ソフトウェア工学 31－1
（1983. 8. 31）

SPQL評価法：サンプル数の決定法と実験結果

大場 充（日本IBM サイエンス・インスティチュート）

1．はじめに

ソフトウェア品質評価において信頼性は重要な要素である．そのソフトウェアの信頼性を決定づける要因として重要なものに，ソフトウェアのリリース（出荷）時に残存するバグ数と，ソフトウェアの実使用環境とが考えられる．残存バグ数が多くとても，ソフトウェアの使用環境が試験で十分カバーされていれば新たなバグの発見はない．しかしこの点はソフトウェアの生産者にとっては，ソフトウェアの実使用が始まるまで未知である．従って，生産者が管理できるパラメータとしては，残存バグ数だけが問題となる．特に利用者数の多いシステム制御プログラム等では，その保守費用はほぼ残存バグ数に比例する．

従って，ソフトウェアの品質保証・品質管理においては，開発工程におけるバグの混入防止と，試験工程におけるバグの除去が重要となる．同時に，ソフトウェア中に残存するバグ数を簡便な方法で精度よく推定する技術が必要となる．そのような方法として，従来からソフトウェアの信頼度成長曲線を応用した信頼度推定法が研究されてきた．[1)2)3)] ところが，その評価はテストの質に強く依存するため，テストの質が十分でない場合にはきわめて『あまい評価』となる欠点がある．この欠点を改善する方法として，捕獲・再捕獲法と信頼度成長の推定法を組合せた品質評価基準SPQLが提案された．[4)5)]

本報告では，このSPQLの推定に必要となる制御欠陥数（ソフトウェアに埋込む既知のバグの数）の決定法とその実験の結果について議論する．

2．SPQL

SPQL（Software Product Quality Level）は，残存バグ数の精度よい推定のために，捕獲・再捕獲法の考え方を応用したものである．[5)] すなわち，ソフトウェアの試験開始前に制御欠陥と呼ぶ既知のバグを混入させ，試験中は真の欠陥（真のバグ）の発見過程と同様に，制御欠陥の発見過程を追跡する．この2種類の欠陥の発見過程を信頼度成長曲線によって分析し，試験を無限時間継続したときに発見可能な真の欠陥数N_iと制御欠陥数N_cを求める．ここで，試験終了時（または評価時）までに発見・除去された真の欠陥数をm_iとし，試験の量に関する指標γを以下のように定義する：

$$\gamma = \frac{m_i}{N_i}. \qquad (1)$$

また，試験開始時に埋込んだ制御欠陥の総数をM_cとし，試験の質に関する指標αを以下のように定義する：

$$\alpha = \frac{N_c}{M_c}. \qquad (2)$$

ここで，ソフトウェアの品質評価尺度SPQLは，αとγの積として以下のように与えられる：

$$SPQL = \alpha \times \gamma. \qquad (3)$$

(3)式から明らかなように，SPQLは，全ての制御欠陥が（無限時間の試験で）発見可能なとき，換言すれば試験の質が十分高いとき，γに等しくなる．すなわち，真の欠陥の信頼度成長曲線のみによるバグの除去率に等しくなる．逆に，制御欠陥の一部が発見可能ではないときには，SPQLはγよりも小さい値となる．すなわち，真の欠陥の信頼度成長からはバグの除去率が100％だと言えても，ある種のバ

グが未発見なことが予測されるので，品質評価は1（欠陥除去率が100％）以下の値となる．

3．信頼度成長モデル

ＳＰＱＬの推定にはソフトウェア信頼度成長の推定を必要とする．ここではソフトウェアの信頼度成長モデルとして，以下の3つを用いる：

1）Goel-Okumotoの指数型成長モデル，
2）遅れ型Ｓ字成長モデル，
3）加速（習熟）型Ｓ字成長モデル．

Goel-Okumotoの指数型成長モデルとは，ソフトウェアの累積発見バグ数が試験時間に対して，図1（a）の曲線のように増加する場合のモデルである．このモデルは以下の平均値函数$m(t)$をもつ：

$$m(t) = N(1 - e^{-\phi t}) \qquad (4)$$

ただし，Nは試験開始時点にソフトウェア中に残存していたバグの総数，またϕは試験中における単位時間当りのバグの発見率（試験の効率）を示すパラメータである．

遅れ型Ｓ字成長モデルは，ソフトウェアの累積発見バグ数が試験時間に対して図1（b）の曲線のように増加する場合のモデルである．このモデルは，バグの発見とエラーの同定との間に時間遅れがあるような場合に適合することが知られている．[1] このモデルは次の平均値函数$M(t)$によって表現される：

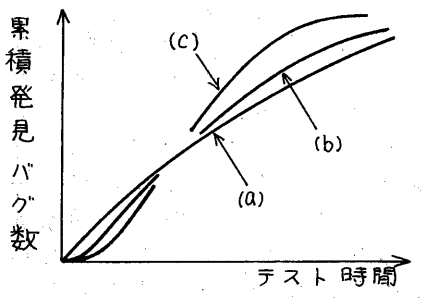


図1．ソフトウェアの信頼度成長曲線

$$M(t) = N[1-(1+\phi t)e^{-\phi t}]. \qquad (5)$$

加速型Ｓ字成長モデルは，ソフトウェアの累積発見バグ数が試験時間に対して図1（c）の曲線のような形で増加する場合のモデルである．このモデルは，ソフトウェア中のバグが相互に関係をもつ（独立でない）場合，試験中に試験担当者の学習による単位時間当りのバグ発見率の向上がある場合，試験中の投入労力（努力）が一定でない場合等によく適合することが知られている．[1] このモデルは以下の平均値函数をもつ：

$$\mu(t) = N \cdot \frac{1 - e^{-\phi t}}{1 + \psi \cdot e^{-\phi t}}. \qquad (6)$$

ただし，ψは：

$$\psi = \frac{1 - r}{r}, \qquad (7)$$

とする．

4．制御欠陥数の決定法

ＳＰＱＬの推定に必要な制御欠陥数M_cは以下のようにして決定できる．試験開始前に埋込む数をM_cとし，試験期間中の制御欠陥発見率の推定値をp（$0 < p \leq 1$）とする．ここで，試験期間中に発見すべき制御欠陥の最少数をLとする．Lは，制御欠陥の信頼度成長曲線の推定に必要なサンプル数であり，一般に20以上である．このとき，M_c個の制御欠陥のうち少くともL個が試験期間中に発見される確率は：

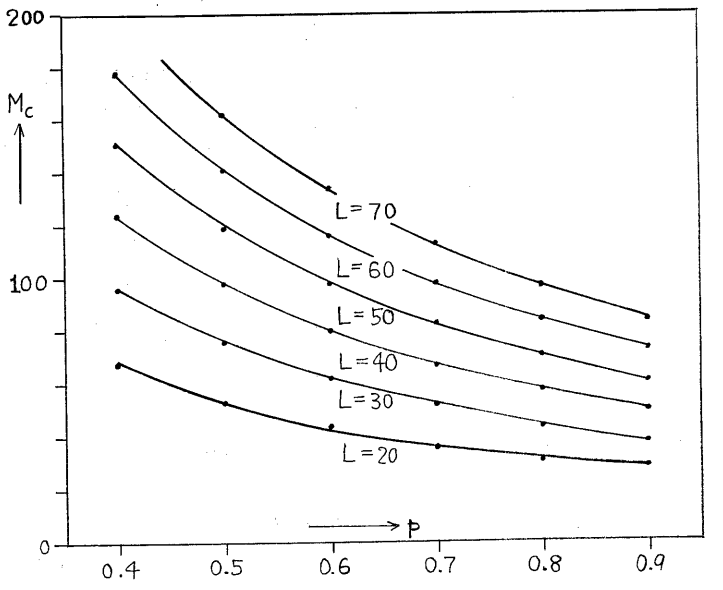
$$Pr\{x \geq L \mid M_c\} = 1 - Pr\{x < L \mid M_c\}, \qquad (8)$$

で与えられる．ここで，$Pr\{x < L \mid M_c\}$は2項分布に従うので：

$$Pr\{x < L \mid M_c\} = \sum_{k=0}^{L} \binom{M_c}{k} p^k (1-p)^{M_c - k}, \qquad (9)$$

を満足する．ここでP_cを信頼度とすれば，制御欠陥数M_cを決定する問題は，p，L，およびP_cをパラメータとして：

$$P_c \leq 1 - Pr\{x < L \mid M_c\}, \qquad (10)$$



を満足するM_cを求める問題となる．図2に$P_c = 0.95$の場合について，各pとLの組合せに対してM_cがどう変化するかを試算した結果を示す．

制御欠陥発見率pを具体的にどう定めるかについては，確立した方法はない．われわれの経験から言えば，試験を担当する作業者と，制御欠陥の作成者の技量のバランスにより，通常の場合0.6から0.9までの範囲の値となる．従って，図2の計算結果より，通常の場合で40前後の制御欠陥が必要となる．

5．実験例

フル・スクリーン・エディタの試験における実験例について以下に述べる．本ソフトウェアは，BASIC言語で書かれた約1000ステップの，パーソナル・コンピュータ上で稼動するスクリーンエディタである．実験では3人の被験者に独立に試験を行わせ，エラー発見に要した時間を計測した．実験に際して被試験プログラムに埋込まれた制御欠陥の総数は45個であり，そのうち3個は真の欠陥を除去するために変更され消失した．従って，試験終了時に有効であった制御欠陥の総数は42個であった．制御欠陥のほとんどは，開発者自身によるデバッグ作業中に発見されたものを，デバッグ完了後に再度ソフトウェア中に戻したもの（残留バグ）であった．ほかに，デバッグ過程で発見されたバグをヒントにして作成したものが少数あった．

図3に被験者Aによる真の欠陥の発見過程とその分析結果を示す．被験者Aは，エディタの試験の経験をもつが，メーカにおけるソフトウェア試験の経験はもっていなかった．このため，他の2人の被験者に比較して，試験の効率が悪く，発見した真の欠陥の総数も少なかった．図3に示されるように，この場合の信頼度成長モデルとしては，遅れ型S字成長モデルが最も適合した．推定によって得られたモデルは：

$$M(t) = 8.4[1-(1+0.017t)e^{-0.017t}], \quad (11)$$

であった．この被験者の場合，試験中

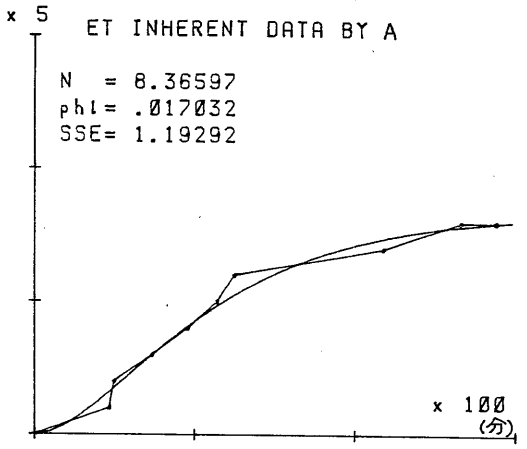


図３．被験者Ａの真の欠陥発見過程

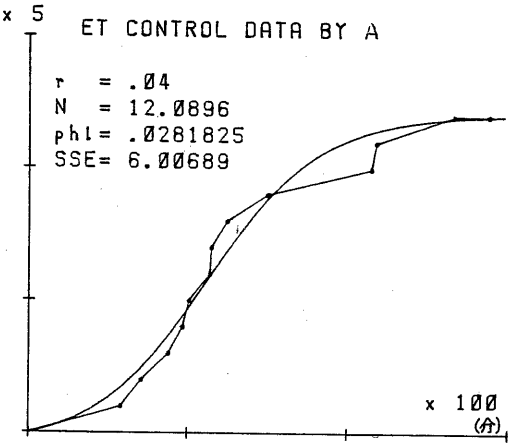


図４．被験者Ａの制御欠陥発見過程

に発見した真の欠陥の総数は８であった．従って，試験の量的尺度γは：

$$\gamma = \frac{8}{M(t_\infty)} = \frac{8}{8.4} \fallingdotseq 0.952, \qquad (12)$$

であった．

図４に，同一被験者による制御欠陥の発見過程とその分析結果を示す．この場合，信頼度成長が試験初期（制御欠陥を10個発見するまで）とそれ以後とで大きく異っていたため，試験初期のデータを除外して分析を行ったところ，ｒパラメータが0.04（ψ=25）の加速型Ｓ字成長モデルが最も適合した．推定によって得られたモデルは：

$$\mu(t) = 12.1 \frac{1 - e^{-0.028t}}{1 + 25.0\, e^{-0.028t}}, \qquad (13)$$

であった．この被験者が試験中に発見した制御欠陥の総数は22個（試験初期に発見された10個を除くと12個）であった．従って，試験の質的尺度αは：

$$\alpha = \frac{\mu(t_\infty)}{32} = \frac{12.1}{32} \fallingdotseq 0.378, \qquad (14)$$

であった．

以上の結果より，同じ被験者による試験のＳＰＱＬ評価値は：

$$SPQL \fallingdotseq 0.952 \times 0.378 \fallingdotseq 0.360, \qquad (15)$$

となる．このＳＰＱＬ評価値より，この試験では，真の欠陥の36％が除去されたと推定できる．従って，真の欠陥の64％が残存していることとなり，結論として当初22.2個の真の欠陥がソフトウェア中に存在し，試験終了後もそのうち約14個が残存していたこととなる．

図５に被験者Ｂによる真の欠陥の発見過程とその分析結果を示す．被験者Ｂは，エディタの仕様書作成の経験をもち，ソフトウェアの研究経歴も長い．この被験者は，現実のテキストをタイプするタスク（作業）をテストケースとして設定し試験を行った．試験所要時間を無視すれば，この被験者が最も多くの真の欠陥を発見した．図５に示

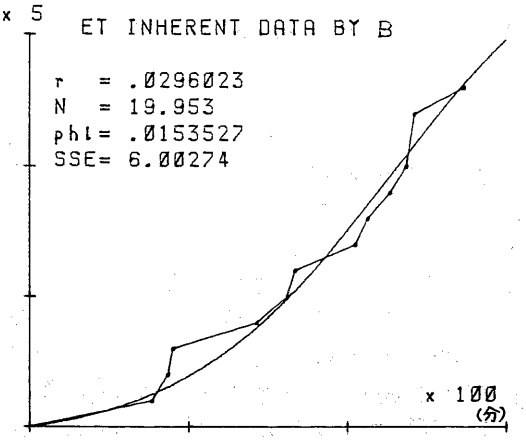


図５．被験者Ｂの真の欠陥発見過程

されているように，この場合信頼度成長は飽和点に達していないが，モデルとしては，加速型S字成長モデル（γパラメータは0.03）が最も良く適合した．推定によって得られたモデルは：

$$\mu(t) = 20.0\,\frac{1-e^{-0.015t}}{1+33.8\,e^{-0.015t}}, \qquad (16)$$

であった．この被験者の場合，試験中に発見した真の欠陥の総数は13であった．従って，試験の量的尺度γは：

$$\gamma = \frac{13}{\mu(t_\infty)} = \frac{13}{20.0} \doteqdot 0.650, \qquad (17)$$

であった．

図6に，同一被験者による制御欠陥の発見過程とその分析結果を示す．この場合も，信頼度成長は試験初期とそれ以後とで異っていた．しかし，その差が被験者Aほど大きくなかったので

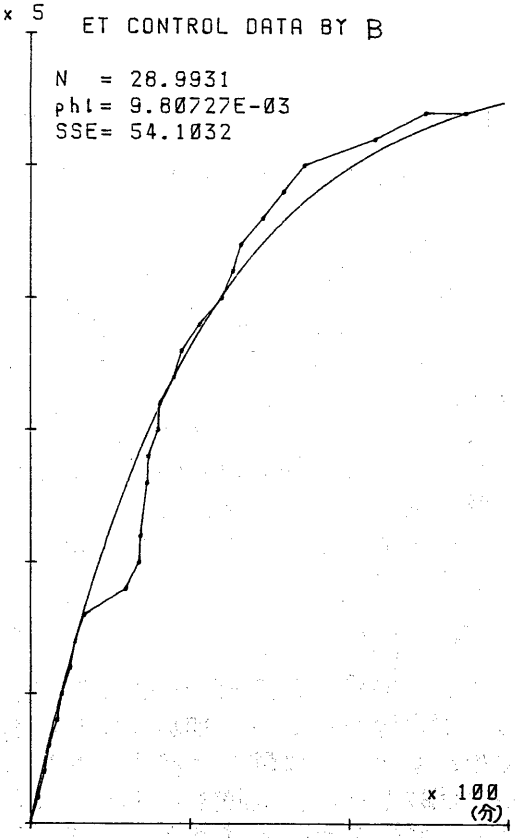


図6．被験者Bの制御欠陥発見過程

サンプル数の観点から1つのデータとして分析した．その結果，信頼度成長モデルとしては，指数型成長モデルが最も良く適合した．推定によって得られたモデルは：

$$m(t) = 29.0\,(1-e^{-0.010t}), \qquad (18)$$

であった．この被験者が試験中に発見した制御欠陥の総数は27個であった．従って，試験の質的尺度αは：

$$\alpha = \frac{m(t_\infty)}{42} = \frac{29.0}{42} \doteqdot 0.690, \qquad (19)$$

であった．

以上の結果より，この被験者による試験のSPQL評価値は：

$$SPQL \doteqdot 0.690 \times 0.650 \doteqdot 0.449, \qquad (20)$$

となる．このSPQL評価値より，この試験では真の欠陥の45%が除去されたと推定される．従って，真の欠陥の55%が残存したこととなり，結論として当初29.0個の真の欠陥がソフトウェア中に存在し，試験終了後もそのうち約16個が残存していたこととなる．

図7に被験者Cによる真の欠陥の発見過程とその分析結果を示す．被験者Cは，ソフトウェア製品に関する品質保証業務を担当する技術者で，アプリケーション・プログラムの試験のほか，

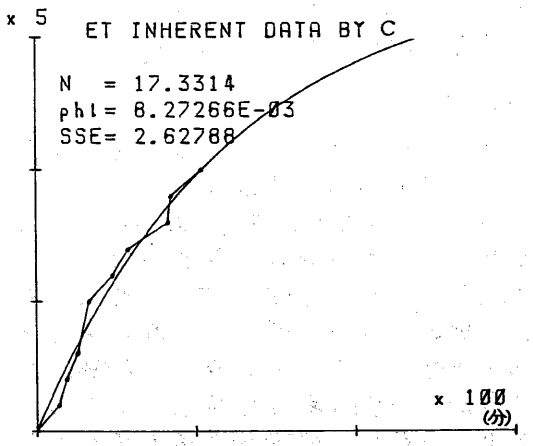


図7．被験者Cの真の欠陥発見過程

パーソナル・コンピュータの基本ソフトウェアの試験の経験をもつ．この被験者は，テスト用データとして，現実のテキストではなく，123456...のような順序をもつ無意味な文字列を利用するなど，試験の方法が他の被験者と異っていた．試験所要時間を考慮した場合，この被験者のバグ発見能力が最も高かった．図7から明らかなように，この場合信頼度成長モデルとしては指数型成長モデルが最も良く適合した．推定によって得られたモデルは：

$$m(t) = 17.3(1 - e^{-0.008t}), \qquad (21)$$

であった．この被験者の場合，試験中に発見した真の欠陥の総数は10であった．従って，試験の量的尺度γは：

$$\gamma = \frac{10}{m(t_\infty)} = \frac{10}{17.3} \fallingdotseq 0.578, \qquad (22)$$

であった．

図8に，同一被験者による制御欠陥の発見過程とその分析結果を示す．この場合，信頼度成長モデルとしては加速型S字成長モデルが最も良く適合した．推定によって得られたモデルは：

$$\mu(t) = 25.3\,\frac{1 - e^{-0.030t}}{1 + 3.5\,e^{-0.030t}}, \qquad (23)$$

であった．この被験者が試験中に発見した制御欠陥の総数は21個であった．従って，試験の質的尺度αは：

$$\alpha = \frac{\mu(t_\infty)}{42} = \frac{25.3}{42} \fallingdotseq 0.602, \qquad (24)$$

であった．

以上の結果より，この被験者による試験のSPQL評価値は：

$$SPQL \fallingdotseq 0.602 \times 0.578 \fallingdotseq 0.348, \qquad (26)$$

となる．このSPQL評価値より，この試験では真の欠陥の35%が除去されたと推定される．従って，真の欠陥の65%が残存したこととなり，結論として当初28.7個の真の欠陥がソフトウェア中に存在し，試験終了後もそのうち約19個が残存していたこととなる．

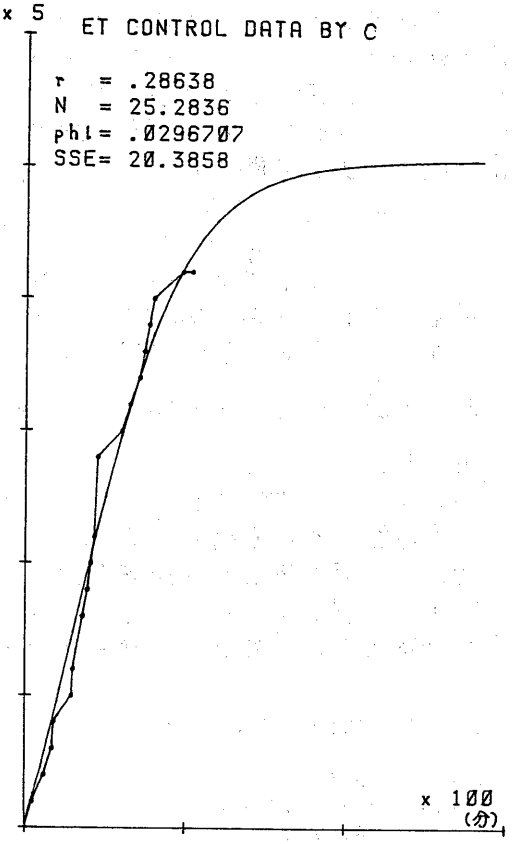


図8．被験者Cの制御欠陥発見過程

6．比較評価

前述の実験例について，SPQLによる総バグ数の推定値と，信頼度成長モデルによる推定値，および捕獲・再捕獲法による推定値との比較評価について述べる．ここで，信頼度成長モデルによる総バグ数の推定値$\hat{M}_i$とは，信頼度成長モデルの推定によって得られたモデルを$f(t)$とするとき，次式で与えられる：

$$\hat{M}_i = \lim_{t \to \infty} f(t) \qquad (27)$$

また，捕獲・再捕獲法による総バグ数の推定値$\hat{M}_i$とは，埋込まれた制御欠陥数をM_c，試験中に発見された制御欠陥数をm_c，試験中に発見された真の欠陥数をm_iとするとき，次式で与えられる：

$$\hat{M}_i = m_i \cdot \left(\frac{m_c}{M_c}\right)^{-1}. \quad (28)$$

表1に実験例についての比較結果を示す．実験中に発見された真の欠陥の総数は，全体で25個であった．従って，表1に示されているように，総バグ数の推定値としては，ＳＰＱＬによる推定が最も精度の良いものであった．表1における相対誤差とは，以下の定義による：

$$e = \frac{\hat{M}_i - M_i}{M_i} \quad (29)$$

ただし，eは相対誤差，M_iは最終的な真の欠陥の総数である．現実にM_iは25以上の値であるから，ＳＰＱＬの相対誤差は表1の値よりも小さいと言える．

7. まとめ

ＳＰＱＬの推定に必要な制御欠陥数の決定法と実験結果について議論した．実験結果からも明らかなように，ＳＰＱＬによる推定は，信頼度成長モデルや捕獲・再捕獲法による推定よりも精度が高かった．特に，他の方法による推定は，ソフトウェアの品質を高めに評価する傾向があることが指摘できる．また，制御欠陥数の決定法については，ほぼ満足すべき結果を得た．

［参考文献］

1) Ohba M. et al, "S-shaped software reliability growth curve: How good is it?" IEEE COMPSAC, Chicago, 1982.

2) 梶山他，「ソフトウェア・エラー発見過程のモデル」，情処学会24回全国大会7P-5，1982年3月．

3) 大場，梶山，「習熟型ソフトウェア信頼度成長モデル」，情処学会ソフトウェア工学研究会資料28-6，1983年2月．

4) 伊土他，「Capture & recapture法による潜在バグの推定法とその応用」，情処学会ソフトウェア工学研究会資料19-1，1981年7月．

5) Ohba M., "Software quality = test accuracy × test coverage," Proc. of ICSE, 1982.

表1．　比較評価

被験者	信頼度成長モデル	捕獲・再捕獲法	ＳＰＱＬ
A	8.4	15.3	22.2
B	20.0	20.2	29.0
C	17.3	20.0	28.7
平均相対誤差	-39.2%	-26.0%	+5.2%
最大相対誤差	-66.4%	-38.8	+16.0%