ソフトウェア品質評価システム　ＥＳＱＵＴ
－ソースコード品質評価メトリクスの検証－

3M-6

平山雅之　山田　淳
(株式会社　東芝　システム・ソフトウェア技術研究所)

1．緒言

ソフトウェアの品質に関しては、テスト工程でのバグに着目した管理が中心となっている．しかしながら、近年ソフトウェアの需要増大にともないソフトウェア生産の分業化、工業化が展開されつつあり、この中でソフトウェアの品質を考える場合には、そのライフサイクルに沿って、各フェーズでのソフトウェアの品質をモニター・分析していく必要があるものと考えられる．

我々は、このような観点から品質評価支援システム～ＥＳＱＵＴ(*1)の研究・開発を進めている．ＥＳＱＵＴはソフトウェアの設計書・仕様書およびソースコードといった成果物に対し、定量化メトリクスを使用してその品質を計測・評価するシステムであり、ソフトウェア生産の工業化を目的とした一貫支援システムＩＭＡＰ(*2)の一環を担うものである．このうち，Ｃ言語で記述されたソースコードの品質評価支援をするシステムとして，ＥＳＱＵＴ－Ｃがある．今回、このＥＳＱＵＴ－Ｃシステムに関して、当社内で試行運用を実施し、ソースコード品質に関するデータ収集・分析を実施したのでその結果について報告する．

2．ＥＳＱＵＴ－Ｃ概要

ＥＳＱＵＴ－Ｃは，Ｃ言語で記述されたソースコードを構文解析し、定量化メトリクスを用いてその品質を計測・分析するものである．

表２・１　ＥＳＱＵＴ－Ｃメトリクス分類

	複雑さモデル	機能モデル	理解性モデル
ｇｏｔｏ文数	○		○
モジュール出口数		○	
条件文数	○		○
最外ブロック数 (**1)		○	
ブロックネストレベル (**2)	○		
実行文数	○	○	○
ステップ数	○	○	○
生数字数 (**3)			○

(**1) 最外ブロック数
　モジュール内で最も外側にある処理ブロックの数を調べる．
(**2) ブロックネストレベル
　モジュール内の処理ブロックのネストレベルを調べる．
(**3) 生数字数
　モジュール内に含まれる数字（置換マクロ，変数は除く）の数を調べる．

ＥＳＱＵＴ－Ｃではソフトウェアの制御構造に関係する品質特性モデル及び品質メトリクスを採用している．これらを表２・１に示す．

ＥＳＱＵＴ－Ｃでは，これらの品質メトリクスにもとずきモジュール毎にその個数を計測して表示する．

各品質特性モデルは次のような意味をもっている．

1）　モジュール複雑さモデル
　　モジュール内処理の複雑さを調べる．
2）　モジュール機能モデル
　　モジュールが実現する機能の大きさを調べる
3）　モジュール理解性モデル
　　モジュールの読み易さ、理解し易さを調べる

3．試行対象・評価手法

今回の試行運用は、当社システム・ソフトウェア技術研究所で作成されたソフトウェア４件に関して実施し、評価を行った．表３・１に各対象ソフトウェアの概要を示す．

表３・１　試行対象プロジェクト

プロジェクト	開発内容	開発機種	開発言語	規模 Kstep	モジュール数
A	移植	J3100	Lattice C	3.5	134
B	新規開発	J3100	Lattice C	7.0	153
C	改良	J3100	Lattice C	5.0	31
D	新規開発	G8050	Unix C	20.0	400

ＥＳＱＵＴ－Ｃにより測定されたデータは、図３．１に示すように、基本統計量分析、メトリクス値分布パターン分析、メトリクス相関分析等を実施し、その特性を把握する．

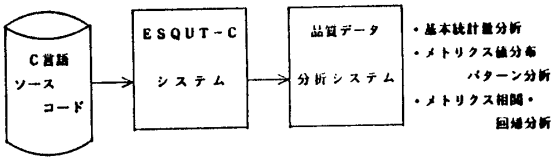

図3．1　ＥＳＱＵＴ－Ｃ　品質評価システム

4．試行結果・考察

ＥＳＱＵＴ－Ｃに採用されている定量化メトリクスの基本統計量を表４・１に示す．これより、８つのメトリクスについてその平均値、分散、標準偏差等の基本統計量は、どのソフトウェアについても、ほぼ近い値となっており、この値を標準値として品質目標に利用することができると考えられる．

図４・１は、条件文数に関して、分布パターンの代表的な例を示したものである．（横軸：メトリクス値、縦軸：モジュール数，横軸スケールはこの図の場合，条件文数10毎のヒストグラム）　この図４・１に示したメトリクス以外でも、ＥＳＱＵＴ－Ｃで採用している８つのメトリクスについては、それぞれ固有の分布パターンをもつことがわかった．

Software evaluation system ESQUT
-Verification of source code quality metrics

Masayuki Hirayama　Atusi Yamada
TOSHIBA CORPORATION

表４・２は、メトリクス間の相関を示したものであるこれより以下に示すメトリクスについては、その相関が高いと言える.

1） 条件文とステップ数
2） 実行文とステップ数
3） 実行文と生数字数
4） ステップ数と生数字数

これら相関が高いメトリクスについては、その回帰曲線を考えることが可能となる. この回帰曲線は、それぞれのメトリクスのペアに固有の値と考えられる. 図４・２は、条件文数とステップ数についてその散布図中に回帰曲線を描いたものである. この図より、この相関関係～回帰曲線を利用して品質評価をすることが可能となる. 即ち、この関係から外れるモジュールについては、他のモジュールと比べた場合, 品質の観点からは異なると言え、問題のある作り方がされたモジュールであると考えることができる.

今回の試行では、研究所において作成された比較的規模の小さいソフトウェアにを対象としたために, ソフトウェアの特性面が類似しており, メトリクス基本統計量、分布パターンメトリクス相関関係等については、余り大きな差は見られないものと考えられる.

5. 結言

本研究では、ソフトウェアのソースコード品質の定量評価について、その一手法を提示した.

今回、ＥＳＱＵＴ－Ｃで用いた定量化メトリクスについては、品質指標として利用できる目標値が定まり、また、これを利用した品質評価手法についても提示することができた今後の課題としては以下の３点が考えられる.

1. 今回得られたメトリクス基準値等については、その試行対象ソフトウェアの特性を反映した値となっていることも考えられるため、この結果をもとに、より広範囲でのＥＳＱＵＴの試行を実施し、一般値を導出することで、ソースコード品質に対する品質標準値を提示する.

2. ソースコード品質とテスト時に発見されるバグとの関連付けを明確にする.

3. ソースコードの品質のみならず設計等より上流工程での成果物品質についても技法・ツールの開発・研究等を通してその品質特性を明らかにしていくと共に, これらの上流工程での品質とソースコード品質との対応・関係についても検討を進める必要がある.

[参考文献]

（１） 平山 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） 山田 他 "ソフトウェア品質評価システム ＥＳＱＵＴ （１）" 情報処理学会第３１回全国大会 １Ｇ－７ １９８５

表４・１ ＥＳＱＵＴメトリクス値 基本統計量

		goto文数	出口数	条件文数	最外ブロック数	ブロックネストレベル	実行文数	ステップ数	生数字数
平均値	A	0.01	0.16	4.78	1.58	1.14	18.84	32.50	19.80
	B	0.00	0.24	3.65	0.91	1.20	12.80	19.62	8.69
	C	0.25	1.51	6.32	1.54	1.83	22.61	45.61	18.58
	D	0.16	2.35	10.62	1.59	1.83	30.60	55.47	27.67
標準偏差	A	0.17	2.37	5.70	1.67	1.03	13.50	24.62	21.24
	B	0.00	0.58	4.66	0.74	1.05	9.89	14.73	9.35
	C	1.26	1.43	6.03	1.52	1.65	21.74	35.27	20.71
	D	1.03	2.02	15.75	1.55	1.46	36.56	66.83	42.95

表４・２ ＥＳＱＵＴ－Ｃメトリクス相関

	goto文数	出口数	条件文数	最外ブロック数	ブロックネストレベル	実行文数	ステップ数	生数字数
goto文数								
出口数								
条件文数		△						
最外ブロック数			△					
ブロックネストレベル			△					
実行文数			●○△	○				
ステップ数			●●○△	○	△	●●●●		
生数字数	△		●○	△		●●	●●	

● r ≧ 0.9
○ 0.9 > r ≧ 0.8
△ 0.8 > r ≧ 0.75

(*1) ESQUT:Evaruation of Software Quality from Usre's viewpoinT
(*2) IMAP: Integrated software Management and Production support system

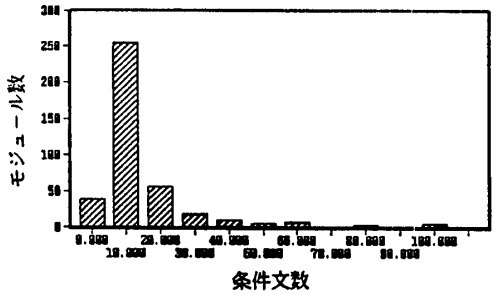

図４・１ ＥＳＱＵＴメトリクス分布パターン
（プロジェクトーＤ）

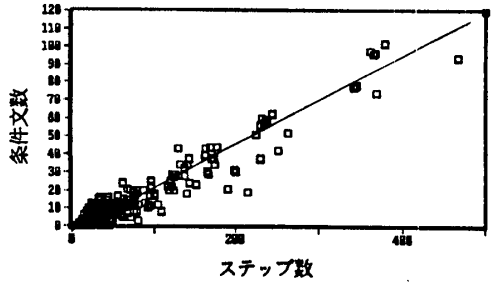

図４・２ ＥＳＱＵＴメトリクスの相関
（プロジェクトーＤ）