

## 分散開発環境におけるソフトウェア信頼性評価モデルに関する考察

02101865 鳥取大学 \*木村光宏 KIMURA Mitsuhiro  
鳥取大学 田村慶信 TAMURA Yoshinobu  
01702425 鳥取大学 山田茂 YAMADA Shigeru

## 1 はじめに

ソフトウェアの開発環境は、これまでホスト集中型のものが多かった。しかし、その後のWS（ワークステーション）やデータ通信ネットワークの進歩によって、現在のソフトウェア開発は、ホスト集中型の開発環境から、分散開発環境へと変わりつつある [1]。また、こうしたデータ通信ネットワークの進歩により、情報ネットワークはますます大規模化し、従来の2層構造のClient/Server System（以下、CSSと略す）に代わり、経営戦略手法の変更が迅速・容易またはclientの増設が迅速・安価といった理由から、3層構造あるいはN層構造のCSSが多用されるようになってきている [2]。本研究では、こうした分散開発環境において、ソフトウェアコンポーネント（ソフトウェア部品）を組み立てることにより、新たなソフトウェアを生産する場合のソフトウェア信頼性を定量的に評価するモデルについて考察する。

## 2 モデルの記述

従来より、多数のソフトウェア信頼性評価モデルが提案され、それらのいくつかのものは実際のソフトウェア開発現場において開発支援ツールの一部として実装され始めている。中でも、テスト工程におけるソフトウェアフォールトの発見・修正過程にNHPP（非同次ポアソン過程）を適用したモデルは、モデルが簡単であるがゆえにその拡張性も高く、多くの研究者らがそれぞれに特徴の異なるSRGM（software reliability growth model）を提案してきている [3]。

本研究で議論するソフトウェア信頼性評価モデルを構築する際の仮定を以下に示す。

- i ソフトウェアコンポーネントには、既存のコンポーネントと新規開発のコンポーネントが混在する。
- ii ソフトウェア故障発生現象に対してNHPP（非同次ポアソン過程）を導入する。
- iii ソフトウェアフォールトは完全に除去できるもの（完全デバッグモデル）とする。

今までの経験から、ソフトウェアコンポーネントを組み合わせて作るソフトウェアシステムをテストすると、累積発見フォールト数の成長曲線は指数形になることが

多く、これに対して新規開発のソフトウェアシステムをテストするとS字形成長曲線のグラフを描くことが多いことが知られている。このことから、ここではNHPPに基づく指数形ソフトウェア信頼度成長モデルと、遅延S字形成長ソフトウェア信頼度成長モデルを導入する。NHPPモデルは、その平均値関数により特徴づけられるので、以下に本研究で考察するモデルにおける平均値関数の構成について述べる。一般に、指数形ソフトウェア信頼度成長モデルの平均値関数  $H_e(t)$ 、すなわち時間区間  $(0, t]$  における総期待発見フォールト数は、

$$H_e(t) = a(1 - e^{-bt}) \quad (a > 0, b > 0), \quad (1)$$

と表され、遅延S字形成長ソフトウェア信頼度成長モデルの平均値関数  $H_d(t)$  は、

$$H_d(t) = a\{1 - (1 + bt)e^{-bt}\} \quad (a > 0, b > 0), \quad (2)$$

と表される。ここで、 $a$  はテスト開始前に潜在していた総フォールト数を、 $b$  は残存フォールト1個当りのフォールト発見率を表す。

以上のことから、次の平均値関数をもつNHPPモデルを考える。

$$H(t) = a\left[\sum_{i=1}^n p_i(1 - e^{-b_i t}) + \sum_{j=1}^m p_{n+j}\{1 - (1 + b_{n+j}t)e^{-b_{n+j}t}\}\right] \quad (\sum_k p_k = 1, p_k > 0, a > 0, b > 0), \quad (3)$$

これは、既存のコンポーネントが  $n$  個、新規開発のコンポーネントが  $m$  個使われたと仮定し、パラメータ  $p_i$  は  $i$  番目のコンポーネントに対するテストの重みを表している。これは、それぞれのコンポーネントについてよくテストされるものとそうでないものがあることを想定している。

## 3 モデルパラメータの推定

モデルパラメータの推定についてのここでの仮定として、すべてのコンポーネントを結合し、インターフェー

スの整合性を確認した後に行われる総合テストにおいて発見されるフォールトの発見・修正作業から採取されるデータが得られるものとする。もちろん、より精密なフォールトの発見・修正過程の把握のためには、各コンポーネントごとのテスト工程でのデータを解析することも考えられるが、一般にコンポーネントの規模が小さくなると、統計的推定手法が適用しづらくなることから、上記の仮定を設けている。また、各コンポーネントの総合テストに関する影響度合いを表す重みパラメータ  $p_i$  ( $i = 1, \dots, n+m$ ) の値は、何らかの方法によって与えることが出来るものと仮定しておく。

本研究では、モデルに含まれる残りの未知パラメータ ( $a, b_1 \sim b_n, b_{n+1} \sim b_{n+m}$ ) の推定方法として最尤法を適用する。ソフトウェア開発のテスト工程で所定のテスト時間間隔  $(0, t_k]$  において発見された総フォールト数  $y_k$  に関する  $n$  組のフォールト発見数データ  $(t_k, y_k)$  ( $k = 1, 2, \dots, n$ ) が観測されたものとする。このときの、式 (3) の平均値関数  $H(t)$  をもつ NHPP モデルの対数尤度関数は、

$$\ln L = \sum_{k=1}^n (y_k - y_{k-1}) \cdot \ln[H(t_k) - H(t_{k-1})] - H(t_n) - \sum_{k=1}^n \ln[(y_k - y_{k-1})!], \quad (4)$$

となる。ここで、各パラメータ ( $a, b_1 \sim b_n, b_{n+1} \sim b_{n+m}$ ) の最尤推定量を求めるために、 $a$  および  $b_i, b_{n+j}$  について式 (4) を偏微分して、

$$\frac{\partial \ln L}{\partial a} = \frac{\partial \ln L}{\partial b_i} = \frac{\partial \ln L}{\partial b_{n+j}} = 0, \quad (5)$$

と整理することにより得られる  $(n+m+1)$  元非線形連立方程式を、それぞれのパラメータについて数値的に解いてモデルパラメータ ( $\hat{a}, \hat{b}_1 \sim \hat{b}_n, \hat{b}_{n+1} \sim \hat{b}_{n+m}$ ) を推定する。

#### 4 数値例

ここでは、新規開発コンポーネント数を  $m = 2$ 、既存のコンポーネント数を  $n = 2$  とした簡単な場合について数値例を示す。パラメータ  $p_i$  を、( $p_1 = 0.05, p_2 = 0.05, p_3 = 0.45, p_4 = 0.45$ ) とし<sup>1</sup>、実測データを解析すると、図 1 に示すような平均値関数  $H(t)$  の最尤推定値  $\hat{H}(t)$  と、その 95% 信頼限界を得た。ここで、他のモデルパラメータは、

$$\hat{a} = 378.12, \quad \hat{b}_i = 0.4654, \quad \hat{b}_{2+j} = 0.1788,$$

<sup>1</sup>重みパラメータ  $p_i$  に関する考察については、紙面の都合上ここでは割愛する。

と推定された ( $i, j = 1, 2$ )。次に、期待残存フォールト数  $n(t)$  の推定結果を図 2 に示す。テスト時刻  $t$  におけるソフトウェア内の期待残存フォールト数は、確率変数  $\{N(\infty) - N(t)\}$  の期待値を考えることにより、

$$n(t) = E[N(\infty) - N(t)] = a - H(t), \quad (6)$$

により表される。したがって、テスト終了時点  $t = t_{19} = 19$  (週) までに発見された総フォールト数は 328 個であるので、テストにより未発見となったソフトウェア内の残存フォールト数は、

$$\hat{a} - 328 = 50.12,$$

より約 50 個と推定される。

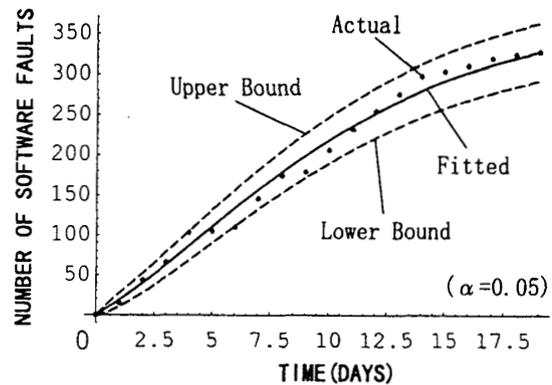


図 1. 信頼性評価結果

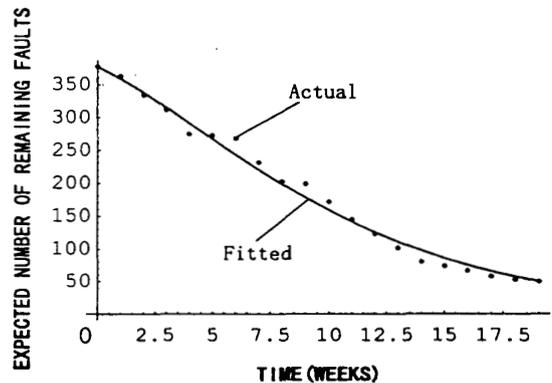


図 2. 期待残存フォールト数の推定結果

#### 参考文献

- [1] 松本 正雄, 小山田 正史, 松尾谷 徹, 「ソフトウェア開発検証技法」, 電子情報通信学会, 1997.
- [2] 木村 光範, 益田 昭彦, 鈴木和幸, 「Client/Server System の Performability に関する研究」, 「信頼性」, Vol.20, No.4, pp.228-229, 1998.
- [3] 山田 茂, 「ソフトウェア信頼性モデル-基礎と応用-」, 日科技連出版社, 1994.

#### 謝辞

本研究の一部は、(財)大川情報通信基金・研究助成(助成番号: 98-09)の援助による。