SEC BOOKS

定量的品質予測のススメ

―ITシステム開発における品質予測の実践的アプローチ

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

Ohmsha

SEC BOOKS

定量的品質予測のススメ

―ITシステム開発における品質予測の実践的アプローチ

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

Ohmsha

刊行にあたって

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)は、大規模・複雑化し、かつ社会生活への影響力が高まっているソフトウェアの信頼性と生産性を向上させるために、2004 年 10 月 IPA に設立されました。

SEC では、産学官連携により広く収集・研究開発してきたソフトウェアエンジニアリング手法や人材育成等について、手法、標準、ガイドブック等の形で成果を取りまとめ、実証実験を行ってきました。これらの成果を広く活用してソフトウェアの信頼性を確保していただくことにより、「安心・安全」な社会生活を実現するための活動を展開しています。

顧客に対するサービスも企業内の業務も、今やソフトウェアなしでは実現できない時代です。「どんなソフトウェアシステムをつくるのか」に関して合意がなされた後に、開発工数や開発工期の見積りが行われます。工数・工期はソフトウェアシステムの開発コストや導入スケジュールに影響をもたらすため、精度の高い見積りを行うこと、そして、その妥当性を図ることは、きわめて重要です。そして、プロジェクトの設計・実装が始まると、データに基づく的確なプロジェクトマネジメントが必要になります。

そうした企業の課題に応えるため、SEC はこれまでに、20 社の協力をいただき、約 2000 件のソフトウェア開発プロジェクトデータを収集し、工数・工期、及び信頼性に関する分析を行ってきました。その成果は、「ソフトウェア開発データ白書」として 2005 年から発刊してきましたが、工数・工期(生産性・規模)に関する定量化に比べて、信頼性(品質)の定量化は十分とは言えない状況でした。すなわち、品質の定量化のためには、上記白書には収録されていない、プロジェクト実行中(インプロセス)のデータが必須となります。

そこで本書では、ソフトウェア品質の定量化に寄与することを狙いとして、インプロセスのデータを前提としたソフトウェアシステムの品質予測の具体的な方法、及びノウハウを紹介しています。本書の手法を社内のデータと有機的に結合することによって、SEC が、設立以来一貫して追究している科学的ソフトウェア開発マネジメントを実現されることを期待しています。

最後に多忙の中、積極的な執筆活動をしてくださった WG2 メンバ、レビューに

多大な協力をしていただいた定量データ部会の皆様、日本ユニシスの飯田志津夫氏、東洋大学の野中誠氏、パブリックコメントにコメントいただいた皆様、並びに SEC 研究員に謝意を表します。

2008 年 9 月

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

はじめに

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)では、ソフトウェア開発の相場観の提供を目的に、「ソフトウェア開発データ白書」を 2005 年から発刊しておりますが、生産性や規模に関する定量化と比べると、品質の定量化は十分とは言えない状況があります。

そこで本書は、「ソフトウェア開発データ白書」を補い、IT プロジェクトのシステム開発において実際に広く実施されているソフトウェアの品質予測の具体的な方法及びノウハウを紹介し、品質の定量化に寄与することを狙いとして発刊しました。

本書の内容には、品質予測の必要性や考え方、システム開発の各工程での品質予測のアプローチや事例が含まれ、企業での実践ノウハウを体系的に整理し、解説を行っています。そして、各工程ごとに参照できるよう、構成しています。

また ANNEX では、ソフトウェア測定プロセスの国際規格の紹介と、ソフトウェア信頼度成長モデルについて解説もしており、品質予測プロセスの構築・改善の基本情報として活用できます。

本書の想定する読者は、IT プロジェクトのシステム開発にかかわるプロジェクト・マネージャや品質管理担当者だけでなく、プロジェクトメンバを含みます。IT プロジェクトのシステム開発はユーザ、ベンダに限らず、関係者の協力なくしては進みません。品質も同様であり、本書が関係者間のコミュニケーションや協力関係の醸成の一助となれば幸いです。

最後に、ソフトウェア開発の品質予測は、単にプロダクトの品質予測のみが重要であるのではなく、プロセスの品質予測、プロジェクトの健全性予測も重要です。安定したシステムによる安心・安全・快適なシステム構築の早期品質予測と改善に、本書が貢献できることを切に望みます。

2008 年 9 月

定量データ分析部会　WG2　一同

Contents

Chapter 1

本書を手にとられた方へ

▶ 1.1　今、品質確保が急務

> プロジェクトを成功に導くためには、ユーザとベンダが信頼関係を持ちパートナーシップを築くことが重要である。プロジェクト遂行においては、ソフトウェアや開発プロセスの状態がタイムリーに把握でき、予測や制御を適切に行うことが求められる。この場合、ソフトウェア開発の関係者は、プロジェクトの状態を定量的に測定し、客観的な判断を下せることが不可欠となる。（『ソフトウェア開発データ白書 2007』，1 章，p.8，2007/8，IPA/SEC）

近年、IT の発展とユーザニーズの多様化等により、ソフトウェアプロダクトを含むシステム開発は、非常な勢いで高機能・複雑化しており、開発期間の制約が厳しい中で高品質の要求に応える必要が生じている。このようなシステム開発プロジェクトを成功に導くためには、ユーザとベンダがそれぞれ役割に応じた責任を担い、相互理解に立った信頼関係を保ちつつ、共にプロジェクトの推進に積極的にかかわっていける環境を築くことが重要となっている。

ソフトウェア開発においても、製造業における工業製品の開発と同様に、開発の工程でプロジェクトを管理するために QCD（Quality, Cost, and Delivery）に関する測定が一般的に行われている。しかしながら、ソフトウェア開発における「品質（Quality）」の測定は、「コスト（Cost）」「納期（Delivery）」の測定と様相が異なっている。「コスト」「納期」がそれぞれ目標値（予定値）と実績値の乖離度で定量的に把握し評価ができるのと異なり、「品質」に関しては、それ自身の状況を一意的に表し評価する標準的な尺度が見当たらない。最終的なソフトウェアプロダクトの品質は、プロダクト適用後の障害の発生件数や影響度で評価されることが多いが、開発途中の品質は、その組織において経験的に測り得た何種類かの品質関連指標を測定して評価されることが常であり、標準化された評価には至っていない。

とはいえ、ソフトウェア開発の現場では、品質をきちんと測定して客観的な判断を得るために代用特性を使って品質を確保しようと取り組んでいる実状がある。

▶ 1.2　品質予測の必要性

ソフトウェアの場合､外見から品質を判定することは困難である。高機能化･複雑化により大規模となったシステムの開発において、ソフトウェア品質の検証と妥当性確認の全てをテスト工程で行うことは、時間的にも費用面からも実現不可能である場合が多い。つまり、テスト工程以前の早い段階で品質を予測し、その都度作り込んでいく必要がある。

また、ソフトウェア開発の一部を外部組織に委託する開発の場合、受託者の設計・製造の過程がブラックボックスであり、受託者より提供されたプログラムを実際に動かしてみて初めてプロダクト品質が判明するような開発形態は、委託者に対し、容認しがたいプロジェクト失敗のリスクを抱えさせることになる。外部委託を用いたソフトウェア開発において、受託者より提供されたプログラムが期待通りに動かず、対策が長引いて納期に問題が生じ委託者が損失を被った事例は決して少なくない。委託者の組織が、設計・製造の過程での品質を把握するプロセスを有している場合でも、受託者の設計・製造の過程において、委託者がその品質を満足に把握し評価できることは稀かもしれない。なぜならば、組織が異なるとソフトウェア開発文化も異なり、開発のプロセスならびにパフォーマンスが異なるため、委託者の物差しで受託者の品質状況の判断ができなくなるためである。

ソフトウェア開発の設計・製造の適切な時点において、期待する最終プロダクト品質を得るために、プロジェクトを監視し制御する活動が強く求められている。設計・製造の段階で、その時点の品質状況を把握し、後工程での品質状況、さらにはプロジェクト完了時点の品質状況を窺い知ることができれば、ソフトウェア品質の確保に向けて、より効果的なプロジェクト制御が可能になるだろう。品質を予測して、適切でよりタイムリーなプロジェクト制御を共通の認識や尺度の下に実現する必要性が高じている。

一言で品質といっても様々である。ISO/IEC9126 が定めた信頼性、保守性、移植性、効率性、機能性、使用性(ユーザビリティ)の 6 つの品質特性とその副特性群や、機能特性／非機能特性の区分等が代表するように、ソフトウェア製品の品質は、ユーザの満足、環境への適合等も含め、幅広い分類に位置付けられる。さらに製品の品質にとどまらず、設計ドキュメント等の中間作業成果

物の品質、プロジェクト活動を計画通りに実施しているかを評価した品質等も、プロジェクトを成功させるために捕捉され管理されている。では、一体どのような品質を扱うことが、プロジェクトを監視し制御する活動に益するのであろうか。

▶ 1.3　本書の目的

本書では、品質予測について考えるこの活動に参加した企業が、実際に取り組んでいる方法を体系的に整理することで、多くのプロジェクトへの監視制御に利用可能な品質に関係する活動のあり方を提示する。機能性と一部の性能／信頼性等に関係する品質、これら品質の定量化と管理・予測への取り組みが、実際の開発現場で進められており、本書が扱う品質もその範囲である。あまねく適用できる「銀の弾丸」は望むべくもないが、本書を IT システム開発プロジェクトの適切な監視と制御、目指す品質の確保に役立てて欲しい。本書で述べる品質予測を、システム開発に臨んでいる組織にソフトウェア開発の設計・製造の段階で利用していただき、プロジェクト改善につなげていっていただくことを期待してやまない。

▶ 1.4　品質予測へのアプローチ

ソフトウェアの品質予測に向けて検討を進めるとき、我々はまず何に眼を向けるべきだろうか？　品質予測に臨む際には、まず、プロジェクトの実績データを収集し、分析し、品質予測のモデルを作成していること。そして、品質に関するプロジェクトの状態を定量的に測定し、客観的な判断を下せる状態にしていることが不可欠である。

特に昨今は、開発期間が短いプロジェクトが多く、品質確保が難しいという実状がある。

したがって、これらの方法で品質の改善箇所を早期に炙り出し、原因を分析し、対策を実施することが重要である。

本書では、まず品質予測の枠組みを第 2 章でとらえる。ソフトウェア開発現場で実施している品質予測の取り組み状況をふまえて、品質にかかわる測定の尺度

等を整理し品質予測のモデルを明らかにした上で、実務的な分析手法を洗い出す。これにより品質予測のための測定・分析のあるべき姿を俯瞰し、実際に測定することができる品質データと、そこにどのような分析手法があるかを明らかにする。

次に、品質予測の実際について第３章で考える。ソフトウェア開発の流れに視点をおき、品質測定ならびに品質予測の実際の活動を、ソフトウェア開発工程のどこで、どのような狙いをもって、どのように実施するのかを説明し事例を挙げる。品質予測のあり方に共通的な理解を得ることを目指したい。

▶ 1.5　本書が扱うテーマ

本書が扱うテーマは、ソフトウェア開発における「品質予測」である。

品質を議論する際、テスト実績から品質評価を行うことは当然であるが、テスト工程における品質予測以上に、要求分析や設計といった上流工程での品質予測が、より大きい予測効果を得るために大切であろう。本書では、品質予測を３つに分類している。

Ⅰ　要求分析・設計における品質予測
　設計工程（要件定義、基本設計、詳細設計）における設計時の品質予測
Ⅱ　プロダクトの品質予測
　製作やテスト工程（単体テスト、結合テスト、総合テスト）における品質予測
Ⅲ　プロジェクトの品質予測
　「良い品質のプロダクトは健全なプロジェクト運営から生まれる」という前提をもとにしたプロジェクトの品質予測

第２章「品質予測の考え方」では、品質予測の考え方をはじめとして、品質予測に必要となる測定項目、運用モデル、分析手法等を説明する。第３章「品質予測の実際」では、ソフトウェア開発の各工程で実際に行われる品質予測の活動を、事例を交えて説明する。なお、実際の開発現場で測定している品質データの状況は、『ソフトウェア開発データ白書 2007』（2007/8,IPA/SEC）を参照いただきたい。

Chapter 2

品質予測の考え方

第 2 章では、品質予測の考え方をはじめとして、品質予測の概念や考え方を説明する。

2.1 節では、品質予測の全体的な枠組みを示し、2.2 節では品質予測の基本事項を示す。2.3 節では、品質予測で用いるモデルと分析手法を示す。なお、詳細なモデルの適用事例は第 3 章で説明する。

▶ 2.1　品質予測の枠組み

品質予測とは、測定した品質データをもとにモデルを適用して現工程、次工程及び最終的な品質の傾向を分析、評価することである。

品質予測を行うために必要な作業はデータを蓄積し、分析・モデル化する部分と個々のプロジェクトで分析・予測する部分に分かれる。全体の流れを図表 2.1-1 に示す。

(1)【品質予測モデル作成、改善】

・データ蓄積：

　各プロジェクトの生産活動で測定される品質、規模、工数、プロジェクトプロフィール(業種、業務等)といった、様々なデータをプロジェクト横断的に蓄積する。

・分析・モデル化：

　蓄積された複数のプロジェクトデータをもとにデータ分析を行い、品質を予測するためのモデルを作成する。さらに、モデルを適用して導き出した予測値と実績値の乖離等の情報をフィードバックすることによりモデルの改善を図る。

(2)【特定プロジェクトの品質予測】

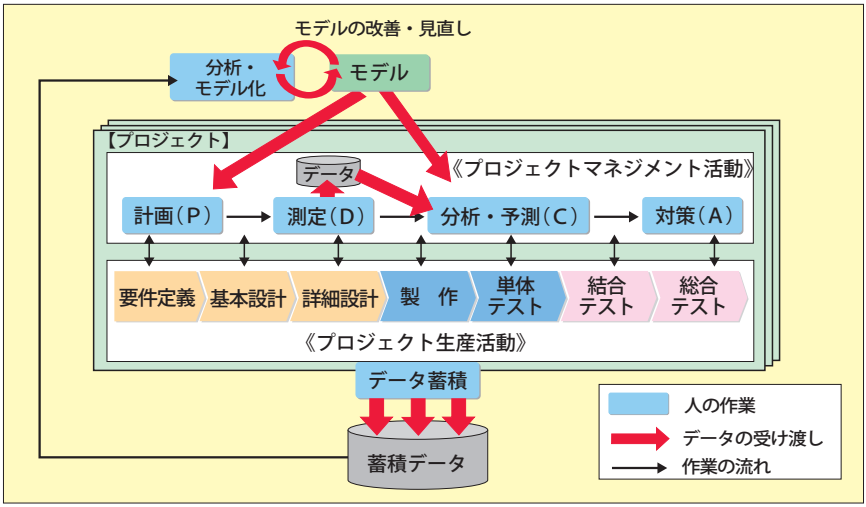
・測定：

　プロジェクト生産活動の各工程で品質を把握、予測するために欠陥数、規模、工数等を測定する。

・分析・予測：

　測定したデータに品質予測モデルを適用し、品質の把握、及び品質予測を

行う。その結果に基づいて、品質評価、及び対策を実施する。



図表 2.1-1 品質の測定と予測

▶ 2.2 品質測定の基本事項

▶▶ 2.2.1 測定単位（品質管理単位）

一般的にソフトウェアは設計段階で段階的に詳細化され、テスト段階では段階的に統合、組み上げられる。このとき、品質の善し悪しを把握したり、必要に応じて改善を実施したりする単位を品質管理単位と呼ぶ。この品質管理単位は通常はソフトウェアの構成単位と対応している。品質管理のための測定単位も基本的にこれと一致する。

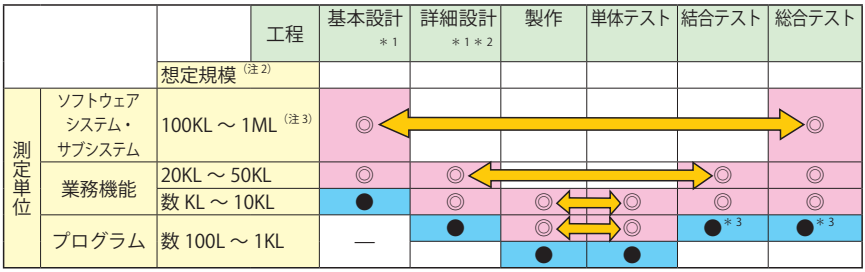
測定単位を細かくして品質データ（欠陥数等）を測定することにより、詳細な品質管理、分析を行うことができる。その測定値を集計することにより当該工程の品質管理単位での品質管理、分析が行える。ただし、測定単位を小さくすることにより測定に費やすコストも増加するため、最小測定単位の設定にはコストとのトレードオフを考慮する必要がある。

したがって、それぞれの工程に合った品質管理単位を選択することが必要である。

品質管理上の測定単位の例を図表 2.2-1 に示す。

また、隣り合う工程間(「基本設計工程と総合テスト工程」、「詳細設計工程と結合テスト工程」、「製作工程と単体テスト工程」)での品質把握、品質予測の連続性を確保するために、隣り合う工程間で同一の品質管理単位を含むように選択することが重要である(図表 2.2-1 の「◎」)。

なお、V 字モデル(注 1)の対応する工程同士の測定単位も合わせておくことが必要である(同じく図表 2.2-1 の「◎」)。



図表 2.2-1　測定単位例

● その工程完了時の最小の測定単位
◎ その工程で主に着目する品質管理単位
― 対象外
＊ 1 品質管理項目の測定に際しては、設計の各工程における最小の測定単位ごとに測定する
＊ 2 詳細設計において、プログラム分割以後はプログラムが品質管理の最小単位である
＊ 3「結合テスト」、「総合テスト」で発生した障害の対処を行う場合、欠陥が存在するプログラムの修正を行うことが多いため、最小測定単位を「プログラム」に設定している

(注 1) 開発対象をシステム構成要素に分解する過程(設計)と統合する過程(テスト)を対応させた表現を V 字モデルという
(注 2) ここでは、想定規模の例として LOC 数を示している
(注 3) 2.2-1 ②を参照

測定単位の階層は以下の通りに考えるとよい。なお、各測定単位の規模目安は本書で想定したものである。

①ソフトウェアシステム

品質管理の対象となるソフトウェアシステム全体を指す(以降、文脈上、混乱しない場合はソフトウェアシステムを単にシステムと呼ぶ)。

・マルチベンダシステム等の場合、顧客ビューと 1 ベンダビューではソフトウェアシステムの範囲が異なる場合がある。顧客にとって、n 社のベンダの担当部分全体をソフトウェアシステムと呼ぶ場合があるのに対して、ベ

ンダは担当部分をソフトウェアシステムと呼ぶ場合も少なくない。「ソフトウェアシステム」の範囲を明確にすると同時に、両者を混同しないよう区別して扱う必要がある。

②サブシステム

ソフトウェアシステムを業務の論理的なまとまり、または違いに基づいて複数に分割した単位をサブシステムと呼ぶ。例えば在庫管理サブシステム、顧客管理サブシステム、管理会計サブシステム等々。

- 1 サブシステムは、Lines Of Code（LOC）ベースで 200KL ～ 300KL 程度を目安とする（200KL ～ 300KL のシステムでは数 10KL 程度を 1 サブシステム、1ML を超えるシステムで 100KL ～ 200KL 程度を 1 サブシステムとすると扱いやすい）。

本書では、プログラムのコード行数の単位を LOC は「L」、KLOC は「KL」、MLOC は「ML」と表記する。

③業務機能

サブシステムを、業務的処理の論理的なまとまり、または違いに基づいて複数に分割した単位を業務機能と呼ぶ。例えば、在庫引当業務、棚卸業務、顧客登録業務、与信チェック業務、等々。

- 1 業務機能は数 KL ～ 10KL 程度を目安とする。
- 手順的なオンライン業務の場合、1 画面の入力から応答が返るまでを 1 業務機能ととらえるのに近い。また、バッチ業務の場合は、ジョブ（複数のジョブステップからなるひとまとまり）を 1 業務機能ととらえるのがよい。
- サブシステムが 100KL ～ 200KL 程度の場合は、業務機能に複数階層を設け、階層 1 では 1 業務機能を 20KL ～ 50KL 程度、階層 2 では数 KL ～ 10KL 程度を目安とする等の工夫が必要である。

④プログラム

ソースコードレベルの単位。

- 1 プログラムは実コーディングベースで 100L ～ 1KL 程度を目安とする（個人ごとに作成、メンテナンスする単位を想定）。なお、Java 等では、実行上のマシンインストラクションをほとんど生成しないインターフェースクラス等、数 10L 未満の小さなクラスの数が増加する傾向にある。したがって、意味のあるひとまとまりのクラス群で、ソースコード行数の合計が上記となるような単位をプログラムとすることを推奨する。

▶▶ 2.2.2　測定量

品質測定では、欠陥数を含む各種の欠陥の属性情報から生産量等を含む各種の情報を対象にする。しかし、測定対象を無闇に増やすことは測定コストを上昇させ、さらにその分析や予測のコストを増加させるので好ましくない。つまり、目的を絞り、それに必要なものだけを測定対象にして、コストを考慮した測定をすることが重要である。

ここでは、代表的な基本測定量 *1（単一の属性とそれを定量化するための方法とで定義した測定量）と導出測定量 *1（複数の基本測定量の値の関数として定義した測定量）の例を図表 2.2-2 に示す。

なお、図表 2.2-2 の「測定方法／算出方法」は例として記載している。

また、以下の基本測定量、導出測定量を導出するための記録項目、基本測定量の測定事例については巻末の ANNEX C に記載する。

対象工程	測定量	単位	測定方法／算出方法 例
全工程	規模	FP LOC	Function Point（FP）は測定手法、LOC は測定ルールを明確にする。
	作業工数	人時	
設計工程	レビュー回数	回数	
	レビュー工数	人時	Σ　各レビューアのレビュー実施時間
	レビュー対象規模	ページ数	レビュー対象ドキュメント量（A4 換算ページ数）
	レビュー指摘件数	件数	レビュー記録票の指摘事項数
	レビュー指摘密度	件数÷ FP, LOC 件数÷ページ数	レビュー指摘件数÷規模 レビュー指摘件数÷レビュー対象規模
	レビュー工数密度	人時÷ FP, LOC 人時÷ページ数	レビュー工数÷規模 レビュー工数÷レビュー対象規模
	レビュー指摘効率	件数÷人時	レビュー指摘件数÷レビュー工数
テスト工程	欠陥数	件数	障害連絡票の欠陥数
	欠陥密度	件数÷ FP, LOC	欠陥数÷規模
	テスト項目数	項目数	テスト仕様書の項目数
	テスト密度	項目数÷ FP, LOC	テスト項目数÷規模

図表 2.2-2　代表的な基本測定量と導出測定量

＊イタリック体は導出測定量

＊1「基本測定量」、「導出測定量」の詳細については、参考文献[1]「JIS X 0141：2004 ソフトウェア測定プロセス」を参照

コラム

「障害」と「欠陥」の違い

品質管理、予測を行う場合、「障害」、「バグ」、「不具合」といった用語を使用することが多い。しかし、これらの用語は、定義を明確にして利用していることは少なく、類似する別用語(障害、故障、不具合、バグ、エラー。英語ではfault、failure、defect、error 等)を使用したり、同じ用語でも別の意味で使われていることも多い。

これらの用語については各団体、協会で定義をしているが、それぞれ異なった意味で定義されていることも少なくない。

本書では、用語の意味を統一させるために、「IEEE Std 610.12」を参考にしているが、「欠陥」、「障害」については以下のように定義して、使用している。

・障害 (failure)

システムが期待されるサービス(service)を提供しなくなること。

欠陥の混入が原因となって、実行結果が仕様や要求等で定義された「期待される結果」と異なる現象として表出し、検知されたものを指す。障害は実行結果として観測できる現象であり、第三者としても観測可能である。

通常、我々が定義せずに使っている、「障害」、「トラブル」の概念に最も近い。障害と呼ぶ場合、ソフトウェア起因以外にハードウェアや人為的ミスに起因する場合も含むことが多い。

・欠陥(fault)

構成要素の異常。障害の原因。

技術者、プログラマの誤認や誤解等、作業者の認識にかかわる問題が原因となって、仕様、設計、プログラム等の成果物上の記述内容に現れた誤りや矛盾、表現上の問題を指す。欠陥は第三者にとって観測可能なものである。

通常、我々が定義せずに使っている「バグ」の概念に最も近い。

なお、JIS X 0014:1997　情報処理用語—信頼性、保守性及び可用性では、failure を故障、fault を障害としている。

コラム

尺度の定義

尺度とは、対象(データ)の特性を数値で表すための連続的または、離散的な値の集合である。

尺度は名義尺度 (nominal scale)、順序(序数)尺度(ordinal scale)、間隔(距離)尺度 (interval scale)、比(比例)尺度 (ratio scale)の 4 つに分類できる。

それぞれの説明を以下に示す。

・**名義尺度**とは、単に区分・分類するために用いられる尺度である。

対象に数値を割り当てるが、区分・分類するためだけに用いられる数値であり、数値の大小に意味はない。一般的な例では、血液型　1：A 型　2：B 型　3：O 型　4：AB 型　等があり、ソフトウェア開発の例では、障害連絡票の欠陥原因分類　1：アプリミス　2：環境ミス　3：データミス　4：オペレーションミス　5：既存障害　6：その他　等がある。

・**順序尺度**とは、数値の大小関係、順序のみに意味がある尺度である。

数値間の大小関係は存在するが、数値間の間隔や比率には意味がない。

一般的な例では、満足度調査等の　1：非常に満足　2：満足　3：普通　4：やや不満足　5：不満足　等があり、ソフトウェア開発での例では、障害連絡票の欠陥の重大度　1：重度　2：中度　3：軽度　等がある。

・**間隔尺度**とは、数値の差のみに意味がある尺度である。

対象に割り振られる数値は順序尺度の性質を全て満たし、尺度の単位当たりの間隔が尺度のどの位置でも等しくなる。

値の和差には意味があるが比率には意味がなく、尺度の絶対零点は存在しなくてよい(負の値も使える)。

例えば、摂氏での温度等がある(−10℃から 10℃に温度が上昇した場合、温度 20℃の上昇であるが、温度が 200％上昇したとは言わない)。

＊ソフトウェア開発の世界では間隔尺度に該当するものの例はあまり見当たらない

・**比尺度**とは、数値の差と数値の比率に意味がある尺度である。

間隔尺度の持つ順序情報と等間隔性に加えて、絶対零点が存在する。

一般的な例では、身長、体重、金額、等があり、ソフトウェア開発での例では、ソフトウェアの規模(FP、LOC)が比尺度の代表的例である。

名称	零点	順序	等間隔	連続性
名義尺度	×	×	×	×
順序尺度	×	○	×	×
間隔尺度	×	○	○	○
比尺度	○	○	○	○

各尺度の特徴

＊測定量とは測定の結果として値が割り当てられる変数である。

▶ 2.3　品質予測で用いるモデルと分析手法

▶▶ 2.3.1　品質予測のモデル整理

品質予測で利用するモデルは、予測する対象、目的によって異なる。例えば、「現状の品質から現工程完了の品質予測」、「現工程の品質から後工程の品質を予測」、「現工程の品質から最終成果物の品質予測」等があり、それぞれで使用する基本測定量、導出測定量も異なってくる。ここでは、品質予測で使用するモデルの概要、目的について説明する。

本書では、モデルを相互排他的な分類ではなく、実用性や現場での使用頻度を考慮して分類した。したがって、軸の尺度の取り方によっては別のモデルとみなすことができる場合もある(例えば、関数モデルのグラフによって分けられる領域に着目すればゾーンモデルとみなすことも可能である)。

(1) 閾値モデル

【概要】

・ある測定量に対する尺度の閾値(UCL（上部管理限界線 Upper Control Limit）／ LCL（下部管理限界線 Lower Control Limit)によって分類するモデル

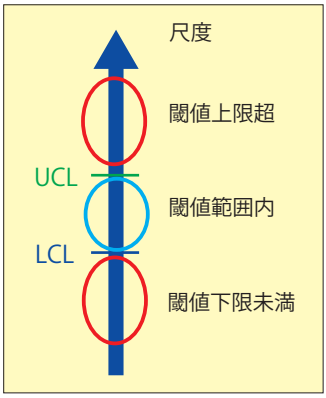
＊基準となる閾値は複数の基本測定量から導き出した導出測定量を用いることも可能

【用途】

・測定値と閾値(UCL ／ LCL)との比較から品質を予測する

【例】測定単位ごとの欠陥密度(縦軸：欠陥密度、横軸：測定単位)

＊閾値モデルは、後述のゾーンモデル、トレンドモデルの一種でもあるが、実際の現場で使用される頻度が高いため基本モデルの 1 つとして扱う。



図表 2.3-1　閾値モデル

(2) ゾーンモデル

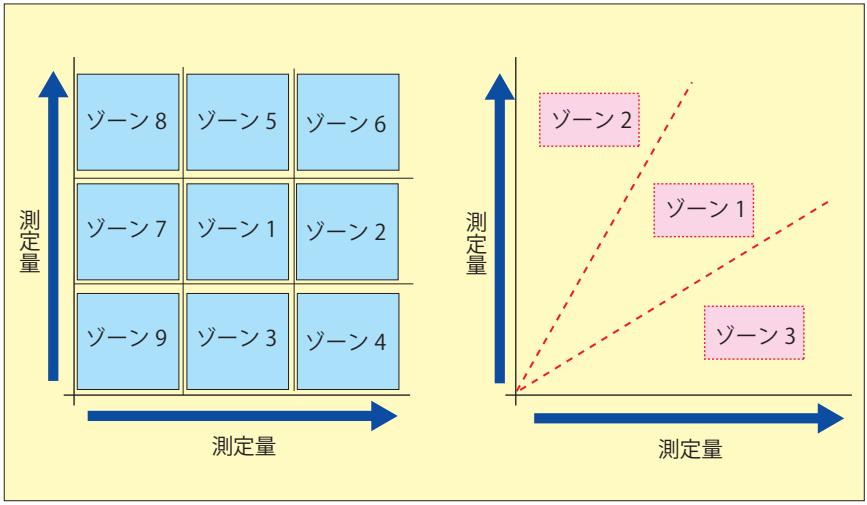
【概要】

・複数の測定量に対するそれぞれの組からなる空間をゾーンに分類するモデル

【用途】

・データから直接、特徴や傾向が見えない場合、次のステップの対策を絞り込みたい場合に測定値がどのゾーンに属するかで品質を予測する

【例】欠陥密度とテスト密度の組合せ
（縦軸：欠陥密度、横軸：テスト密度）



図表 2.3-2　ゾーンモデル

(3) 関数モデル

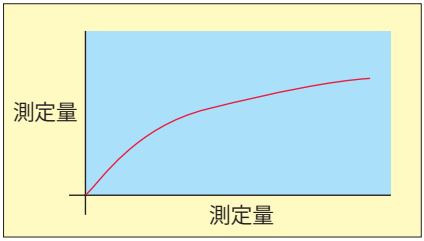
【概要】

・n 個の測定量の値の関係を統計的な回帰式（近似関数）で表すモデル

【用途】

・過去から現時点までの品質測定値から以降の品質を予測する

【例】欠陥数と日付の関係（縦軸：欠陥数、横軸：経過週）
（ソフトウェア信頼度成長曲線等＊以降、「信頼度成長曲線」と表記する）



図表 2.3-3　関数モデル

(4) トレンドモデル

【概要】

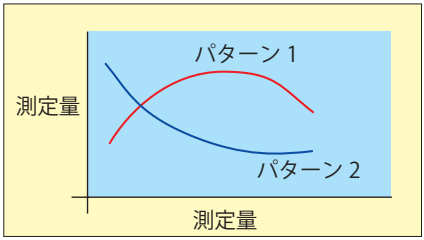
・ある測定量の時間的推移のパターンを分類するモデル

【用途】

・測定値がどのような推移のパターンを辿るかで品質を予測する

【例】最終品質の予測(縦軸：欠陥密度、
横軸：工程)

＊トレンドモデルは、X軸を経過時間にした場合の関数モデルの一種でもあるが、実用性を考慮して基本モデルの1つとして扱う。敢えて、両モデルの違いを説明すると、関数モデルが「関数」という式に着目するモデルであるのに対して、トレンドモデルは傾向のパターンに着目しているモデルであると言える。



図表 2.3-4 トレンドモデル

(5) チェックリスト

【概要】

・有識者のノウハウをあらかじめリスト化するモデル

【用途】

・チェック状況から、品質(プロジェクトを含む)を予測する

【例】レビュー指摘チェックリスト、リスクチェックリスト

大分類	小分類	内容	評価	備考
全体	網羅性	・・・・・・・・・・・・	■	
		・・・・・・・・・・・	□	
		・・・・・・・・・・・・	■	
		・・・・・・・・・・・・	□	

図表 2.3-5 チェックリスト

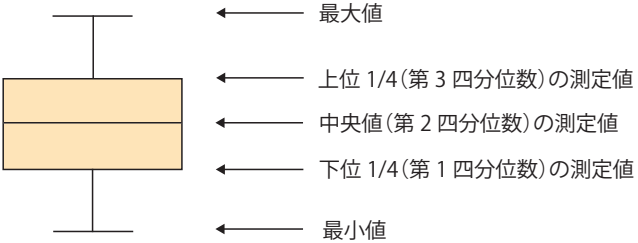
コラム

モデル化時の注意点

工業製品等で品質のモデル化を行う場合、測定値は分散(σ^2)が小さい(分布の山が急峻な)正規分布となることが多いため、管理限界に$\pm 3\sigma$を用いて分析を行うケースが多い(管理幅を$\pm 3\sigma$に設定するということは全体の 99.7%が管理幅に含まれるため、管理幅外にはみ出す値(不良品)は 0.3%しか存在しない)。これに対して、ソフトウェア開発では複数の要因(特に人的要因)が大きく影響するため、測定値に大きなバラツキが出る(すなわち分布の山がなだらかな)傾向があり、しかも正規分布にならないことが多い。したがって、実際の現場では管理限界$\pm 3\sigma$では範囲が広過ぎて利用できないことが多い。そこで例えば、四分位点を利用して下限値(第 1 四分位点の値)と上限値(第 3 四分位点の値)を決める等の方法が使われる。四分位点とは、測定値を昇順または降順に並べたとき、全体を四分割する 3 箇所の境界点のことである。測定値が昇順である場合、下から 25%番目を第 1 四分位点、50%番目を第 2 四分位点(この値が中央値)、75%番目(つまり上から 25%番目)を第 3 四分位点と呼ぶ。この方法では、中央値の$\pm$ 25%幅を管理幅としていることになる(経験的には$\pm$ 20%～$\pm$ 30%幅、場合によっては$\pm$ 50%幅に設定する)。

ただし、管理限界を極端に狭くすると管理幅外にプロットされるデータが多く発生し、それぞれの分析に費やすコストも多くなるため、管理限界を設定する時は品質とコストのトレードオフを考慮して設定する必要がある。

また、分析に影響を及ぼす程の外れ値が管理限界内に存在する場合は、単に分析対象外にするのではなく、データ内容を調査して分析対象の可否判断をする等の考慮が必要である。



四分位点の説明(箱ひげ図)

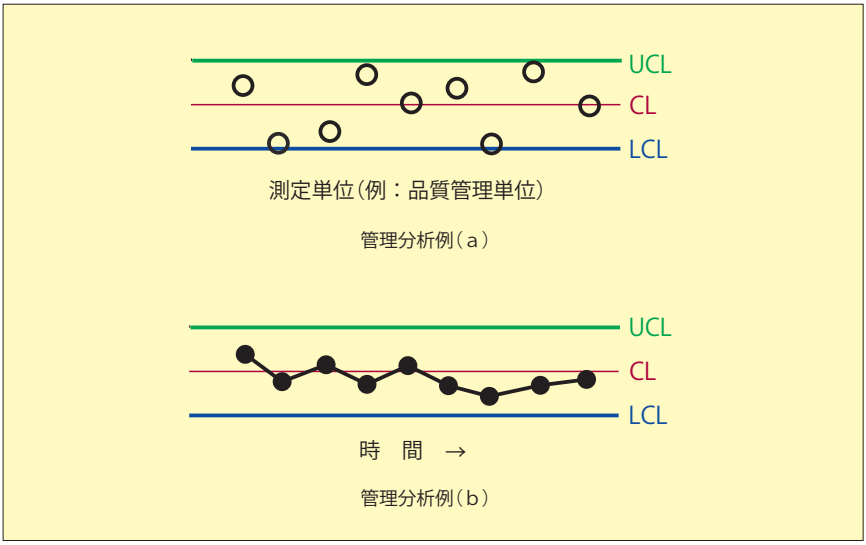
▶▶ 2.3.2　管理図分析の使用例

管理図分析とは、閾値モデルを使用して、データの分布が UCL と LCL に対してどの位置にプロットされるかを見て、データが正常値であるか外れ値であるかを判断する分析方法である。

通常は、図表 2.3-6 (a)のように横軸に測定単位(品質管理単位等)を取り、CL(中心線 Center Line)と外れ値の UCL と LCL の 3 本の線を引き、その上にデータをプロットする。例えば、UCL と LCL は SEC のソフトウェア開発データ白書では中央値からの± 25%に設定している。

また、図表 2.3-6 (b)のように横軸に時間を取り、1 つの指標に対する特定の測定単位のデータの傾向性を見る場合にも利用される(例えば、プログラムの欠陥数等)。

データの分布が UCL と LCL の中にあり、中央値の線に対して、片方向に多く出現する等の傾向性がないかを確認する。



図表 2.3-6　管理分析例

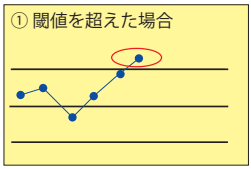
コラム

管理図の見かた

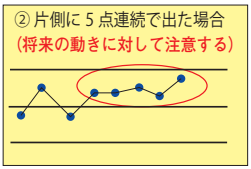
管理図で、実績値が以下の傾向を示す場合は、品質不良の恐れがある。傾向が見える対象の状況の調査・分析を行い、品質に関する問題の有無を判断する必要がある。

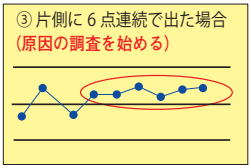
（1）　閾値を超えた場合
（2）　中心線の片側に連続して実績値が現れる場合（７点以上）
（3）　実績値が中心線に対して、片側に多く出る場合（８割以上）
（4）　実績値が連続して上昇、または下降する傾向がある場合

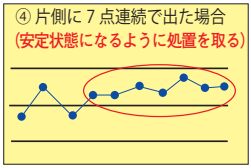
(1) 閾値を超えた場合



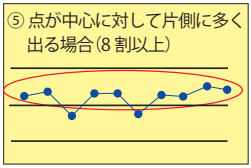
(2) 連がある場合 (7 点以上)
管理図の中心線の片側に連続して点が現れることを連といいます。



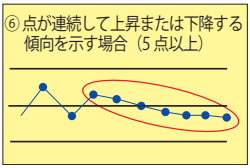




(3) 点が中心に対して片側に多く出る場合 (8 割以上)



(4) 点が連続して上昇または下降する傾向を示す場合 (5 点以上)



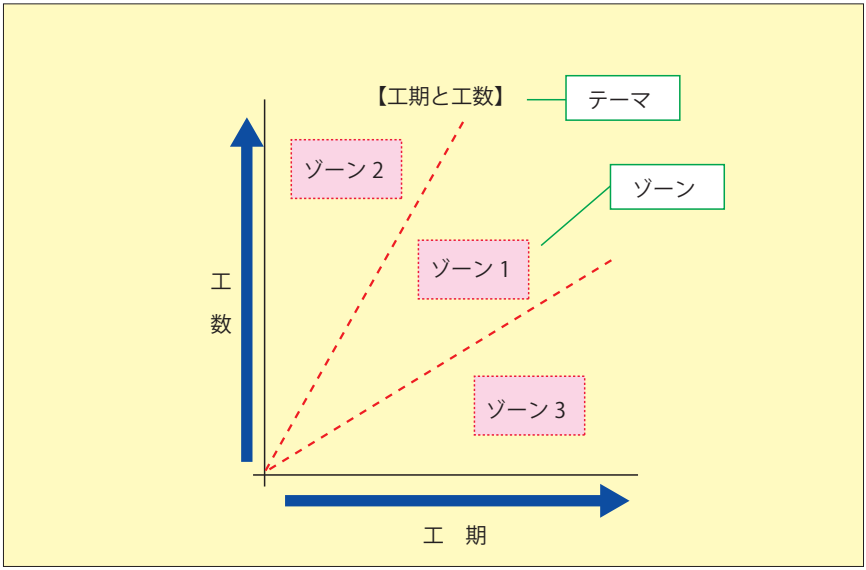
出典：「わかりやすい品質管理(改訂版)」(稲本 稔著、理工学社　1991 年) 参考文献[26]

▶▶ 2.3.3　ゾーン分析の使用例

ゾーン分析とは、与えられた分析のテーマをある特徴に着目した視点によってゾーンに分割し、ゾーンごとに分析を行うものである。この概略を図表 2.3-7 に示す。この図表 2.3-7 は縦軸と横軸の指標によって、データの分布をいくつかのゾーンに分割している。

ゾーンに分割することにより、全体では見えにくかったテーマの特徴が見やすくなり、全体に行う施策と比較してゾーンごとに行う施策はきめ細かく対応できるようになる。また、どのゾーンにも属さないデータについてヒアリングすることで、理由を明確にして、処理をどのようにするかを決定する。

例えば、図表 2.3-7 に、工期と工数の関係をテーマとしてゾーン分析を行う方法を示す。縦軸に工数を取り、横軸を工期とする。これによって、例えば、工数が大きく工期が短いゾーン、工数が小さく工期が長いゾーン、工数、工期とも適正なゾーンに分割できる。これらのゾーンに入るデータをプロジェクト単位で分析し、施策を与える。例えば、工数が大きく工期が短いゾーンに入るプロジェクトの業種別やアーキテクチャ別等の特徴を分析し、それに対応する施策を与えることができる。



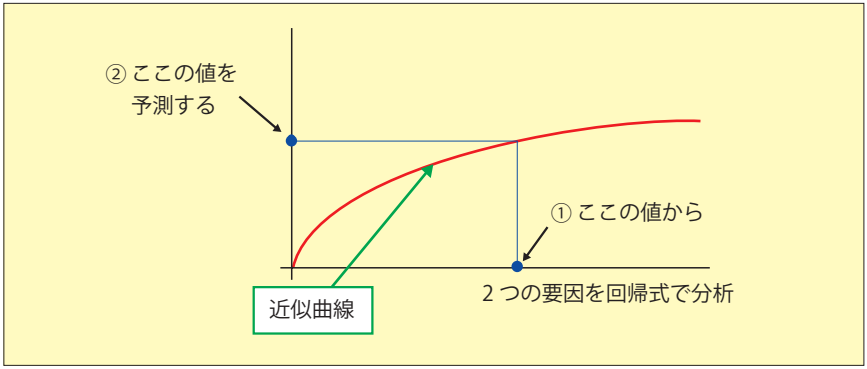
図表 2.3-7　ゾーン分析例

▶▶ 2.3.4　回帰分析の使用例

回帰分析とは、関数モデルを使用して、2 つのデータ列の関係を回帰式と呼ぶ近似曲線で代替することで予測を行う分析である。図表 2.3-8 は、横軸と縦軸のデータ列の分布を近似曲線で表現している。

この回帰分析では、データの分布を数式で表現し、その数式を曲線で近似することでデータの分布を明確にすることができる。また、データの範囲にない箇所でもデータの出現を予測することができる。

例えば、欠陥密度と期間の対のデータから近似曲線(回帰式)を見つけ、その曲線を欠陥密度と期間の関係として分析する。また、あるプロジェクトのデータがこの近似曲線から離れたものになれば、そのプロジェクトに対して、原因をヒアリングする等の対応を行う。



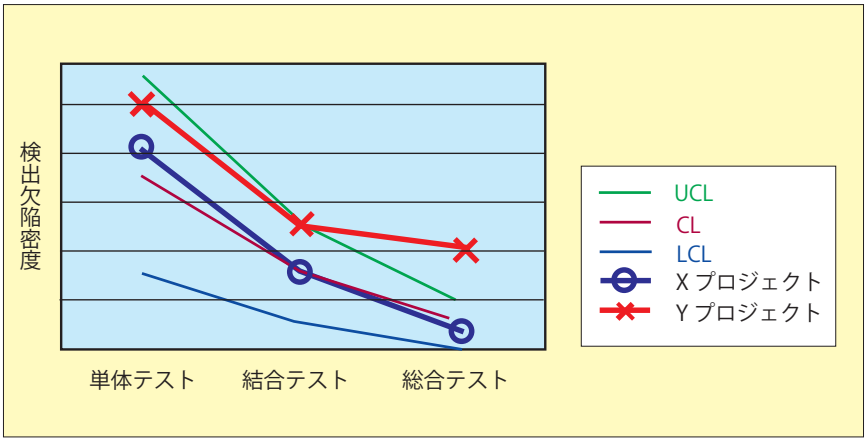
図表 2.3-8　回帰分析例

▶▶ 2.3.5 トレンド分析の使用例

トレンド分析とは、過去のプロジェクトの実績データの時間的なパターンと、現在のプロジェクトの実績データのトレンドを比較し、過去のプロジェクトの最終品質と同等な結果となるかを予測する分析である。

例えば、品質の良いプロジェクトの欠陥密度のデータを収集し、そのデータを統計処理し、テスト工程のトレンドモデルを作成する(単体テスト・結合テスト・総合テストと UCL・CL・LCL のトレンドのモデル化)。このトレンドモデルにプロジェクトの実績データをプロットすることにより、X プロジェクトは管理限界の中に入って推移しているので最終品質は良いと判断される。逆に Y プロジェクトでは、結合テスト時点で単体テストから結合テストにおいて

UCL を超える傾向があり品質が懸念されることが判断される。また総合テスト時点では、結合テストから総合テストにおいて UCL を逸脱してしまっており、品質に問題があると判断できる(図表 2.3-9)。



図表 2.3-9　トレンド分析例

▶▶ 2.3.6　チェックリスト分析の使用例

チェックリストは、与えられたテーマに対してチェックする項目をリストにしたものである。これは、今までに述べてきた分析手法とは異なり、プロジェクトやプロダクトが異常になっていないかを判断するリスク対策の手法の 1 つである(図表 2.3-10)。

チェックリストで、チェックする項目を明示することで、プロジェクト管理やプロダクト管理に対して、一般的な注意が喚起される。

例えば、品質についてのチェックリストには、規模に比較してテスト工数が少なすぎないか等が入る。また、チェックリストに重み付けを追加することによりポイント化することができ、その合計ポイントは閾値モデルに適用することができる。

要求分析のレビュー指摘チェックリスト						
大分類	小分類	レビュー指摘事項	評価	重み	ポイント	備考
全体	完全性	記載内容の範囲についての記述があり、明確か	○	A	1.2	
		要求の網羅性について記載があるか	○	B	1.0	
		要求に漏れがないかの確認をしているか	×	A	0.0	
	無矛盾性	内容に矛盾はないか	○	A	1.2	
		要求の粒度は揃っているか	×	B	0.0	
	非曖昧性	主語が明確であるか	○	C	0.8	
		事実と推測が分離しているか	○	B	1.0	
		数値表現できるところは数値で表現しているか	○	A	1.2	
					計 6.4	

図表 2.3-10　チェックリストの例

＊評価(○：1、×：0)、重み(A：1.2、B：1.0、C：0.8)

＊ポイント＝評価×重み

▶▶ 2.3.7　モデル利用上の注意点

上記分析モデルのうち、特に管理図分析や曲線近似分析を利用するときの注意点を下記に挙げる。

(1) 管理限界の設定

モデル化に際しては、まず管理限界を明確にする必要がある。ここでいう管理限界とは管理図分析であれば UCL や LCL であり、曲線近似分析であれば与えられた独立変数に対して従属変数がとり得る上位及び下位の統計的な信頼限界である。これらの管理限界はこれまで組織で蓄積したデータをもとに、同一条件のプロジェクトのデータから統計的に算出すべきである。例えば、同じ部門、同じ開発内容、同じ開発要員で開発を行う場合(パッケージ開発等)は、類似プロジェクトの実績が蓄積されていることが多く、管理限界が明確な値で定義しやすいと考えられる。

しかし、もし十分な蓄積データがない場合は、①類似のプロジェクトの集合から類推する、②既存の品質管理ツールで設定されている値を用いる、③品質管理担当者やプロジェクト管理者の経験から暫定値を設定する等の方法でとりあえずの値を設定し、組織内のデータの蓄積と共に値を修正することが現実的であろう。

(2) 入力データのチェック

モデルを適用しようとする入力データは、可能な限りその妥当性をチェックすることが望ましい。特に、管理限界を大きく超える「特異な」データが現れた

場合は、そのデータを提出したプロジェクトにヒアリングし、データ作成までに何らかの誤りがなかったか、あるいは例外的な条件が含まれていないかどうかを確認することが必須である。もし、例外的な条件が含まれている場合は、現在利用しようとしているモデルによる管理が妥当かどうかを再検討する必要がある。

なお、「特異な」データであるかどうかの一般的な判断基準は存在しない。例えば、母集団が正規分布に従う場合、$\pm 3\sigma$ をわずかに超えたデータは管理限界を超えたデータではあってもモデルの適用対象外のデータとは言えないことが多い。それに対して、$\pm 6\sigma$ を超えるものは明らかに特異なデータとみなすことができるであろう。

(3) アクションの策定

管理限界を超えたデータを示すプロジェクトに対して、どのようなアクションを起こすか、ということを考えるべきである。対象となるプロジェクトは、あくまで統計的に算出した管理限界を超えたというだけであり、(2)で述べたような「特異な」プロジェクトではない。

アクションの策定にあたっては、設定した管理限界が管理のしやすさを考慮したものであって、あくまでも統計的な指標すなわち目安でしかないことに注意する必要がある。管理限界の範囲内であれば問題はなく、管理限界の範囲外であれば問題があると断定できるわけではない。

アクション策定の方針例としては、次のものが考えられる。

① 現物の再点検によって、なぜ管理限界を超えたか論理的に説明でき、かつ点検によって品質上の問題がないのであれば、とりあえず良しとして以降経過を注意する。

② 現物の再点検によって、なぜ管理限界を超えたか論理的に説明がつかないならば、担当者にヒアリングする等、さらに点検範囲を広げて原因を追究する。

③ 原因が判明し、品質上の問題があるならば対策(品質改善策)を実施する。

④ 原因がわからない場合は、管理限界を再点検する。

(4) 蓄積データベースへのデータの追加

(1)で述べたような適正な管理限界を設定するためには、信頼できる蓄積データが必要である。したがって特別の理由がない限り、モデルを利用したプロジェクトのデータも蓄積データベースに追加することが望ましく、これが次回の管理限界の再設定に役立つ。

Chapter 3

品質予測の実際

第 3 章では、ソフトウェア開発で実際に行われる品質予測の活動を、事例を交えて説明する。3.1 では要求分析や設計で多用されるレビューを中心にした品質予測の方法を示し、3.2 ではプロダクトの品質を確認するテスト系の品質予測を説明し、3.3 でプロジェクトの品質予測について紹介する。

以下に紹介する測定項目や分析を全て行うのではなく、効果に見合った測定・分析コストだけをかけるようにしていただきたい。

▶ 3.1　要求分析・設計における品質予測

ここでは要求分析と設計時における品質予測を紹介する。要求分析と設計時には、レビュー結果による品質予測を行う。その目的やアプローチ、測定項目、予測方法や注意点等について触れる。

▶▶ 3.1.1　目的と狙い

要求分析・設計の品質予測は、以下の目的を達成するために実施する。

(1) 現工程の品質予測に基づく品質改善施策の立案と実施

要求分析や設計時に各種のレビューを行い、その結果から、現工程におけるプロダクトの品質を予測する。さらにこの予測をもとに現工程において、改善や修復するための施策を立案し実施する。

(2) 後工程の品質予測に基づく品質改善施策の立案と実施

レビュー結果から後工程のプロダクト品質を予測して、品質を確保するために計画を見直す等の品質改善施策を立案し実施する。

なお、品質の予測は数値の評価に加え、欠陥の種別等の現象や原因も加味し、評価する。

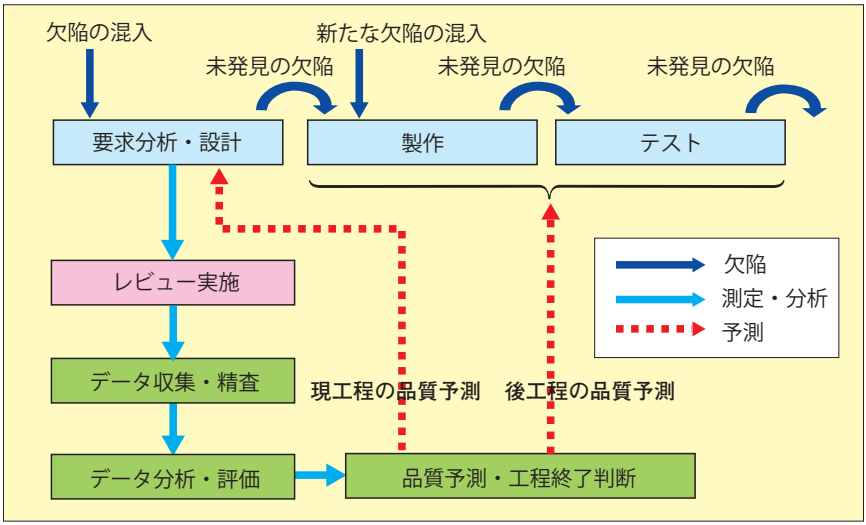
(1)、(2)が要求分析・設計の品質予測の目的であるが、プログラムを実行することができるテストの品質予測と比較して、これらの目的を達成するのはより困難である。これは、レビューはテストとは異なり人間的要素が強く、測

定基準を設定したとしても、測定データに曖昧性が多く、バラツキが大きいことが原因である。つまり要求分析・設計の品質予測では、このような人間的要素を、レビューアの質を揃えることや、チェックリストを活用すること等でぶれの少ない数値に変換することがポイントになる。

レビューには上記で述べたような性質があるため、レビューの対象となる要求分析や設計工程におけるプロダクト品質だけでなく、レビューの実施方法(レビュープロセス)の妥当性が問題となり、その評価も合わせて行う必要がある。

▶▶ 3.1.2 アプローチ

図表 3.1-1 に要求分析・設計の品質測定と予測の流れを示す。



図表 3.1-1 要求分析・設計の品質測定と予測の流れ

図表 3.1-1 に示すように、レビューから得られるデータの分析と評価を行うことにより、要求分析や設計時に混入された欠陥、レビューによって排除できなかった欠陥を推定する。つまり、レビュー結果のデータによって要求分析・設計の品質予測を行うと共に、後工程における欠陥の混入を予測する。

以下に要求分析・設計の品質予測のアプローチを示す。

(1) レビュー実施

レビューの目的は、要求分析・設計の網羅性と内容の適切性の確認と参加者

の情報の共有であり、第２の目的として、参加者のスキル向上、レビュー対象とその工程の承認等がある。

ここでは会議形式で実施するレビューの実施例を示す。レビュー会議は責任者や司会、レビューを受けるメンバ、レビューア等数人のメンバが一堂に集まって、レビュー対象のプロダクトに対して、インスペクション、ウォークスルー等の手法を用いて議論をする会議である。

レビューはレビューアの人間的要素が強いため、レビューの平準化を図るために、レビューの目的に応じたチェックリストを用意することが望ましい。チェックリストは、レビューを行うときに定常的にチェックする事項と、特定の内容をチェックする事項とを分類して作成する。またチェックリストの各項目に重み付け(点数付け)を行うことで、レビュー指摘内容の定量化を行うこともできる。

ただし、チェックリストに記載されていることを表面的に確認する形式的なレビューや、チェックリストの項目の見直しが定期的に実施されず古いままのチェックリストを使用するレビューではレビューが形骸化する危険がある。レビューの形骸化を防止するためには、資質と経験に裏付けられた能力の高いレビューアを参加させてレビューを実施し、かつチェックリストの項目を見直すことを推奨する。

(2) データ精査

レビューデータ(レビュー指摘の重要度や件数等)は、レビューアの主観による評価が大きく影響し、レビュー結果がレビューアによって異なることがある。また、レビュー指摘の記録は、分類等が不十分で精査されていないデータとなっている場合が多い。例えば、１つのレビュー指摘で複数の事を指摘している場合や、指摘内容もその影響度等により、指摘の軽重があるが、その軽重も基準があっても主観的な評価になっていることが多い。このため、レビュー指摘等のデータを精査して、品質予測に使えるようにすることが必要となる。

データを精査する方法として、レビュー指摘内容のうち単なる文言の誤り(誤字脱字)を分離し、本質的指摘にフォーカスする方法や、仕様の抜け等のレビュー指摘種別の影響度(重大度)をランク分けし、データの重み付けをする方法がある。

定量的品質予測を行うためには、まずレビュー指摘の情報を定性的なものから定量化することが必要になる(定量化の方法は 3.1.3)。例えば、レビュー指摘の重要度等の測定で主観的評価を排除するために、レビュー評価のガイドラインを設ける。

コラム

IT プロジェクトのシステム開発における レビュー実施の注意点

(1) 発注者と受注者のレビュー実施の注意点

- 発注者、受注者双方が説明責任を果たすことが、システム開発において品質を確保する重要なポイントである。よって、発注者は受注者との役割分担を明確にし、プロジェクト(レビュー等)に積極的に参画する。
- レビュー結果は、発注者と受注者の間で齟齬が起きないように、合意プロセスと承認ルールを明確にし、それに基づいて行動する。
- 発注者は、ステークホルダの合意認識(レビュー回答の承認等)を自らの仕事と心得る。
- 要件定義は発注者の責任であるので、要件定義のレビューは、発注者と業務部門と IT 部門が、協力して進める。

※経営者が参画する要求品質の確保 (参考文献[24])より

(2) レビュー会議での注意点

- レビューを単なる説明会にしないように、レビュー方針を設定し、それに対する問題意識を持って行う。例えば、レビューの目的に対応したチェックリストを用意して、レビューの観点に抜けがないようにレビューを実施する方法もある。
- レビューの参加者や責任者、司会、レビューアのキーマンを決め、役割分担を明確にする。
- レビューを効率よく行うために、レビュー対象の資料をあらかじめ配布し、参加者は事前にチェックする。
- レビュー指摘を記録し、解決状況等を管理する。
- レビュー対象に近すぎないメンバを入れ第三者の視点でレビューする。レビューが何度も繰り返される場合は、新しい視点を持つレビューアを加えることも効果的である。
- 上記を含めた、レビューに関する記録を保管する。これにより後工程で参照することができる。また類似のプロジェクトも参照ができるようになる。

ちなみに、文章のわかりやすさは影響度が低いと考えられがちだが、オフショア開発を利用する場合等においては大きな問題になることがあり、プロジェクトの特性を考慮したガイドライン作成が重要である。

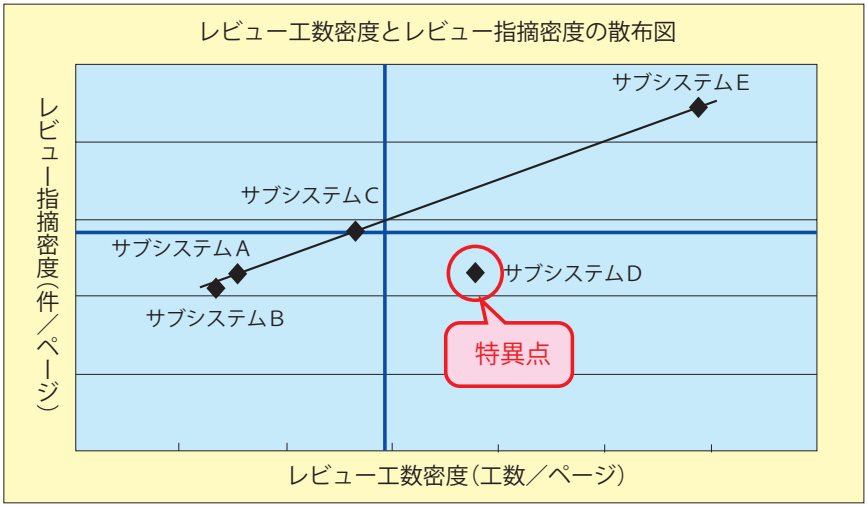
これまで述べてきたデータ精査の注意点をまとめると以下のようになる。

・データの抜けや記述誤り(記述データの桁違い等)を点検し、不適切なものはヒアリング等を行い見直す。
・レビュー指摘やその解決状況を定量化する。
・主観的評価を排除し、測定データをレビューアに依存しない客観的なデータにする。

(3) データ分析

精査したレビューデータを 2 章で述べた閾値モデルやトレンド分析、ゾーン分析、回帰分析等で分析する。データ分析では、指摘対象の欠陥が混入した工程と発見したレビュー時の工程との距離等の数値化や、レビュー指摘内容の各種情報から、レビュー指摘のデータを分析する。レビュー指摘は件数だけでなく、レビュー指摘種別や重大度等で調整する方法もある。

例えば、レビュー工数密度とレビュー指摘密度の散布図の傾向から回帰分析を用い、特異点があるかを分析する(図表 3.1-2)。特異点があるときは、これについてヒアリングを行い、原因を明確にする必要がある。



図表 3.1-2　レビューデータの分析例

具体的なデータ分析方法を以下に示す。

(a) 分析結果をもとに品質の傾向性を読み取る

2 章で述べた分析手法を用い、データの傾向性を読み取る。例えば、レビュー指摘の種別や対象とするサブシステムの特質に合わせて、それを領域に分けてゾーン分析を行う。レビュー指摘がどのレビュー指摘種別に多く出現しているか、どのサブシステムに多く出現しているか等のゾーンごとの傾向性を読み取る。

なお、データの傾向性は、レビュー対象やプロジェクトに最適な分析手法を用いる。例えば、経過に意味があるときは近似曲線や管理図を使う。2 軸間の相関に特定パターンの傾向がある場合はゾーン分析を使う。

(b) さらに読み取った傾向性が妥当であるかを別のデータを使って確認する

(a)で読み取った品質傾向が妥当であるかを、別のデータを使って確認する。例えば、レビュー指摘件数が通常より少ない傾向性の場合、レビュー工数や参加者のスキル、レビュー指摘内容を確認して、レビュー指摘件数が適切であるかを判断する。レビュー工数やレビューアのスキルが十分満足できるのに、レビュー指摘件数が少なかった場合には、そのプロダクトの欠陥が少なかったと判断できる。

(c) 確認された傾向性をもとに品質分析を行う

品質分析を効率よく行うために方向付けを行う。例えば、ある品質管理単位での測定対象が、他の測定対象と比べ、品質の傾向性に大きく差がある場合は、その測定対象を分離して品質分析を行う等の方針を設定する。

(4) 品質予測と工程終了判断

データ分析結果から現工程や後工程の品質やコストを予測する。さらに最終成果物の品質予測を行う。この予測をもとに後工程の計画見直し等、プロジェクト運営に役立てる。

例えば、全体のレビュー指摘密度に注目して、それが一定の範囲にないときは、現工程の品質確保ができていない傾向性が出ていると判断する。これが後工程の製作工程に悪影響を及ぼすと予測し、後工程の計画を見直す等の手段を講じる。

以下に品質予測と予測に基づいた対応の実施の注意点を挙げる。

- 予測が、計画の達成を阻害すると判断される場合は、その原因を追究する。対応は原因が判明してから検討する。ただし、原因が判明せずに作業が一方的に進むことを防止するために、状況によっては暫定的な対処

も必要である。

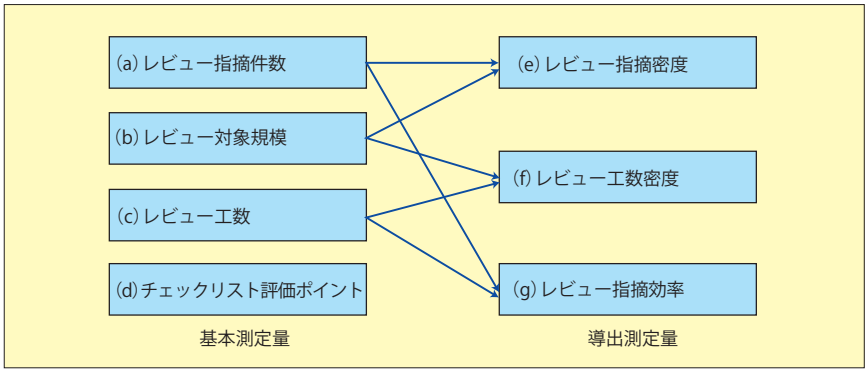
・予測に基づいた対応は、コストに見合った効果のある対応を立案する。

▶▶ 3.1.3　方法（項目、手法）

要求分析、設計工程時の品質測定とその予測の方法について説明する。最初にレビューアによる主観評価をどのように客観的な評価に変換するかを測定項目の面から述べる。

(1) 分析と測定項目

分析する項目とそのための測定項目には以下のものがある。



図表 3.1-3　品質の測定項目の関係

品質測定の項目は、基本測定量と導出測定量があり、矢印は基本測定量から導出測定量が導出されることを意味する。単に件数の多い少ないだけでは、規模によって多くて良い場合、少なくて良い場合があるので、レビュー対象規模もしくはレビュー工数当たりの件数の密度で評価することに意味がある。

ここでは全てを測定し分析することを勧めているのではなく、測定コストを意識して、前述した目的やプロジェクトに応じて、適切な項目のみを収集する。ただし、評価の妥当性を検証するためには、複数の測定項目を活用し相互に検証する必要がある。

以下に基本測定量と導出測定量の各測定項目について説明する。

(a) レビュー指摘件数（基本測定量）

レビュー指摘件数とはレビューで指摘された欠陥の件数である。

レビュー指摘件数は、レビュープロセスとレビュー対象の分析を容易に行え

るように、誤り分類(誤字脱字、欠陥等の分類)で集計できるよう識別しておくことが望ましい。

(b) レビュー対象規模(基本測定量)

レビュー対象規模とは、レビュー対象を表す規模である。「レビュー対象物」には、開発コスト、システム規模、ドキュメント等があり、それぞれに適した測定項目(¥、FP:Function Point、LOC:Lines Of Code、UP:Usecase Point、文書のページ数、文字数等)がある。

(c) レビュー工数(基本測定量)

レビュー工数は、レビューアのレビューに要した時間である。

$$\text{レビュー工数} = \sum \text{各レビューアのレビュー実施時間}$$

有識者以外(育成等を目的とした要員)のレビューアのレビュー工数は、レビュー工数から除外する、または適切な係数で補正することが望ましい。

(d) チェックリスト評価ポイント(基本測定量)

チェックリストによる測定は、手順通りに行われているか、または手順に抜けがないかを確認するために使われる場合や、定性的な情報を各チェックに点数を付けることで定量化する場合がある。それをチェックリスト評価ポイントという。

要求分析や設計工程で必要となる手続きが手順通りになされているかを確認するために、あらかじめ用意されているチェックリストに基づいて、その合否をチェックする。チェックリストの例を図表 3.1-4 に示す。ただし、これによる品質測定や予測は、チェックリストに記載されている内容に大きく依存する。

要求分析のレビュー指摘チェックリスト

大分類	小分類	レビュー指摘事項	評価	重み	ポイント	備考
全体	完全性	記載内容の範囲についての記述があり、明確か	○	A	1.2	
		要求の網羅性について記載があるか	○	B	1.0	
		要求に漏れがないかの確認をしているか	×	A	0.0	
	無矛盾性	内容に矛盾はないか	○	A	1.2	
		要求の粒度は揃っているか	×	B	0.0	
	非曖昧性	主語が明確であるか	○	C	0.8	
		事実と推測が分離しているか	○	B	1.0	
		数値表現できるところは数値で表現しているか	○	A	1.2	
					計 6.4	

図表 3.1-4　チェックリストの例
＊評価(○:1、×:0)、重み(A:1.2、B:1.0、C:0.8)
＊ポイント=評価×重み

(e) レビュー指摘密度(導出測定量)

(a)のレビュー指摘件数を(b)のレビュー対象規模で正規化した導出指標である。

レビュー指摘密度 = レビュー指摘件数 ÷ レビュー対象規模

これは、レビュー対象規模当たりのレビュー指摘件数であり、要求分析や設計時のプロダクト品質を表す数値として利用する。レビュー対象規模に対して、指摘されている件数が許容範囲にあるかを確認する。

(f) レビュー工数密度(導出測定量)

(c)のレビュー工数を(b)のレビュー対象規模で正規化した密度である。レビュー対象規模当たりにどれだけのレビュー工数をかけているかの数値になる。

レビュー工数密度 = レビュー工数 ÷ レビュー対象規模

これは、技術が新規または新規の対象分野であること等、要求分析や設計の困難さが大きいと判断したとき、十分に規模当たりのレビューが行われているかをチェックするために使用する。

(g) レビュー指摘効率(レビューパフォーマンス)(導出測定量)

(a)のレビュー指摘件数を(c)のレビュー工数で正規化した密度である。これは工数当たりのレビュー指摘能力を表す。

レビュー指摘効率 = レビュー指摘件数 ÷ レビュー工数

(2) 測定・分析・予測

(1)で測定した品質項目をどのように使って、現工程のプロセスやプロダクトの評価や後工程の予測を行うかを説明する。

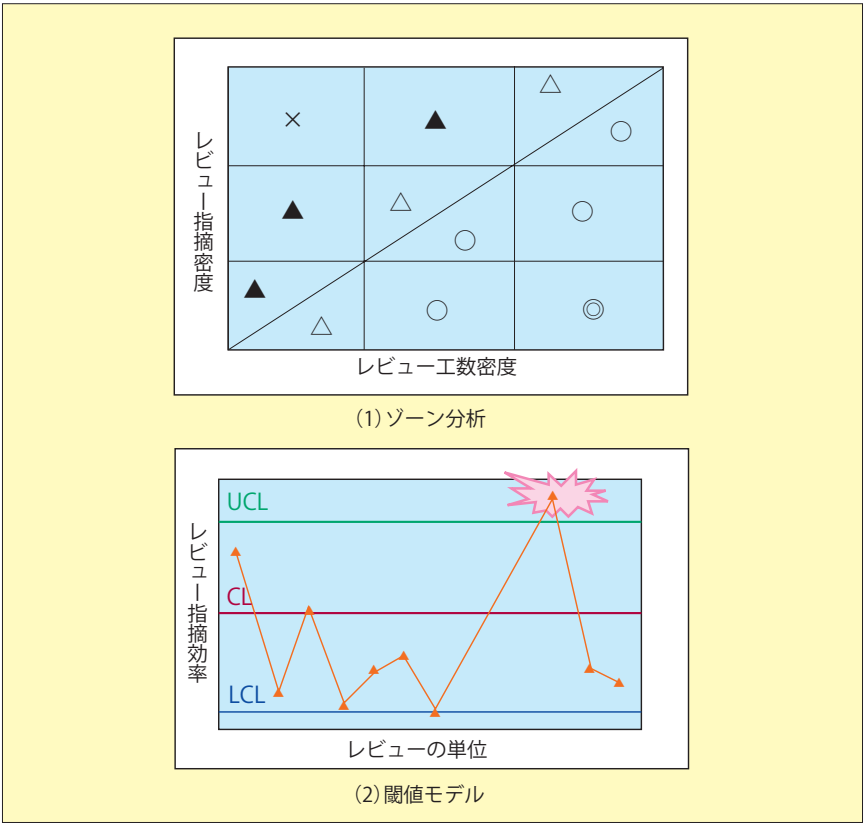
まず、最初に第 2 章で説明したモデルを選択し、それに応じた分析手法を用いて分析する。この分析結果から、データの有意差等を見て予測する。

図表 3.1-5 にゾーン分析や閾値モデルによる典型的な分析方法を示す。ゾーン分析ではレビュー工数密度を横軸にし、レビュー指摘密度を縦軸にするグラフを作成する。グラフの各領域をある特徴に着目した視点によってゾーン分割する。そしてゾーンごとにデータの傾向性を読み取り、レビュー時の品質施策を立案し実施する。例えば、レビュー工数密度が小さいにもかかわらずレビュー指摘密度が高いときは品質が悪いと分析・予測して、品質向上のために品質

施策を立案し実施する。他のゾーンも対応する施策を実施する。

ゾーン分析は多次元の情報を使うが、これを一次元に還元して、一度に分析する方法として、閾値モデルによる分析・予測がある。例ではレビュー指摘密度とレビュー工数密度から求めるレビュー指摘効率の値を一次元に還元した数値であるレビュー指摘効率をもとに分析を行う。レビュー指摘効率が一定の範囲にあるときにはレビューは正常に動作していると予測できるが、閾値の範囲外にあるときには、何らかの欠陥がレビュー対象またはレビューにあるとして、該当レビュー対象やレビュープロセスの再調査を行う。

さらにサブシステムの特質を領域に分けてゾーン分析する方法や、値の変動を分析するために管理図分析を行う方法がある。そしてレビュー指摘の種別に注目して、上記の分析を行うことがある。



図表 3.1-5　典型的なレビュー分析

具体的な手順を以下に述べる。

(a) 測定項目の一覧表をもとにした測定とデータの精査

測定項目の一覧表を作り、これをもとに測定する。

例えば、レビューアの人数や参加時間、レビュー指摘件数とその指摘種別、原因工程、発見工程、原因種別等の測定項目と、レビュー対象規模を求めるためにドキュメントページ数や開発予定規模の一覧表を作成する。この測定項目の一覧表をもとに測定を実施する。また、測定したデータを精査するために、データの抜けや記述誤り(桁違い等)を点検する。

(b) 測定データの一次元化と正規化

複数の測定項目から意味のありそうな項目を集計(一次元化)する。例えば、以下の式によりレビュー指摘の量を計算することができる。

$$\text{レビュー指摘量} = \sum(\text{レビュー指摘の重大度} \times \text{レビュー指摘対象の混入の工程と発見の工程との距離による係数})$$

レビュー指摘の重大度は、大きな仕様抜けや誤り等、後の工程での影響が大きいものを重大度が大きいとし、その係数を大きくする。またレビュー指摘対象が混入した工程とそれを発見した工程との距離が大きければ、修正するコストがかかると推測できるので、この掛け算でレビュー指摘量を計算する。

さらに分析しやすいようにデータの正規化を行う。例えば、レビュー指摘密度のように、ある単位当たりにデータを加工し分析しやすくする。

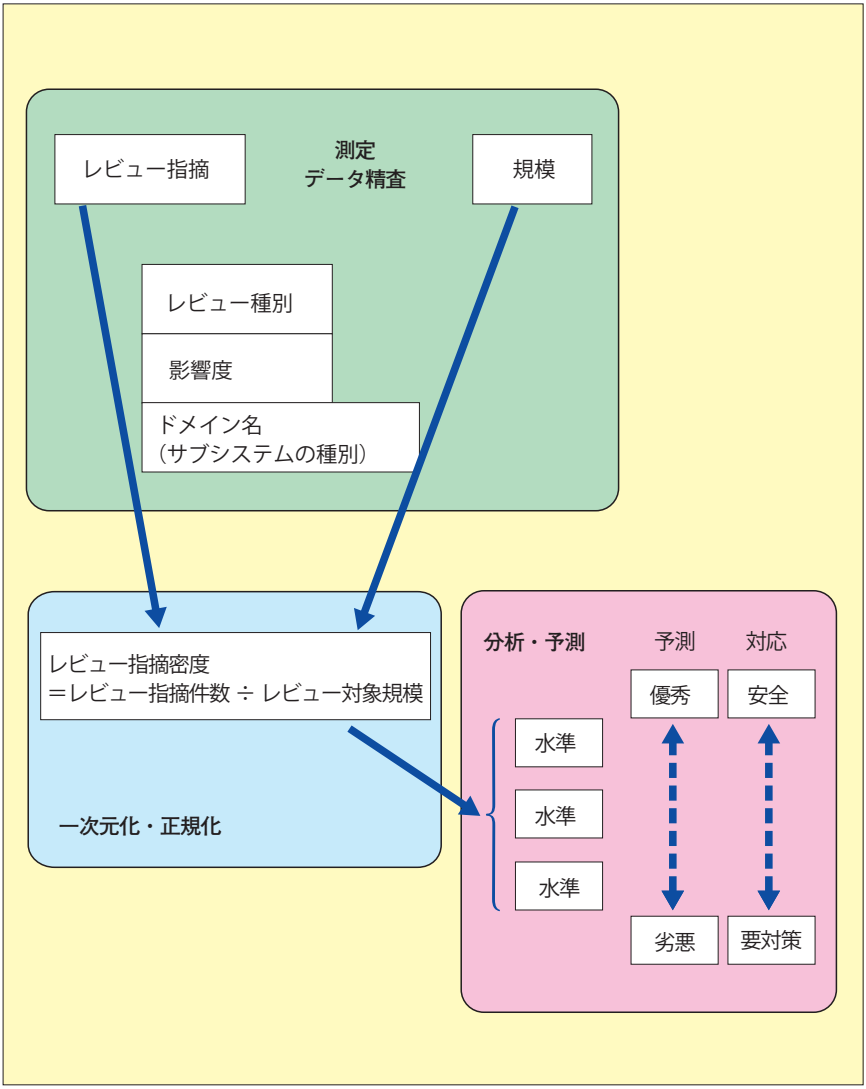
(c) 分析・予測の実施

(a)の測定データや(b)の一次元化・正規化した数値をもとに分析・予測を行う。しかしこれらの数値だけではレビュー対象の品質が良いか悪いかの判断は難しい。そこでこの数値に対して、いくつかの閾値を設け、閾値によって予測の指針となるべきものをゾーンごとに定義する。そして数値と過去の品質データとの比較、サブシステムごとの有意差をもとに、品質の善し悪しを予測する。例えば、レビュー指摘密度は過去の品質データから3件／KL以上では注意すべき品質であると定義する。この基準と数値とを比較し品質の善し悪しを予測する。

ここでは測定や精査、予測について、(a) ～ (c)の典型的な例を中心にして述べてきた。この例だけでなく、最初に紹介したように各種のゾーン分析やトレンド分析、管理図法等を用いて分析し、予測することもできる。しかし、レ

ビューはテストと異なり、関係者のスキルによってバラツキが大きくなるため、測定データの精査が非常に重要である。

これまで述べてきたレビューの測定・データ精査、一次元化・正規化、分析・予測の流れを図表 3.1-6 に示す。



図表 3.1-6　典型的なレビュー測定から品質予測まで

コラム

仕様変更による影響度合いの定量化

IT システム開発において、上流工程における要件の定義、要件の確定は重要な成功要因となっており、要件の曖昧さ等による仕様変更がプロジェクトの納期遅延、コスト増加につながっている。そのことは、「経営者が参画する要求品質の確保」（参考文献[24]）で十分に語られている。

しかし、その「要件の確定度合い」や「要件の曖昧さ」を定量的に把握することは非常に困難な作業である。そこで、本書では少し視点を変えて、仕様変更がシステム開発にどのくらい影響を与えるかを定量的に把握することで間接的に要件定義の品質をとらえる考え方と測定量の例を紹介する。

【目的】

仕様変更の発見工程、混入工程、重大度の係数から算出されるこの導出測定量で要件定義の品質を間接的に把握する。この導出測定量は設計工程でレビュー密度等の指標と同様に利用することができる。例えば、この導出測定量が目標値を超えた場合には、要件定義をやり直す等の判断に利用する。

【基本測定量と導出測定量】

この測定量は、仕様変更が発見された工程と混入工程の距離と仕様変更の重大度を掛けることにより、手戻りの「長さ」と「範囲（幅）」を考慮した測定量を算出する。発生した仕様変更ごとの測定量の総和を対象となるシステムの規模で割ることにより相対的な指標を算出する。本書では、この導出測定量を「仕様変更密度」と定義する。

(1) 仕様変更の発見工程

発見された工程の係数は以下のように設定する。

要件定義：0、基本設計：1、詳細設計：2、製作：3、単体テスト：4、結合テスト：5、総合テスト（ベンダ）：6、総合テスト（ユーザ）：7、運用：8

(2) 仕様変更の混入工程

混入工程の係数は以下のように設定する。

要件定義：0、基本設計：1、詳細設計：2、製作：3、単体テスト：4、結合テスト：5、総合テスト（ベンダ）：6、総合テスト（ユーザ）：7、運用：8

＊本コラム内では、混入工程は常に「要件定義：0」となるが、便宜上、全ての工程の重みを記載している

(3) 仕様変更の重大度

仕様変更内容の重大度の係数は以下のように設定する。

大：5、中：3、小：1

(4) 規模

仕様変更の対象となるシステムの FP 数，LOC 数。

(5) 仕様変更密度

$$\left(\sum_{i=0}^{n}(\text{発生工程の係数}i - \text{混入工程の係数}i) \times \text{重大度}i\right) \div \text{規模}$$

n は対象の件数

＊ここで示した重み(仕様変更の重大・影響度合 大: 5 等)は参考値であり、利用される場合は、各組織、プロジェクトの特性に応じて重みを決めることが望ましい。

＊ここでの「仕様変更」とは、顧客要求仕様確定後の変更、追加

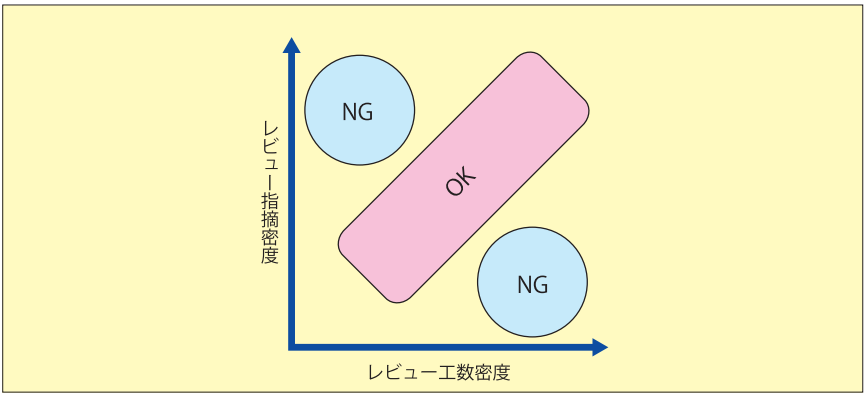
▶▶ 3.1.4　品質評価・予測の適用領域と分析指針

ここでは、要求分析・設計の品質評価・予測の適用領域と分析の観点と対応指針について述べる。

(1) プロダクトの品質の予測(現工程、後工程)

プロダクトの品質の予測・評価は、以下の指針で判断する。

- レビュー工数密度に対し、レビュー指摘密度が大きすぎるときは、プロダクトの品質が悪いと予測し、指摘内容を分析する。
- レビュー工数密度に対し、レビュー指摘密度が小さすぎるときは、プロダクトの品質が良い、またはレビューが不適切と予測し、実情を調査する。

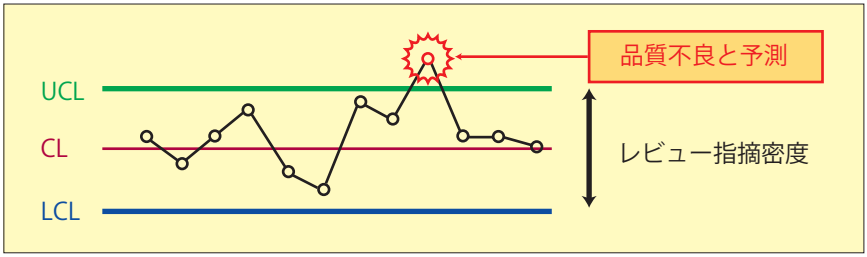


図表 3.1-7　プロダクト品質の予測・評価の指針

レビュー指摘密度やレビュー指摘効率を使って、プロダクトの品質を予測する。プロダクト品質予測から、以下の対応を検討する。

- プロダクトの品質が一定の水準にあることを検証する。
- プロダクトの品質が一定の水準に満たない場合、プロダクトの品質改善のための改善策を検討する。
- プロダクト品質を考慮した、納期遅延やコスト増大等のプロジェクトのリスクマネジメントを行う。

例えば、ドキュメントのレビュー指摘密度を管理図で管理した場合、管理限界(UCL または LCL)を逸脱したドキュメントの品質に問題があると予測する(図表 3.1-8)。問題があると予測したドキュメントに対しては、そのドキュメントの指摘内容を分析する。分析内容に問題がある場合は是正処置を行う。管理限界は組織的に統計処理された値を使用してもよいが、事業分野で蓄積した値やプロジェクトの実績値を使用した方が精度の向上が期待できる。プロジェクト内の実績値を基に管理限界を設定する方法は、最初のドキュメントのレビュー実績(3 回以上)を使用して標準偏差を算出し、管理限界を設定する。

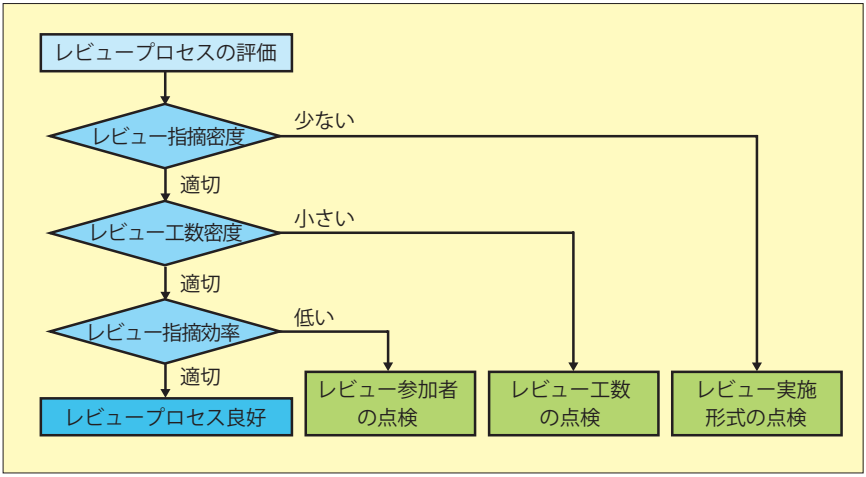


図表 3.1- 8　レビュー指摘密度の管理図

(2) レビュープロセスの評価(現工程)

レビュー指摘件数、レビュー工数密度、レビュー指摘効率を使って、以下の観点でレビュープロセスの評価を行う。

- レビュー指摘密度が低いときは、レビューが形式的に行われている可能性がある。
- レビュー工数密度が低いときは、適切な時間をかけてレビューが行われていない可能性がある。
- レビュー指摘効率が低いときは、レビューアの能力、もしくはレビューアの体制が適切でない可能性がある。



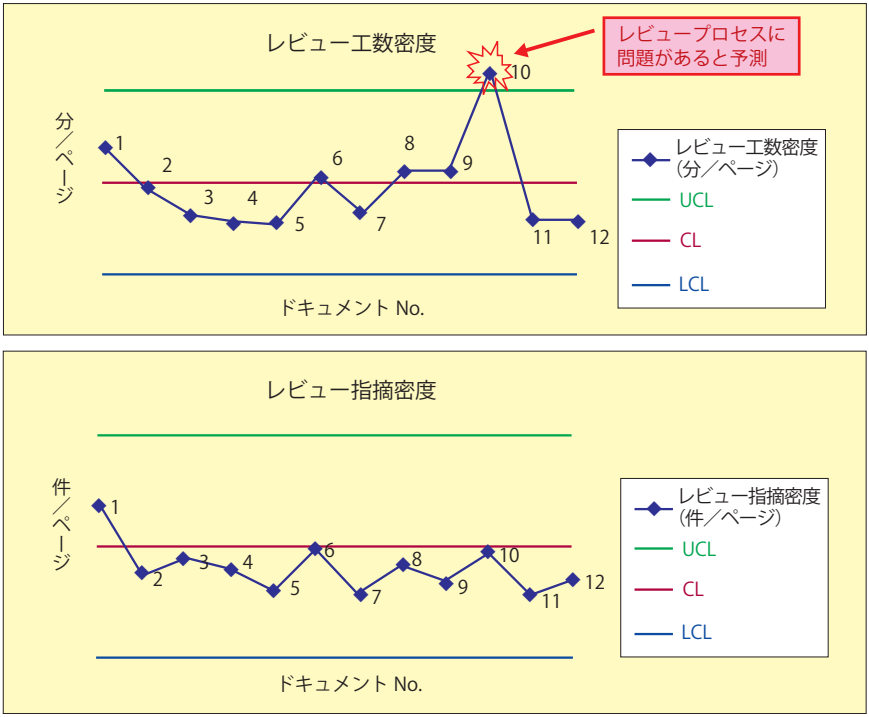
図表 3.1-9　レビュープロセスの評価フロー

レビューが不適切と思われる場合は、レビュー指摘内容の合理性(誤字脱字や質問、誤解が多くないか)の点検、レビュー工数の妥当性、レビュー参加者(レビューア)の適切性等の分析を行う。分析結果に問題がある場合は、レビュープロセスを見直し再度レビューを行うことが望ましい。

例えば、ドキュメントのレビュー工数密度とレビュー指摘密度を管理図で管理した場合、レビュー指摘密度は管理限界(UCL または LCL)内で収まっていても、レビュー工数密度が管理限界を逸脱したドキュメントのレビュープロセスに問題があると予測する(図表 3.1-10)。問題のありそうなレビュープロセスは、レビュー対象のボリュームに見合ったレビューを実施しているのか？ 必要なメンバがレビューに参加しているのか？ 等の調査を行う。調査内容に問題がある場合は是正処置を行う。プロダクトの品質の予測と同様、管理限界は組織的に統計処理された値を使用してもよいが、事業分野で蓄積した値やプロジェクトの実績値を使用した方が精度の向上が期待できる。プロジェクト内の実績値をもとに管理限界を設定する方法は、最初のドキュメントのレビュー実績(3 回以上)を使用して標準偏差を算出し、管理限界を設定する。

(3) 潜在誤り予測(後工程)

信頼度成長曲線やゾーン評価等により、プロダクトの潜在誤りを予測する。フィールド品質目標の達成が危ういと予測した場合は、品質確保の観点から追

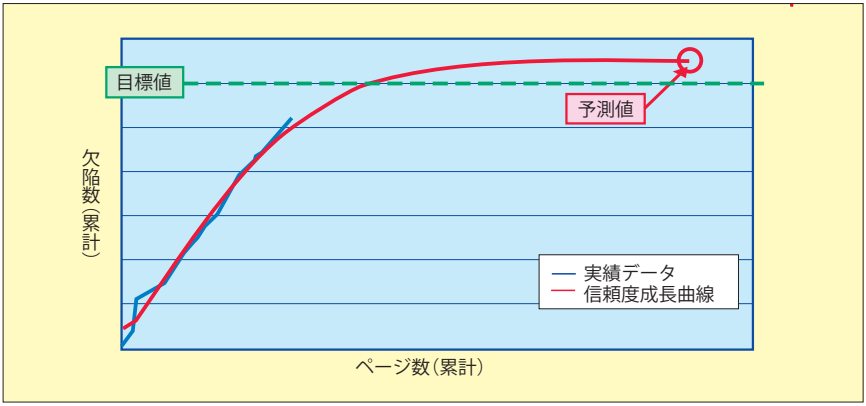


図表 3.1-10　レビュー指摘密度の管理図

加 QA の立案・実施や、最終品質確保のための品質確保の要員強化や品質改善施策の追加等、プロジェクト計画の見直しを行う。

組織的に統計処理したデータやモデルの活用は有効であるが、自部門や事業分野ごとのデータを蓄積し、統計処理やモデル化したものを活用した方が予測精度は向上する。

例えば、信頼度成長曲線を活用してドキュメントの欠陥数を予測する。横軸にドキュメントのレビュー実施ページ数、縦軸に欠陥数として信頼度成長曲線モデルから最終欠陥数(予測欠陥数)を予測する(図表 3.1-11)。予測欠陥数が摘出目標欠陥数を下回るようであれば、その予測欠陥数を目標欠陥数に追い込むため、レビュー工数の追加や、レビューアの質の向上等の改善施策を計画する。



図表 3.1-11　ドキュメントの信頼度成長曲線

▶▶ 3.1.5　要求分析・設計の品質予測のまとめ

(1) 要求分析・設計の品質予測の注意点

今まで述べてきた、品質予測を行うための実務的な注意点をまとめる。

レビュー工程	注意点
レビュー実施時	• レビューを単なる説明会にしないように、レビュー方針を設定し、それに対する問題意識を持って行う。 • レビューの参加者や責任者、司会、レビューアのキーマンを決め、役割分担を明確にする。 • レビューを効率よく行うために、レビュー対象の資料をあらかじめ配布し、参加者は事前にチェックする。 • レビュー指摘を記録し、解決状況等を管理する。 • レビュー対象に近すぎないメンバを入れ第三者の視点でレビューする。レビューが何度も繰り返される場合は、新しい視点を持つレビューアを加えることも効果的である。
データ精査時	• データの抜けや記述誤り（桁違い等）を点検し、不適切なものはヒアリング等を行い見直す。 • レビュー指摘やその解決状況を定量化する。 • 主観的評価を排除し、測定データをレビューアに依存しない客観的なデータにする。
データ分析時	• 値が許容範囲内にあるときでも、プロダクト品質が保証されているとは言えない（レビュー指摘内容の分析が前提）。 • 統計的方法を用いて機械的に特異点を判断する。特異点は調査し原因を追究し対処する。 • 閾値は、過去またはサンプリングの品質データに基づき設定する。 • 分析結果の妥当性を確認するために、別の観点から多角的に分析結果の妥当性を検証する。
予測時	• 実測値の傾向が過去の品質傾向と異なる場合は、その原因を調査・分析する。 • 予測結果は、有識者（同様のシステムの経験者や品質のスペシャリスト等）の意見も参考にする。
予測に基づく対応実施時	• 予測結果への対応は、原因を分析してから適切な対応策を立案し実施する。 • 対応策は、対応コストに見合った効果がある対応策を立案する。

図表 3.1-12　要求分析・設計の品質予測の注意点

(2) 品質の測定項目別の予測指針

今まで述べてきた、品質の測定項目別の予測指針をまとめる。

測定項目	意味	目的	予測指針
レビュー指摘件数	プロダクトの障害指摘件数	プロダクトのレビュー障害指摘件数の測定(データ)	・少ないときはプロダクトの品質が良いまたは、レビューが不適切と予測し、レビュー実施内容を調査する。
レビュー指摘密度	規模当たりの障害指摘件数	プロダクト品質を表す指標(サンプル内またはガイドラインとの比較)	・大きいときはプロダクトが悪いと予測し指摘内容を分析する。 ・小さいときはプロダクト品質が良いまたは、レビューが不適切と予測し、実情を調査する。
レビュー工数密度	規模当たりのレビュー工数	レビュープロセスの妥当性評価	・小さいときレビューが十分行われていないと判断し、レビュープロセス(時間やレビュー実施者等)を調査する。
レビュー指摘効率	レビューのパフォーマンス	レビュー能力の評価 レビュー結果の評価	・高いときはレビュー対象の品質が悪いか効率的なレビューが実施されている。 ・低いときは、レビュー対象の品質が良いか、レビューアの能力／体制が適切でない可能性がある。

図表 3.1-13　品質の測定項目別の予測指針

(3) プロダクトとレビュープロセスの品質予測・評価の指針

今まで述べてきた、プロダクトとレビュープロセスの品質予測・評価の指針をまとめる。

適用領域	測定項目	品質予測と評価の指針
プロダクトの品質予測(現工程、後工程)	レビュー指摘密度 レビュー指摘効率	・レビュー指摘密度が大きいときは、プロダクトが悪いと予測し、指摘内容を分析する。 ・レビュー指摘効率が小さいときは、、レビュー対象の品質が良いか、レビューアの能力／体制が適切でないと予測し、実情を分析する。
レビュープロセスの評価(現工程)	レビュー指摘件数 レビュー工数密度 レビュー指摘効率	・レビュー指摘件数の絶対値が少ないときは、レビューが不適切と予測し、レビュー実施内容を調査する。 ・レビュー工数密度が小さいときはまたは、レビュー指摘効率が高いときは、レビューが十分行われていないと予測し、レビュープロセスを調査する。
潜在誤り予測(後工程)	プロダクトの欠陥数からの潜在誤り予測数	・信頼度成長曲線等の潜在誤り予測数が計画より大きいときは、プロダクトの品質確保が困難と予測し、品質確保の観点から追加 QA やプロジェクト計画を見直す。

図表 3.1-14　プロダクトとレビュープロセスの品質予測・評価の指針

▶▶ 3.1.6　事例：レビュー評価によるドキュメントの品質予測

(1) 目的と狙い

欠陥検出のデータのみでなく、レビュー投入工数や欠陥の内容から、ドキュメントの品質予測を行う。

(2) 考え方

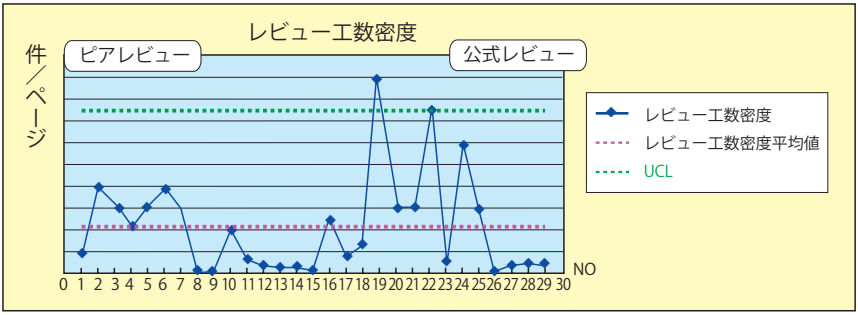
ドキュメントごとのレビューの投入工数と欠陥検出状況から、レビューが十分に行われているか(レビューの質)と、レビューが十分に行われドキュメントの欠陥が適切に検出されているか(ドキュメント品質)を予測する。また、同一ドキュメントのピアレビュー(有識者により積極的に誤りを摘出するレビュー)と公式レビュー（対象物を公式化するレビュー)の比較からドキュメント品質を予測する。

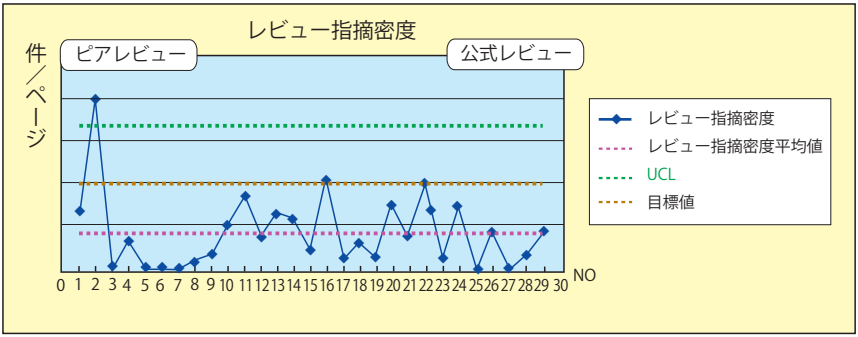
(3) 方法(項目、手法)

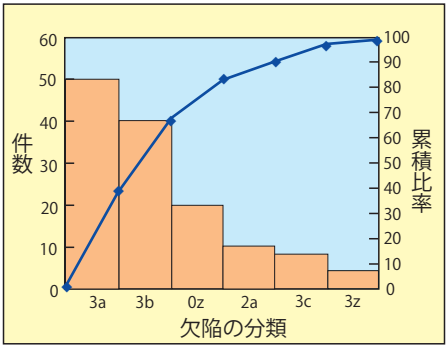
レビュー工数密度、レビュー指摘密度、欠陥分類、レビュー体制と品質目標について、以下の観点で評価を行い、ドキュメントの品質を予測する。

評価項目	評価観点
レビュー工数密度	・レビュー対象のボリュームに見合ったレビュー工数を費やしているか？ ・平均値に比べ極端に時間を費やしているのはなぜか？ ・平均値に比べ極端に時間を費やしていないのはなぜか？
レビュー指摘密度	・レビュー対象のボリュームに見合った指摘が行われているか？ ・平均値に比べ極端に指摘率が高いのはなぜか？ ・平均値に比べ極端に指摘率が低いのはなぜか？ ・レビュー工数密度とレビュー指摘密度のトレンドに傾向があるか？
欠陥分類	・適切なレベルの指摘が行われているか？ ・誤字／脱字ばかりではないか？ ・欠陥か、欠陥ではないか？ ・どこで欠陥が作り込まれたか？ ・欠陥の影響範囲はどこまでか？
レビュー体制と品質目標	・必要なメンバがレビューに参加しているか？ ・品質目標値の達成状況はどうか？

図表 3.1-15　評価項目と評価観点







【レビュー工数密度】
ピアレビューに比べ公式レビューに十分な工数がかけられている。
NO19 と NO22 は、エンドユーザのレビューが広範囲にわたり上限値を超えている。

【レビュー指摘密度】
ピアレビューに比べ公式レビューのレビュー指摘密度が大きい。レビュー工数密度が高く、その影響で増えていると思われる。

【欠陥の分類】
上流段階のシステム設計の誤りが 24% 発生しており、発生部分の再点検が必要。

図表 3.1-16　レビュー評価によるドキュメントの品質予測事例

(4) 効果

・レビューの質が伴ったドキュメント品質の予測が可能
・機械的に管理限界を逸脱したものを分析する方法より、分析対象の選定の効率が良い
・品質に不安のあるドキュメントに対して欠陥分類から対応策の検討が可能

▶▶ 3.1.7　事例：レビュー不足の予測＆是正

(1) 目的と狙い

レビューでのページ当たりの指摘件数の管理限界を設定し、その実績からレビュー対象の品質を予測する。

(2) 考え方

基本設計や詳細設計では，レビューの対象となるサブシステムやアプリケーションごとに業務の理解度や技術的な難易度に差が出ることがある。しかし、そのサブシステムやアプリケーション単位のチームメンバは同一なので、チームごとの能力はほぼ一定となる。よって、パフォーマンスがある領域よりも大きく逸脱した場合、そこに何らかの問題が潜んでいると推測できる。このパフォーマンスを監視することにより、レビュー対象の品質を予測する。

(3) 方法（項目、手法）

設計工程の品質予測は、大きく以下の 2 点を実施した。

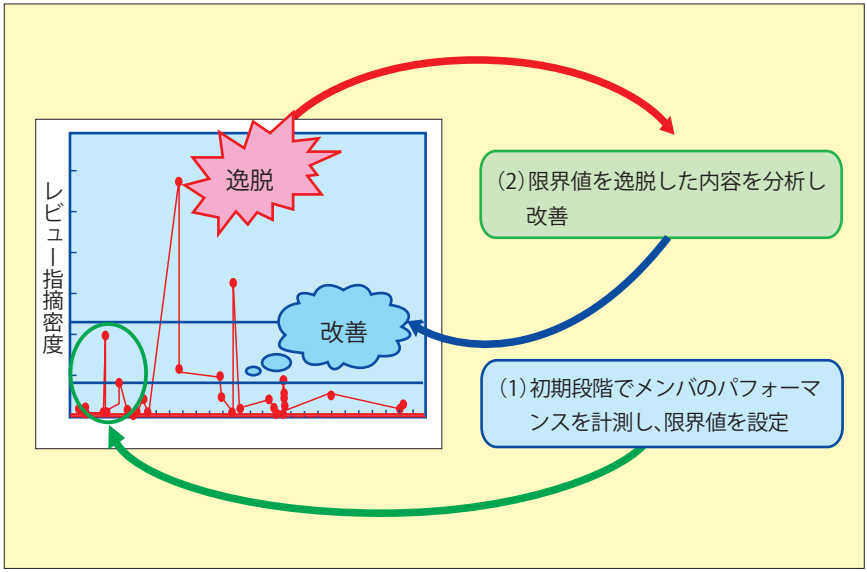
・チームメンバの能力範囲を設定
・パフォーマンスの逸脱監視と改善

まず、チームメンバの能力範囲を明確にするため、初期のレビュー指摘密度（レビュー指摘件数／ページ）を測定し、レビューメンバの能力が高い確率で収まる範囲（3σ）を限界値として設定した。この限界値をもとに管理図を用いてドキュメントごとのレビュー指摘密度を監視することにより、品質問題が潜在しているレビュー対象やレビュー方法の特定を行った。

(4) 効果

能力をもとにしたレビュー指摘密度を監視することにより、レビュープロセスの是正が行えた。

原因分析を行った結果、レビューの直前工程の質に問題があり、多くの欠陥が混入したままレビューに入ったことと、レビューアの質に問題があることが判明した。対応策として、お客様への追加ヒアリングの実施や、有識者による



図表 3.1-17　パフォーマンスの逸脱監視と改善

レビューの追加等の是正処置を早期に実施することができた。その結果、プロジェクト後半での仕様変更の多発等の問題を防止することができた。また、原因分析と改善をプロジェクト実行中に行ったことにより、問題を検出したメンバ以外の人にもレビュー不足の原因と改善策について周知することができ、同様の問題の再発を防止することができた。

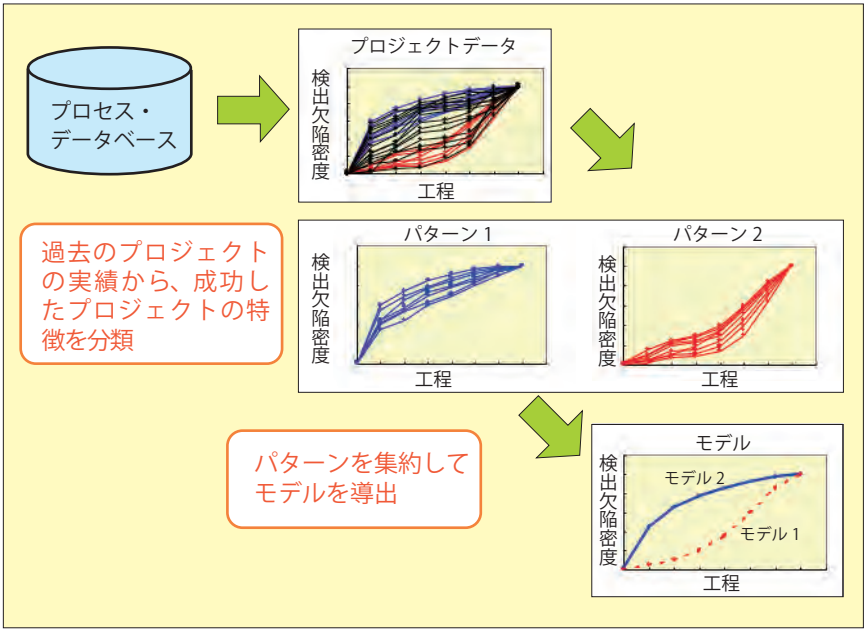
▶▶ 3.1.8　事例：プロセスパフォーマンスモデルを活用した潜在誤り予測

(1) 目的と狙い

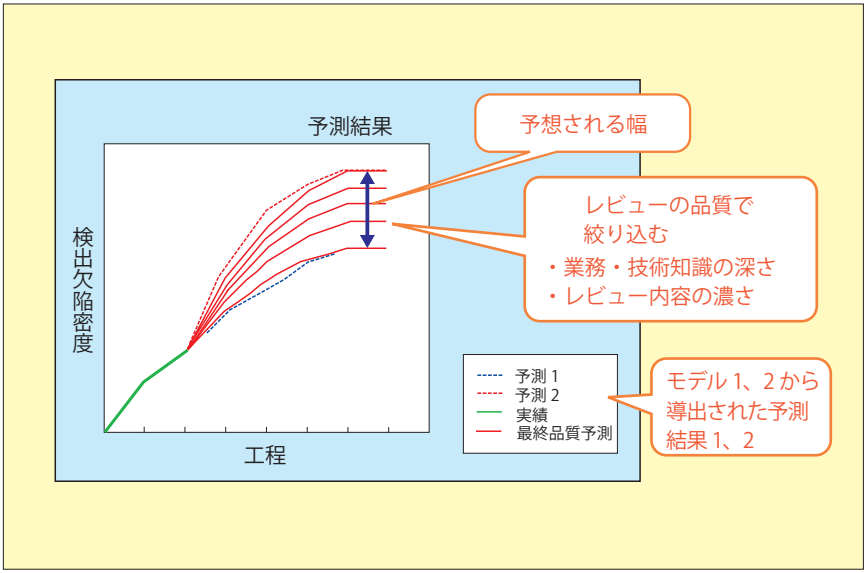
プロジェクトの最終品質目標の達成予測を目的に、工程の検出欠陥密度の実績から最終品質の予測を行う。

(2) 考え方

成功プロジェクトと失敗プロジェクトの測定値に差があるという前提において、成功プロジェクトの工程ごとの測定値のトレンドをモデル化する。プロジェクトメンバの特性上、前半の工程で欠陥を摘出して後半の工程で落ち着くパターンと、後半の工程で誤りを徹底的に検出するパターンの 2 パターンの傾向を関数モデル化する(図表 3.1-18)。



図表 3.1-18　プロセスパフォーマンスモデルの考え方



図表 3.1-19　最終品質予測の考え方

前述の２つの関数モデルにプロジェクトの実績値を入力することで、モデル１とモデル２の最終品質候補値が予測される。この２つの予測された最終品質候補値に、業務・技術知識の深さ、レビュー内容の濃さで補正して、プロジェクトの最終品質を予測する（図表 3.1-19）。

(3) 方法（項目、手法 [把握、予測]）

成功プロジェクトの誤り検出率のトレンドモデル（上流偏重モデルと下流偏重モデル）に、プロジェクトの測定値（検出欠陥密度：件／ KL）をインプットし、プロジェクト完了段階の目標品質達成度を予測する。

予測判定は、予測値が、カットオーバー後の残存誤り目標以下の場合は品質良好。目標以上の場合は品質不良と予測する。

(4) 効果

上流段階から、カットオーバー時の最終品質予測が可能となり、プロアクティブな品質改善施策のフィードバックが行えた。その結果、手戻り作業が減少しコストの抑制につながった。

適用結果から以下の改善が必要であることも判明した。

・モデルのレパートリー充実

　導入部門の事業特性やプロジェクト特性に合った複数のモデルの設定及び、最適なモデルの選定が可能な仕組みの確立

・予測精度の向上

　予測精度の検証の自動化とシミュレーション機能の拡充

・定性的パラメータの追加

　ツールの利用度、ドキュメントの再利用率等のプロジェクトの複雑性要因の追加

・品質データ収集の自動化

　開発活動から自動的にデータを収集する仕組み

▶ 3.2　プロダクトの品質の予測

▶▶ 3.2.1　目的と狙い

テスト工程の品質測定は以下の目的を達成するために実施する。

(1) 現工程作業の制御

測定したソフトウェア品質から、あとどのくらいのテストを実施すれば現工程を終了できるソフトウェア品質に到達できるかを予測する。その際、サブシステム、作成者、チームごとに見た場合の品質に偏りがある場合、改善のためテスト計画見直しや前工程の再実施等の品質向上策を実施する。

(2) 後工程・最終品質の予測

測定したソフトウェア品質から後工程や稼働時の最終品質を予測する。それにより後工程の作業内容設定や作業スケジュール調整を行い、最終品質が要求レベルに達するようにする。

▶▶ 3.2.2　アプローチ

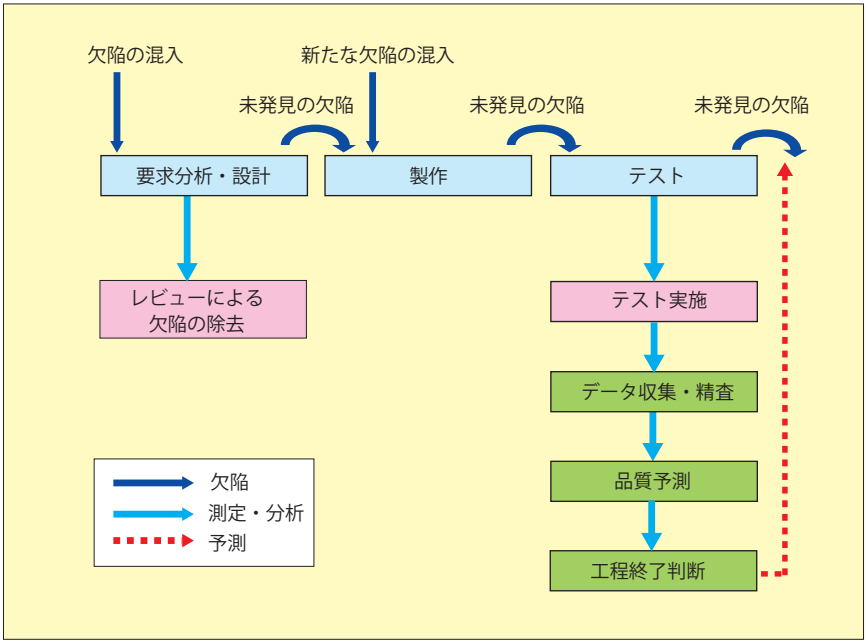
テスト工程の品質予測とは、設計から作成に至る様々な段階で混入され、結果として作成までで除去できなかった欠陥を、テスト工程によりどのくらい除去できたか、またあとどのくらい残っているかを予測することである。そのためには、その工程ではどのレベル(深さ・範囲)までテストするかを事前に計画しておくことが必要である。

また、テストが単体テスト、結合テスト、総合テストと進む程、発見された欠陥に対して、その原因の特定から対策の検証まで、膨大な手戻り工数が必要になる。したがって、テスト工程全体を通して「実施している工程で発見可能な欠陥はできるだけそこで発見し、除去する」ということが重要である。

(1) テスト実施

テストを実施する目的は、①欠陥を検出する②対象ソフトウェアの品質レベルが十分であることを確認し、その情報を示すことである。そして、『テストとは、エラーを見つけるつもりでプログラムを実行する過程である。』(『ソフトウェアテストの技法』G.J.Myers，1980/03，近代科学社)という意識が重要である。

テストの作業は、テスト実行の前後にも存在する。例えば、計画、コントロール、テスト条件の選択、テスト項目の設計、実行結果のチェック、テスト完



図表 3.2-1　品質測定と予測の流れ

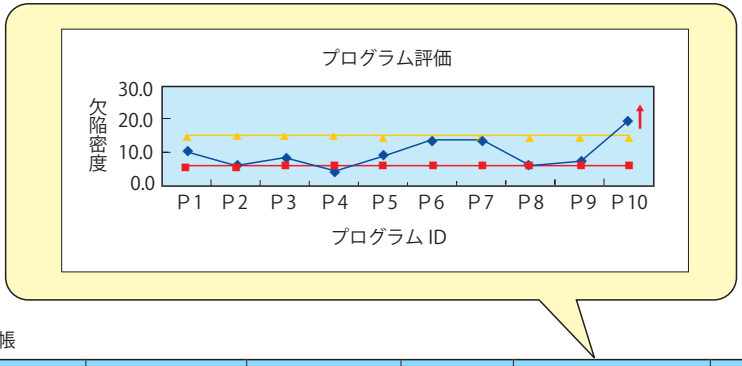
了基準の検証、テストプロセスやテスト対象システムの報告、テストの終了や収束がある。

テスト実施時の留意点として以下が挙げられる。

・テスト項目は、予想される出力または結果を定義しておくことが必須である。
・テスト項目は、正しい予想ができる入力条件だけでなく、誤った(予想しない)場合も考えて用意する必要がある(例：入力データ・ゼロ件、マイナスデータ)。
・良いテスト項目とは、重大な欠陥を検出できる確率が高いものである。
・テスト実施に際して、人的や時間的な制約があるため、効率的／効果的に実施する必要がある。

(2) データ精査

低品質の管理単位(サブシステムやプログラム)はテストを繰り返して品質を上げるよりも、設計書やコーディングを最初から見直した方がよい場合が多い。そこで、まず品質の悪い管理単位を、閾値モデルを用いて特定する。その際、品質を表す指標として欠陥密度を使用する。



管理台帳

プログラム ID	開発環境(LOC)	カバレッジ(%)	欠陥率(件)	欠陥密度(件／KL)	評価
P1	330	80	4	12.1	—
P2	750	60	5	6.7	—
P3	1000	40	8	8.0	—
P4	780	90	4	5.1	—
P5	630	80	6	9.5	—
P6	480	60	7	14.6	—
P7	930	40	13	14.0	—
P8	600	90	4	6.7	—
P9	350	40	3	8.6	—
P10	350	40	8	22.2	↑

図表 3.2-2　低品質対象物の特定

例えば欠陥密度が管理限界から飛び出しているプログラム(↑または↓のプログラム)を特定し、これらのプログラムの欠陥密度を管理限界内に抑えるために、検出した欠陥を分析し、内容によっては、設計の見直し、コードの見直し等を実施する。なお、このような対応をするためには、検出した欠陥について、それがどういう内容か、どの工程で作り込まれた欠陥か等を分類し記録しておく必要がある(分類の仕方については 3.2.3 (1) (d)参照)。

(3) 品質予測

テスト工程を制御するためには、工程の最後になって初めて品質状況を評価するのではなく、テストの進捗と同期して評価する必要がある。

コラム

ホワイトボックステスト

プログラムの内部構造(制御フロー)を調査してテスト項目を用意する。

ホワイトボックステストでは、パス・カバレッジの確保に主眼を置くことが多い。なお、通常次の３種のカバレッジを使用する。

・C0：命令網羅(ステートメントカバレッジ)
　コード内の全てのステートメントを少なくとも１回は実行する。
・C1：分岐網羅(ブランチ・カバレッジ)
　コード内の全てのブランチを少なくとも１回は実行する。
・C2：条件網羅(コンディション・カバレッジ)
コード内の任意の条件の組み合わせ全てを少なくとも１回は実行する。

これ以外にも、判定条件／条件網羅(Decision / Condition coverage（DC/CC))、複数条件網羅(Multiple Condition coverage（MCC)、経路網羅(Path coverage（PC)等、様々な定義がある。

ブラックボックステスト

仕様書／設計書からテスト項目を作成し、プログラムの内部構造と動作には一切関知しない。

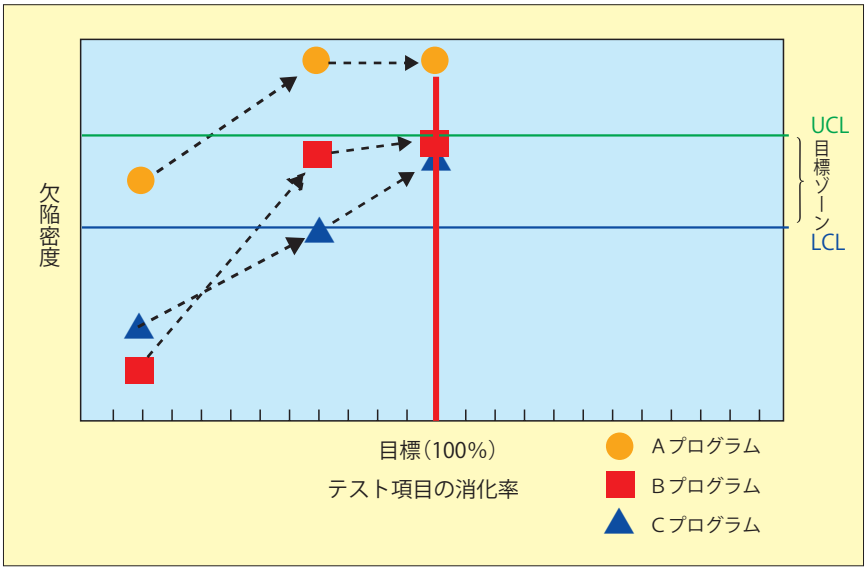
なお、代表的なテスト項目を設計する手法として同値分割、境界値分析、テストシナリオを設計する手法として因果グラフ、欠陥推測がある。

・同値分割
　テスト対象の入力データが取りうる値の範囲の中から、同じ結果が得られる範囲を同値クラスとし、分割されたクラスを代表する値を選択する方法。
・境界値分析
　入力データと出力データを同値クラスに分割し、それぞれのクラスの端(境界)の値を選択する方法。
・因果グラフ(原因ー結果グラフ)
　入力と出力の関係をグラフに表現し、デシジョンテーブルに展開して、テスト項目を設計する方法。
・欠陥推測
　直感と経験から起こりそうな欠陥の型を推測して、テスト項目を設計する方法。

(a) 品質のトレース(品質の動的把握)

品質を表す指標として、欠陥密度を使用し、ゾーンモデルを使ってトレースする。

例えば、図表 3.2-3 に示す通り工程内のテストの進捗に合わせて、測定量の値をプロットすることで、品質の改善状況を把握することができる。



図表 3.2-3　品質のトレース

① 目標ゾーン設定

必要なテスト項目を洗い出した後、管理単位ごとのテスト密度が自組織の基準値を満たしていることを確認する。

次に、欠陥密度の目標ゾーンを設定する。

② プロットとゴール予測

定期的に欠陥密度及びテスト項目の消化率を測定しプロットする。

この際、欠陥密度は、欠陥数÷規模で、テスト項目の消化率は考慮しない。

また、テスト項目の消化率は、実施済テスト項目÷当初想定テスト項目とする。

プロットした値から、現在の傾向が続いた場合、テスト項目消化率が 100 %になったときに欠陥密度が目標ゾーンに入るかを予測する。

③ 目標ゾーンから逸脱しそうなプログラムの欠陥分析＆対策

目標ゾーンを下回ると予測される場合、テスト項目を精査し、内容が妥当で

あればテストを継続する。目標ゾーンを上回ると予測される場合、欠陥を分析し対策を実施する。

④ 対策の評価

前回プロットと今回プロットを結んだ推移線が目標ゾーンの方向に向かう場合、改善効果が出ていると判断できる。対策を継続し、目標ゾーンを目指す。

推移線が目標ゾーンに向かっていない場合は、改善効果があまり出ていないと判断し、前回行った対策を見直す。

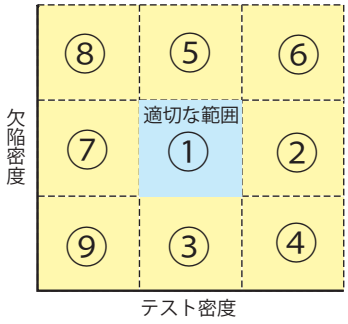
⑤ 目標ゾーンの精度向上改善

信頼度成長モデルにより欠陥が出尽くしたと判断できる時点のゾーンを記録しておき、次回プロジェクトの“ゾーン設定”に使用する。この改善サイクルを回すことにより、目標ゾーンの精度を向上させる。

(b) ゾーン判定

「ゾーン判定」が有効となる条件を次に述べる。

・テストの網羅性が確認されていること。
・当該組織で過去の実績データが蓄積・評価されていること。
・今回の開発プロセスが、過去に実施した開発プロセスを踏襲していること。



ゾーン	評価	品質
①ゾーン	一応品質は良好、テスト結果も予想どおり	良
②ゾーン	テスト結果がやや悪、テスト内容点検	低 ↑ 点検順位 ↓ 高
③ゾーン	テスト内容が適切か点検	
④ゾーン	テスト効率がやや悪、テスト内容点検	
⑤ゾーン	前工程の品質確保不足、内容点検	
⑥ゾーン	前工程の品質確保不足、内容点検	
⑦ゾーン	テスト不足、前工程の品質確保不足、内容点検	
⑧ゾーン	テスト不足、前工程の品質確保不足、内容点検	
⑨ゾーン	テスト不足、内容点検	

図表 3.2-4　ゾーン判定

(4) 工程終了判断

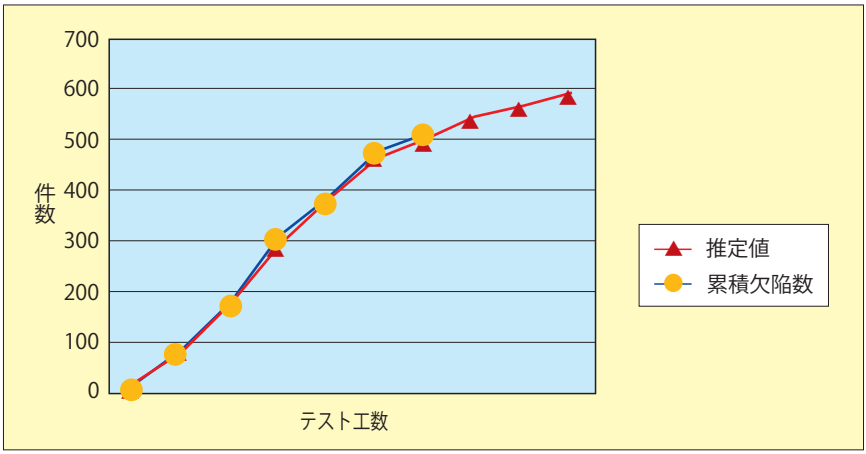
現工程の終了を判断するためには、関数モデルや閾値モデルを使用して、複数の導出測定量を総合的に評価する必要がある。

(a) 信頼度成長モデル

信頼度成長モデルを使用して、テスト期間中に検出した欠陥の状況から、工程終了時の品質や最終品質(稼働時の品質)を予測する。これはある時点までの

欠陥数の時間的変化から、その時点の残存欠陥数、及び最終欠陥数を予測し、工程終了判断に利用するものである。

工程の終了条件として、最終欠陥数に達するまでテストを実施するのは効率面からも無理がある。一般的な品質を求められるシステムでは信頼度成長曲線において、あらかじめ設計したテスト項目を全て実施しても 95%に到達しないときは、95%に達するまで追加テストを実施するべきである。なお、高い品質を要求されるシステムにおいては、それ以上のテストが必要である。



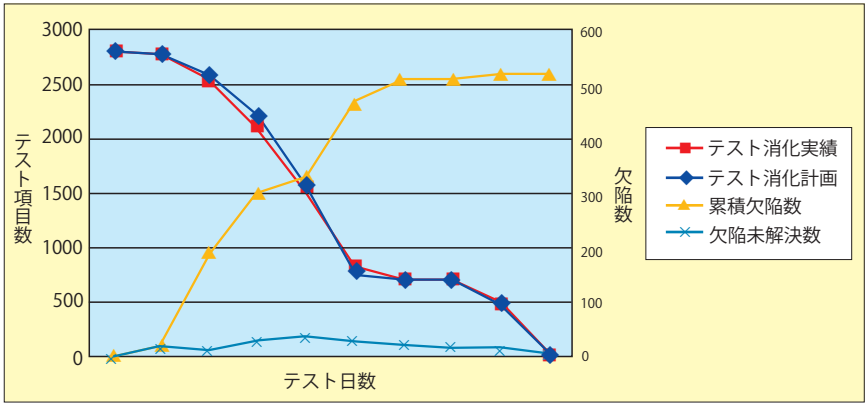
図表 3.2-5　信頼度成長モデル

プロジェクト全体の信頼度成長モデルと、各サブシステムの信頼度成長モデルを 1 つのグラフに重ね合わせ、プロジェクト全体が 95%に達する時期より遅く 95%に達するサブシステムを発見し、てこ入れするという利用方法がある（横軸設定時の注意点は 3.2.4 (6)を参照されたい）。

(b) 総合的な終了判断

信頼度成長モデル使用にあたっては、「3.2.4　適用上の注意点」で述べているが、テストに極端な偏りがないことや、テストの網羅性が十分確保されている等々注意すべきことがあり、信頼度成長モデルだけでテストの終了を判断するのは不十分である。信頼度成長モデルに加えて、残テスト項目数や、累積欠陥件数、及び未解決欠陥件数等を総合的に判断して評価する必要がある。

これらの品質評価指標を 1 つのグラフに表示し、品質の評価を行う。



図表 3.2-6　品質評価グラフ

累積欠陥件数が、テスト開始と同時に出始め、テストの後半では新たな欠陥が発生せず、平らな曲線になっており、未解決欠陥数が0件のグラフが理想的である。

テストの進捗が予定通りでないと、テスト消化実績がテスト終了日に向かっていない傾向を示す。また、プロジェクトの問題解決能力が低いと、欠陥未解決数が減少せずに増加する傾向を示す。いずれの場合にも、原因を究明し対応策を打つ必要がある。

▶▶ 3.2.3　方法（項目、手法）

(1) 基本測定量

テスト系のプロダクト品質予測時に使用する基本測定量について説明する。

(a) 規模

全体的な傾向を把握するには、LOC でも FP でも大差ないが、品質不良箇所を絞り込むには LOC の方が向いている。欠陥密度等の導出測定量では規模情報が必要である。LOC の場合、最終的にはプログラム単位まで規模情報を測定できるため、品質不良箇所をプログラム単位まで分解、絞り込みができる。これに対して、FP の場合アプリケーション境界を決めて測定するため、アプリケーション境界よりも細かな単位に絞り込むことができない。

また、人間の注意力には限界があるため、「作成した成果物の量に比例して誤りを作り込む」と考えられる。したがって、テスト工程では成果物の量を適切に測れるプログラムのコード行数(L)を規模とする。

プログラム規模を LOC で測定する場合、プログラムの機能量を表すものは論理ステップであり、論理ステップ数を測定する必要がある。しかし、プログラミングの表記法は自由度が大きく、厳密に論理ステップを測定するのは困難である。例えば大部分のプログラムは、1 行に 1 論理ステップを記述しており、ソースコードの物理的な行数(半自動生成行で手を加えた行も含む)から、コメント行・空白行・自動生成行を減じたものを、論理ステップとみなしても実用上大きな違いは出てこない。

なお、GUI ツール等で生成した成果物の量は LOC とは別に扱うことが多い。

(b) テスト項目数

品質を正しく予測するためにも、テスト項目数のカウントは重要である。重複した無駄なテスト項目が多い場合、誤った品質評価を下すことがある。したがって、テスト項目を設計するにあたり、

- テスト項目とテスト対象部分の対応を明確にし、テスト部分の重複や漏れをなくすことで、最低限のテスト項目数でより多くの部分を網羅するテスト項目を作成する
- より多くの欠陥を見つけられるテスト

等の点に留意する必要がある。

テスト項目数をカウントする場合、テスト対象の入り口から出口に至るパスのいずれかを最後まで通過するテスト項目、つまり、論理的に意味を持つテスト項目を 1 件とカウントする。

例えば、

- 単体テスト：モジュール単位でテスト項目を設定する。
- 結合テスト：業務処理アプリケーションの機能ごとに連結した単位でテスト項目を設定する。
- 総合テスト：システム全体でテスト項目を設定する。

(c) 欠陥数

検出した欠陥をカウントする場合の留意点を次に記述する。

- 仕様書／設計書通りに動作しない場合は、全て欠陥としてカウントする。ただし、業務上支障のない軽微な誤りや誤字脱字は基本測定量には含めない。
- テストを実施して見つかった欠陥、及び第三者によるコードレビューで見つかった欠陥をカウント対象とするが、作成者のセルフ・チェック(いわゆるデバッグ)で見つかった欠陥はカウントしない。

・複数の欠陥が同一原因により発生していたことが後で判明した場合は、カウントしない。
・機能の欠落は 1 件とカウントするが、重要障害として別管理し、発生した根本原因を究明して再発を防止する。

(d) 検出した欠陥の分類

検出した欠陥の分類は以下のように行う。

・どういう内容か
　仕様不良(要件未定義、機能・設計漏れ、設計誤り)
　コード不良(規約違反、コード漏れ、効率不良)
　テスト手順の不良
・原因は何か
　検討不十分、理解不足、不注意、環境設定誤り、他システム原因
・どの工程で作り込まれたか

なお、設計者、あるいはプログラマに起因する欠陥もあるので、そのプログラムの設計担当者、プログラム担当者を明確にしておく必要がある。

(2) 導出測定量

(a) テスト密度

基本測定量のテスト項目数を、同じく基本測定量の規模(KLOC)で正規化して求める。

テスト密度 ＝ テスト項目数 ÷ 規模(KL)

機能規模測定法に習熟した組織において、テスト項目数を機能規模測定可能な部分と非機能要件の部分に分けて把握できる場合は、基本測定量の規模(FP)で正規化して求めることができる。

(b) 欠陥密度

基本測定量の検出欠陥数を、同じく基本測定量の規模(KLOC)で正規化して求める。

欠陥密度 ＝ 欠陥数 ÷ 規模(KL)

正規化の規模として FP を使用する場合には、テスト密度の項で記述した点に留意する必要がある。

▶▶ 3.2.4　適用上の注意点

(1) 自組織の基準値を蓄積

目標とする欠陥密度やテスト密度を設定する前提条件は、自組織の基準値が確立していることである。基準値は、プロジェクトの母体組織で、開発するアプリケーションの業種、開発種別、言語別に、蓄積しておくことが望ましい。

(2) プロジェクトにふさわしい管理限界を設定

過去のプロジェクトの実績データベースが存在するのであれば、そのデータベースを利用して、目標とする管理限界を設定する。実績データベースがない場合、管理対象の欠陥密度を一定期間測定し、測定した値を統計処理して決定する。管理限界はプロジェクトの品質目標を勘案して決定する必要がある(2.3.1 のコラム「モデル化するときの注意点」を参照)。

(3) ゾーンの過信は禁物

ゾーン内に収まっていたとしても、品質は良好と過信せず、テスト項目を正しく設定しているか、カバレッジは十分か、重大な欠陥を検出していないか、欠陥を検出したモジュール以外にも同種欠陥は潜んでいないか等をレビューする必要がある。

(4) 適切なテスト項目の設定

ゾーン分析では、欠陥密度とテスト密度の組み合わせで、品質を評価しており、適切なテスト項目の設定が重要である。例えば、同値項目は本来 1 項目とカウントすべきであるが、正しいテスト項目設計が行われていないと、複数項目をカウントすることがあり、テスト項目の水増しとなる。正確な品質予測を行うためには、テスト設計はきわめて重要であり、ホワイトボックステストのカバレッジと、ブラックボックステストでは“同値分割”、“境界値分析”、“原因－結果グラフ”、“欠陥推測”等の手法を駆使して、正しくテスト項目を設計する必要がある。

(5) プロジェクト特性に合わせた品質目標の設定

品質目標は一意に決まるものではない。ユーザの求める品質要求や、品質への投資コストの大小により、品質目標が変わってくる。また、即時性が求められるリアルタイム処理や即時更新を必要としないバッチ処理等アプリケーションの種類により、品質目標は異なってくる。プロジェクトの特性に合致した品質目標を設定すべきである。

(6) 信頼度成長モデルの適用条件の考慮

信頼度成長モデルが機能するためには、テストに極端な偏りがないことが重要であり、あるサブシステムの部分だけが通るデータでテストした後、別のサブシステムの部分だけが通るデータでテストすると不連続なカーブになり最終品質の予測が難しくなる。同様に、テストの網羅性が低いと、十分な品質になる前に飽和現象が発生し誤った評価を下すことになるので注意が必要である。実業務に即したシナリオテストを繰り返して精度向上を図る際に、どのくらい繰り返せば現工程を終了できるかを測るツールとしては有用である。

変曲点を持つモデルでは、モデルが適切に機能するためには、変曲点後のデータが必要である。モデルの種類によって異なるが、想定作業量の 60％以上のデータが必要と言われているので、工程の前半で利用しても高い精度は得られない。累積欠陥数曲線が飽和に達しているかどうかでテストの十分性を評価する。

また、信頼度成長モデルでは、テストの進捗に合わせて欠陥数を累積していく。この累積欠陥数が数学モデルの曲線に正しく乗るためには、横軸に指定した単位間で同じテスト密度になるように、単位を設定することが理想である。しかし現実には難しく、横軸としてテスト日数やテスト消化率、テスト工数を取ることが多い。

したがって予測の解釈には注意が必要である。

(7) 欠陥を分類

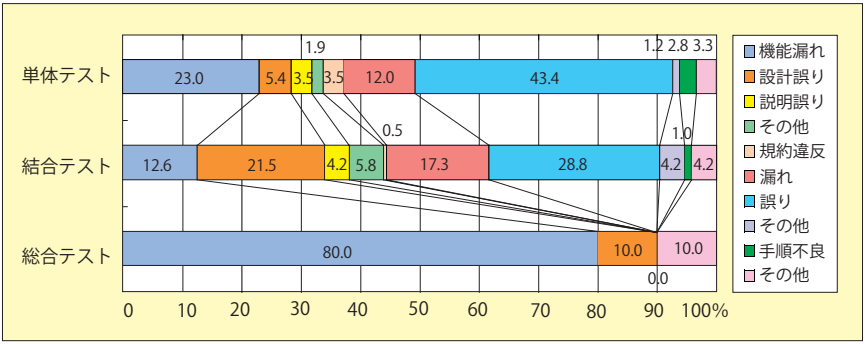
検出した欠陥の傾向を分析するために、欠陥が混入した工程の分類、欠陥の分類(仕様の不良かコードの不良か等)、原因の分類(仕様の記述が曖昧、あるいは仕様は正しいが誤解していた等)等を、行う必要がある。

欠陥の傾向分析を行い、必要な対策を実施し、その改善効果を評価して品質の向上に努める必要がある。

例えば、プログラムの欠陥をテストだけで全て取り除こうとすると、膨大な量のテストを行う必要があり、きわめて非効率である。効率的に対応するには、欠陥を検出した場合、その部分を修正するだけでなく、類似の欠陥をコードレビューにより取り除くべきである。

また、結合テスト工程の欠陥分析の結果、単体テストで見つけるべき欠陥が多発する場合は、一旦結合テストを止め、単体テストをやり直す、あるいはプログラムの見直しを行う等の対策が必要である。

サブシステム A									
欠陥分類		単体テスト		結合テスト		総合テスト		合　計	
仕様不良	機能漏れ	98	23.0%	24	12.6%	8	80.0%	130	20.7%
	設計誤り	23	5.4%	41	21.5%	1	10.0%	65	10.4%
	説明誤り	15	3.5%	8	4.2%			23	3.7%
	その他	8	1.9%	11	5.8%			19	3.0%
	小計	144	33.8%	84	44.0%	9	90.0%	237	37.8%
コード不良	規約違反	15	3.5%	1	0.5%			16	2.6%
	漏れ	51	12.0%	33	17.3%			84	13.4%
	誤り	185	43.4%	55	28.8%			240	38.3%
	その他	5	1.2%	8	4.2%			13	2.1%
	小計	256	60.1%	97	50.8%			353	56.3%
その他	手順不良	12	2.8%	2	1.0%			14	2.2%
	その他	14	3.3%	8	4.2%	1	10.0%	23	3.7%
	小計	26	6.1%	10	5.2%	1	10.0%	37	5.9%
合　計		426	100.0%	191	100.0%	10	100.0%	627	100.0%



図表 3.2-7　欠陥傾向の分析

▶▶ 3.2.5　事例：ゾーン分析事例

品質管理ツールを使ったゾーン分析実施事例を紹介する。４つの業務機能の結合テストが完了した時点での品質報告から抜粋したものである。

(1) サブシステムの品質評価

- サブシステム全体の分析結果が上記のように目標ゾーンに入っているから品質がよいと判断するのは粗すぎる。
- この例では、業務機能ごとにゾーン分析をすることで、業務機能Ａ、Ｂ、Ｃの品質が目標ゾーンから外れていることがわかる。

・この際、どの管理単位で分析するかはテスト工程により異なる。すなわち、単体テストの場合はプログラム単位に分析を行うが、結合テストではテストをした単位である業務機能ごとに分析を行う(図表 2.2-1 を参照)。

(2) 業務機能の欠陥状況分析

以下、結合テストを例に、欠陥状況分析の方法を述べる。

① どのゾーンに属しているかで対応を検討する。

・業務機能A(◆)は、欠陥密度が大きいため、品質に問題がある可能性が高い。
・業務機能B(■)はテスト量が不足しているので、テストを継続すべきであるが、テスト量が少ない割には欠陥密度が高く、品質にかなり不安がある。闇雲にテストを継続するのではなく、これまでの欠陥内容を分析し対策を講じた後にテストを行わないと矢印のように、目標ゾーンを超えてしまう恐れが大きくなる。
・業務機能C(▲)はテスト量が不足しているためテストを継続すべきである。テスト密度と欠陥密度割合から、テストを継続することで適切な範囲に入ってくると予測される。
・業務機能D(▼)は目標ゾーンに入っているので、品質はよいと考えられる。

図表 3.2-8 の例では、サブシステム全体(●)としての品質はテスト密度、欠陥密度は品質目標ゾーンに入っているが、サブシステムを構成する業務機能Aから業務機能Dの品質を個別にプロットすると、サブシステム全体を一括りで見たときとは随分異なる状況になる。このように、品質を評価する際には適切な管理単位で分析する必要がある(図表 2.2-1 参照)。

② 次に欠陥内容を整理し(図表 3.2-7 参照)どういう欠陥が多いのか、欠陥が混入した工程はどこか、欠陥が混入した理由は何かを分析する。

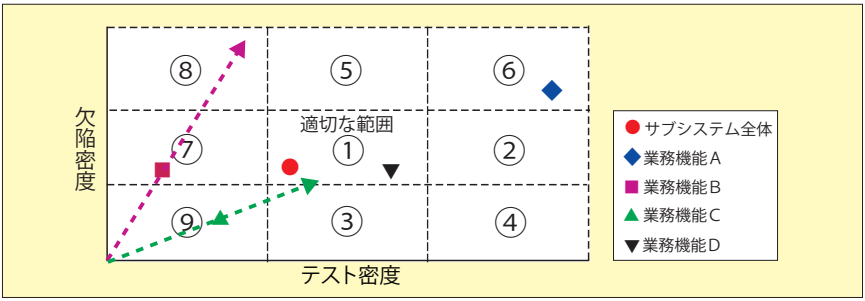
・コード不良が多かった場合は、本来単体テストで発見されなければいけない欠陥であるから、単体テスト結果を見直し、必要に応じて単体テストを再実施する。
　また、この場合はプログラム単位で欠陥密度を測り、特定の人やチームに欠陥が偏っている場合は、そこに対する指導や重点的な見直しも実施する。

・仕様不良が多かった場合、この業務機能の設計を見直す必要がある。モジュール間のインターフェース設計に欠陥があり、結合テスト工程で欠陥の山を築くのはよくあることである。

業務機能ID	規模(KLOC)	(結合テスト)テスト密度 達成率	目標	実績	(結合テスト)欠陥密度 達成率	目標	実績	品質評価(ゾーン位置から選択)
A	19.20	136.3%	80.0	109.00	177.8%	9.0	16.00	前工程の品質確保不足、テスト内容点検
B	9.90	75.0%	80.0	60.00	115.6%	9.0	10.40	テスト不足、前工程の品質確保不足、内容点検
C	4.80	85.3%	80.0	68.20	58.9%	9.0	5.30	テスト不足、内容点検
D	10.14	112.2%	80.0	89.72	111.1%	9.0	10.00	一応品質は良好、テスト効率も計画通り
サブシステム全体	44.04	101.3%	80.0	81.00	103.3%	9.0	9.30	
			↑基準(100%)			↑基準(100%)		

<目標ガイド>	テスト密度	テスト密度(達成度)*	欠陥密度	欠陥密度(達成度)*
結合テスト	80.0 ～ 100.0	100%～125%	9.0 ～ 12.0	100%～133%

補足：テスト密度(件／KLOC)、欠陥(件／KLOC)
＊：プロジェクト目標に対する目標ガイドの範囲

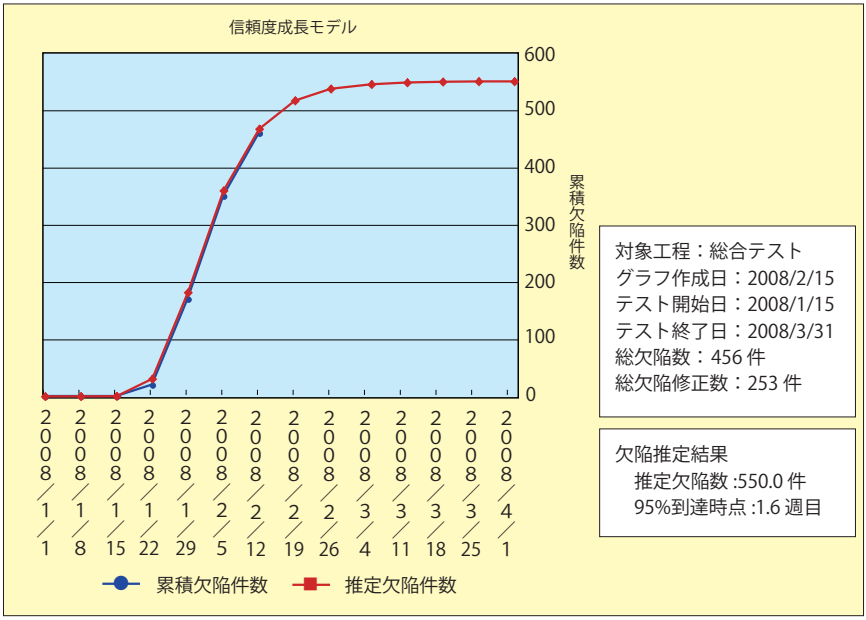


図表 3.2-8　ゾーン分析事例

▶▶ 3.2.6　事例：信頼度成長モデル

同じく品質管理ツールを使った信頼度成長モデル事例を紹介する。

総合テストの品質レビューに提出された資料である。信頼度成長モデルを使って累積欠陥数から、残存欠陥数やテスト終了時期を予測している。レビューでは、テスト項目の網羅性、ゾーン分析結果、テスト消化状況、重大欠陥の有無と対処状況等から、最終判断を行っている。



図表 3.2-9　信頼度成長モデル事例

▶▶ 3.2.7　事例：受入れテストでの品質予測

(1) 目的と狙い

ゾーン評価により納品物の品質予測と、その推移から品質確保の状況の監視を行う。

(2) 考え方

納品物のテストの推移と残存欠陥予測数から、納品物の品質の収束をとらえる。

(3) 方法（項目、手法）

外部の協力会社に発注したプログラムの品質の予測を、ゾーン評価と品質収束率を活用して実施した事例を紹介する。

プログラム納品時から強化テストまでの品質予測の経緯を以下の①～③に示す。

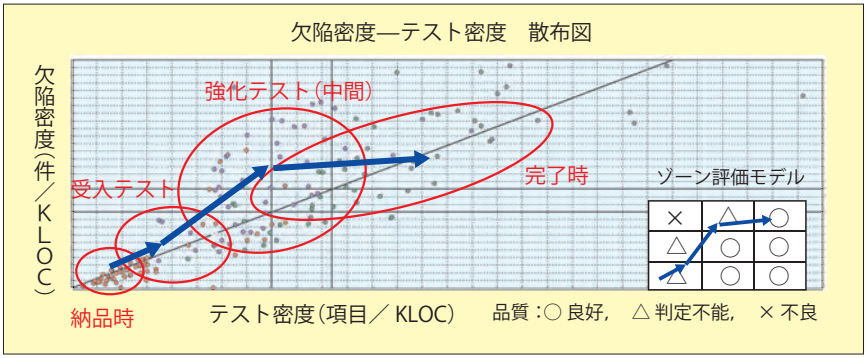
① 納品時のゾーン評価では品質判定不能領域にあり、受入テストを実施しゾーン評価を実施したがほぼ同様の判定結果であった。

② 強化テストを実施し中間時点で評価した結果、品質判定不能領域であったが、品質良好の領域に近づいていることが確認できた。

③ 強化テスト完了時点では、品質良好の領域に入り、プログラム品質良好

と判定することができた。

本事例におけるゾーン評価の推移を図表 3.2-10 に示す。



図表 3.2-10　受入テストでの品質改善例

また上記の各工程での欠陥数と残存欠陥予測量をもとに、各工程での品質収束率から品質が収束に向かっていることを監視した。

工程	欠陥数（A）	残存欠陥予測量（B）	合計欠陥数（C=A ＋ B）	品質収束率（D=A ／ C）
納品時	130 件	793 件	923 件	14%
受入テスト	295 件	663 件	958 件	31%
強化テスト（中間）	999 件	396 件	1395 件	72%
完了時	1123 件	190 件	1313 件	86%

図表　3.2-11　各工程での品質収束率

(4) 効果

これらのゾーン評価と品質収束率から、プログラムの品質確保ができているかを予測することができた。

▶▶ 3.2.8　事例：パフォーマンス測定からの品質目標状況の予測＆是正

(1) 目的と狙い

品質を作り込むための品質目標の達成度の予測をテスト工程の最後で評価を行うのではなく、テスト中のプロセスパフォーマンスのデータから、テスト進捗中に工程の最終品質目標の達成度を予測する。

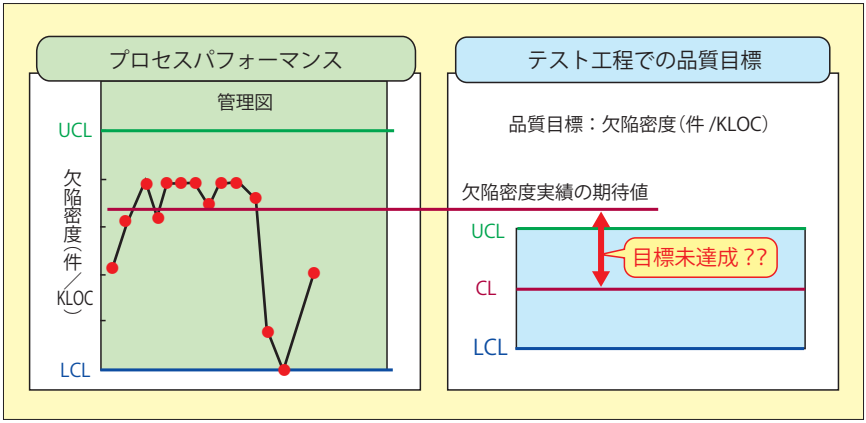
(2) 考え方

テスト期間中の実績値の最終予測値（その時点での実績値の平均等）が、テス

ト工程の品質目標範囲に収まっているかを監視することにより、テスト工程の最終実績が、品質目標を達成可能か予測する。

(3) 方法(項目、手法)

・テストでの欠陥密度の実績値を収集する。
・欠陥密度の実績値から工程終了時の欠陥密度の最終予測値を計算する。
・最終予測値と品質目標を比較する。



図表 3.2-12　パフォーマンスの最終予測値から品質目標状況を予測した事例

(4) 効果

本事例では、工程の途中で、工程の最終の品質目標が達成できるか予測できた。

単体テストでの適用事例としては、単体テストの実績データはプロセスパフォーマンスの管理限界内に分布していたが、工程の品質目標の上限値を大幅に上回っていたため、このままでは品質目標の達成が危ういと予測した。この結果を受け単体テスト中に中間評価を行い、現状の課題抽出と原因分析を行った。その結果、単体テストで検出された欠陥の原因の多くが、コードレビューで摘出すべき誤りが十分摘出できていない理由であったため、後半にテストを行うプログラムについては、コードレビューをやり直すこととした。その時点では大きな手戻り作業であったが、その後の結合テスト／総合テストの品質が安定し、プロジェクト全体で見ると効率的な施策を打つことができた。

▶ 3.3　プロジェクトの品質予測

前章までは、プロジェクトで測定した定量データから、次工程の成果物に与える影響や、プロダクトの最終品質をいかに予測するか、という視点でまとめてきた。

ここでは視点を少し広げ、良い品質のプロダクトは健全なプロジェクト運営から生まれる、という前提のもとに、プロジェクト品質の予測について考える。

▶▶ 3.3.1　目的と狙い

プロジェクトが「①健全な運営状態を維持できているか」を、定量データによって把握し、結果として「②計画通りの目標を達成できるか」を予測する。これを以下では「プロジェクト品質」の予測と呼ぶ。

これによりプロジェクトの問題化を未然に防止できるようにし、プロダクト品質への影響を最小限に抑えることを狙いとする。

▶▶ 3.3.2　アプローチ

予測を行うためには、まずプロジェクトの現状を正しく把握する必要がある。予測モデルを作るためのアプローチは次の順番になる。

① 定量データによりプロジェクトの現状をとらえる。
② プロジェクト途上の状況が最終結果に及ぼす影響を蓄積データから分析する。
③ 因果関係について仮説を立て、定量的な裏付けからモデル化する。

▶▶ 3.3.3　方法（項目、手法）

(1) プロジェクトの現状をとらえるための定量データ

プロジェクトの状況は刻々と変化する。その変化は、プロジェクトによって収集される様々なデータに、直接的あるいは間接的に反映されてくる。その中から、特に悪い方向に向かう変化をいかに早い段階でとらえるかが重要となる。

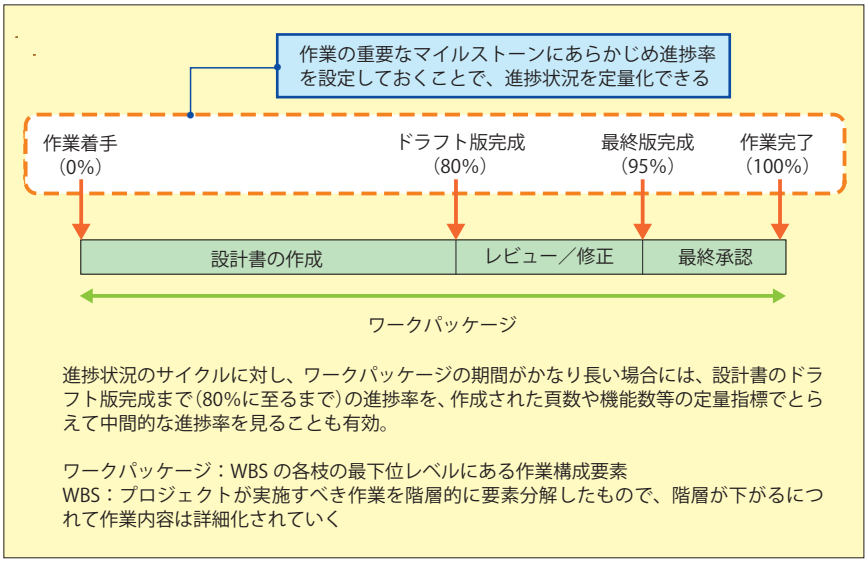
変化をとらえるには、計画値と実績値との差異を検証するのが一般的である。そのためには収集するデータ項目の定義、収集方法と測定タイミングをあらかじめ定義しておくことはもちろん、計画を定量的に立案することが前提となる。

(a) 進捗状況

計画したスケジュール通りにプロジェクトが進行しているかを見る。ただ漠然と遅れているとか予定通りとか言うのではなく、成果物の作成状況を定量的に測定し、その遅れ度合いを日数で示したり、特定のマイルストーンに達するべき日付に予定通り到達しているかを見ることが重要である。

例えば、プログラミング工程であれば、予定したステップ数に対してどのくらいコーディングできたか、予定した機能数のうちどのくらい作り終えたか、ドキュメントのページ数が予定に対してどの程度作られたか、といった定量指標を用いる。テスト工程においては、設定されたテスト項目の消化予定数に対する消化実績数、予測した欠陥発生数に対する発生実績数等の定量指標を用いる。

また、設計書のドラフトが完成する日、レビューが終わる日、最終承認が終わる日等をあらかじめマイルストーンとして決めておき、その通りに進んでいるかをチェックするのも有効である。



図表 3.3-1　進捗状況の把握

(b) コスト状況

重要でありながら、プロジェクトでは評価やコントロールが難しい指標である。プロジェクトでは、納期や品質を守るためにリソースの追加投入が優先され、コストが計画内に収まったか超過したかは結果論で評価されることが多い。

しかしながら、プロジェクトの途上で、特に進捗と合わせて見ることで、プロジェクトの状況を把握できる有効な指標であることには間違いない。

例えば、計画値以上のコストを費やしている場合は、プロジェクト状況が悪化してリカバリーに要員を投入しているとか、メンバが残業して対応している等の可能性がある。その一方で、リソースを先行投入することで進捗が前倒しに進んでいる、というプラス要因も考えられる。

計画値よりもコストを費やしていない場合は、単に要員が予定通りアサインできていないとか、計画に余裕を盛り込みすぎている等の可能性がある。その一方で、進捗が遅れていて予定通り作業が開始されていないということも考えられる。

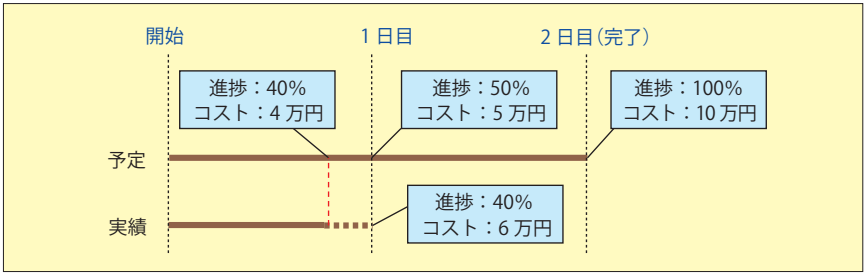
このように、コストはプロジェクトの進捗と合わせて見る必要があり、単にコストが予定より多いから超過、少ないから問題ないとは判断できない。これを解決する 1 つの手段が、アーンドバリュー・マネジメント(EVM)である。

(c) アーンドバリュー・マネジメント(EVM)

もともと米国の政府機関で採用され広がった手法で、計画されたコストとスケジュールに対する実績との差異から、プロジェクトの状況を定量的に把握する。

考え方は、いたってシンプルである。例えば A さんが、2 日間をかけて 10 万円のコストの仕事をすると仮定しよう。このとき EVM では、1 日経過した時点で進捗は 50%、コストは 5 万円消費すると考える(作業時間とコスト消費が均一に進む前提)。

ところが実際は、1 日経過した時点で全体作業の 40%しか進捗しておらず、コストは 6 万円消費してしまったとしよう。

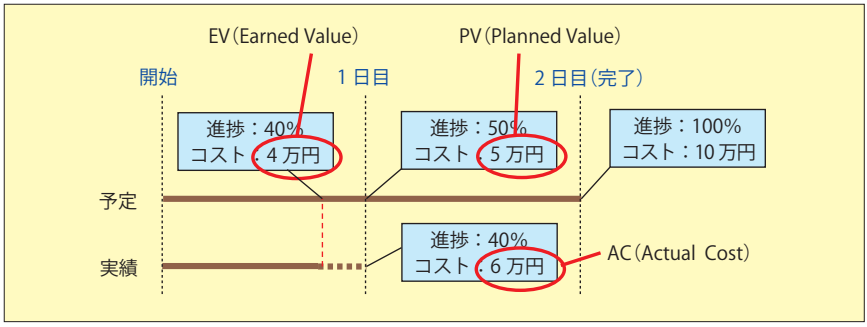


図表 3.3-2　EVM の考え方 1

進捗だけを見ると、10%の遅れを取り戻すだけの生産性アップでキャッチアップできるように思える。しかし、EVM 的に見るとどうだろう。進捗が 40%ということは、本来 4 万円のコスト消費で収まるべきだが、実績では既に

６万円を消費している。すなわち、スケジュール的には１万円の遅れだが、コスト的には２万円の遅れが生じていると見る。言い換えれば、10％の遅れを取り戻す生産性アップでは、スケジュールはキャッチアップできてもトータルコストはキャッチアップできないということである。

EVM では、計画通り実施できた場合のコスト（もともとの予算）を PV（Planned Value）と呼び、実績として消費したコストを AC（Actual Cost）、実績進捗で消費しているはずのコストを EV（Earned Value）と呼ぶ。



図表 3.3 -3　EVM の考え方２

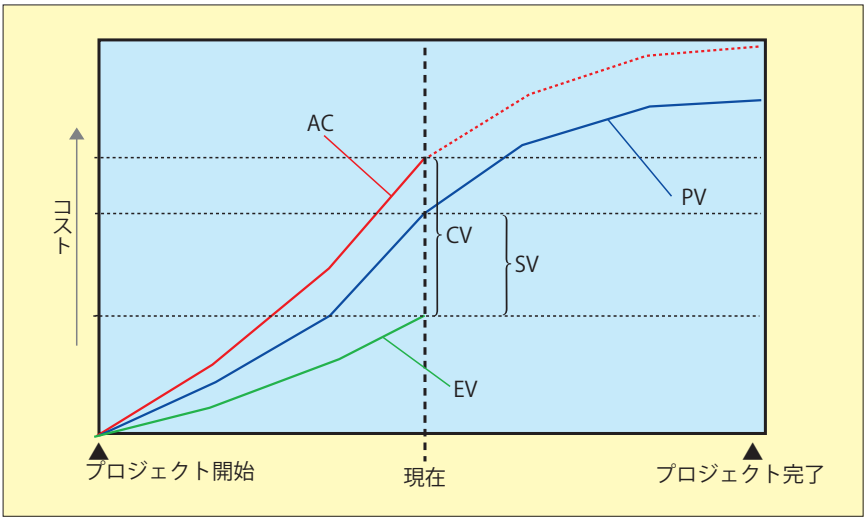
スケジュール面の状況を示す指標として SV（Schedule Variance）があり、SV ＝ EV － PV で算出される。コスト面では CV（Cost Variance）があり、CV ＝ EV － AC で算出される。よって、上記の例では SV は－１万円、CV は－２万円となる。

この２つの指標から何がわかるのだろうか。下の表で説明しよう。

例	CV	SV	解　説
1	－２万	－１万	上記の例：スケジュールもコストも超過しており、プロジェクトとして深刻な問題が発生していると考えられる
2	±０	－１万	スケジュールは遅れているが、その分コストも消費していない：要員調達ができていない等の可能性あり
3	±０	１万	スケジュールは進んでおり、その分コストも消費している：先の作業を早めに片づけている可能性あり
4	－１万	±０	スケジュール通りだが、コストを余分に消費している：生産性が悪い分、残業して頑張っている可能性あり
5	１万	±０	スケジュールは予定通りで、コストも予定程使っていない：リスクとして見込んだ余裕を使わずに済んでいると考えられる
6	１万	１万	スケジュールが進んでおり、コストも予定より少なく済んでいる：そもそも計画に余裕を見込みすぎていると考えられる

図表 3.3- 4　CV と SV の解説

このように、EVM を導入することで、単なる進捗管理では見えないことが把握できるようになる。特に上記 4 番の例のように、進捗管理では予定通りとしか判断できない事象も、EVM ではその背景にあるコスト超過が見えてくる。



図表 3.3-5　EVM の効果

このように効果の高い EVM ではあるが、現実にはソフトウェア開発の場では、あまり活用されていないのが実態である。なぜなら、EVM を確実に行うためには、

①プロジェクトが精度の高い計画を立て、担当者 1 人のレベルまで落とす
②上記レベルの作業 1 つひとつについて、予定コストをきちんと入力する
③作業者が、作業が完了する度に、滞りなく実績を入力する

ということを実践する必要があり、曖昧な要件から見えないものを作るソフトウェア開発プロジェクトにとっては、かなり高いハードルになっているからである。仕様変更が多く、スケジュールを日々組み替えながら進むようなプロジェクトでは、その度に予定コストを見直して更新するのも大変な作業である。

(d) 仕様変更状況

通常は、構築するシステムの仕様を要件定義工程で確定させ、それを実現するための計画を立案し、その計画に沿ってプロジェクトを進行させる。仕様変更は、確定したはずの仕様を変えるということなので、少なからずプロジェク

トの健全性に影響を与え、場合によっては実施した作業に手戻りを引き起こしたり、計画そのものを変更する必要性を生じさせたりする。

仕様変更が多発し、プロジェクトの途上で取り入れていくと、システム全体としての整合性が失われ、プロダクト品質に多大な影響を与える。多くの場合、コミュニケーションロス等により変更が伝わらず、テスト段階になって不整合が発覚する等、次第に混乱し始め、プロジェクトは健全性を失っていく。

仕様変更は、その内容や発生時期等により影響度合いが大きく異なるため、事前に発生予定数を計画することは難しい。また、仕様変更には、ユーザの追加要望によるものだけでなく、問題課題や欠陥の解決に伴う内部要因で発生するものもある。

したがって、仕様変更の発生数、受入数はもちろん、プロジェクトへの影響度を定量的に把握することが重要となる。できれば影響度の基準をプロジェクト横断的な視点で定義し、影響度がプロジェクトの健全性に与えるインパクトを過去の事例からとらえられるようにする(3.1.3 のコラム参照)。

(e) 問題課題状況

大きく、マネジメント系の課題(例：サブシステム間で発生した欠陥の対応責任者が不明確なまま放置されている)とプロダクト系の課題(例：夜間バッチがオンライン開始時間までに完了しない)に分けられる。後者はプロダクト品質そのものであるが、前者も少なからずプロダクト品質に影響を与える。

問題課題については、その発生数や解決数だけでなく、プロジェクトの健全性にかかわる影響度を合わせて評価する。また、問題課題を工程ごとに管理し、次工程に持ち越した残課題の状況をとらえることも重要である。

(f) 労務状況

プロジェクトメンバの労務状況(残業時間や病欠等の状況)は、プロジェクト品質を示す重要な指標となる。特定のメンバに作業が集中しているとか、トラブル等で徹夜している等、マクロでは見られない状況がわかることもある。

昨今は、メンタルヘルスの問題でプロジェクトから離脱してしまうようなケースも、プロジェクトの健全性を損なう大きなファクタとしてとらえる必要がある。

(g) プロセスの遵守度

モダン・プロジェクトマネジメントにおいては、各工程で実施する作業を標準プロセスとして定義し、それに沿ってプロジェクトを推進する。標準プロセスには、作業内容(WBS)だけでなく、そのアウトプットは何か、実施基準は

何か、等も定義する。これにより、作業の実施漏れを防止し、責任分担を明確にすることができる。

標準プロセスから逸脱する行動を繰り返しているプロジェクトには、何か特別な事情があるはずで、プロジェクトの健全性にも影響を与える可能性がある。

例えば、プロジェクトの歯車が合わなくなると、会議が予定通り開催されず延期が多くなるとか、記録が残されなくなる、承認プロセスが省略される、報告が滞る、といった現象となって現れてくる。

このような状況を把握するには、内部監査やプロセスQA(Quality Assurance)等によって定期的にプロセスの遵守状況をチェックすることが重要であり、一般的にはチェックリストによって確認する。

(2) プロジェクト品質の予測モデル

ここでは、当該プロジェクトの過去の振る舞い、あるいは過去のプロジェクトから蓄積されたデータに基づいて、予測モデルを作る方法について考える。

一般論で言えば、プロダクト品質の予測に比べると、プロジェクト品質の予測はまだ十分に確立されていないのが実情である。なぜならプロジェクト品質は、プロジェクトマネージャの力量やユーザの参加度合い、体制、スケジュール、環境等の様々な要因が複雑に影響するためである。しかも、それらの要因の多くが定量的に測定することが難しい(要員の力量、体制の弱さ、ユーザの理解度等)。

ただ、世の中に認知され確立されるまでは至らないまでも、企業ごとに、その特性に応じた定義や分析を行い、試行したり実践したりしている手法がいくつかある。また、「このようなマネジメント系の課題が出始めるとプロジェクト品質が悪化する」といった定性的な判断指標を明文化したり、チェックリストの結果を定量化してプロジェクト品質の評価点を付けているケースもある。

例えばある企業では、蓄積された成功プロジェクトの実績データをもとに、その振る舞いをモデル化し、そのモデルとの乖離を見ることでプロジェクトの健全性を予測し、実際のプロジェクト運営に役立てている。

多くの成功プロジェクトでは内在する誤りを早い段階で潰しているのに対し、失敗プロジェクトではレビューの甘さが後工程で露呈する傾向にある。このモデルは、その傾向がプロジェクト全般に当てはまることを前提とし、レビューやテストにおける欠陥密度でプロジェクトの遂行能力を定量化しようというものだ。

具体的な指標として、レビューにおいては 1 ページ当たりの欠陥数、テスト

においては KLOC 当たりの欠陥数を用い、成功プロジェクトのパターンを抽出し、モデル化する。

後は、実プロジェクトのデータを測定し、このモデルからどの程度逸脱しているかを見ればプロジェクト品質を予測できるという理屈である。

このモデルの特徴は、単に逸脱しているか否かだけではなく、逸脱の仕方についてもパターン化し、その傾向から取るべきアクションを整理している点にある。このプロジェクト品質の予測モデルについては、3.3.6 事例に詳しく紹介する。

▶▶ 3.3.4　適用領域

プロジェクトは、人の力量や経験、振る舞い等に大きく依存するため、論理的な裏付けや定型的な予測モデルを確立するのが難しい。モデルが確立しても、それに当てはめる定量データをいかに着実に収集するか、基準を統一するか、という課題が常につきまとう。ある環境下で有効な手法が、別の環境では機能しないこともある。

確固たるモデルの確立までには至らずとも、多くのプロジェクトでは、プロジェクトの計画段階でリスクを洗い出し、予防対策や顕在化対策を立案する。しかし、プロジェクトの途上でリスクの状況を見直したり、新たなリスクを抽出したりすることは忘れがちで、顕在化した問題課題の解決に目がうばわれてしまうことが多い。

「この程度の遅れなら、過去の経験からしてもキャッチアップは十分できる」とか「このままのペースなら、仕様変更をもう少し受け入れてもカットオーバーまでには何とかなるだろう」という見込みを立てるだけでなく、逆に「遅れがキャッチアップできなかった場合」「受け入れた仕様変更が大きな手戻りを起こした場合」というリスクとしてもとらえ、プロジェクト品質やプロダクト品質にどのようなインパクトを与えるのか、その顕在化確率はどの程度なのかを明確にすることが重要である。

このような定性的な予測でも、プロジェクトの問題化防止には有効である。まずは、測定した定量データからとらえられた状況変化に対し、その原因を踏まえた懸念事項を整理し、過去の事例をもとに仮説を立ててみる。

例えば進捗が予定に対して1ヶ月遅れている」という現象に対し、「作業が増えている」のか、「手待ち状態」なのかによって、懸念されるリスクも打つ手も大きく違ってくる。作業が増えているなら当初の規模に抑える努力をする必要があるし、

手待ち状態ならボトルネックを解消することを一番に行わなければならない。

「進捗が予定に対して 1 ヶ月遅れている」場合、今発生している現象、すなわち「遅れている」「作業が増えている」「手待ち状態になっている」「仕様変更が止まらない」「要員がアサインできていない」といったそれぞれの現象と関連する過去の実績や経験が、仮説を立てる上で重要になってくる。すなわち、蓄積された過去のプロジェクトの実績や経験が、それぞれの現象に対応付けられ、チェックリスト、対策案、あるいはモデルという形になるのである。

プロジェクトの品質の予測は未開拓の部分が多く、今後も研究が進み、有効なモデルが生み出されていくだろう。これらのモデルを適用する際は、自社の特性やプロジェクトの状況を大きくとらえ、各モデルの本質をよく理解して取り組むことが重要である。できれば自社の実プロジェクトで検証を行い、予測するためのデータ収集労力に見合うだけの有効な結果が得られるかどうかを見極めてから本格導入することが望ましい。

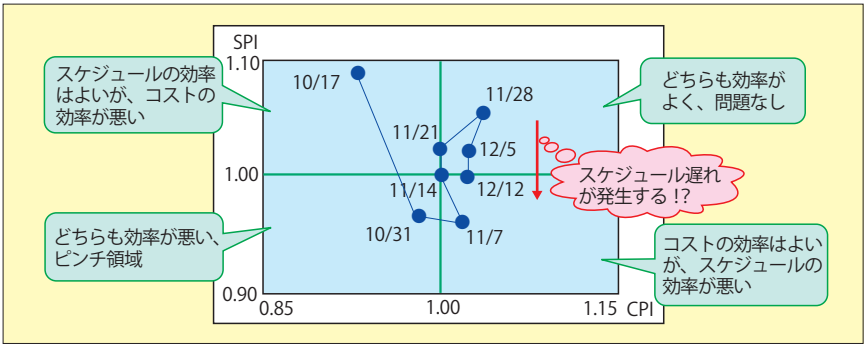
▶▶ 3.3.5　事例：EVM を活用したプロジェクト品質予測

(1) 目的と狙い

プロジェクトの品質予測を目的に、プロジェクトのスケジュール効率とコスト効率の実績からプロジェクトの状態を予測する。

(2) 考え方

スケジュール効率(SPI)とコスト効率(CPI)の直交座標から、プロジェクトの現在の状況と、そのトレンドからプロジェクトの今後の状態予測を行う(図表 3.3-6)。



図表 3.3 -6　プロジェクト品質予測モデルの考え方

SPI と CPI の状態	プロジェクトの状態
SPI も CPI も 1.0 以上の場合	・計画よりも生産性が高く、問題ないため現状を維持 ・ただし、計画よりもあまりに効率良い場合は、計画自体に問題がないか、または品質に問題ないかを確認する必要がある
SPI は 1.0 以上であるが CPI は 1.0 以下の場合	・スケジュールには余裕があるため、スピードを落としてでもコスト抑制が必要 ・深夜残業や休日出勤等で対応している等、計画に無理がないかを確認する必要がある
CPI は 1.0 以上であるが SPI は 1.0 以下の場合	・コストには余裕があるため、コストを追加投入してでもスピードアップが必要 ・ただし、コスト計上が正しく行われていない可能性もあり、確認する必要がある
SPI も CPI も 1.0 以下の場合	・スケジュール及びコストとも問題であり、原因と対策が明確な場合は、スコープの変更を含めて、計画の見直しを検討する

図表 3.3-7　SPI と CPI から見たプロジェクトの状態

＊ SPI や CPI が 1.0 以下の場合は、原因分析を実施し基本的にはプロジェクト管理の中で解消させる。なお対応策は効率指標(CR=SPI × CPI)も考慮して検討するとよい。

(3) 方法(項目、手法 [把握、予測])

プロジェクトの SPI と CPI 実績を定点観測し予測する。

以下に SPI や CPI が悪化した場合の対処事例及び、パートナーのコスト実績の把握方法を示す。

(a) コスト効率が悪化してしまった場合

一般的には、コスト効率は悪化すると、ほとんど挽回できないため、悪化させないように最優先で監視する。

悪化してしまった場合は、以下の対応を検討する。

・プロジェクトのコントロールによる挽回
　悪化原因を分析し、プロジェクトの進め方による挽回を検討する。

・コンティンジェンシー予備費の切り崩し
　完了時総コスト予測(EAC)がコンティンジェンシー予備費(あらかじめプロジェクトのコストに織り込み済みの費用)で対応できるかどうかを検討する。

・マネジメント予備費の投入
　コンティンジェンシー予備費で対応できない場合は、マネジメント予備費(プロジェクトのコスト外の費用)の使用について検討する。

(b) スケジュール効率が悪化してしまった場合

まずはクリティカルパスに影響がないかを点検する。

クリティカルパスに影響がある場合は、クラッシングやファストトラッキング等を行い、所要期間を短縮する。このとき、生産性低下やリスク増大になることが多いため、注意する必要がある。

(c) パートナーのコスト実績の把握方法

コストに関してはパートナーからの報告が拒絶されることがあり、プロジェクトの進捗に伴うコスト(AC)の把握が困難となる。しかし、AC を把握することでパートナーの生産効率が把握できるため、パートナー先のパワー不足やプロジェクトの体制強化等の対策を前もってに行うことができる。よって、パートナーに PV と EV のみを報告してもらい、AC は以下のいずれかで代替しプロジェクトの品質を予測する。

AC 代替方法	EVM 計算結果	説　明
AC ＝ PV	CPI ＝ SPI	コスト責任はパートナー先だが、進捗遅れにより当社にコスト悪化のリスクが発生する為、遅れ状況を加味する運用
AC ＝ EV	CPI ＝ 1	コスト責任は全てパートナー先にあるとして、進捗にかかわらず、コストは一定とする運用

図表 3.3 -8　AC 代替のプロジェクト予測方法

(4) 効果

以下に、EVM 適用の効果例を示す。

・早い段階で計画漏れを検知

プロジェクトの初期段階で、SPI と CPI が良すぎる傾向にあることに気付いた。原因を分析してみた結果、WBS の洗い出しが不足していたことが判明した。これにより、早い段階での WBS 見直しを行うことができた。

・計画の定量化への意識改善

EVM を導入し定量的な管理を行うためには「計画をきちんと立て、計画をもとにマネジメントしていかなければならない」という意識が、プロジェクトメンバに浸透してきた。

・予測による損失増大の防止

設計の途中段階において、大幅な納期遅延とコスト超過が発生することが予測結果として出た。これをトリガに分析した結果、お客様からの要求

により仕様がいつまでも確定しないことが原因であることが判った。早速対応を取り、お客様に対して、EVM のデータを元に論理的に説明することで、仕様の限定と早期確定を説得した結果、損失の拡大を防止できた。

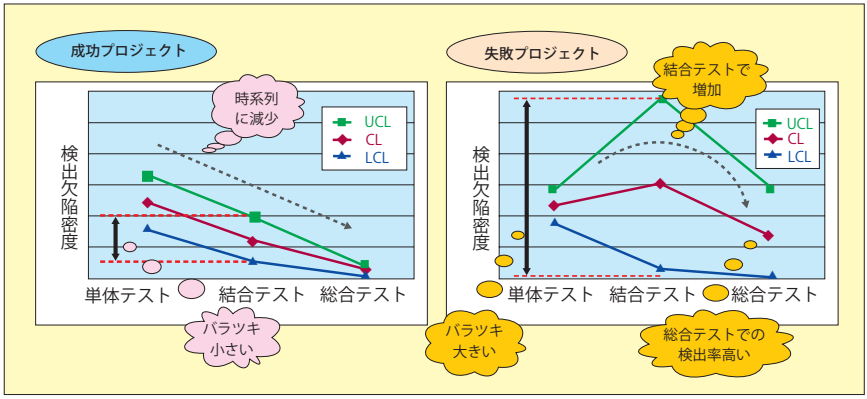
▶▶ 3.3.6　事例：プロセスパフォーマンスベースラインを活用したプロジェクト品質予測

(1) 目的と狙い

プロジェクトの最終品質目標の達成予測を目的に、システムの品質安定化を狙い、工程の欠陥密度のトレンドから品質予測を行う。

(2) 考え方

品質目標達成プロジェクト(成功プロジェクト)と品質目標未達成プロジェクト(失敗プロジェクト)のテスト工程のトレンドに差があるという前提において、成功プロジェクトの工程ごとの欠陥密度のトレンドをモデル化する(図表 3.3-9)。このモデルとプロジェクトの実績値のトレンドを比較することにより、プロジェクトが成功に導かれているかを予測する。



図表 3.3- 9　プロジェクトの最終品質予測の考え方

(3) 方法(項目、手法[把握、予測])

成功プロジェクトの欠陥密度のトレンドモデル(期待値± 3σ)と、プロジェクトの実績値のトレンドを比較し、プロジェクトの成否の傾向を予測する。測定尺度の例は図表 3.3-10 に示す。

対象工程	検出欠陥密度
要件定義・設計・製作	件／ページ
テスト	件／KL

図表 3.3-10　工程と検出欠陥密度の尺度例表

予測判定は、実績値が 3σ の範囲で推移していれば成功プロジェクトに限りなく近いと予測し、3σ を逸脱している場合は失敗プロジェクトに限りなく近いと予測する。

(4) 効果

プロダクトの実績からプロジェクトの成否が予測できるようになり、プロアクティブな品質改善施策のフィードバックが行えた。しかし、適用結果から以下の改善が必要であることも判明した。

- 収集データの信頼性向上

  データ項目の定義と収集データにギャップがある。また、データの精査や外れ値を考慮してベースラインを設定する必要がある。

- 標本数の確保

  プロジェクトの特性等条件を絞ってパフォーマンスベースラインを設定しようとすると、標本数が少なくなり信頼性に問題が発生する。

- 生産性データの収集拡大

  品質に関するデータは蓄積されているが、生産性に関連するデータの蓄積が弱い。また、プロジェクトの生産性として外部委託先の生産性データの収集が必要。

- データのバリエーションの拡大

  LOC 以外に FP 等のデータも蓄積し、パフォーマンスベースラインのバリエーションを増やす必要がある。

- 類似システムの実績値からのパフォーマンスベースラインの導出

  言語やシステム特性による条件でのパフォーマンスベースラインの導出はできるが、プロジェクトが複雑で、メンバのスキルやプロジェクトの置かれている環境等によりパフォーマンスが左右される。類似システムの抽出精度の向上が必要。

ANNEX

▶A　ソフトウェア測定プロセス

ソフトウェア測定プロセスについては、2002 年 7 月に国際規格：ISO/IEC 15939 が制定された。2004 年 6 月には国際規格を完全翻訳したものが JIS X0141 として制定されている（参考文献[1]）。ソフトウェア測定プロセスの実践を支援する書物として、Practical Software Measurement（参考文献[2]）がある。これは ISO/IEC 15939 を検討した ISO/IEC JTC1 SC7/WG13 のコンビーナを務めた John McGarry 氏らが中心となって書き下ろしたものである。その邦訳版が「実践的ソフトウェア測定」（参考文献[3]）である。

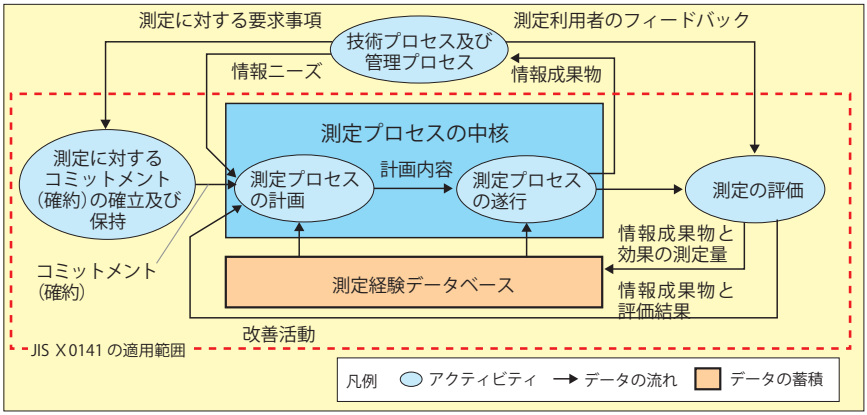
ここでは、上記邦訳版をベースにソフトウェア測定プロセスの概要を紹介する。なお、図表は JIS X0141 から引用している。

▶▶ 1. 規格の中核となるモデル

測定プロセスモデルと測定情報モデルが規格の中核である。

(1) 測定プロセスモデル

測定プロセスモデルは、PDCA の概念に基づいている。図表 A-1 では、技術プロセスや管理プロセスの目的や課題を達成するために必要となる情報（情報

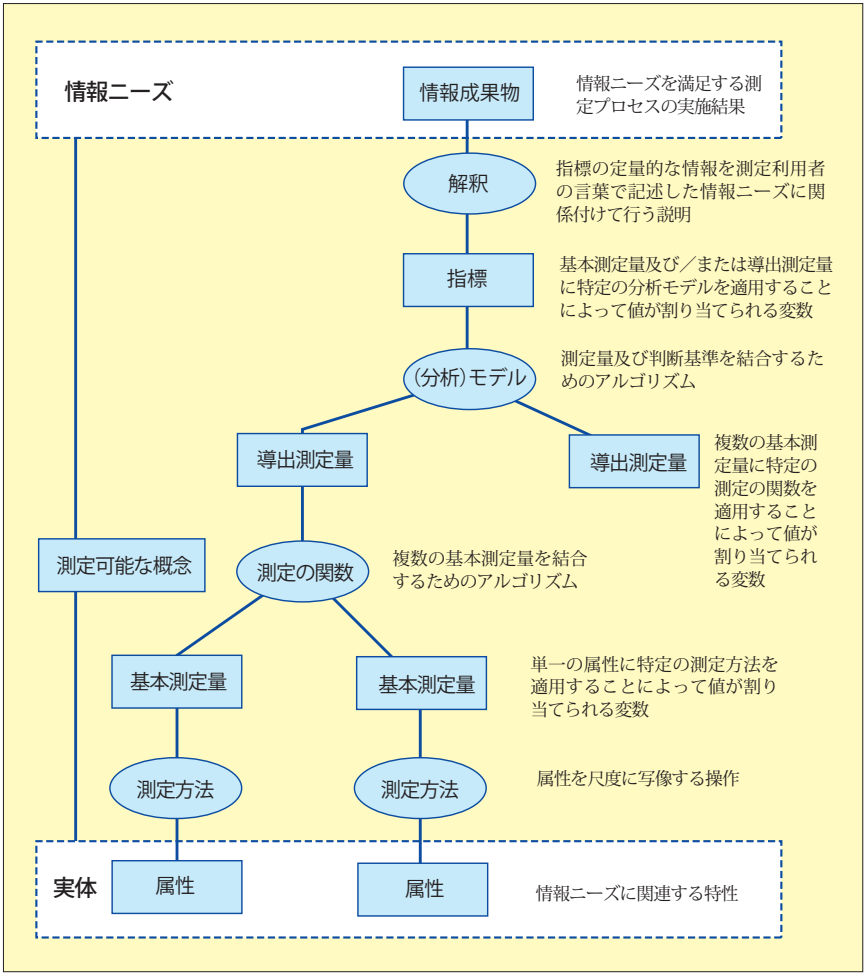


図表 A-1　測定プロセスモデル [1]

ニーズと呼ぶ)に基づいて測定を実施し、分析結果をこれらのプロセスにフィードバックすることを示している。

(2) 測定情報モデル

測定情報モデルは、測定要素間で矛盾のない測定の枠組みを示すためのものである(図表 A-2)。その実例を図表 A-3 に示す。



図表 A-2　測定の情報モデルにおける主要な関係図 [1]

情報ニーズ	生産物の設計時の品質を評価する。
測定可能な概念	生産物の品質
関連する実体	1. 設計文書 2. 設計検査報告書
(測定可能な) 属性	1. 検査の対象となる設計文書の本文 2. 検査で発見した欠陥のリスト
基本測定量	1. 設計文書 X の規模 2. 設計文書 X における欠陥数
測定方法	1. 設計文書の本文の行数を数える。 2. 報告書で列挙された欠陥数を数える。
測定方法の類型	1. 客観的 2. 客観的
尺度	1. ゼロから無限大までの整数 2. ゼロから無限大までの整数
尺度の類型	1. 比尺度 2. 比尺度
測定の単位	1. 行数 2. 欠陥数
導出測定量	検査による欠陥密度
測定の関数	設計文書ごとに欠陥数を規模で割る。
指標	設計の欠陥密度
モデル	欠陥密度の値を用いて，プロセスの平均及び管理限界を計算する。

図表 A-3 “ 品質 ” に対する測定の事例

＊図表は JIS からの引用であるが、この表の場合、1. は「設計文書 X」、2. は「設計文書 X の検査報告書」に関して述べている。

▶▶ 2. 測定の目的

測定の具体的な目的としては次のようなものが考えられる。

・提案されたプロジェクトの計画作成と評価の助け
・計画目標に対する実績の客観的な追跡
・プロセス改善の判断と投資の指針
・市場からの要求に対するビジネスと技術全体の実績の評価

▶▶ 3. 測定の効果

測定を実施することにより次のような効果が期待できる。

・情報交換を効率的に行う：組織全体に対する客観的な情報を提供することができる。また、受注側と発注側の組織間での意思疎通が促進される。
・プロジェクト固有の目標を追跡する：プロジェクトのプロセスと製品の状態を正確に把握できるようになる。
・問題を早めに特定し対応策を取る：リスクを客観的に特定することができるようになり、既存の問題をより詳しく評価・優先順位付けすることができるようになる。
・重要なトレードオフの判断を下す：コスト、スケジュール、機能、品質、性能のトレードオフの判断を支援する。
・判断を正当化する：各レベルの管理者に対して見積りや計画の根拠及び最良の選択の有効な根拠を示すことができる。

▶▶ 4. 測定の計画作成

測定プロセスは構造的で再現性のあるものでなければならない。そのためにもきちんとした測定計画を立案することが大切となる。測定計画の立案は次の順で行われる。

(1) 情報ニーズの特定と優先順位付け
(2) 測定量の選択と特定
(3)プロジェクトプロセスへの統合

測定プロセスを成功させるためには、プロジェクトの意思決定者の情報ニーズに直接関係する測定データの収集・分析・報告を行うことが重要となる。情報ニーズの特定にあたっては例えば次のようなことを考慮すべきである。

・リスクアセスメントの結果

・工数、スケジュール、品質の見積りに影響を与えるプロジェクトの前提や制約
・プロジェクトの成否を左右するような効果が大きい技術の有効性
・厳しい外部要件に対する実現可能なレベルの程度
・経験を持つチームの知見

測定量の選択にあたっては、例えば次のような基準を設けるのがよい。
・測定の有効性：プロセスや成果物の特性を直接測定し、必要な洞察を与える測定量か。
・領域の特性：例えば組み込み型システムではメモリ使用量を応答時間より優先することがある。
・プロジェクトの管理プラクティス：既存の管理プラクティスを手直しすることで測定要件を満たすようになるか。
・費用と利用可能性の関係を把握する。
・ライフサイクルのどの範囲で適用するか。
・システムコンポーネントの規模や対象製品：選択した測定量が適用できるか。

▶▶ 5. 測定の実施

実際の測定は次のような手順で実施する。
(1) データの収集と処理
(2) データの分析
(3) 提言の作成

データ収集時には、データの収集漏れや誤ったデータを収集することが少なくないので注意をする必要がある。

データの分析については次節で述べる。

提言の作成では次のような情報を盛り込むとよい。
・プロジェクトの全体的な評価
・具体的な問題、リスク、不足している情報の特定
・特定した根本的なリスクと、問題を解決するための代替アクションについての提言
・利点と欠点を併記して記述
・想定できる新しい課題

▶▶ 6. 分析の技法

分析の技法は次のようなものに分類される。

・見積り－規模、工数、スケジュール、品質等を対象
・実現可能性の分析－プロジェクト計画や再計画の作成時に実施
・実績の分析－計画に対するプロジェクトの実際の達成状況を評価

見積りについては、見積り部会で議論されており、またソフトウェア工学の教科書でよく取り上げられているので、ここでは特に記述はしない。

(1) 実現可能性の分析

実現可能性の分析では次の３つの状況を精査する必要がある。

・プロジェクトで優先度の高い情報ニーズに関連する測定量について、その実現可能性を分析する。
・優先度の高い情報ニーズに対しては、統合分析モデル(*1)の上流にあたる個々の要素も調査する。
・プロジェクト計画の個々の要素が、互いに一貫性を保っているかどうかを評価する。

（*1）統合分析モデルは、製品の規模と安定度、リソースとコスト、スケジュールと進捗、製品品質、プロセスパーフォーマンス等を構成する詳細なプロジェクト運営指標間の関係を示したモデルである。

(2) 実績の分析

実績の分析においては、例えば次のようなことに留意するとよい。

・プロジェクトで優先度の高い情報ニーズについては直接的な測定量で測定することが重要である。例えば、コストが最大の関心事であれば、コスト指標を常時監視する。
・主要な情報ニーズの上流に位置する情報ニーズは先行指標とみなせるため、常時監視の対象とする必要がある。
・２つの関連する指標を合わせてみることにより問題を検出できる場合がある。
・異常なデータ値が混在する場合は、その異常値を調査すべきである。

実績の分析プロセスでは、計画と実績を比較し、乖離があればその影響を評価する必要がある。また影響結果を予測して、問題があると考えられる場合は代替案を検討する必要がある。

▶▶ 7. 測定の評価

測定がひとまわり終了した後で、必ず、測定成果物、測定プロセスそのものの評価を行い、経験データベースを更新する必要がある。また、改善事項を決定して実施しなければならない。

▶▶ 8. コミットメントの確立と維持

測定の成功には組織的なコミットメントと責任の明確化が重要である。測定の実施にあたっては、リソースの準備や測定プログラムの進捗レビュー等も重要となる。

測定プログラムを成功させるいくつかの助言を次に示す。

- 小規模で開始する(主要な情報ニーズに絞った数個の測定量から開始する)。
- 測定コストの最小化を図る。
- 管理者がコミットメントを表明する。
- 全ての関係者に適切な訓練を実施する。
- 測定に基づいてコミュニケーションを促進し、その有効性を関係者に理解してもらう。

▶▶ 9. ソフトウェアで成功を収めるための測定

一流の実績を上げているソフトウェア企業では、測定プログラムを管理プロセスと技術プロセスに高度に統合し、測定プログラムを支援する企業文化を有している。うまく測定プロセスが回りだした組織では成功の証しとして次のような現象が見られる。

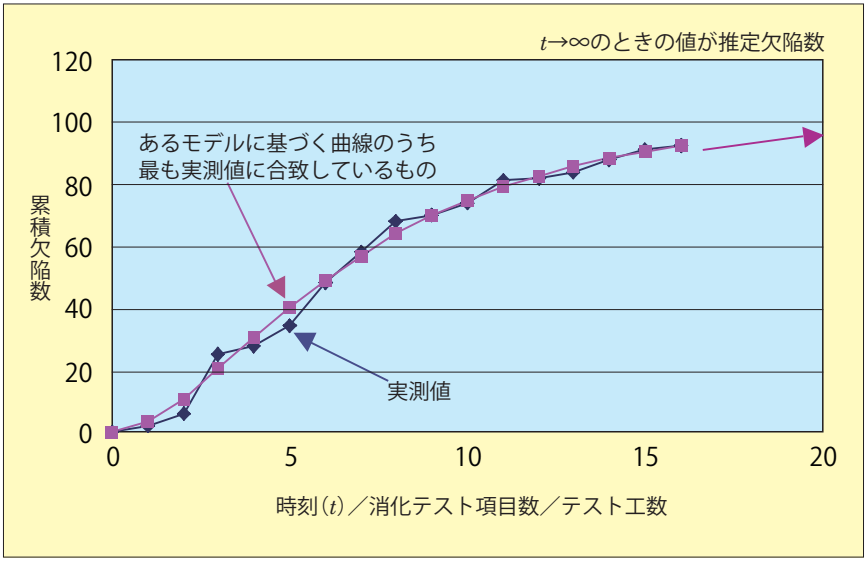
- データが自動的にかつ自然に集まる。
- データを広く利用できる。
- 人々は意思決定の際の基盤とするためにデータを得ようとする。
- 失敗は非難よりも理解につながる。
- 数値目標は、合理的な計画に裏付けられている。
- 定期的に測定プロセスを改善している。

▶B　ソフトウェア信頼度成長モデル

信頼度成長モデルによるソフトウェアの欠陥の予測は、最も初期の段階ではゴンペルツモデルが代表格であった。しかし、予測が外れるケースも少なくなかったため、ソフトウェア開発現場での評価は高いものとは言えなかった。しかし、その後の研究から種々のモデルが提唱された。現在では、こうしたモデルを統合的に扱う研究やツールの普及が進み、予測精度は格段に向上してきている。

▶▶1. ソフトウェア信頼度成長モデルとは

ソフトウェア信頼度成長モデル(SRGM)は、主にテスト工程で用いられ、与えられた累積欠陥データの傾向から総欠陥数を推定するものである。



図表 B–1　ソフトウェア信頼度成長モデル

図表 B-1 の横軸は、経過時刻だけでなく、消化テスト項目数や作業工数等を取ることが多い。あるモデルに基づく曲線のうち、与えられた累積欠陥データ(実測値)に最も合致するもの(モデルインスタンス)を選んで、それが $t \to \infty$

のときに示す値によって総欠陥数を推定するというものである。

例えば、指数形モデルでは、総欠陥数を N、欠陥検出速度を b とした場合、時刻 t での累積欠陥数 を次の式で表す。

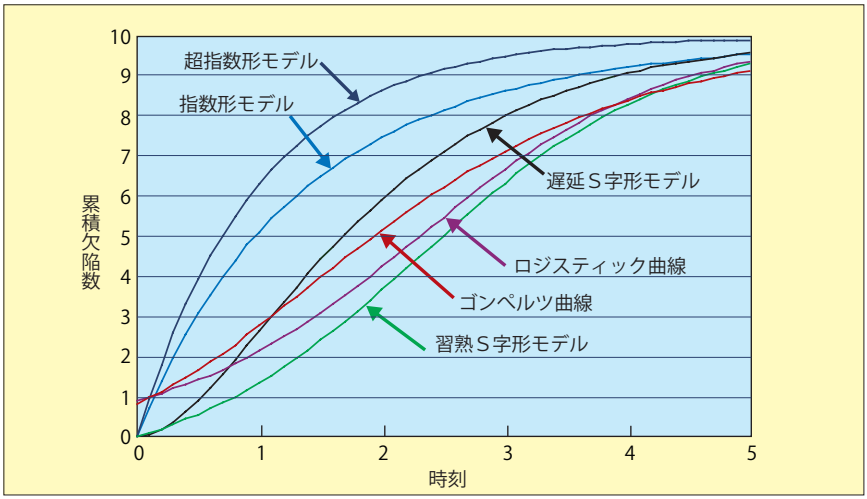
$$y=N(1-e^{-bt}) \qquad (1)$$

ここで最も実測データに適合するようなパラメータ N と b を求める。このときのパラメータ N の値が推定総欠陥数となる。

▶▶ 2. 既存の代表的な信頼度成長モデル

既存の代表的な信頼度成長モデルとしては、指数形モデル[4]、超指数形モデル、遅延 S 字形モデル[5]、ゴンペルツ曲線、シフトゴンペルツモデル(この資料では山田が提案した「ゴンペルツ曲線を用いた確率的ソフトウェア信頼度成長モデル」[6]をこのように呼ぶことにする)、ロジスティック曲線、習熟 S 字形モデル[7]等がある。これらのモデルは関数形が定まっており、与えられたデータに最も適合するように関数のパラメータを定めることにより、最適なモデルインスタンスを決定する。

図表 B-2 に上で述べた既存の代表的な SRGM を示す。図 B-2 に示されている各 SRGM に対応する曲線は、各モデルに対してそれぞれある特定のパラメータ値を与えて作成したモデルインスタンスであり、各モデルの 1 例に過ぎない。



図表 B–2　代表的なソフトウェア信頼成長モデル

▶▶ 3. 統合モデル

既存の代表的な SRGM を統合して記述できるモデルとして統合モデル[8] [9]や一般化ガンマソフトウェア信頼性モデル[10]がある。いずれのモデルも、形状に寄与する２つのパラメータの値を変えることにより累積欠陥データの形状に対応した様々な曲線を生成できるため、幅広い累積欠陥データに単独で対応できる。以下では、統合モデルを例にとってその様子を紹介する。

統合モデルでは、パラメータを変えることによって、指数形モデル、超指数形モデル、遅延 S 字形モデル、ゴンペルツ曲線、シフトゴンペルツモデル、ロジスティック曲線、習熟 S 字形モデルを表すことができる。

統合モデルの曲線の原型は次の式で表される。

$$y=N(1-\gamma\lambda e^{-\beta t})^{\frac{1}{\gamma}}=N(1-\gamma e^{-\beta(t-t_0)})^{\frac{1}{\gamma}} \qquad (2)$$

ここで N は総欠陥数、β は欠陥の検出速度を表すパラメータである。また、γ と λ（または t_0）は曲線の形状を表すパラメータである。

式(2)において 、$\gamma = 1$、$\lambda = 1$（$t_0 = 0$）とすると指数形モデルの式と同じになる。$\gamma = -1$ とするとロジスティック曲線と同じに、$\gamma \to 0$ とするとゴンペルツ曲線と同じになる。遅延 S 字形モデルは統合モデルの解ではないが、$\gamma = 0.463$ とした曲線と遅延 S 字形モデルが示す曲線の誤差は最大でも 0.6%程度となり、統合モデルの解が遅延 S 字形の非常に良い近似を与える。

シフトゴンペルツモデルと習熟 S 字形モデルは、それぞれゴンペルツ曲線とロジスティック曲線を $t = 0$ のとき $y = 0$ となるように y 軸方向に平行移動したものである。それに対応して統合モデルでは、式(2)を $t = 0$ のとき $y = 0$ となるように y 軸方向に平行移動し、かつ $t \to \infty$ のときの総欠陥数が N となるように定数倍した次の解も提供している。

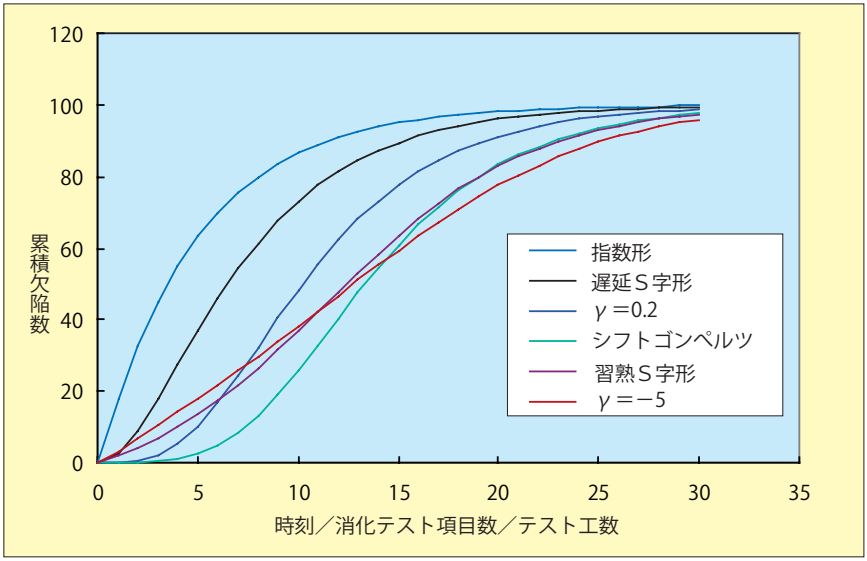
$$y= N\frac{(1-\gamma\lambda e^{-\beta t})^{\frac{1}{\gamma}}-(1-\gamma\lambda)^{\frac{1}{\gamma}}}{1-(1-\gamma\lambda)^{\frac{1}{\gamma}}} \qquad (3)$$

なお、統合モデルについては市販ツールも存在し、これを利用すれば、既存の単独のモデルのみを使用した場合に生ずるモデル不適合による誤差の増大の問題や、既存の複数のモデルを適用する場合に生ずる最適モデル選択作業の煩雑さから解放され、安定した精度で残存欠陥数の推定ができる。

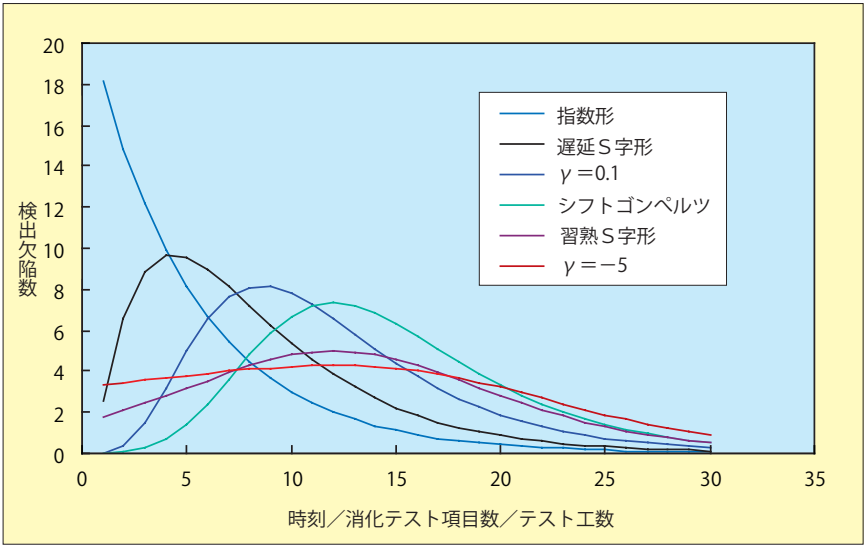
▶▶ 4. 統合モデルの解の形状の変化 [11]

式(2)と式(3)で示した統合モデルの解の形状をパラメータγの観点から比較したグラフを図表 B-3 と図表 B-4 に示す。図表 B-3 は累積欠陥数に対応する曲線の形状であり、図表 B-4 はそれを微分した、単位時間(または消化テスト項目数または単位テスト工数)当たりの検出欠陥数の変化を示している。図表 B-3 からわかるようにγの値を変化させることによって、単位時間当たりの検出欠陥数の立ち上がりと立ち下りの勾配を変化させることができる。γの値がマイナス 1 の場合、立ち上がりと立下りの勾配は同じであるが、γがマイナス 1 より大きいときは立ち上がりの勾配が急となり、γがマイナス 1 より小さいときは立下りの勾配が急となる。

このことは、あるモデルだけを利用することはある特定の傾向をもつ曲線のみを使って総欠陥数を推定することになり、推定誤差が大きくなることを示している。例えばシフトゴンペルツモデルだけを使うと立ち上がりが立ち下りより急な検出曲線だけを使うことになり、習熟 S 字形モデルが適合するような累積欠陥データに対しては推定誤差が大きくなる。統合モデルは既存の代表的な SRGM をカバーしているので、統合モデルを用いることにより図表 B-3 と図表 B-4 で示したような累積欠陥データを全て適用対象とすることができる。



図表　B-3　既存の代表的 SRGM のグラフ



図表 B-4　検出欠陥数の変化

▶C　必須となる記録項目、測定事例

必須となる記録項目 (例)

No		区分	定量化に必要なレベル	項目名	属性	収集対象工程						説　明
						基本設計	詳細設計	製作	単体テスト	結合テスト	総合テスト	
1	設計	レビュー実績	◎	レビュー日時		○	○	○				含　開始時刻、終了時刻
2				場所		○	○	○				
3			◎	レビューア延時間		○	○	○				参加レビューア数×レビュー時間
4			◎	参加者		○	○	○				レビューア／レビューイ／他がわかるようにする
5			△	レビュー対象規模		○	○					A4 換算ページ数
6			○	結論(決定事項)	テキスト	○	○	○				全て列記
7			○	アクションアイテム	テキスト	○	○	○				全て列記
8		指摘実績	◎	指摘内容／変更内容	テキスト	○	○	○				欠陥の内容と訂正内容(方針)
9			○	指摘箇所	テキスト	○	○	○				ドキュメント上の修正箇所情報
10			◎	指摘者		○	○	○				
11			◎	対象品質管理単位 ID ＊1		○	○	○				指摘の対象となる最小品質管理単位
12			◎	指摘観点の分類コード	コード	○	○	○				1：漏れ 2：誤り 3：改善 4：その他
13			◎	原因工程		○	○	○				欠陥が混入した工程
14			△	原因分類コード	コード	○	○	○				1：確認漏れ 2：誤解釈 3：設計漏れ
15			○	対処者		○	○	○				指摘事項のフォロー上は必須
16			○	対処完了予定日		○	○	○				同上
17			○	対処完了実績日		○	○	○				同上
18			○	確認日付		○	○	○				同上

・定量化に必要なレベル
　◎：収集必須項目
　○：定量的品質管理上は必須ではないが、管理上は必要な項目
　△：定量的品質管理上、あった方がよい項目
　空：プロジェクトにて任意
・属性　自明なものは省略
＊1：図表 2.2-1 の「●(最小の品質管理単位)」に相当する

No		区分	定量化に必要なレベル	項目名	属性	収集対象工程						説　明
						基本設計	詳細設計	製作	単体テスト	結合テスト	総合テスト	
1	共通	障害情報	◎	発生日					○	○	○	
2			○	発行日					○	○	○	
3			◎	障害処理管理No.					○	○	○	
4			○	発行元グループ					○	○	○	
5			○	発行者／確認者					○	○	○	
6			△	テスト項目ID					○	○	○	
7			△	発生工程					○	○	○	通常は現工程。品質強化作業の場合は強化対象の工程
8			○	発見手段	コード				○	○	○	1：机上 2：マシン
9			◎	事象分類コード	コード				○	○	○	1：システム異常 2：アプリ結果異常 3：性能劣化の区分
10			◎	事象概要	テキスト				○	○	○	障害事象の記述
11			◎	影響度					○	○	○	システム全体／特定業務等
12			◎	原因分類 一次	コード				○	○	○	1：アプリミス 2：環境ミス 3：データミス 4：オペミス 5：既障害 6：その他
13			△	原因分類 二次、三次					○	○	○	一次原因の細分類
14			◎	原因工程					○	○	○	欠陥が混入した工程
15			◎	障害原因	テキスト				○	○	○	欠陥(障害の原因)の記述
16			◎	対象品質管理単位ID[*1]					○	○	○	指摘の対象となる最小品質管理単位
17			△	対象プロセス名					○	○	○	同上
18			◎	対処方法	テキスト				○	○	○	要／不要
19			◎	水平展開要否	コード				○	○	○	水平展開すべきか否かの判定
20			○	水平展開影響箇所					○	○	○	
21			○	対処予定日					○	○	○	指摘事項のフォロー上は必須
22			○	対処完了日					○	○	○	同上
23			○	回答者					○	○	○	同上
24			○	完了確認日／確認者					○	○	○	同上
25			○	対処分類(資源・プログラム名)					○	○	○	同上

参考文献

［1］ JIS X 0141:2004 ソフトウェア測定プロセス

［2］ John McGarry, David Card, Cheryl Jones, Beth Layman, Elizabeth Clark, Joseph Dean and Fred Hall：Practical Software Measurement, Addison-Wesley, p.277, 2002.

［3］ 古山恒夫，富野壽監訳：実践的ソフトウェア測定，共立出版，p.250，2004.

［4］ Goel, A.L. and Okumoto, K.：Time -Dependent Error-Detection Rate Model for Software Reliability and Other Performance Measures, *IEEE Trans. Rel.*, Vol. R-28, No. 3, pp. 206-211, 1979.

［5］ Yamada, S., Ohba, M. and Osaki, S.：S-Shaped Reliability Growth Modeling for Software Error Detection, IEEE Trans. Rel., Vol. R-32, No. 5, pp. 475-478,1983.

［6］ 山田茂：ゴンペルツ曲線を用いた確率的ソフトウェア信頼度成長モデル，情報処理学会論文誌，Vol. 33, No. 7, pp. 964-969, 1992.

［7］ Ohba, M.：Software Reliability Analysis Modeles, IBM J. Res. Dev., Vol. 28, No. 4, pp. 428-443, 1984.

［8］ 古山恒夫，中川豊：ソフトウェア信頼度成長曲線に関する統合モデルと有効性の検証，情報処理学会ソフトウェア工学研究会，Vol. 97-10，pp.73-80, 1994.

［9］ Furuyama, T. and Nakagawa, Y.：A Manifold Growth Model that Unifies Software Reliability Growth Models, Int. J. of Reliability, Quality and Safety Engineering, Vol. 1, No. 2, pp. 161-184, 1994.

［10］ 岡村寛之，安藤光昭，土肥正，一般化ガンマソフトウェア信頼性モデル，電子情報通信 学会論文誌 (D-I)，vol. J87-D-I，no. 8，pp.805--814，2004.

［11］ 古山恒夫：ソフトウェア信頼度成長モデルにおける統合モデルの曲線の形状と実測データへの適用，ソフトウェアエンジニアリングシンポジウム 2007，pp.119-129，2007.

［12］ 21 世紀へのソフトウェア品質保証技術―日科技連ソフトウェア品質管理研究会 10 年の成果，日科技連出版社，ISBN 4-8171-6039-X，1994.

［13］ CMU/SEI, Capability Maturity Model Integration (CMMISM) Version1.1 Staged Representation

［14］ 財団法人 日本規格協会，JIS Q 9001:2000 品質マネジメントシステム 要求事項

［15］ 山田茂，高橋宗雄：ソフトウェアマネジメントモデル入門，共立出版，ISBN 4-320-02635-7，1993.

［16］ 中前雅之，桑田すみれ：定量的プロジェクト管理の改善に繋がるレビュー結果の層別，プロジェクトマネジメント学会，2007 年度，春季研究発表大会予稿集，pp.160-164，2007.

［17］ 細谷和伸，山本美子：見積り精度向上のための見積り標準モデル作成における

考慮点，プロジェクトマネジメント学会，2007 年度，春季研究発表大会予稿集，pp.101-104，2007.
[18] 桑田すみれ，岩切博，大曽根一将：実績データに基づく品質目標の達成度予測，プロジェクトマネジメント学会，2006 年度，春季研究発表大会予稿集，pp.268-271，2006.
[19] 岩切博，大曽根一将，藤原良一，中前雅之：高度化プロセスにおける定量的プロジェクト管理の実践と効果，プロジェクトマネジメント学会，2006 年度，春季研究発表大会予稿集，pp.389-393，2006.
[20] 神崎光司，藤野友也，平井規郎，中前雅之：データ分布を考慮したパフォーマンスベースラインの近似算出の有効性，プロジェクトマネジメント学会，2005 年度，春季研究発表大会予稿集，pp.193-197，2005.
[21] 神崎光司，藤原良一，中前雅之，平井規郎，藤野友也：統計分析による類似プロジェクト抽出手法の提案，プロジェクトマネジメント学会，2005 年度，秋季研究発表大会予稿集，pp.170-174，2005.
[22] 尾崎正博，中前雅之，山本美子：EVM 実践における落し穴とその解決策，プロジェクトマネジメント学会，2004 年度，春季研究発表大会予稿集，pp.113-117，2004.
[23] 神崎光司，藤原良一，中前雅之，伊藤満男，平井規郎：CMMI レベル 4 実現のためのパフォーマンスベースラインの提供とその活用，プロジェクトマネジメント学会，2004 年度，春季研究発表大会予稿集，pp.128-132，2004.
[24] IPA/SEC：経営者が参画する要求品質の確保～超上流から攻める IT 化の勘どころ～第 2 版，オーム社，ISBN 4-274-50076-4，2006.
[25] SQuBOK 策定部会編：ソフトウェア品質知識体系ガイド－ SQuBOK Guide －，オーム社，ISBN 978-4-274-50162-3，2007.
[26] 稲本稔：わかりやすい品質管理(改訂版)，理工学社，ISBN 4-8445-2269-8，1991.
[27] IPA/SEC：共通フレーム 2007 ～経営者、業務部門が参画するシステム開発および取引のために～，オーム社，ISBN 978-4-274-50156-2，2007.
[28] 冨永章：解説：アーンド・バリュー・マネジメント，プロジェクトマネジメント学会，ISBN 4-902378-00-0，2003.
[29] Project Management Institute：プロジェクトマネジメント知識体系ガイド第 3 版，Project Management Institute，ISBN 1-930699-75-1，2005.
[30] 土屋雅士，藤原良一：試験における Zone 評価の考え方，プロジェクトマネジメント学会，2008 年度，春季研究発表大会予稿集，pp.217-220，2008.

Index

た行

は行

ま行

ら行

【執筆・監修者】

定量データ分析部会　WG2（定量的品質予測）

服部　克己（日本ユニシス株式会社　定量データ部会 WG2 リーダ）
五味　　弘（沖ソフトウェア株式会社　定量データ部会 WG2 サブリーダ）
（以下五十音順）
小田　雅一（TIS 株式会社）
男澤　　康（日立ソフトウェアエンジニアリング株式会社）
合田　治彦（富士通株式会社、定量データ分析部会主査）
中村　　裕（リコーソフトウェア株式会社）
八谷　貴則（富士通株式会社）
藤原　良一（三菱電機インフォメーションシステムズ株式会社、定量データ分析部会副主査）
古山　恒夫（東海大学）
奥　　保正（ソフトウェア・エンジニアリング・センター）
三毛　功子（ソフトウェア・エンジニアリング・センター）

編 者 紹 介

独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター
2004年10月に独立行政法人 情報処理推進機構（IPA）内に設立されたソフトウェア・エンジニアリング・センター（SEC）は，エンタプライズ系ソフトウェアと組込みソフトウェアの開発力強化に取り組むとともに，その成果を実践・検証するための実践ソフトウェア開発プロジェクトを産学官の枠組みを越えて展開している。

［所在地］ 〒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
電話 03-5978-7543，FAX 03-5978-7517
http://sec.ipa.go.jp/index.php

SEC BOOKS
定量的品質予測のススメ
～IT システム開発における品質予測の実践的アプローチ～

平成 20 年 10 月 1 日　第 1 版第 1 刷発行
平成 20 年 11 月17日　第 1 版第 2 刷発行

編　者　独立行政法人 情報処理推進機構
ソフトウェア・エンジニアリング・センター
発 行 者　竹 生 修 己
発 行 所　株式会社 オ ー ム 社
郵便番号　101-8460
東京都千代田区神田錦町 3-1
電 話　03 (3233) 0641(代表)
URL　http : //www.ohmsha.co.jp/

表紙デザイン　下野　剛(志岐デザイン事務所)　　組版　高陵社　　印刷・製本　報光社
ISBN 978-4-274-50206-4　Printed in Japan

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

ソフトウェア開発見積りガイドブック
～IT ユーザとベンダにおける定量的見積りの実現～
A 5 判・240頁

ソフトウェア改良開発見積りガイドブック
～既存システムがある場合の開発～
A 5 判・172頁

ソフトウェアテスト見積りガイドブック
～品質要件に応じた見積りとは～
A 5 判・224頁

共通フレーム2007
～経営者，業務部門が参画するシステム開発および取引のために～
B 5 変形版・320頁

経営者が参画する要求品質の確保
～超上流から攻める IT 化の勘どころ～
第 2 版
A 5 判・128頁・CD-ROM 付き
※【事例検索システム】URL : https : //sec.ipa.go.jp/enterprise/index.php

プロセス改善ナビゲーションガイド
～なぜなに編～
A 5 判・124頁

プロセス改善ナビゲーションガイド
～プロセス診断活用編～
A 5 判・160頁

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～
A 5 判・208頁

組込みシステムの安全性向上の勧め（機能安全編）
A 5 判・72頁

もっと詳しい情報をお届けできます.
◎書店に商品がない場合または御注文の場合も右記宛にご連絡ください.

ホームページ　http://www.ohmsha.co.jp/
TEL/FAX　TEL.03-3233-0643　FAX.03-3293-6224

オーム社/雑誌局

ISBN978-4-274-50206-4

C3055 ¥1143E

9784274502064

定価(本体1143円【税別】)

1923055011434

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

SEC−TN08−004