

習熟型ソフトウェア信頼度成長モデル

大場 充, 梶山 昌之

(日本アイ・ビー・エム(株))

1. はじめに

現在, 大規模ソフトウェアの新規開発プロジェクトでは, 開発工数全体の約50%が, テストに費されていると言われる。にもかかわらず, ソフトウェアにはエラーが潜存する。大規模ソフトウェアではテスト期間中に除去できなかったエラーとその副作用によって, 再現性の小さいエラー現象が発生することもある。そのようなエラーは, その原因究明が容易でないことから, 修復作業に多大なコストを要する。このような問題を開発工程で未然に防止するためには, ソフトウェア開発・管理技術の導入・改善が重要となる。ここでは, そのようなソフトウェアの品質管理のための道具の一つとして, ソフトウェア信頼度成長モデルについて議論する。

ソフトウェアの信頼度成長モデルを考えるとき, 既にハードウェア分野で確立している理論を応用しようとする立場がある。しかし, そのような理論の適用に際しては, ソフトウェアの特徴を考慮したアプローチが必要となる。そのような条件を満足するソフトウェア信頼度成長モデルとしては, Musaの実行時間モデル¹⁾, Littlewoodのバイジアン・モデル²⁾, そしてGoelらによる非定常ポアソン過程(NHPP)モデル³⁾等が知られている。これらは全て, Jelinski と Moranda によって提唱された以下の仮定に基づくモデルである。すなわち, 『ソフトウェア・エラーの発見率は, 当該ソフトウェアに潜存しているエラーの総数に比例する』とする仮定である。これは, 今ん個目までのエラーが発見済みであると仮定するとき, 発見率 r は n をパラメータとす

る次式で与えられるとする:

$$r(i) = \alpha(N-i), \quad (1)$$

ただし, N はソフトウェアに潜存する総エラー数, α は比例定数とする。

(1) 式の仮定によるモデルは, 図1の曲線 a に代表される指数型信頼度成長曲線をもつ。しかし, 現実のソフトウェア・エラー・データは, 図1の曲線 b に代表されるS字型信頼度成長であることが多い。本報告では, ソフトウェア・エラーの発見過程に関する古典的な Jelinski - Moranda の仮定を改善したS字型信頼度成長曲線をもつ, 習熟型信頼度成長モデルを提案する。また, Rome Air Development Center (RADC) のソフトウェア開発プロジェクトで収集されたエラー・データにモデルを適用し, Goel - Okumoto の NHPP モデルによる推定結果等との比較を行う。

2. ソフトウェア・エラーの発見過程

ソフトウェア・エラー発見に関する(1)式の自然な仮定にもかかわらず, 現実の信頼度成長が図1の曲線 b のようなS字型となる原因として以下のようなものが考えられる。すなわち, (1)ソフトウェア・エラー発見の定義の間

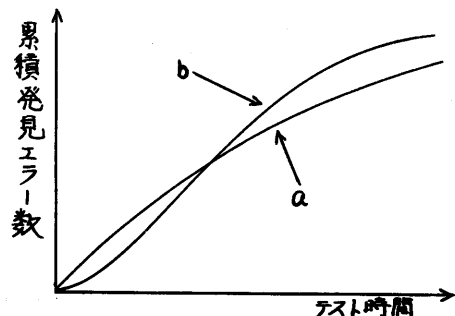


図1 ソフトウェア信頼度成長曲線

題, および, (2) テストの均一性の問題である。

ソフトウェア・エラー発見の定義の問題とは, 「何をもってソフトウェア・エラーを発見した」と言うかの問題である。これには, ソフトウェア・エラーを現象と考え, エラー現象発見をもってエラー発見とする立場がある。また, ソフトウェアのエラー現象を発生させる欠陥をエラーと考え, 欠陥の発見をもってエラー発見とする立場がある。後者では, エラーの原因が究明された時点, または, エラーの原因が除去された時点, をエラー発見時点とする。一般には, まずテストによってエラー現象が観測され, その後エラー解析を通してエラーの原因が究明されるという過程を辿るので, エラー原因が究明された時点, または, エラー原因が除去された時点, をエラー発見時点とする場合, エラー発見時間は遅れを含むことになる。(図2参照) この遅れが, S字型の信頼度成長の原因の一つと考えられる。すなわち, エラー現象の発見は(1)式に従い, さらにエラー原因の究明(発見)過程として(2)式を仮定する:

$$s(j) = \rho(i - j), \quad (2)$$

ただし, 現在までに j 個のエラーが原因究明を終り発見されているとする。 ρ は比例定数である。山田, 大場, 尾崎, 武田らは, そのようなエラー原因発見過程モデルとして, $\phi \approx \rho$ のとき(3)式を平均値函数とするNHPPモデルが定式化できることを示した。⁴⁾す

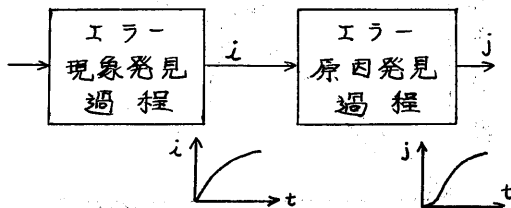


図2 エラー原因発見過程モデル

なめち, $\eta(t)$ を時刻 t までに発見(原因究明)済みの総エラー数とすると, $\eta(t)$ は:

$$\eta(t) = N \{ 1 - (1 + \lambda t) e^{-\lambda t} \}, \quad (3)$$

で与えられる。

次に, テストの均一性の問題とは, テスト期間中のテストチームによるエラーの現象発見努力およびその効果についての一様性か(1)式では暗に仮定されている。しかし, 現実のテストでは, エラーの現象発見努力は一定に分布してはいないだけでなく, エラーの現象発見能力も一定ではない。一般に, エラー現象発見努力は, 図3のようにテストの後半でピークに達する傾向がある。これが実行時間でなく, カレンダー時間データの場合S字型信頼度成長が観測される原因の一つであると考えられる。また, テストチームのソフトウェアに対する習熟という観点から考えると, エラーの現象発見能力はエラーを発見すればするほど向上する。この習熟によるエラー発見能力の向上もエラーの現象発見がS字型信頼度成長をする要因の一つであろう。

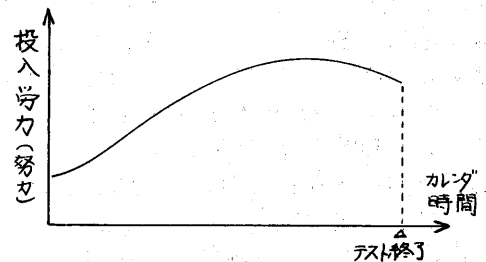


図3 エラー現象発見努力の時間分布

3. 習熟型信頼度成長モデル

N をソフトウェアに潜在するエラーの総数, $\eta(t)$ を時刻 t までに発見されているエラーの数とする。この場合, エラーとは観測されたエラー現象を言うものとする。ここで, エラーの発見

率 r が累積発見エラー数に比例して向上すると仮定すれば、(1) 式の仮定は次のように改善される：

$$r(i) = \phi \frac{i}{N} (N - i), \quad (4)$$

ただし、 i は累積発見エラー数である。ところで、(4) 式の r は単位時間当りの累積発見エラー数の増分を示すので、(4) 式は次のように書換え可能である：

$$\frac{d}{dt} m(t) = \phi \frac{n(t)}{N} (N - m(t)). \quad (5)$$

(5) 式は、ロジスチック曲線としてよく知られた S 字型の成長曲線となる。

一般的には、エラーには累積発見エラー数には独立な、発見の比較的容易なエラーと、発見には習熟を必要とするエラーとが混在する。従って、(5) 式を一般化した (6) 式が現実をよく説明すると考えられる：

$$\frac{d}{dt} n(t) = \phi \left\{ \lambda + (1 - \lambda) \frac{n(t)}{N} \right\} (N - n(t)), \quad (6)$$

ただし、 λ は定数とする ($0 \leq \lambda \leq 1$)。この (6) 式のモデルでは、 $\lambda = 0$ のときロジスチック型信頼度成長曲線となり、 $\lambda = 1$ のとき指数型信頼度成長曲線となる。(6) 式を $n(t)$ について解くと、習熟型信頼度成長曲線のモデルとしての平均値函数 $n(t)$ が求められる。すなわち：

$$n(t) = N \frac{(1 - e^{-\phi t})}{(1 + \lambda e^{-\phi t})}, \quad (7)$$

ただし、

$$\phi = \frac{1 - \lambda}{\lambda}, \quad (8)$$

とする。

(7) 式で与えられる習熟型信頼度成長モデルのパラメータは、最少 2 乗法または最尤法によって推定することができる。すなわち、最少 2 乗法では：

$$sse = \sum_{i=1}^K \left\{ y(t_i) - N \frac{1 - e^{-\phi t_i}}{1 + \lambda e^{-\phi t_i}} \right\}^2, \quad (9)$$

なる sse を最小化する N 、 ϕ 、および λ を探索すればよい。また、最尤法で

は：

$$L = \prod_{i=1}^K \frac{\{n(t_i)\}^{y(t_i)}}{y(t_i)!} e^{-n(t_i)} \quad (10)$$

で与えられる尤度を最大化する N 、 ϕ および λ を探索すればよい。ただし、ここでは、 t_i ($i = 1, 2, \dots, K$) の各観測時点において、 $y(t_i)$ なる発見エラー数が観測されているとする。

4. 適用例と比較評価

習熟型信頼度成長モデルを RAD C の 2 つのプロジェクトで収集されているソフトウェア・エラーのデータに適用した結果について議論する。

第 1 例は、RAD C Project 1 のテスト時に観測されたエラー現象発見データである。ソフトウェアは約 1300 K スタップの大きさで、テストは約 1 年間行われており、このテスト期間における投入労力は最低時と最高時で約 2 倍の開きがあった。このテストで発見されたエラーの総数は 2657 であった。図 4 に、カレンダー時間を基にしたデータと最尤法による推定結果を示す。本例の場合、信頼度成長は指数型をしており、図 5 に示した Goel-Okumoto のモデルによる推定と大きな差はなかった。これは、テストへの投入労力の時間分布が第 2 例に比較して一様であった。

× 500 RAD C PROJECT 1 TIME/ERROR DATA

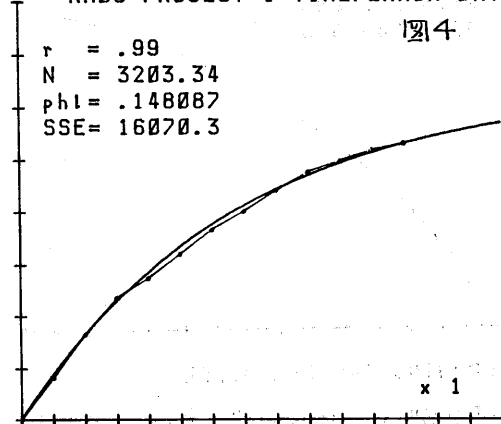
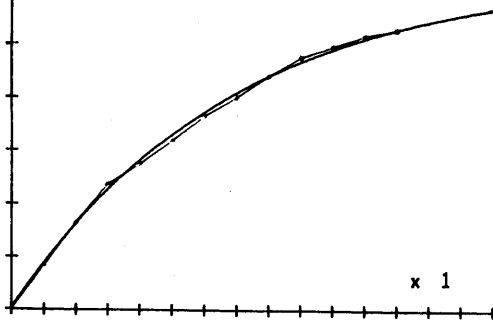


図 4 LEARNING S-SHAPE MODEL

x 500 RADC PROJECT 1 TIME/ERROR DATA

N = 3242.54
phi = .142632
SSE = 13354.2

図5



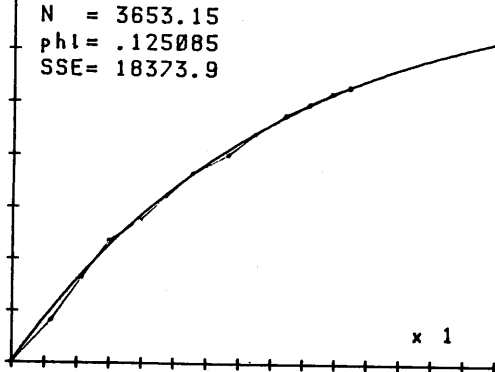
EXPONENTIAL MODEL

めと考えられる。また図6に実際に投入された労力(時間)を基にしたデータと最尤法による推定結果を示す。この場合も信頼度成長は明らかに指数型である。本例の場合、Nの推定値にモデルによる差が小さい。すなわち、ポアソン過程を応用した Jelinski-Moranda のモデルでは、Nの推定値はカレンダー時間で3228、労力で3721となっている。しかしながら本例の場合、全てのモデルでカレンダー時間と労力とのデータの差による差がある。

x 500 RADC PROJECT 1 EFFORT DATA

r = .99
N = 3653.15
phi = .125085
SSE = 18373.9

図6



LEARNING S-SHAPE MODEL

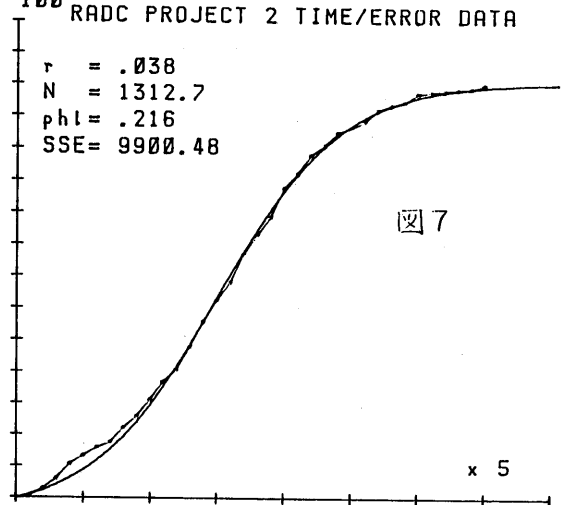
第2例は、RADC Project 2のテスト時に観測されたエラー現象発見デー

タである。ソフトウェアは約120 K ステップの大きさで、コンパイラ言語で書かれている。テスト期間は約3年間で、この期間における投入労力は最低時と最高時で約25倍の開きがあり、ピークは後半にカタヨリをもっている。このテストで発見されたエラーの総数は1301であった。図7に、カレンダー時間を基にしたデータと最尤法による推定結果を示す。また図8に、最尤法による推定結果を示す。本例の場合、

x 100 RADC PROJECT 2 TIME/ERROR DATA

r = .038
N = 1312.7
phi = .216
SSE = 9900.48

図7

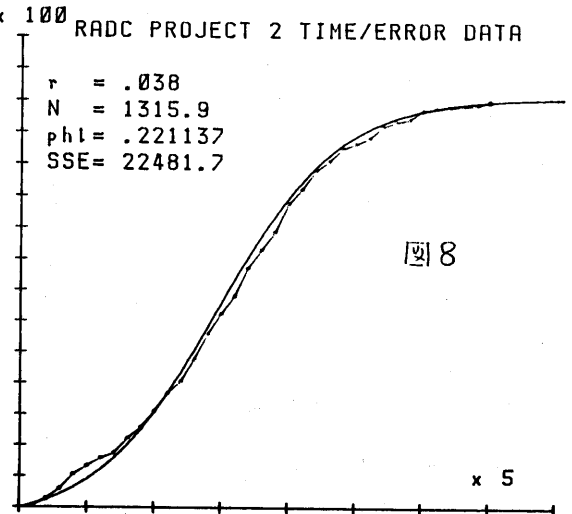


LEARNING S-SHAPE MODEL

x 100 RADC PROJECT 2 TIME/ERROR DATA

r = .038
N = 1315.9
phi = .221137
SSE = 22481.7

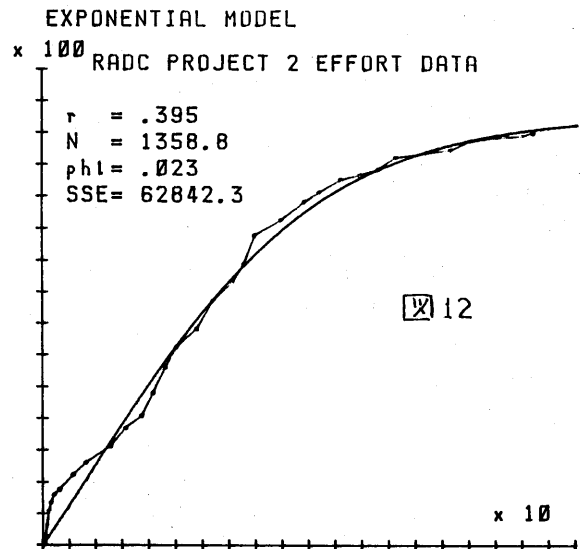
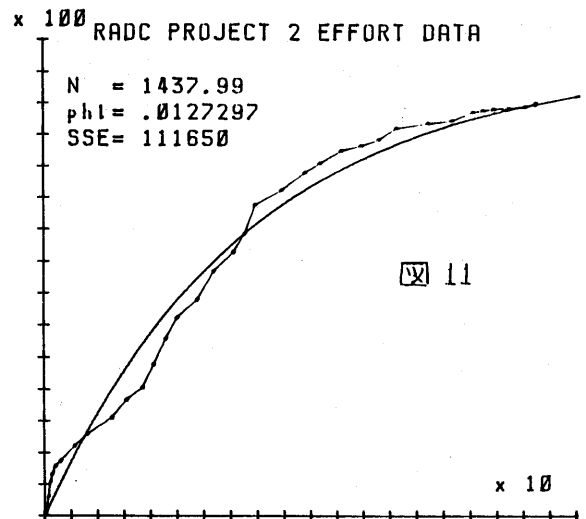
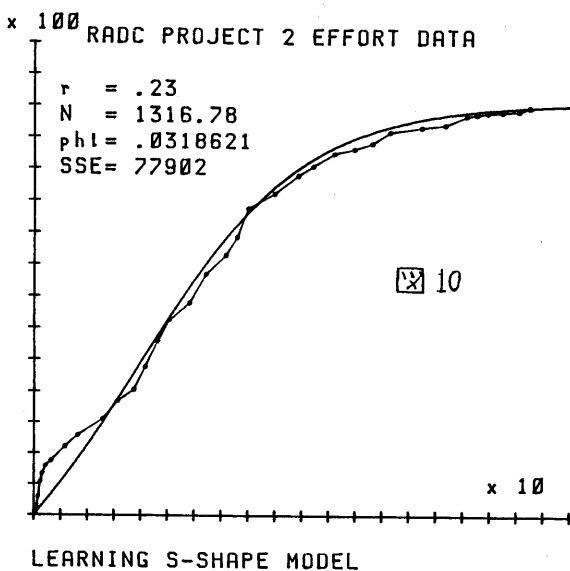
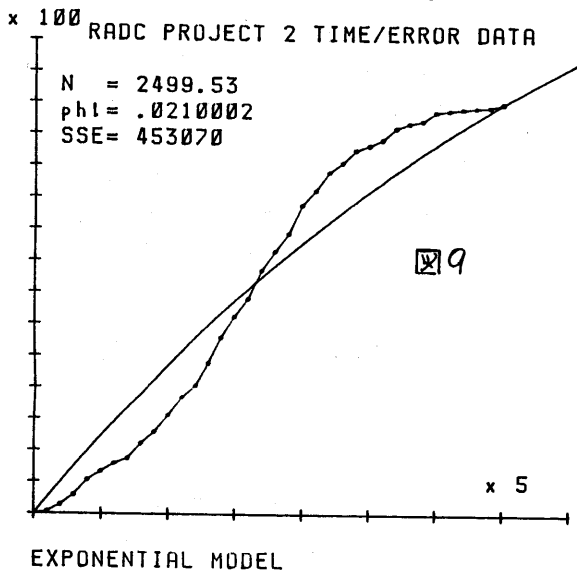
図8



LEARNING S-SHAPE MODEL

習熟型信頼度成長モデルでは推定法による差はなかった。しかし、指数型成長モデルでは、NHPPと定常ポアソン過程による推定法の違いによる差があった。図9からも明らかのように、本例の場合指数型成長モデルは適合しない。指数型成長モデルが適合しない理由としては、本例の場合テストへの投入労力の時間分布が一様でないことが第1に考えられる。

図10に投入された労力(時間)を基



にしたデータと最尤法による推定結果を示す。この場合、実際の信頼度成長は、指数型に近いともS字型に近いとも言える。図11に指数型のNHPPモデルによる推定結果を示す。推定曲線とデータとの誤差2乗和では、よりS字型に近い成長とすることができ。特に、後半のデータに対する適合度を重視する場合、S字型となる。図12に示した最尤法による推定結果では、最尤法の場合よりも指数型成長に近くなっている。本例の場合も、投入労力

を基にしたデータの場合には、モデルやパラメータの推定法による差がカレンダー時間を基にしたデータの場合よりも小さくなっている。

5. 結 論

以上のように、習熟型のソフトウェア信頼度成長モデルは、テストに投入される労力などエラー発見努力がテストの全期間にわたって一様に分布していないとき、エラー現象発見時点のデータに対してよく適合する。特に、カレンダー時間を基にしたエラー現象発見データに対して有効であることは、前述の第2例によく示されている。

逆に、Musa や Littlewood のモデルの場合のように、実行時間を基にしてエラー現象発見時間が与えられる場合、エラー発見努力の時間分布は、多くの場合習熟効果等によるエラー現象発見能力の向上を無視すれば、一様となるので、指数型信頼度成長モデルが適用可能となる。実際に、テストへの投入労力を基にした発見時間データの場合、指数型成長モデルのデータへの適合性は大きく改善されている。

ところで、前述の第2例では投入労力を基にした発見時間データの場合でも、相対的にS字型の信頼度成長が適合した。このことは、エラー発見努力の時間分布の均一性以外に、エラー発見率を左右するパラメータがあることを暗示している。実際にその主たる要因が、テストチームの習熟によるエラー発見能力の向上かどうかは、今後の興味ある研究課題である。

6. おわりに

本報告では、エラー発見努力や能力が、テスト期間中にわたり均一に分布していないことによって生じると考えられるソフトウェアのS字型信頼度成長について議論した。今後は、推定精

度の改善を目指したモデルの改良が重要となる。

最後に、本研究を行うにあたり御教示いただいた、広島大学工学部尾崎俊治助教授、およびIBMのL.A. Belady氏に謝意を表します。

参考文献

- 1) Musa J.D., "Validity of Execution-Time Theory of Software Reliability," IEEE Trans. Reliability, vol. R-28, 1979
- 2) Littlewood B., Verrall J., "Likelihood function of a debugging model for computer software reliability," IEEE Trans. Reliability, vol. R-30, 1981
- 3) Goel A.L., Okumoto K., "Time-dependent Error detection rate model for software reliability and other performance measures," IEEE Trans. Reliability, vol. R-28, 1979
- 4) Ohba M., Yamada S., Takeda K., Ozaki S., "S-shaped software reliability growth curve: How good is it?" IEEE Proc. COMPSAC 82, 1982.
- 5) 梶山, 大場, テストの習熟を考慮したソフトウェア信頼度成長モデル, 情報学会 25回全国会議論文集 1E-3, 1982
- 6) Littlewood B., Verrall J., "A Bayesian reliability growth model for computer software," IEEE Symp. on computer software reliability - New York, 1973.