解　説

ソフトウェアの品質定量評価とテスト管理†

東　基衛††　保田勝通†††

1. はじめに

コンピュータの技術や生産技術の進歩による，コンピュータの高性能化，低価格化によるコンピュータの普及は，ソフトウェアの品質に関する関心を呼び起こした．

ソフトウェアの品質に対する関心の高さは，この分野の図書や，国際会議，シンポジウムなどの急増で知ることができる．この分野の国際会議としては，当初 ICSE (International Conference on Software Engineering) や，COMPSAC (IEEE ComputerSociety's Computer Software and Application Conference) のようにソフトウェア工学の一分野として扱われていたが，最近では，応用技術よりも基礎技術を重視する傾向の強いヨーロッパでも，ソフトウェアの品質に関する国際会議は多い．ECSQ (European Conference on Software Quality)，はその一例である．

ソフトウェアの品質や生産性に関する共同プロジェクトやコンソーシアムも盛んである．アメリカの SPC (Software Productivity Consorsium) やヨーロッパの ESPRIT (European Strategic Programme for Research and Development in Information Technology) はその例である．

ソフトウェアの品質に関する研究は，当初はソフトウェアの信頼性の研究として発足した．前述の ICSE の前身が高信頼性ソフトウェアに関する国際会議であったことはその一例である．現在でもチェコでは毎年ソフトウェアの信頼性に関する国際会議が行われている．また，イギリスではロンドンの City University と Newcastle Upon Tyne 大学には CSR (Center for Software Reliability) がある．

しかし，最近ではソフトウェアの信頼性以外の品質特性にも注目するようになってきた．たとえばソフトウェアの問題のより多くは，保守に関するものであることから，保守性に関する関心も大きく保守性に関する国際会議が行われたりしている．

また，対話型ソフトウェアの普及，特に Macintosh の成功は，ソフトウェアのユーザインタフェース，つまり使いやすさに関する関心を一気に高めた．最近のイギリスの BBC で放送された (91-12-1) Dream Machine という番組でも，同じハードウェアを用いて，いかにソフトウェアが違うと使いやすさが違うかを見せている．

このような背景から，今日の品質評価は，信頼性はもとより，いろいろな特性について行う必要がある．本稿では，以下に品質評価技術とテスト管理技術の現状の展望を試みる．

2. 品質評価

2.1 ソフトウェアの品質評価技術の現状

2.1.1 概　要

ソフトウェアの品質評価に関連した技術としては，ソフトウェアメトリックスがあるが，これは，一般にソフトウェアを測ろうとするもので，複雑さのメトリックスや，モジュールに関するメトリックスのように品質に深い関係のあるものもあれば，ファンクションポイントのように，コストに関係の強いものもある．ソフトウェアメトリックスの分野では，ベルリン大学の Zuse の書いた "Software Complexity, Measurement and Methods"[1] 及び City University の CSR の N. Fenton の書いた "Software Metrics, A rigorous approach"[8] が注目される．

品質評価技術は，品質に関する概念，品質構造モデル，メトリックス，評価方法などに分けるこ

† Software Quality Evaluation and Test Management by Motoei AZUMA (Waseda University) and Katsuyuki YASUDA (Hitachi Ltd.).

†† 早稲田大学理工学部
††† (株)日立製作所

とができる．品質保証でよくいわれる，レビュやテストは，評価方法の一分野と考えられる．

ソフトウェアの品質評価技術，計測技術については最近研究が盛んに行われているが，以下にアメリカのメリーランド大学の GQM 及びヴァージニア工科大学の OPA ならびに ヨーロッパの ESPRIT プロジェクトの現状を紹介する．

2.1.2 メリーランド大学の GQM

メリーランド大学 (University of Maryland) はアメリカのワシントン DC に近い College Park という町にあり，場所柄か，政府関係の研究が多いようである．ここのコンピュータ科学科の V. Basili, D. Rombach らは，TAME[2] というプロジェクトで，GQM (Goal, Question, Metric) というパラダイムに基づいた評価技術を開発している．Basili はすでに何度も訪日し，日本でもよく知られており，また TAME 及び GQM についてもすでにこれまでに，紹介されているので詳しくは触れないが，GQM は，簡単に言えば，品質の目標に基づいて多くの質問を用意し，それらの質問に対応したメトリックスを用いようとするものである．目標設定のためにはテンプレートが，また質問とメトリックスを設定するためにはガイドラインが開発されている．目標は，プロセスに関するものと，プロダクトに関するものがあり，たとえば“信頼性に関連してテストプロセスを……”というように具体的に設定する．目標は副目標に細分化する．質問はたとえば“最終製品の複雑さはどれだけか？”というように副目標に対応して定められる．このアプローチの成功の鍵は，どのような質問とメトリックスを用意できるか，また，いかにこの技法の利用者が GQM をそれらの中から選定，利用するかについてのガイドの適切さ，分かりやすさにあるといえよう．

2.1.3 ヴァージニア工科大学の OPA

ヴァージニア工科大学 (Virginia Polytechnic Institute and State University : VT) は，アメリカ南部 Virginia 州の Blacksburg という小さな町にある．

VT では，Dr. Nance と Dr. Arthur が中心になって，US Navy と共同でソフトウェアの品質評価技法の研究[3] を行っている．彼らのアプローチは，OPA (Objective, Principles, Attributes) という．この方法では，まず，ソフトウェア品質の目標 (Objectives) を明らかにする．目標は，Adaptability, Correctness, Maintainability, Portability Reliability, Reusability 及び Testability である．最後の 2 者はどちらかというと開発者の要求を反映したものであるが，ほぼ ISO/IEC 9126 の品質特性に対応する．

原理 (Principles) は，良いソフトウェアを開発するための原理であり，階層的分解 (Hierarchical Decomposition)，情報隠蔽 (Information Hiding)，段階的洗練 (Stepwise Refinement) など，ソフトウェア工学を学んだ人にはお馴染みの原理が並んでいる．属性 (Attributes) は，これらの各原理がその向上に寄与するとみられるソフトウェアの属性で，複雑さ (Complexity)，強度 (Cohesion)，結合度 (Coupling)，読みやすさ (Readability)，行動の可視性 (Visibility of Behavior) などがリストされている．O, P, A の間の関係は，線で示されているが，どれくらいの関係かは，文献調査に基づいて，調査対象文献数に対する関係ありと認めている文献数の分数で示されている．したがって 1 に近いほど関係は強いことになる．

各属性がどれくらい良いかを示す指標として，SQI (Software Quality Indicators) が用いられる．SQI は，各属性に対して何種類か用意されている．値は 1 から 10 までの範囲で 10 点満点である．SQI にはコードを測るものとドキュメントを測るものがあり，前者は現在 66 種類，後者は 32 種類ある．O, P, A 及び SQI の関係を図-1 に示す．

このほか，VT には，Dr. S. Henry[10] が Nance らのプロジェクトとは別にソフトウェアメトリックスの研究を精力的に行っている．

2.1.4 ESPRIT

ESPRIT は，EC 諸国が，出資して作っているプロジェクトで，実際には多くのプロジェクトの集合体である．その仕組みは，参加各国がそれぞれ出資し，ジュネーブにある ESPRIT の本部に集める．EC 内の企業または大学，研究所などは，毎年この ESPRIT 本部にプロジェクトの提案を行うことができる．提案には，厳密にはどのような規則や制限があるのかは知る由もないが，大抵は数カ国にわたり，また，企業，大学などの異種の機関を包含している．また，提案に対する採用の比率は年々高くなる傾向にあるが，ほぼ 10 対

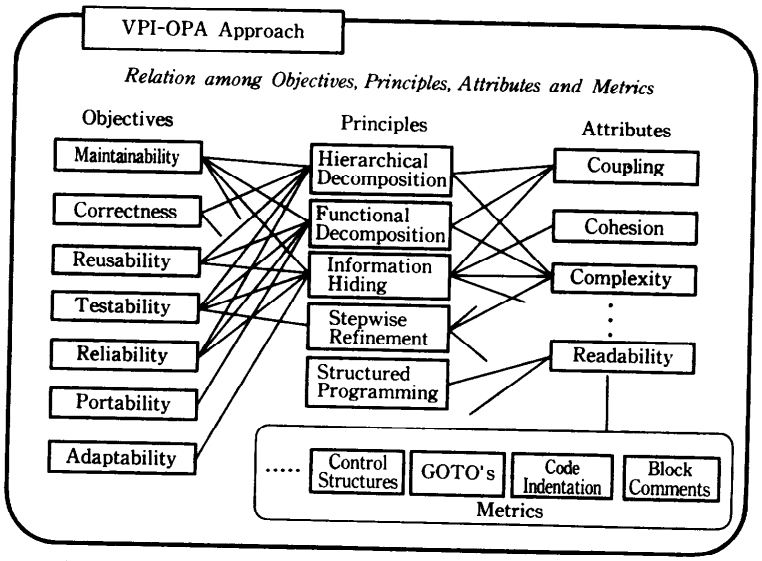

図-1 VPI の Objectives, Principles, Attributes とメトリックスの関係

1といわれている.

ESPRITは現在第2期目に入り, ESPRIT 2と呼ばれている. これは, 92年3月で終了する. 現在は第3期の提案を募集, 審議中である. 第1期の成果としては PCTE (Portable Common Tools Environment) が知られている. これはその後改訂され, PCTE-2 として, ECMA が標準化に取り組んでいる.

ソフトウェア品質評価に関連したプロジェクトとしては, METKIT, SCOPE, AMI などがある. METKIT (Metrics Education Kit) は, ソフトウェアメトリックスの教育を行うためのプロジェクトと同名の METKIT というキットを開発しており大学などの教育機関用モジュールと企業内教育を目的とする産業界用モジュールがある. 各モジュールはそれぞれ教育者用と受講生用の教材から構成される. 教材は構造化されており, 全体像の教育から始まり, 次第に詳細化される仕組みである.

AMI (Applying Metrics Industry) は新しいプロジェクトで, Metrics Users Handbook を開発するとともに実際にコンサルティング活動を行おうとしている.

ESPRIT には多くの企業, 大学, 研究機関が参加しているが, この分野の主な大学, 研究機関としては, イギリスではロンドンの City University と Newcastle Upon Tyne 大学の両 CSR (Center for Software Reliability), グラスゴーの Strathcryde University 及び筆者の滞在した, SouthBank Polytechnic, ドイツでは GMD などがあげられる. また, イギリスの企業では, NCC (National Computing Center), BT (British Telecom), GEC-Marconi Software System, コンサルタント会社の Brameur, SEMAGroup, イタリアでは, DIDA*EL, フランスでは BUll, Verilog などがあげられる.

2.2 国際標準と日本の INSTAC/STD の評価技術

ISO/IEC JTC1/SC7 は, "9126: Evaluation of Software—Quality Characteristics and Guideline for Their Use" (ソフトウェア製品評価のための品質特性とその利用の手引き) の標準化の推進を行った (1991 年 12 月発行).

この規格は, 品質特性をソフトウェアの品質を利用者の視点から評価するための特性すなわち属性の集合として定義し, 機能性 (Functionality), 信頼性 (Reliability), 使用性 (Usability), 効率性 (Efficiency), 保守性 (Maintainability), ならびに移植性 (Portability) の6つを取り上げ, 評価プロセスとともに標準化したものである. 各品質特性はさらにいくつかの副特性に詳細化されるが, ISO/IEC 9126 ではこれらの副特性は, ANNEX にし, 参考にとどめている. これらの用語は, 従来研究者や企業, 学会の中の一コミュニティがそれぞれ定義なしに, あるいは勝手に定義して使用していた. これらの定義を標準化した結果, ソフ

トウェアの品質に関係する人は，ソフトウェアの品質という曖昧な言葉の代わりに，より具体的な言葉を用いて要求を定義したり，評価したりできるようになった．

評価プロセスモデルの特長は，計測，評定，及び総合評価の3段階に分けたことである．計測は，ソフトウェアの一つの品質特性，または副特性に対応する計測対象を客観的に測ろうとするものである．計測した結果が良いか悪いかは，ソフトウェアの種類，利用者に関係なく決まるものもあるが，その多くはそれらとの関係で決まる．たとえば応答時間が1秒といっても，何の応答かによって非常に良い値かもしれないし，または使いものにならないくらい遅いかもしれない．計測結果がどれだけ良いかの判断が評定である．また，総合評価は特性相互間や，特性とコストや納期など特性以外の要因とのトレードオフを考慮した評価である．

この 9126 関連プロジェクトを支援するため，日本では日本規格協会の中に設置された情報処理技術標準化調査検討委員会（NISTAC）の中の対応する委員会（STD，委員長：大野豊）に作業グループ（WG 5，主査：東基衛）を設置し検討を行ってきた．STD は毎年1回報告書[4),5)]を出版してきているが，その WG 5 はこれまで，ソフトウェアの品質の構造，品質を利用者視点からみた 9126 の品質特性を詳細化した副特性，及びそれらに対応するメトリックス，ならびに内部特性とそれに対応するメトリックスなどを開発定義してきた．ISO/IEC 9126 の品質特性が利用者視点から特性を選定したのに対し，内部特性は，たとえばモジュール性や追跡可能性のように，開発者のみに関心のあるような特性を選定したものである．91 年度は最終年度として，その総まとめを行っている[6)]．

この WG ではさらに提案されたメトリックスを分析した．その結果，広義のメトリックスと狭義のに分けることを提案している．一般に，ある特性の良し悪しは，たとえば KLOC (Kilo Lines of Code；ソースコード千行）あたりの障害数，のように比率などの数式で表わされる．これを指標 (Indicator) という．指標の数式を構成する KLOC のような要素をデータエレメントという．データエレメントを計測するための尺度と方法を狭義のメトリックスという．広義のメトリックスは，指標を包含する．

指標には，外部特性すなわち品質特性及び副特性に対応するものと内部特性に対応するものがある．これらを用いて，品質特性や外部特性の良し悪しを知ることができる．しかし，開発途中では，多くの場合品質特性に対応した指標は，使用できない．この場合内部特性の良し悪しから，品質特性の良し悪しを予測する必要がある．このためには内部特性と品質特性の関係が明らかでなければならないが，現在のところヴァージニア工科大学のアプローチと同様，関係の有無が明らかにされているにすぎない．

2.3 品質評価の方法とその実際

SC7/WG6 では，現在実際に 9126 を利用する際の手引きの開発に取り組んでいる．これは，基本概念の解説，Metrics Plan，GMRA-1 (Guide to Measurement, Rating and Assessment：開発者用）と，GMRA-2（購入者用）から構成される．しかし，まだ原案の原案というような状態で，実用のレベルに到達するまでにはまだしばらく時間がかかると思われる．

現実問題としては，コストや納期の問題もあるため，いくら詳しい計測，評価が可能な技術が開発されたとしても，無制限に使用するわけにはいかない．

SQMAT[7)] (Software Quality Measurement and Assessment Technology）は，詳しさのレベルを3通りに分けた計測方法により，重要な特性を重点的に詳しく計測，評価することを提唱している．これにより，相対的に重要度が高い特性が十分に評価される利点だけでなく，重要度が低いとされる特性が評価されることなく出荷されることを防ぐことに意義がある．

品質評価の進め方は，現時点ではこれでないといけないといえるようなものはないが，品質特性の定義については，ISO/IEC 9126 に準拠してゆくことが望ましい．すでに，ヨーロッパではいくつかのソフトウェアの品質に関する国際会議に出席したが，国際標準の出版を待たずに，利用が始まっていることを示す多くの発表に出会った．品質評価を推進するための要点としては，次のようなものがあげられる．

- 機能要求だけでなく，全ての品質特性につい

て品質要求を検討し仕様化する.

● 品質要求に基づいて，プロジェクトの計画時に，メトリックス利用計画を加える.

● プロジェクトの進行にともない記録すべきデータを明らかにする．後からでは採れないか，採るのが困難なデータがある.

● 品質評価は，重点指向にし，重要な特性は厳密に評価する．しかし，全ての特性をなんらかの形で評価する.

● GQM，OPA または INSTAC/STD/WG 5 のどのアプローチを採用することも可能である.

● メトリックスは，当面その精度がはっきりしているものから使用してゆく.

● 主観的な評価を併用する.

3. テスト管理

3.1 テストの考え方

（1） テストの位置づけ

図-2 は，ウォータフォール型開発モデルにおけるテストプロセスの位置づけを概念的に示したものである．各設計プロセスと各テストプロセスは対応している．テスト工程は，内部仕様に対応したモジュールテスト，外部仕様に対応したプログラムテスト及び要求仕様に対応したシステムテストの三つに大別される.

（2） テスト方法論[11]

テスト方法論には，ソフトウェアを外部仕様のみからテストするブラックボックステストとソフトウェアの内部仕様に基づいてテストするホワイトボックステストの二つがある．この二つのテスト方法論はおのおの一長一短があり，いずれか一方だけでは不十分であり，両者を併用することが必要である．図-2 に，各テストプロセスとテスト方法論の典型的な対応付けを示す.

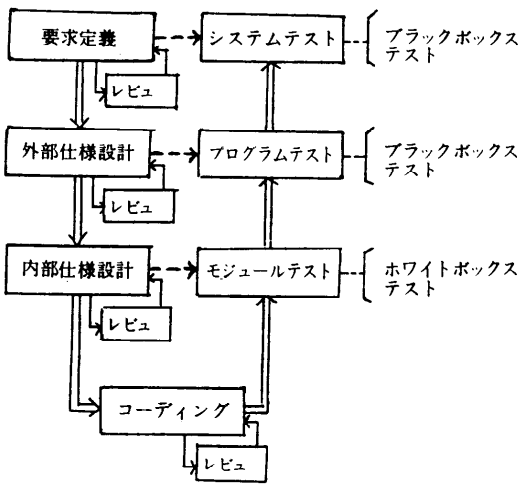

図-2 テスト工程の位置づけ

（3） レビュとテスト

レビュには，デザインレビュ，ウォークスルー，コードレビュ/コードインスペクションなどいろいろの種類がある．いずれにせよ，テストと並んで各プロセスの検証に欠かせない手段である．それではレビュとテストの関係はどう考えるべきであろうか．作り込まれた不具合は，可能なかぎり，早い段階で除去すべきというのが品質管理の鉄則である．すなわち，設計・製造工程の各レビュで徹底的に不具合を摘出し，後工程に持ち込まないようにすべきである.

（4） テストの定義[12]

ソフトウェアのテストとは：

定められた基準にしたがって，ソフトウェアが仕様どおりに正しく動作することを確認し，保証する．または，正しく動作しない部分を摘出する

ことをいう.

3.2 テスト技術の現状

3.2.1 テスト技術[13),14)]の体系

テスト技術は，テスト設計技術とテスト実行技術及び評価・予測技術に大別される．テスト設計技術には，ブラックボックス方式とホワイトボックス方式の二つがある．ブラックボックス方式は，テスト対象のソフトウェアをブラックボックスとみなすテスト方式である．すなわち，ソフトウェアの内部構造は考えず，ソフトウェアの入力と出力のみに着目するテスト方式である．このような観点でとらえたソフトウェアを論理構造モデル[15]と呼ぶ．この論理構造モデルは，入力の組合せだけで一義的に決定されうる空間モデルと，入力相互間に順序（時間）関係がある時間モデルに大別される.

一方，ホワイトボックス方式は，テスト対象のソフトウェアの内部構造に着目したテスト方式である．すなわち，実際に実現されるソフトウェアの構造すなわち物理構造モデル[15]を意識して実施するテスト方式である．この場合，ソフトウェアの制御構造に着目した制御構造モデルに基づくテ

スト方式と，データ構造に着目したデータ構造モデルに基づくテスト方式がある．テストの実施にあたっては，上記の2方式を併用することが必要である．

テスト実行技術は，直接実行方式とシミュレーション実行方式に分けられる．後者の方式は組込型ソフトウェアなどでテスト環境をシミュレーションし，ソフトウェア論理のテストを行うものである．直接実行方式は，テスト実行支援と結果検証支援で構成される．モジュールテストやプログラムテストについては，最近の CASE[16]（コンピュータ支援ソフトウェア工学）ツールにより，ワークステーション上でおおむね実行可能となっている．デグレードの防止とテスト効率の向上のためには，テスト資産の再利用を行う繰り返しテスト支援の充実が重要である．

評価・予測技術は，テスト自体の網羅性の評価技術と，対象ソフトウェアに対する信頼性予測技術からなる．前者の技術は，ホワイトボックス方式の制御フローに基づく計測尺度とツールが実用化され普及している．後者の信頼性予測は，信頼度成長曲線による予測が主体であり，各種モデルが提案されている．開発テストと別に独立したテストグループが存在する場合には抜取評価法による予測も有力な手法である（表-1 参照）．

なお，本項で取り上げた技術は，テスト管理の前提となる技術という観点に立ち，実用化された技術に限定していることをお断わりしておく．

表-1 テスト技術の体系

区分	方式	技術区分
テスト設計技術（テストケース設計技術）	ブラックボックス方式	空間モデル（組合せ論理）
		時間モデル（順序論理）
	ホワイトボックス方式	制御構造モデル
		データ構造モデル
テスト実行技術	直接実行方式	テスト実行支援
		結果検証支援
	シミュレーション実行方式	シミュレーション支援
評価予測技術	テスト網羅性評価	ホワイトボックステスト網羅性尺度
	信頼性予測	信頼度成長曲線による予測
		抜取評価法予測

3.2.2 テストケース設計技法[13),17)]

ブラックボックス方式すなわち外部仕様書に基づくテストケースの設計の場合，テストケース要因の網羅的抽出技術と，合理的最小組合せ技術が鍵となる．通常，外部仕様が文章で記述されているため，同値分割法，限界分析法という入力域の分割・解析により設計する．例外処理，特に障害処理解析には FTA（フォールト・トリー解析）などが有効である．一方，仕様書が，ソフトウェア構造に基づき，決定表あるいは状態遷移図のような形式で記述されている場合には，より精度の高いテストケースの設計が可能となる．その場合，一般にテスト要因の組合せが膨大になり，全組合せを実施することは現実的でない．そのため，原因結果グラフ技法[13)]，実験計画法に基づく技術[17)]や AGENT 技法[18)]などでは，合理的に組合せ数を減少する方式を採用している．こうして作成されたテストケースに対して，さらに過去の障害事例をフィードバックすることが重要である．

ホワイトボックス方式すなわち内部仕様書に基づくテストケース設計の場合，制御構造に基づくテストカバレージ尺度を基準とする方法が一般的である．これにデータ構造及びデータフロー分析に基づくテストケースを加えることが望ましい．以上をまとめたのが表-2 である．

3.2.3 信頼性予測技術[20)~23)]

ソフトウェア信頼性の実現レベルの評価は次の二つに大別される．第一は，故障の現象として表われるエラー発生間隔による評価である．第二は故障の原因である，発見されたエラー数による評価である．信頼度の経時的な変化を信頼度成長と呼び，信頼性予測にあたっては，信頼度成長の分析に基づく予測が重要である．

（1） 故障発生間隔の予測

時間計測モデルと呼ばれる確率・統計モデルが提案されている．古典的なモデルとして有名な Jelinski と Moranda 及び Shooman やこれを発展させた Musa のモデルはソフトウェア故障の発生時間分布を指数分布とする指数ハザードレートモデルと呼ばれるものである．さらに故障の発生時間分布をワイブル分布とするモデルや，ベイズ統計学に基づくパレート分布による Littlewood らのモデルがある．

表-2　テストケース設計法[10]

区　分	構造モデル	テストケース設計法	
		仕様記法例	テストケース設計法
ブラックボックス方式	空間モデル（組合せ論理）	文章記述決定表	同値分割法 限界値分析法 障害解析法（FTA 他） 原因結果グラフ技法 実験計画法
	時間モデル（順序論理）	状態遷移図 機能図式	AGENT 技法
ホワイトボックス方式	制御構造モデル	モジュール内	C0：全ステートメント C1：全分岐条件
		モジュール間	S0：全モジュール呼出し S1：モジュールの全コール関係
	データ構造モデル	構造内	D0：全データ要素の設定・参照
		構造間	D1：データ階層の親子関係の組合せで全データ要素の設定・参照

（2）　残存エラー数の予測

この一つは，我が国で 1970 年代から研究・適用されてきたロジスティック曲線やゴンペルツ曲線[24]を適用する決定論的モデルであり，現在でも開発現場では品質管理の手段として多用されている．

第2は，個数計測モデルと呼ばれる．発見されたソフトウェアエラー数に基づく確率・統計モデルである．これは，指数型分布のほかに，非同次ポアソン過程（NHPP）[25]に基づく各種の提案が行われている．我が国でもモデルの仮定をより現実的なものに拡張したS字型モデルがいくつか提案されている．さらにテスト過程をモデル化した超幾何分布モデル[26]が提案されている．

第3は，サンプリング法を改善し，残存エラー数を精度よく推定する方法である．埋め込みエラーによる捕獲・再捕獲法や2段階エディット法[20]が提案されている．2段階エディット法は，二つの独立したテストグループに別々にテストをさせ，共通して発見されたエラーから残存エラーを摘出する方法である．

3.3　テスト管理の実際

3.3.1　テスト工程の品質管理[27]

テスト工程における品質管理は，まず内在する不良の予測，すなわち摘出すべき不良の目標値を設定することから始まる．テストを開始すると不良の発生状況を監視し，管理限界を超えた場合はさらに不良の詳細分析を行う．

この結果，品質改善の必要性があれば，積極的に品質向上施策を推進する．この活動は図-3 に示すように稼働品質も含む大きな品質目標値管理サイクルの一部に位置付けられており，評価データの蓄積，稼働品質の予測も重要な作業になっている．

（1）　不良予測と目標値管理

不良予測については，時系列的な不良の累積曲線を成長曲線などに当てはめる手法を用い，これに基づく不良摘出推移を評価している例が多い．テスト開始前に上限，下限の二つの曲線からなる標準的な不良管理曲線，及び不良摘出目標値を設定する．これは過去の類似プログラムの実績を参考に，当該プログラムの特徴，プロジェクトの特徴などを加味し設定する．この作業は一連の品質管理の手順のうちでも重要なものである．

テストが進み，時系列の不良累積実績が出てきた時点で，この不良累計実績をもとにして，その後の不良発生予測を行う．この予測値，あるいは実績値そのものが，テスト開始時点に設定した管理曲線の上限，下限を超えた場合は，詳細な品質分析を行い，問題点を追及する．これに基づき品質向上施策を立案し，推進する．テストの後半では，不良実績に基づく不良発生予測曲線の，収束値に対する現在の実績比，すなわち不良収束率を重点的に監視する．

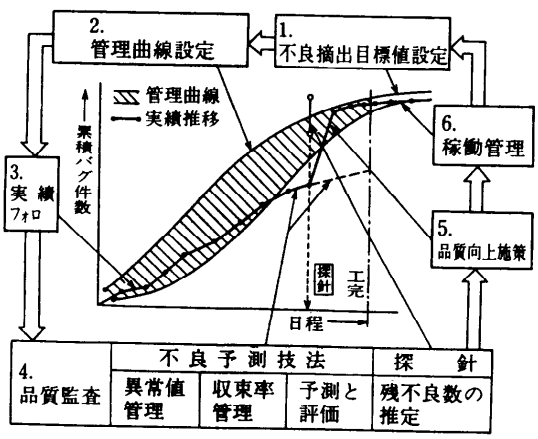

図-3　品質目標値管理の概念[17]

（2） 探　針[24]

探針とは，テスト段階における品質を，テスト部隊とは別のグループ，通常は検査部門が実際にサンプリングテストをして測定・評価するものである．このときの摘出不良件数から，潜在不良件数を統計手法で推定するとともに，不良内容の分析・評価により，弱点を具体的に指摘する．

これにより，テスト段階での不良の先取りを加速し，品質向上施策への指針を与えようというものである．

探針は，基本的には検査項目をサンプリングし，その結果から全体の母不良率を求め，残不良件数を推定する方法である．

（3） テスト推捗管理

テスト工程においては，特に進捗管理と品質管理は密接な関係がある．品質が悪いとテストが進まず，テストが進まないと品質も向上しない．したがって，テスト項目の消化状況と不良発生状況及びテストカバレージ率などを同一グラフ上に重ね合わせて，相互の関係を検討することが大事である．いずれにしても，眼に見える管理，ビジュアル化が重要である．

3.3.2 テスト工程の信頼性評価尺度[24]

テスト工程の管理を行うためには，テスト工程の進捗管理は当然であるが，テストの過程で所定の品質が得られているかどうかの見極めが重要である．適切な管理を行うためには，適切な尺度を用いて計測し評価する必要がある．表-3 に示した尺度例は，開発現場で実際に使用されているものである．ここでは，テスト対象のソフトウェアの信頼性品質特性のうちの成熟性及び可用性に関するものの一部と，テスト自体の品質の尺度を示した．

3.3.3 テスト完了基準

ソフトウェア製品を出荷するにあたってのテスト完了基準は，本来残存不良がなくなったことを見極める基準であるべきだが，それは一般に困難なため，以下の観点から総合的に判断する．

（1） 予定したテストあるいは検査がすべて終了

（2） 摘出された不良あるいは問題点の対策完了

（3） 最終工程での品質水準が合格基準を達成

（4） 最終工程での品質が収束し，安定している

これらの判断の前提としてテスト自体の質が一定以上であることが保証されていることが必要である．実際の出荷にあたっては信頼性以外の品質特性の評価や提供ドキュメント品質についても評価が完了している必要があることはいうまでもない．

表-3 テスト工程の信頼性評価尺度例

項番	品質特性	適用工程			計測尺度	定義
		開発	検査	稼働		
1	成熟性	○	○	○	障害密度	障害件数/規模（Kステップ）
2		○	○		障害率	障害件数/テスト項目数
3		○	○	○	故障率	故障件数/マシン時間
4		○	○	○	平均故障発生間隔	稼働時間/故障件数
5		○	○	○	平均ダウン発生間隔	マシン時間/ダウン件数
6		○	○		障害収束率	累積摘出障害数/推定障害総数
7			○		探針不良率	探針不良件数/探針実施項目数
8		○	○		現象別障害比率	現象別障害件数/全摘出障害件数
9		○	○		原因別障害比率	原因別障害件数/全摘出障害件数
10	テスト品質	○	○		テスト密度	テスト項目数/規模（Kステップ）
11		○	○		テスト実施率	実施済テスト項目数/実施予定テスト項目数
12		○			テストカバレージ率（C0）	実行済ステートメント数/全ステートメント数
13		○			テストカバレージ率（C1）	実行済分岐数/全分岐数
14	可用性		○	○	稼働率	実稼働総時間/予定稼働時間
15			○	○	平均復旧時間	復旧に要した総時間/ダウン回数
16				○	重症不良件数頻度	件/月

3.3.4 テスト管理事例

開発過程をとおして，不良件数に基づく信頼性の管理・評価を行う事例を2件紹介する．

（1） 品質会計制度[27),28)]

これは，設計・コーディング工程で作り込んだバグを負債とみなし，各工程でのレビュ及びテストによるバグ摘出によってこの負債を返済し，負債が0になった時点で出荷可とするものである．図-4 にこの考え方を示す．

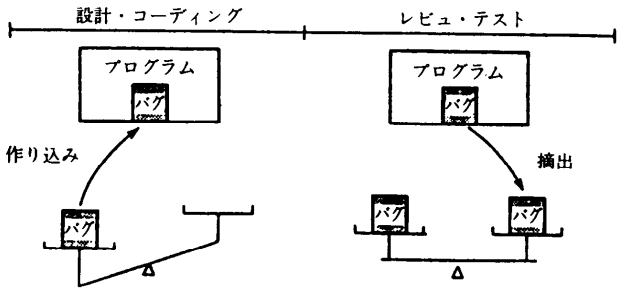

図-4 品質会計制度[28)]

まず，各工程での作り込みバグ数を予測し，貸方に記す．当該工程を実施するとレビュやテストで除去したバグ数を借方に記録する．工程の区切り目において勘定を締める．貸方と借方の差が未摘出バグであり，これは次工程に引き継ぎ，その貸方に記載する（図-5 参照）．

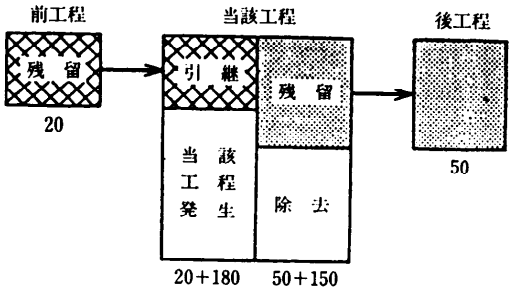

図-5 工程における品質会計[27)]

（2） ソフトウェア品質評価システム (SQE)[29)]

これは，設計から検査までの開発工程を中心に信頼性に着目した評価システムである．このSQE は図-6 に示すとおり，信頼性設計と品質目標値設定を軸とする設計品質評価サブシステムと，テスト段階の目標値管理を含むテスト品質評価サブシステム及び総合品質評価サブシステムの三つからなる．テスト品質評価サブシステムは図-3 を適用した事例である．

3.4 テストの研究動向と課題

現在テスト技術の分野では，データフロー解析の研究が大きなテーマとなっているが，まだ実用技術という段階には至っていない．しかし，このような観点に立ったレビュやテストケース設計は有効であろう．さらに，通信分野など形式記述法が導入されつつある分野では，テストケースの自動生成のようなテスト分野への応用研究が期待される．

より現実的な課題としてはCASEツールにおけるテスト機能の充実があげられる．その一つとしてリポジトリの有効活用によるテストケース・テスト結果の蓄積と再利用があ

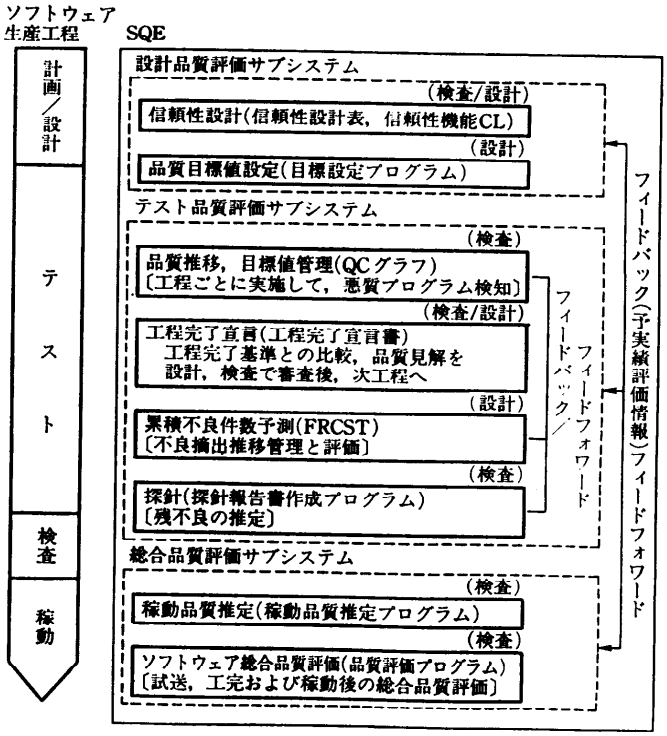

図-6 SQE 概要[29)]

る．最近進歩の目ざましいGUI（グラフィカル・ユーザ・インタフェース）による画面出力を正確かつ効率的に検証するための出力結果検証支援の強化が当面の課題である．さらに近年急速に進展しつつある分散開発環境下でのテストを統合的に管理（品質，工程管理）する方式の実現があげられる．

4. おわりに

ヨーロッパにいて，よく聞く意見は，「日本ではメトリックスを実際に利用してよくデータを採っている．われわれもメトリックスを利用し，品質管理をきちんと行わないと，日本にいつまでも追いつくことはできない．」というもので，実態を知る者としては何やらこそばゆい感じがしないでもない．

アメリカでは，日本のほうが進んでいるというクスマノ氏の意見と，日本はまだそれほど進んでいないとするSEIの意見が分かれて話題となっているが，ある分野では進んでおり，別の分野では遅れているというのが実態と思われる．

現在の時点では，むしろ熱心な企業，または熱心な管理者に率いられる部門またはプロジェクトとそうでない所の差のほうが大きいのではないだろうか．つまり，日本と欧米の間には，自動車産業や，家庭電気産業のような優劣はまだ存在しないといえよう．品質評価の分野でいえば，メトリックスの利用や自動化でアメリカがやや勝り，細かい管理改善で日本がやや勝るというのが実感である．一方，テストについていえばテスト技術の研究ではアメリカが先行し，開発現場でのテスト管理ではやや日本が勝るという印象である．いずれにしても，これからの品質管理への取り組みいかんが今後の優劣を決定するといっても過言ではない．ただこれまでの他の産業分野との違いは，彼らもこれまでの教訓に学んでおり，真剣に取り組んでいるということである．

なお，本稿の執筆にあたっては2. 品質評価を東，3. テスト管理を保田が分担した．

参考文献

1) Zuse, H.: Software Complexity, Measurement and Methods, Welter de Gruyter (1991).
2) Basili, V.R. and Rombach, H.D.: The TAME Project: Towards Improvement Oriented Software Environments, IEEE Trans. Software Engineering (June 1988).
3) Arthur, J.D. and Nance, R.E.: Developing an Automated Procedure for Evaluating Software Development Methodologies and Associated Products—A final report, Technical report SRC-87-007, Systems Research Center and Virginia Tech. (1987).
4) 東 基衛他：STD-WG5（ソフトウェア品質特性）分科会報告，平成元年度ソフトウェア開発・システムの文書化標準化調査研究報告書，日本規格協会（1990）．
5) 東 基衛他：STD-WG5（ソフトウェア品質特性）分科会報告，平成二年度ソフトウェア開発・システムの文書化標準化調査研究報告書，日本規格協会（1991）．
6) 東 基衛他：STD-WG5（ソフトウェア品質特性）分科会報告，平成三年度ソフトウェア開発・システムの文書化標準化調査研究報告書，日本規格協会(1992)．
7) 東 基衛，砂塚利彦：ソフトウェア品質計測/保証技術（SQMAT），品質，Vol. 16, No. 1 (1986).
8) Fenton, N.: Software Metrics, A Rigorous Approach, Chapman & Hall (1991).
9) McCall. et al.: Factors in Software Quality, RADC report TR-367 (1977).
10) Henry, S.M. and Selig, C.L.: A Metric Tool for Predicting Source Code Quality from a PDL Design, Proc. Workshop on Software Design Metrics, Melbourne, Florida (1988).
11) 保田勝通：テスト・品質保証技術の現状と課題，情報処理，Vol. 28, No. 7, pp. 873-879 (1987).
12) 菅野文友(編)：ソフトウェアの品質管理，日科技連出版社（1986）．
13) Myers, G.J.(著)，長尾 真(訳)：ソフトウェア・テストの技法，近代科学社（1980）．
14) 玉井哲雄他：ソフトウェアのテスト技法，共立出版社（1988）．
15) 片岡雅憲：ソフトウェア・モデリング，日科技連出版社（1988）．
16) 竹下 亨：CASE概説，共立出版（1990）．
17) 石井康雄(編)：ソフトウェアの検査と品質保証，日科技連出版社（1986）．
18) 野木兼六他：ソフトウェアテスト項目作成支援システム，日立評論，Vol. 66, No. 3, pp. 9-32 (1984).
19) Beizer, B: Beyond Testing, Quality Week 1991 Conference Proceedings, Paperl-2, Software Research, Inc. (1991).
20) 山田 茂他：ソフトウェアの信頼性，ソフト・リサーチ・センタ（1990）．
21) 大場 充：ソフトウェア信頼性モデル入門，情報処理，Vol. 31, No. 12, pp. 1623-1630 (1990).
22) 当麻喜弘：ソフトウェア信頼性モデルと評価，電子情報通信学会誌，Vol. 73, No. 5, pp. 461-466 (1990).
23) 山田 茂：ソフトウェアの品質評価に関する考え方と動向，情報処理，Vol. 32, No. 11, pp. 1198-

1202 (1991).
24) 菅野文友：ソフトウェア・エンジニアリング，日科技連 (1979).
25) 尾崎俊治：非定常ポアソン過程モデル，情報処理，Vol. 31, No. 12, pp. 1631-1640 (1990).
26) 当麻喜弘：超幾何分布にもとづくソフトウェア残存フォールト数推定モデル，情報処理，Vol. 31, No. 12, pp. 1641-1646 (1990).
27) 森口繁一(編)：ソフトウェア品質管理ガイドブック，日本規格協会 (1990)，第 17 章 pp. 341-370.
28) 寺本雅則他：応用ソフトウェアの高信頼化技術，電子情報通信学会誌，Vol. 73, No. 5, pp. 492-496 (1990).
29) 古賀恵子他：ソフトウェア品質評価システムの開発，ENGINEERS, pp. 18-21 (1986年8月).
30) Boehm, B. W. et al.: Quantitative Evaluation of Software Quality, Proc. 2nd ICSE, Vol. 2, pp. 592-605.

(平成4年1月31日受付)

東　基衛（正会員）

1939年生．1963年早稲田大学理工学部卒業．同年日本電気(株)入社．1987年同社を退社．現在早稲田大学理工学部教授(工業経営学科)．ソフトウェア工学（特に品質管理，メトリクス，要求定義，標準化），ユーザインタフェースなどの研究に従事．編著書は，日経品質管理文献賞を受賞したソフトウェアの品質管理と生産技術（1988, 日本規格協会）及びソフトウェア品質管理ガイドブック（1990, 日本規格協会）の他，コンピュータソフトウェアの標準化（1976, 日本経済新聞社），システム分析マニュアル（1991, 日本工業新聞社），ユーザインタフェースの設計（1988, 日経マグロウヒル），ANSI/IEEE ソフトウェア規格集(1988, 日本規格協会)など多数．日本経営工学会，日本経営情報学会，IEEE, ACM 各会員．ISO/IEC JTC 1/SC 7/WG 6（主査），情報処理学会情報規格調査会/SC 7 専門委員会（委員長）他各種委員会委員．

保田　勝通（正会員）

昭和17年生．昭和41年電気通信大学電気通信学部通信機械工学科卒業．同年(株)日立製作所入社．以来大型コンピュータ用基本ソフトウェアの開発，主としてソフトウェア検査に従事．平成2年よりソフトウェア生産技術推進に従事．現在，同社情報システム開発本部生産技術部部長．ISO TC 176（ソフトウェア品質保証），JTC 1/SC 7（ソフトウェア生産技術）の国内委員会を通じて，ソフトウェアの品質管理・生産技術の標準化活動に参加．