ソフトウェア工学 71−11
（1990. 2. 9）

ソフトウェア品質モデルの体系化とその適用評価

平山雅之　佐藤弘行　帥雍　匠
山田　淳　津田淳一郎

（株）東芝　システム・ソフトウェア技術研究所

従来より，ソフトウェアの品質については，品質の定量評価が大きな問題となっている，品質の定量評価の代表的な研究としては，McCabe,Halstead らによる評価指標の研究があげられる．しかし，これらは，ソースコードの定量評価を中心としたものである．我々はソースコードのみでなく，設計の品質評価も含めたソフトウェア品質定量評価技法に関して，新たに品質モデル・メトリクスを設定し，品質定量化支援システム　ESQUT の研究を進めている．本報告では，このシステムの品質モデル概要，品質定量化技法の概要，および実際のプロジェクトへの適用・評価事例について述べ，さらに従来技法であるMcCabe,Halstead の評価指標との関係について考える．

Establishment of software quality models and their evaluations

Masayuki Hirayama　Hiroyuki Sato　Takumi Kusanagi
Atushi Yamada　Junichiro Tuda

Systems and Software Engineering Laboratory,Toshiba Corporation
70 Yanagi-cho Saiwai-ku Kawasaki,210 Japan

This paper covers a software quality quantification technique. The software quality quantitative estimation has been a major issue these days. Typical studies of this subject include those of estimate indexes by McCabe,Halstead,and others which center on quantitative estimate of source code. We have introduced quality metrics in the software quality quantitative estimation technique embracing quality estimate of design as well as source code in studying a quality quantification support system. This paper outlines a quality quantification technique in this system,describes examples of its application to actual projects and of its evaluation, and considers its relationship with the conventional technique of estimate indexing by McCabe and Halstead.

1．序言

従来，ソフトウェアの品質については、テスト工程での試験に着目したアプローチや，あるいは、Halstead,McCabe に見られるように，特定工程での品質メトリクスを採用した，品質定量化の試みを主体として進められてきた．しかしながら，ソフトウェアの品質は特定の工程だけでの，評価では十分とはいえない．我々は，ソフトウェア生産の分業化・工業化に対応すべく，ソフトウェア生産一環支援システムIMAPの研究を進めている．このIMAP(*1)のなかでの，ソフトウェア品質の捕えかたとして，ソフトウェア・ライフサイクルに基づいた，品質の一環した定量・評価技術の確立を目指している．

本報告では，これらを踏まえ，特に，品質特性モデルとそれに基づく品質メトリクスの設定，また，これらを当社内での実プロジェクトに適用した適用例をもとに，以下の事項に関しての考察を進める．

1．品質特性モデルとソフトウェア・ライフサイクルの関わり，各工程での品質モデル
2．各工程における品質の等価性
3．品質評価・支援システム　ＥＳＱＵＴ(*2)　の品質モデルとその特徴
4．品質評価・支援システム　ＥＳＱＵＴ　の適用事例とその評価

また，このＥＳＱＵＴの適用事例に関しては，ソフトウェア・ライフサイクルの中の設計および作成工程に関して，品質の等価性に関する考察もおこなう．

2．品質定量化のための品質モデル

ここでは，ソフトウェア品質を定量評価するための品質モデルについて，検討をくわえる．

2・1　品質管理の流れ

ソフトウェアの管理を考える場合，以下の２点がポイントとなる．

管理サイクル　　Plan,Do,Check,Action

管理対象　　管理の対象となるアイテム

品質	品質管理
納期	納期管理
コスト	コスト管理

ソフトウェアの管理を考える場合，その管理対象としては，品質・納期・コストの３アイテムが，基本となる．

品質管理の場合，管理対象をさらに詳細化すると，作業の品質と成果物の品質の２側面に分けて考えることができる．ここでいう，作業とは，ＤＲ，ＷＴあるいは，プロジェクト計画など，直接、成果物（製品）として捕えることは出来ないが，成果物と密接に関係するものをさす．

次に，管理サイクルについて考える．管理対象を品質とした場合の，管理サイクルは，以下のように定義される．

Plan　　『管理モデル』
どの様な開発モデル，管理モデルを採用し、また，どの様な，品質メトリクスを採用して，品質管理を行うかを定める．

Do　　『計画技術』
品質要求定義，品質機能展開および品質メトリクスの目標値の設定技術

Check　　『モニタリング技術』
品質特性値計測，品質状況の確認，品質のビジュアル化などの技術

Action　　『分析評価技術』
品質の評価・分析，診断技術および高品質誘導技術，診断結果のフィードバック

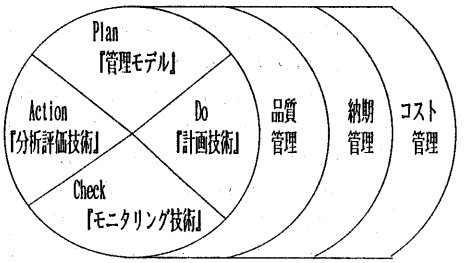

図２・１　品質管理サイクル

(*1)IMAP : Integrated software MAnagement and Production support system
(*2)ESQUT: Evaluation of Software Quality from Usre'sviewpoinT

2・2　品質特性モデル

従来より，品質保証技術のポイントとして考えられているものとして，品質特性モデルがあげられる．図2・2に示したものは，このうちの代表的なものである．この品質特性モデルは，『2・1　品質管理の流れ』の中で、管理モデルおよび計画技術に関係していると考えることができる．

品質特性モデルを反映して，品質メトリクス・モデルの検討および品質要求定義・機能展開等が実施される．

しかしながら，従来の品質特性モデルについては，製品としてのソフトウェアを強く意識しており，例えば『ソフトウェアの信頼性』といった場合，製品としてのソフトウェアの信頼性をさすものでしかない．

しかしながら，製品の品質については，製品の作成過程での品質要因によって，その大半が決定してしまうため，製品に関する品質特性を、その作成過程でどこまで，作り込み・定量評価するかがポイントとなる前述の『管理の流れ』は、基本的に，ソフトウェア開発の各工程それぞれに存在するものと考えられる．この意味では、品質特性モデルについても，開発の各工程毎に検討を加える必要があるものといえる．

品質の作り込みについては，ソフトウェア作成の各工程で行うことが原則となり，ソフトウェア品質を意識した，ソフトウェア開発プロセス・モデルの研究等が行われている．（参考文献　※1，2）

また，この品質作り込みについて，より積極的なアプローチとして提案されているものとして，要求仕様段階での品質要求定義の考えかたがあげられる．しかしながら，この品質要求定義の手法は，第三者による品質管理の立場からは，技術的に難しい面が多い．

2・3　品質の等価性

ここでは，品質特性モデルの拡張と品質の等価性について，検討を加える．

品質特性モデルは，製品としてのソフトウェアを前提としたものであるが，この特性は，ソフトウェアの開発過程の中から生み出されたものである．従って，ソフトウェアの開発過程においても，同様の，あるいは類似した品質特性を設定することは可能であると考えられる．ここで，特に，成果物の品質に着目すると，ソフトウェアの特性として，次の2点が考えられる．

(1) 中間成果物がそのまま製品としての性質を合せ持つもの
　例えば，仕様書などのドキュメント等

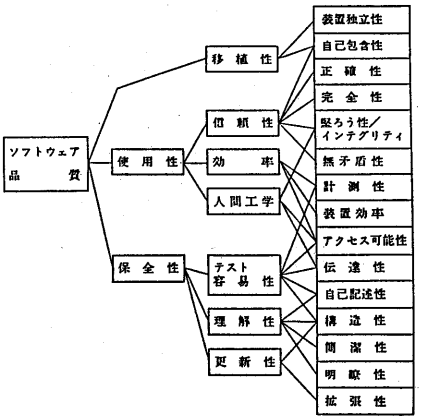

図2・2　ソフトウェア品質特性

(2) ある工程の中間成果物が，次の工程の中間成果物へと加工され，その品質についてもそのまま受継がれるもの
　例えば，ソースコード等

この関係を示したものが，図2・3である．このような，中間成果物の特性を考慮したうえで，その品質について考える場合，ソフトウェア開発工程における品質特性モデルの位置付けは，図2・4のようになる．

ここで，図2・3の中間成果物の特性と，図2・4の品質特性モデルの位置付けを合せて考えると，ソフトウェアの品質についての，等価性の概念を導くことが可能となる．

ソフトウェアの品質等価性

ソフトウェアの品質については，その開発の各工程間での等価性が成立する．即ち，工程ｉでのソフトウェアの品質は，工程ｉ＋１についても，そのまま引継がれる．

ソフトウェアの品質劣化とは，この工程間での品質等価性が，崩れた場合に起きるものと考えることができる．（即ち，バグが入りこんだと考えれる）

このソフトウェアの品質等価性を前提とした場合，工程ｉおよび工程ｉ＋１の品質を，計測し，その間の品質劣化を調べることにより，バグの入込みを察知することが可能であるといえる．

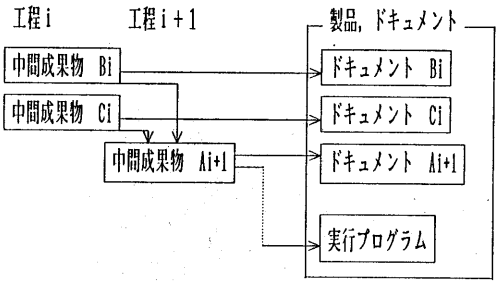

図2・3　中間成果物の性質

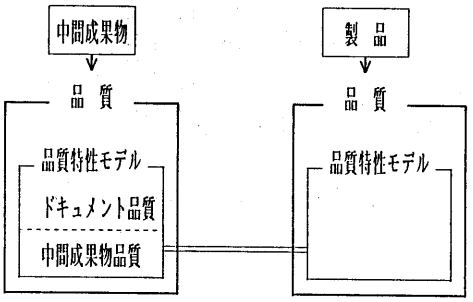

図2・4　品質の等価性

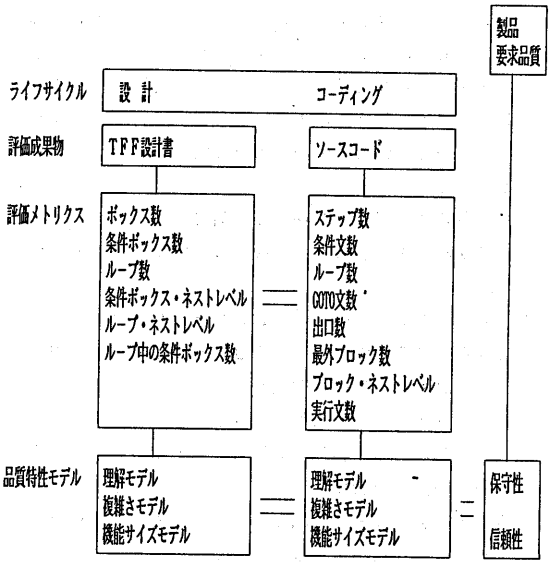

図3・1 品質モデルの相似性

3．ソフトウェア品質評価支援システム　ＥＳＱＵＴ

ここでは，品質評価支援システム　ＥＳＱＵＴ　について，その構成および品質モデルの構成を示す．

3・1　ＥＳＱＵＴ　システム構成

品質評価支援システム　ＥＳＱＵＴ　は，ソフトウェア・ライフサイクルを通して，各工程での（中間）成果物の品質を定量評価・支援し，品質の管理サイクルの実現をはかるものであり，以下の様な特徴をもつ

(1) 各工程の成果物毎に，その品質を測定・評価する手法・ツール群より構成される．
(2) 各ツールは，測定する成果物に合せた品質メトリクスを採用し，品質の定量化・可視化をおこなう．『モニタリング技術』
(3) 各工程ごとの品質特性モデルを設定し，採用するメトリクスは，この品質特性モデルに基づくものである．『管理モデル，計画技術』
(4) 作成中の（中間）成果物についての適用が可能であり，品質作り込みによる，品質管理が容易となる．『分析・評価技術』

図３・１に，ＥＳＱＵＴの品質モデル及びメトリクスの例を示す．この品質モデルは，原則として，図２・１の製品に関する品質特性モデルをベースに，誘導したものである．　ＥＳＱＵＴでは，特に，『ソフトウェアの信頼性，保守性』に重点をおいた，品質モデルの構成となっている．

また，ＥＳＱＵＴでは，要求仕様，設計，作成の各工程に対応して，ＥＳＱＵＴ－ＳＰＣ，ＥＳＱＵＴ－ＴＦＦ，ＥＳＱＵＴ－Ｃの各技法・ツールが用意されているが，各工程で採用している品質モデルについては，前述の『品質の等価性』を意識したモデル構成となっている．従って，設計から作成の途中で品質劣化がおきた場合，ＥＳＱＵＴ－ＴＦＦ，Ｃでの計測・評価結果を比較することで，品質劣化を確認することが可能となる．

3・2　ＥＳＱＵＴ－ＳＰＣ

ＥＳＱＵＴでは，（要求）仕様の品質を計測・評価するシステムとして，ＥＳＱＵＴ－ＳＰＣを用意している．ＥＳＱＵＴ－ＳＰＣでは，自然言語記述の仕様書及びＤＦＤ（Data Flow Diagram ）について，品質を計測・評価するものであり，現在，開発の途中にある．使用する品質モデル・メトリクスは，他工程でのＥＳＱＵＴ－モデル，メトリクスとの整合性を保っている．

3・2　ESQUT－TFF

プログラム設計工程での設計書の品質を計測・評価するシステムである．対象となる設計書は，設計記述法　TFF(*3)　により記述されたものである．TFFはモジュール内部仕様・外部仕様，データ仕様等の作成を支援するもので，これにより，設計の標準化をおこない，結果として，設計の理解性・保守性を高めるものである．図3・2に，TFFによるモジュール内部仕様書の例を示す．

ESQUT－TFFでは，このTFF記述の設計書のなかで，設計の信頼性・保守性の観点から，特に，設計の単純性・独立性に着目し，TFFに記述されている，図形言語から形式的に評価できる，図3・1に示す6つのメトリクスを採用している．また，これらのメトリクスは，設計段階での制御構造にも深く関係しておりソースコードに関して，従来提案されていた，Halstead,McCabe のメトリクスとの対比を考えることも可能である．

ESQUT－TFFの特徴としては，次の3点をあげることができる．

(1) 設計品質を形のうえから測定することが可能
(2) 設計品質の低い，危険度の高いモジュールを探索することが可能
(3) 設計段階での品質改善・品質作り込みを誘導することが可能

3・4　ESQUT－C

ESQUT－Cは，C言語で記述されたソースコードについて，その品質を計測・評価するシステムである．使用する品質モデルは，ESQUT－TFFのモデルに準拠し，同様に，ソフトウェアの信頼性・保守性に着目し，理解性，単純性を中心としている．

採用しているメトリクスは，図3・1に示す8つであり，これらは，ESQUT－TFFのメトリクスとも深い関係をもっている．

4．ESQUTによる品質定量化事例

ここでは，実際のソフトウェア開発プロジェクトにESQUTを適用して，品質定量化をおこなった事例について報告する．

4・1　対象プロジェクト

今回のESQUT適用対象のプロジェクトとしては実際のシステム製品開発を行なっている，ソフトウェア部門を中心とした，複数プロジェクトを選択した．

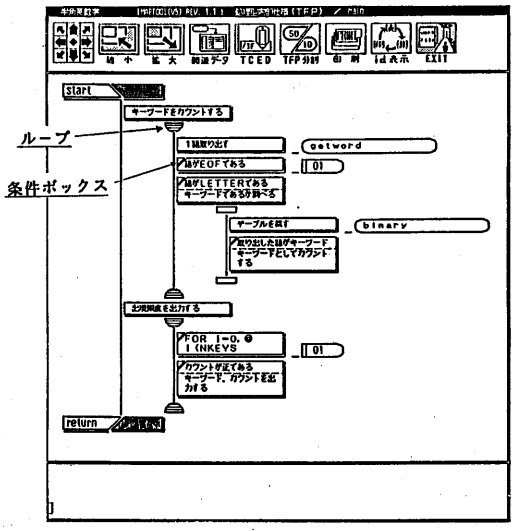

図3・2　モジュール内部設計書

(*3)TFF　：　Techinical Fomula for
Fifty steps/module design

表4・1　ESQUT　適用対象

試行対象ＪＯＢ	組込みタイプのマイコンソフト
試行期間	H01-7 ～ H01-11
対象プロジェクト 開発環境	設計　TFF 言語　C マシン　Ｊ３１００
規模	約１６Ｋバイト 約２０人月

この対象プロジェクトについて，ESQUTを適用し，品質の定量化をおこなった．本報告では，このうちの一例として，表4・1に示すジョブでの適用事例・結果について報告する．また．今回の適用実験では，設計工程以降についてESQUTを適用し，定量評価を行なった．

４・２　品質管理限界値

ＥＳＱＵＴ－ＴＦＦ，Ｃについては，過去の適用実験により，各メトリクスについて，メトリクス値の値分布（平均値，標準偏差），およびこれをもとにした品質管理限界値を設定してある．今回の適用実験でのポイントの一つとして，これらの値の検証をおこなった．図４・１，図４・２は，それぞれ，ＥＳＱＵＴ－ＴＦＦ，Ｃについて，各メトリクスの平均値，管理限界値について，過去の事例よりもとまった基準値と今回の実験の値を比較したものである．

これより，各メトリクスともに，多少の差異はあるが，いずれも，基準値と実験値とは近い値をとることが確認された．これより，ＥＳＱＵＴ－ＴＦＦ，Ｃについて設定した基準値（平均値，標準偏差，管理限界値）は，プロジェクトの差異に関係なく，どのようなプロジェクトについても，共通に適用できることが確認された．

４・３　メトリクス相関

ＥＳＱＵＴ－ＴＦＦのそれぞれのメトリクスについて，個々の相関の強いメトリクスの組合わせを表４・２に示す．メトリクスの相関についても，過去の適用例で明らかになっていた組合せについて，同様に相関が高いことが明らかになった．

また，この相関の強い組合わせについては，回帰分析を行った結果，回帰直線の係数についても，過去の事例に近い値となった．この回帰直線の係数は，ＥＳＱＵＴのメトリクスのもつ特性値の一つであると考えることができる．

この回帰直線と値の分布を利用して，品質問題モジュールの探索に利用することもできる．（図４・４）

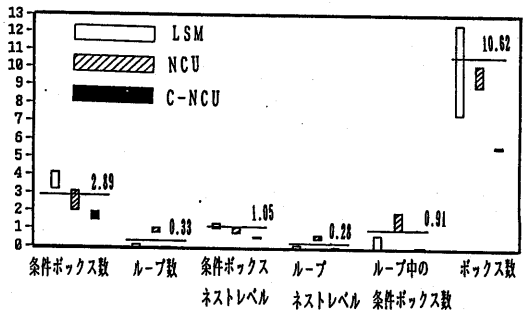

図４・１　ESQUT メトリクス値（平均値）の分布

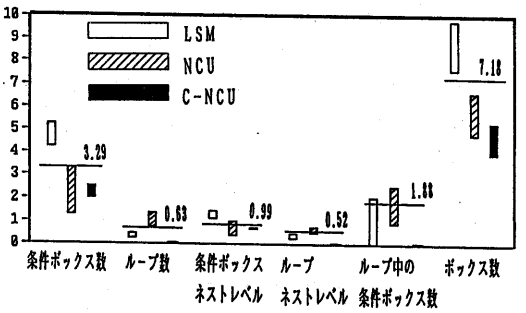

図４・２ESQUT メトリクス値（標準偏差）の分布

表４・２　相関の高いメトリクスの組合わせ

条件ボックス数	ボックス数 条件ボックス・ネストレベル
ループ数	ループ・ネストレベル

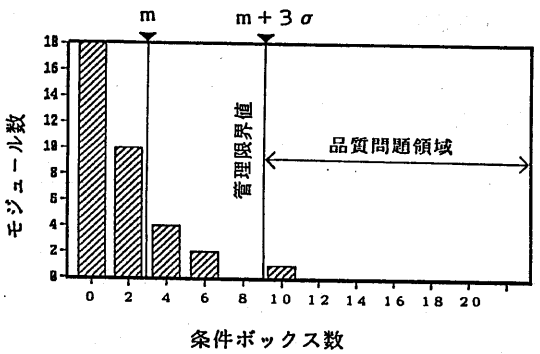

図４・３　条件ボックス分布傾向

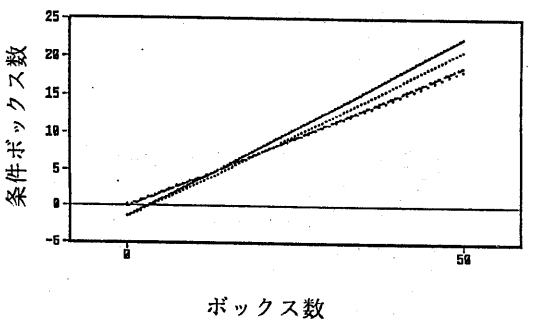

図４・４　ボックス数と条件ボックス数の関係

4・4　設計品質とソースコード品質の等価性

ＥＳＱＵＴ－ＴＦＦとＥＳＱＵＴ－Ｃでは，前述したように，その品質モデルについての共通性を保持している．このため，ＥＳＱＵＴ－ＴＦＦ，Ｃの計測結果から，品質劣化を検討することが可能である．図４・５は，ボックス数とステップ数の関係を示したものである．過去の事例より，ボックス数の約３～４倍がソースコードのステップ数になることが分っている．この例では，図中の黒く塗ったモジュールについて，この関係から外れ，設計通りのコーディングが行われていないことがわかる．

また，図４・６は，条件ボックス数と条件文数の関係を示したものである．通常，条件ボックス数と条件文数は，一致するものであるが，この例では，黒く塗潰したモジュールについて，設計された条件以外にもソースコード作成段階で，条件文が作り込まれていることがわかり，作成段階での品質劣化がおきていると判断できる．

このように，ＥＳＱＵＴ－ＴＦＦ，Ｃを併用することで，工程間での品質劣化を分析・評価することが可能となる．

4・5　ＥＳＱＵＴ　メトリクスの有効性

今回の適用実験では，ＥＳＱＵＴメトリクスによる品質管理の有効性を検証するため，各モジュールについて，設計工程での問題点，ソースコード作成以降の問題点をモニタし，これとＥＳＱＵＴ計測結果から考えられる問題モジュールとの関係について検討を加えた．

表１　に示したプロジェクトについて，設計工程終了後に，何等かの問題点が発生し，設計の修正・変更を行ったモジュールは，全体の約１５％あり，このうちの，６２％については，ＥＳＱＵＴの計測結果からも問題点があるものと予測できたものであった．　また，このＥＳＱＵＴの計測結果について，今回の実験で，特に品質管理指標として有効であると考えられたものとしては，条件ボックス数を処理ボックス数で割った，条件ボックス密度に相当するものが，有効であることが確認された．この指標は，各モジュールの制御構造の複雑さを，表現する指標の一つと考えられる．

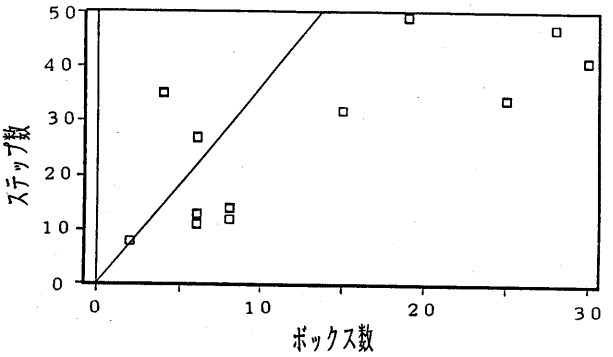

図4・5　ボックス数とステップ数の関係

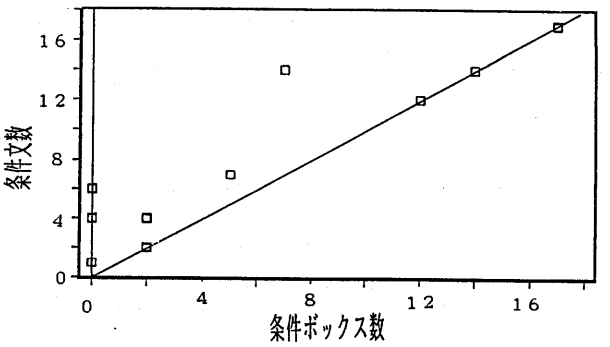

図4・6　条件ボックス数と条件文数の関係

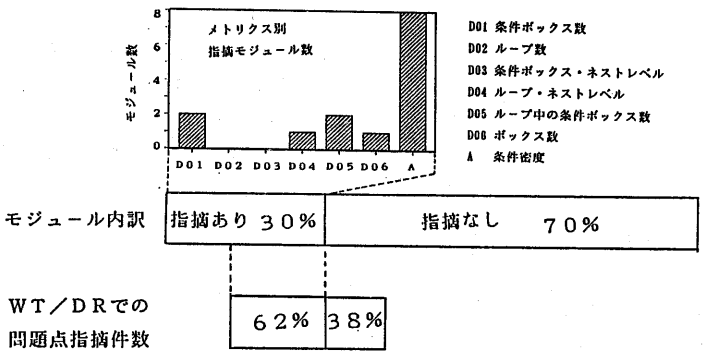

図4・7　ESQUT　メトリクスの有効性

4・6　ＥＳＱＵＴメトリクスの検証

ここでは，ＥＳＱＵＴで採用しているメトリクスと他のメトリクスとの関係を考える．代表的な品質指標としては，Halstead,McCabe の品質指標があげられるしかし，これらはソースコードの品質指標の一つである．ここで，ＥＳＱＵＴ－ＴＦＦの品質メトリクスとこれらのメトリクスの関係を示したものが，図４・７である．Halstead,McCabe のメトリクスとも，ＥＳＱＵＴ－ＴＦＦの処理ボックス数，条件ボックス数との相関は極めて高いことが確認できる．

これより，設計工程で，ＥＳＱＵＴ－ＴＦＦによる品質計測を行うことにより，その設計に基づくソースコードの複雑さ（Halstead,McCabe メトリクス値）を予測することも可能となる．（但し，この場合，設計からソースコード作成の仮定で，品質劣化がおこらないことを前提としている）

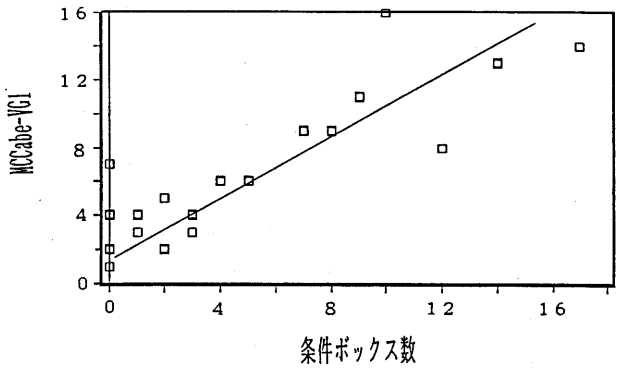

図4・8　条件ボックス数とMcCabe-VG1　の関係

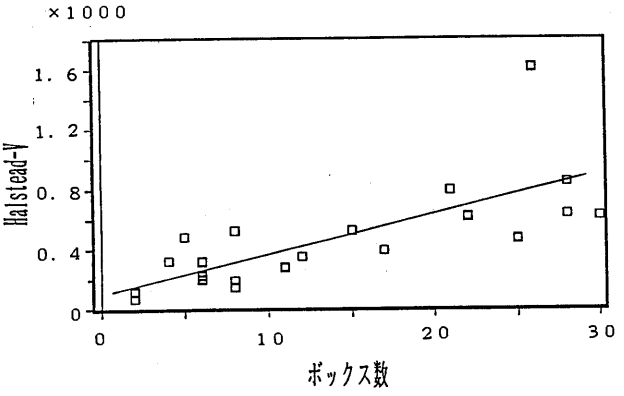

図4・9　ボックス数とHalstead-V　の関係

5．結言

以上，ソフトウェア品質モデルの体系化およびＥＳＱＵＴによる品質計測・定量化の事例についての報告をおこなった．以下に，本報告のまとめを示す．

(1) 製品ソフトウェアの品質特性モデルをもとに，ソフトウェア・ライフサイクルの各フェーズでの品質モデルの在り方を検討した．
(2) 各工程での品質特性モデルのベースとして，品質等価性の概念を誘導した．
(3) 上記の品質特性モデルより，ＥＳＱＵＴの品質モデルを誘導し，実際の品質評価技法としての確立をはかった．
(4) ＥＳＱＵＴ－ＴＦＦ，Ｃを実際のプロジェクトに適用し，その品質メトリクス値の特性値の検証をおこなった．
(5) 品質評価・支援システムとしての，ＥＳＱＵＴの有効性を確認した．
(6) 品質等価性の概念について，一部，設計とソースコード作成工程間での検証をおこなった．

6．今後の課題

今回のＥＳＱＵＴ適用事例では，設計工程・作成工程を主体としたものであったが，今後は，要求仕様の段階でのＥＳＱＵＴ－ＳＰＣによる品質定量化および要求仕様工程と設計工程間における品質の等価性についての検証も進めていく予定である．

7．参考文献

(1) 平山，山田　他　“モジュール設計段階における品質評価の一手法“
情報処理学会　第３６回全国大会　5L-5 1988
(2) 平山，山田　“ソフトウェア設計品質メトリクスの検証“
情報処理学会　ソフトウェア研究会 64-5 pp33-
(3) 平山，山田　他　“ソフトウェア品質評価システム　ＥＳＱＵＴ“
情報処理学会　第３８回全国大会　3M-6 1989