム仕様書のピアレビュー同様，2〜5 回目の検出効率は 1 回目に比べて高く，欠陥が残っている可能性の高いプログラムを選択的にテストできていると考えられる．また，2〜5 回目の工数合計は 1 回目の 31%であった．しかし，早期欠陥検出により統合テストの工数が減少したため，システム開発全体の工数増加にはつながっていない．

5.3　作込欠陥密度の評価

欠陥管理システムを利用していないシステムの作込欠陥密度が測定できていない為，システム化の効果は判断できないが，欠陥管理システムを利用したプロジェクトのプログラム開発の作込欠陥密度を図 12 に示す．縦軸は平均値を 100 とした相対値で，横軸はプロジェクトを示している．また，標準偏差 (σ) を使って $\pm 1\sigma$ の範囲を補助線で示している．プロジェクトごとにばらつきが大きく組織全体で設計・開発プロセスが安定しているわけではない．全体の平均値 (100) 以下の多くのプロジェクトは作込欠陥の削減を実施しているがそのノウハウが他のプロジェクトに横展開されておらず，効果がプロジェクト内に留まっている．ノウハウの共有が進めば組織全体で更に作込欠陥密度

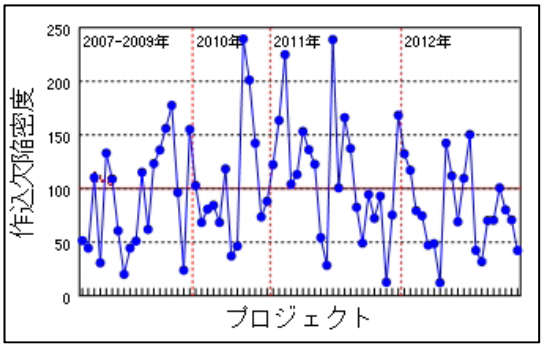

図 12　プログラム開発の作込欠陥密度
Fig. 12　Introduced defect density of each program.

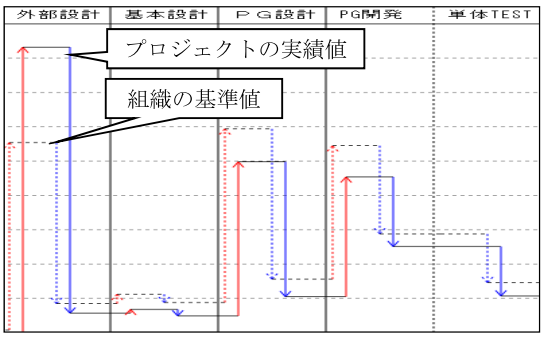

図 13　欠陥フロー図のサンプル
Fig. 13　Example defect flow diagram.

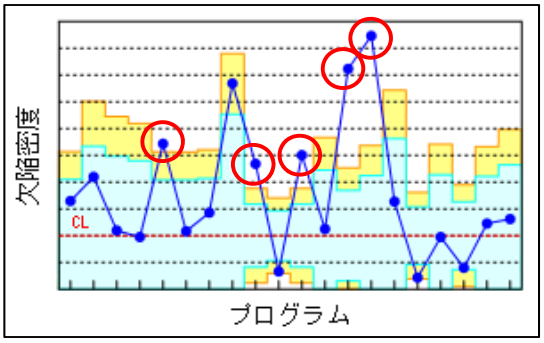

図 14　品質制御なしプロジェクトの u 管理図
Fig. 14　u control chart of the projects without quality control.

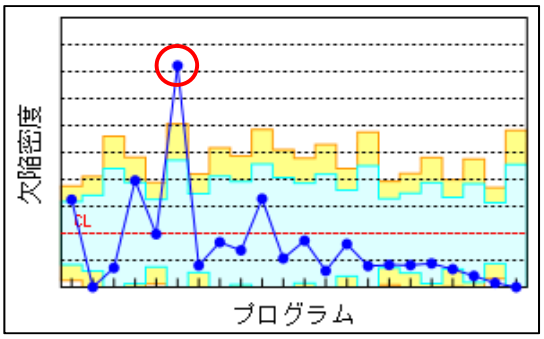

図 15　品質制御されたプロジェクトの u 管理図
Fig. 15　u control chart of the projects with quality control.

を低減できると考えられる．また，作込欠陥密度が高いプロジェクトでは機能的に問題がなくても保守性の観点から改善点を指摘しているものも多く，ばらつきの要因となっている．2012 年のプロジェクトは平均値を下回るものが増えており，今後の改善が期待できる．

5.4　品質管理プロセスの変化

(1) 改善計画の立案

図 13 は実際に作成された欠陥フロー図の一部（外部設計～単体テスト）を拡大表示したものである．プロジェクトの実績値と組織の基準値が示されている．プログラム設計，プログラム開発の作込欠陥数は組織の基準値に比べ，それぞれ 21%，10%少ない値となっている．また，外部設計を含めて全ての工程で流出欠陥が組織の基準値より少なく，うまく品質が制御できていることがわかる．しかし，外部設計では組織の基準値の 1.5 倍の欠陥を作り込んでおり，次回プロジェクトの改善課題となる．このようにシステム開発を開始する際，前回プロジェクトの結果を客観的に評価し，改善計画を立案することができるようになった．

(2) レビュー計画の策定

作込欠陥の削減を行うためには，設計，製造の後，できるだけ早く検証を行い，問題があれば再発防止策を実施する必要がある．しかし，フェーズの最後のほうでまとめてコードレビューや単体テストを行っているプロジェクトもあった．そこで，既存の文書管理システムの進捗管理機能にレビュー，テストの計画，実績の管理機能を追加し，欠陥の再発防止対策がより確実に実施できるようにした．この結果，レビューがより確実に実施できるようになり，リアルタイムに品質の状況が把握できるようになった．

(3) プロセスの監視と制御

図 14 は管理図の異常点に対して迅速に欠陥の再発防止策を実施してないプロジェクトの管理図のサンプルである．20 本中 5 本のプログラム（図中の赤丸）が異常値となっており，検出された欠陥の合計値は組織の基準値に比べて 1.5 倍であった．欠陥管理システム導入前はこのような状況であったと考えられる．一方，図 15 は，毎朝の連絡会等で再発防止策を行ったプロ

ジェクトの管理図である．6 番目に開発されたプログラム（図中の赤丸）は共通部品の利用法が適切でなく多くの欠陥を出しているが利用方法の指導を行うことで以降のプロクラムは品質が安定している．図 15 のように前半で異常値が発生し，後半安定するプロジェクトが多いことから，欠陥管理システムにより多くのプロジェクトで管理図による品質の制御が実施できるようになったと考えられる．

6. むすび

本論文では CMMI レベル 3 の組織がコスト増加を抑制しながら組織全体の品質改善を実施した事例を示した．管理図の作成ツールがあれば管理図の作成は難しくはない．しかし，開発現場でうまく活用するには単に異常判断だけではなく，再レビューを実施すべきかどうかの判断材料としても使える必要がある．成果物ごとの合計値を使った u 管理図を出力することでこの要求を満たすことができ，効率的に欠陥検出ができるようになった．更に構成管理機能を応用して簡単な入力で作込欠陥数が計算できるようにし，欠陥の発生状況がわかるようにした．早期欠陥検出や欠陥再発防止策の実施により成果物の修正工数が削減でき，結果的に増加した品質管理工数を相殺できている．以上の取り組みの結果，従来と同じ開発費で欠陥が 1/2 以下の品質の高いシステムが提供できるようになった．

今回の取り組みでは欠陥管理システムを構築し，外部設計，プログラム設計，プログラム開発の工程に対する統計的品質管理を実施してきた．しかし，外部仕様書の変更は設計上の欠陥と利用部門からの追加要求により実施されるが両者の境界が明確でなく，後者は利用部門の特性によりばらつきが大きいことがわかった．そのため組織全体として基準値を定めることができず，管理図による管理がうまく機能しなかった．今後改善する必要がある．

なお，当組織の一部のグループは品質予測モデルを構築 [17] し，2011 年 6 月に CMMI レベル 5 を達成している．管理図を活用した品質管理の仕組みが CMMI のモデルに整合していることが確認できている．今後，品質改善に取り組む組織の参考になれば幸いである．

文　献

[1] 経済産業省，“情報システムの信頼性向上に関するガイドライン第 2 版,” 経済産業省，http://www.meti.go.jp/committee/materials2/downloadfiles/g90722a07j.pdf, 2009.

[2] 日本情報システム・ユーザー協会 (JUAS), “ソフトウェアメトリックス調査 2012,” JUAS，東京，2012.

[3] M. Cusumano, A. MacCormack, C.F. Kemerer, and W. Crandall, “Software development worldwide: The state of the practice,” IEEE Software, pp.28–34, Nov./Dec. 2003.

[4] A.J. Albrecht, “Function point analysis,” Encyclopedia of Software Engineering, 1:518–524, 1994.

[5] 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)，“9.1 FP 生産性,” ソウトウェア開発データ白書 2012-2013，pp.284–286, 情報処理推進機構，東京，2012.

[6] メアリー・ベス・クリシス，マイク・コンラド，サンディ・シュラム，CMMI 標準教本第 2 版，日経 BP，東京，2009.

[7] 山田　茂，高橋宗雄，ソフトウェアマネジメントモデル入門-ソフトウェア品質の可視化と評価法-，共立出版，1993.

[8] 誉田直美，山田　茂，“高品質ソフトウェア開発を実現する品質会計技法,” プロジェクトマネジメント学会誌，vol.13, no.5, pp.9–14, 2011.

[9] 梶山昌之，合田英二，千野智子，“ソフトウェア開発プロジェクトの計数管理フレームワークによる定量的管理,” プロジェクトマネジメント学会誌，vol.13, no.5, pp.3–8, 2011.

[10] 日本規格協会，JIS ハンドブック 57 品質管理，pp.1100–1122, 日本規格協会，東京，2007.

[11] A. Florence, “CMM Level 4 Quantitative analysis and defect prevention,” CROSSTALK The Journal of Defense Software Engineering, Feb. 2001.

[12] G. Vijaya and S. Arumugam, “Monitoring the stability of the processes in defined level software companies using control charts with three sigma limits,” WSEAS Trans. Information Science and Applications archive, vol.7, Issue 9, pp.1200–1209, Sept. 2010.

[13] 相磯正司，湯浅耕季，鈴木圭一，“ソフトウェア開発におむる統計的プロジェクト管理手法の導入と実践,” FUJIFILM RESEARCH & DEVELOPMENT, no.56, pp.31–34, 2011.

[14] 繁本将憲，“ソースコード品質の俯瞰的評価技法：ソースコード品質評価への Quality Inspection 技法の適用,” プロジェクトマネジメント学会誌，vol.8, no.3, pp.26–31, 2006.

[15] S.H. Kan, “Metrics and Models in Software Quality Engineering,” Addison-Wesley Professional, 2002, (邦訳) 古山恒夫，富野　寿，“ソフトウェア品質工学の尺度とモデル,” pp.116–121, 構造計画研究所，2004.

[16] Western Electric Company, 統計的品質管理ハンドブック，住友電気工業株式会社訳，pp.225–226, 東京河北印刷，1961.

[17] N. Nakamura, S. Takatashi, S. Kusumoto, and K. Nakatsuka, “Approach to Introducing a Statistical Quality Control,” IWSM-MENSURA 2011, pp.297–301, Nara, Japan, Nov. 2011.

（平成 25 年 1 月 11 日受付，5 月 14 日再受付）

中村　伸裕

1988 京都工芸繊維大学・工芸卒．1988 住友電気工業株式会社入社．2011 大阪大学大学院入学，現在，企業内で利用するシステム開発の研究に従事．

高橋　　覚

1990 大阪大学・人間科学卒．1990 住友電気工業株式会社入社．現在，企業内で利用するシステム開発の研究に従事．

楠本　真二　（正員）

1988 大阪大学基礎工学部卒．1991 同大学大学院博士課程中退．同年同大学基礎工学部助手．1996 同講師．1999 同助教授．2002 同大学大学院情報科学研究科助教授．2005 同教授．博士（工学）．ソフトウェアの生産性や品質の定量的評価に関する研究に従事．情報処理学会，IEEE，IFPUG，PM 各会員．