

デスクトップ環境における ソフトウェア信頼性評価法に関する一考察

鳥取環境大学・環境情報学部
情報システム学科
田村 慶信 (Yoshinobu Tamura)
Department of Information
Systems, Faculty of Environmental
and Information Studies,
Tottori University of
Environmental Studies
E-mail: tamura@kankyo-u.ac.jp

鳥取大学・工学部
社会開発システム工学科
山田 茂 (Shigeru Yamada)
Department of Social Systems
Engineering, Faculty of
Engineering, Tottori University
E-mail: yamada@sse.tottori-u.ac.jp

法政大学・工学部
経営工学科
木村 光宏 (Mitsuhiro Kimura)
Department of Industrial and
Systems Engineering, Faculty of
Engineering, Hosei University
E-mail: kim@k.hosei.ac.jp

概要

現在のソフトウェア開発は、ネットワークを基にした分散開発が主流となっている。中でも、世界中の誰もが開発に参加できるという特徴をもつオープン・ソース・プロジェクトは、分散型ソフトウェア開発形態の成功例として特に注目されている。

本論文では、オープン・ソース・プロジェクトの下で分散共同開発されたソフトウェアシステムに対する信頼性評価法を提案する。特に、オープン・ソース・プロジェクトの下で開発された UNIX 系 OS 上で動作する X デスクトップ環境を 1 例にとり、意思決定手法の 1 つである AHP (Analytic Hierarchy Process) 手法と、NHPP に基づくソフトウェア信頼度成長モデル (software reliability growth model, 以下 SRGM と略す) を適用した信頼性評価法について考察する。さらに、実際のフォールト発見数データに基づいた数値例も示す。

1 はじめに

1970 年代から着々と進歩し続けてきた情報技術は、現在では我々の生活に必要不可欠なものとなっている。特に、ハードウェアの技術の進歩により、10 年前とは比べ物にならないほど高性能、ダウンサイジング化されたコンピュータが、企業の業務支援などに使用されている。さらに、近年のインターネットの普及により世界中が同時に新しい情報を得ることができるようになり、実時間的あるいはインタラクティブ性の高い機能の追求へと関心が深まっている。こうした状況から、ネットワークを基にしたソフトウェアの分散開発、およびソフトウェアそのものの分散化がさらに拡大してきた。特に、最近のソフトウェア開発は、クライアント/サーバ・システムや Web プログラミング、オブジェクト指向開発、ネットワーク環境での分散開発といった新しい開発形態が多用されるようになってきている [1, 2]。現在、分散ソフトウェア共同開発は、同一企業内における開発形態から、複数のソフトウェアハウスや同一企業内、複数の企業間での遠隔地間共同開発、さらには、多くの開発者が協調しながら開発を行うオープン・ソース・プロジェクトなどの様々な形態が存在する [3]。

本論文では、オープン・ソース・プロジェクトの下で開発が進められているソフトウェアとして、X ウィンドウシステム用のデスクトップ環境を取り上げる。UNIX 系 OS 上では、さまざまなオープン・ソースのデスクトップ環境がある。デスクトップ環境とは、ユーザに統一したインタフェース・デザイン・操作性などを提供するソフトウェアで、UNIX 系 OS 上にグラフィカルな操作環境を提供してその操作性を高めることが目的とされている。その代表的なものとして、GNOME や KDE などが開発されている。特に、GNOME や KDE は、柔軟で使いやすい GUI を備えている反面、その操作性の重さ、ライブラリの多さ、依存性の複雑さといったことが指摘されている。本論文では、GNOME や KDE よりも軽量であり、見た目や機能ではこれらの有名なデスクトップ環境よりもひけをとらない、Xfce [4] と呼ばれるデスクトップ環境のソフトウェア信頼性評価法について考察する。Xfce は、現在 Gtk+2 をベースにフルスクラッチで書き直され、バージョン 4.0 としてリリースされている。

従来から、ソフトウェア製品の開発プロセスにおけるテスト進捗管理や出荷品質の把握のための信頼性評価を行うアプローチとして、ソフトウェア故障の発生現象を不確定事象として捉えて確率・統計論的に取り扱う方法がとられている。その1つが、ソフトウェア信頼度成長モデル (software reliability growth model, 以下 SRGM と略す) である [5]。これまでに、分散ソフトウェア開発環境を対象とした SRGM が提案されている [6, 7]。特に、分散開発環境においては、アーキテクチャ、マシン構成などの組み合わせに自由度をもつことから、従来のソフトウェア開発環境のようにソフトウェアコンポーネントを単に積み上げていくという開発ではなく、すり合わせて構築するような側面がある。開発規模が大きく、中に含まれる言語やシステムの構成要素が増すに従い、このような形はより一層複雑化する。こうした背景から、現在のところ分散開発環境における有効なテスト方法は提案されていない。

本論文では、オープン・ソース・プロジェクトの下で分散共同開発され、UNIX 系 OS 上で動作するデスクトップ環境の1つである Xfce に対する信頼性評価法を提案する。特に、意思決定手法の1つである AHP 手法を用いて各コンポーネントに対する重要度を推定するとともに、SRGM に基づき各コンポーネント間の相互作用を包括した信頼性評価法を提案する。また、実際のフォールト発見数データに対する数値例も示す。

2 各コンポーネントに対する信頼性評価

2.1 SRGM に基づく信頼性評価

従来から、ソフトウェアの信頼性を定量的に評価する手法として、SRGM による方法がとられている。中でも非同次ポアソン過程 (nonhomogeneous Poisson process, 以下 NHPP と略す) モデルは、実利用上極めて有効でありモデルの簡潔性が高いゆえにその適用性も高く、実際のソフトウェア信頼性評価に広く応用されている。この NHPP モデルは、所定の時間区間内に発見されるフォールト数や発生するソフトウェア故障数を観測して、これらの個数を数え上げる計数過程 $\{N(t), t \geq 0\}$ を導入し、以下の式で与えられる確率変数すなわちポアソン過程を仮定する SRGM である [5]。

$$\Pr\{N(t) = n\} = \frac{\{H(t)\}^n}{n!} \exp[-H(t)] \quad (n = 0, 1, 2, \dots). \quad (1)$$

ここで、 $\Pr\{\cdot\}$ は確率を表し、 $H(t)$ は時間区間 $(0, t]$ において発見される総期待フォールト数、すなわち $N(t)$ の期待値を表し、NHPP の平均値関数と呼ばれる。

本研究では、各コンポーネントについて累積発見フォールト数データの成長曲線の形状により、以下に示す NHPP モデル [5] のうち最適なモデルを適用する。

☐ 指数形 SRGM

☐ 習熟 S 字形 SRGM

さらに、モデルに含まれる未知パラメータの推定方法として最尤法を適用する。上記の NHPP モデルから、種々のソフトウェア信頼性評価のための定量的尺度を導出できる。

2.1.1 指数形 SRGM

残存フォールト 1 個当りのフォールト発見率または残存フォールト 1 個当りのソフトウェア故障発生率は、一様であり、指数形 SRGM は $b(t)$ が時刻 t に関して一定であるので、一定型フォールト発見率をもつと言われる。その平均値関数 $E_i(t)$ は、次式によって与えられる。

$$E_i(t) = a_i(1 - e^{-b_i t}) \quad (a_i > 0, b_i > 0). \quad (2)$$

ここで、 $E_i(t)$ は $i(i = 1, 2, \dots, n)$ 番目のソフトウェアコンポーネントに対して適用された指数形 SRGM の平均値関数であり、時間区間 $(0, t]$ において発見される累積フォールト数の期待値を表す。パラメータ a_i は最終的に発見される総期待フォールト数、パラメータ b_i はフォールト 1 個当りのソフトウェア故障発見率またはフォールト発見率を表す。

2.1.2 習熟 S 字形 SRGM

習熟 S 字形 SRGM の平均値関数は、

$$D_i(t) = \frac{a_i(1 - e^{-b_i t})}{(1 + c_i \cdot e^{-b_i t})} \quad (a_i > 0, b_i > 0, c_i > 0), \quad (3)$$

により与えられる。式 (3) のパラメータ c_i は、各ソフトウェアコンポーネントの特徴を近似的に表すものと仮定する。ここで、 $D_i(t)$ は i 番目のソフトウェアコンポーネントに対して適用された習熟 S 字形 SRGM の平均値関数であり、時間区間 $(0, t]$ において発見される累積フォールト数の期待値を表す。また、 a_i は、各コンポーネントにおいて最終的に発見される総期待フォールト数を表す定数パラメータである。さらに、パラメータ b_i は各コンポーネントにおけるフォールト 1 個当りの発見率、および c_i はテストに対する習熟性を表す習熟係数を表す。

本論文では、各コンポーネントに対して、平均偏差 2 乗和 (mean squared errors, 以下 MSE と略す) を評価基準とした SRGM の選択を行う。MSE は、実測値と推定値との 2 乗誤差をデータ数で平均化したものである。ここで、一定のテスト時刻 t_k までに発生した累積フォールト数 y_k に関する K 組の発見フォールト数データ $(t_k, y_k)(k = 1, 2, \dots, K)$ が観測されているものとする、

$$\text{MSE} = \frac{1}{K - P} \sum_{k=1}^K (y_k - \hat{y}_k)^2, \quad (4)$$

により計算される。式 (4) の $\hat{y}_k$ は、テスト時刻が $t_k(k = 1, 2, \dots, K)$ のときの推定値である。また、 P はモデルに含まれるパラメータ数を表す。平均偏差 2 乗和は、その値が小さいほど実測データに良く適合していることを意味している。

2.2 AHP に基づく重み係数の推定

1970 年代に開発された AHP (Analytic Hierarchy Process) は、主観的判断による意思決定支援に有効な方法として、欧米を中心に経営問題、エネルギー問題、政策決定、都市計画学など様々な分野で広く活用されている [8, 9]。

ソフトウェアの信頼性評価手法の開発において、各コンポーネントでのデバッグの状況やその良し悪しが、システム全体の信頼性に与える影響を考慮しようとする場合、プログラムパス、コンポーネントの規模、フォールト報告者のスキルなどの、様々に絡み合った要因を捉える必要があると考えられる。しかしながら、これらを考慮することは困難であることが予想される。したがって本論文では、こうした複雑な状況下でシステム全体の信頼性に対する各コンポーネントの影響度合いを推定するために、一般には主観的判断の合理的合成方法として知られている AHP を利用し、システム全体の信頼性に対する各コンポーネントの重要度を表す重み係数の推定を行う。特に、適用される評価基準としては、各コンポーネントに対して発見されたフォールトの重要度、コンポーネントの規模、フォールト報告者のスキルといった要因が考えられる。各コンポーネントにおける AHP の評価基準に

対する重み係数を、それぞれ $w_i (i = 1, 2, \dots, n)$ とすれば、一対比較行列は、

$$A = \begin{bmatrix} \frac{w_1}{w_1} & \frac{w_1}{w_2} & \dots & \frac{w_1}{w_n} \\ \frac{w_2}{w_1} & \frac{w_2}{w_2} & \dots & \frac{w_2}{w_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{w_n}{w_1} & \frac{w_n}{w_2} & \dots & \frac{w_n}{w_n} \end{bmatrix}, \quad (5)$$

となる。この一対比較行列から、各評価基準に対する重み係数を次式の幾何平均により求めることができる。

$$\begin{aligned} \alpha_i &= \sqrt[n]{\prod_{j=1}^n x_{ij}}, \\ x_{ij} &= \frac{w_i}{w_j}. \end{aligned} \quad (6)$$

以上のことから、各ソフトウェアコンポーネントに対する重み係数は、

$$p_i = \frac{\alpha_i}{\sum_{i=1}^n \alpha_i}, \quad (7)$$

により与えられる。

3 システム全体に対する信頼性評価

フォールト発見の意味と同様に重要なソフトウェア信頼性要因としては、フォールト発見の独立性に関するものがある。ソフトウェア構造とプログラムパスの観点から考えて、発見されるフォールトには、潜在するフォールトが互いに独立で発見の容易なものと、独立ではなく発見にはその前に他のフォールトの発見を必要とするものとが混在すると仮定した習熟S字形SRGMが提案されている。その平均値関数 $H_c(t)$ は、式(3)と同様にして

$$H_c(t) = \frac{a(1 - e^{-bt})}{(1 + C \cdot e^{-bt})}, \quad (a > 0, b > 0, C > 0), \quad (8)$$

により与えられる。ここで、パラメータ a は最終的に発見される総期待フォールト数、パラメータ b はフォールト1個当りの発見率を表す。本論文では、パラメータ C は各コンポーネントに対して推定されたパラメータ c_i と2.2のAHP手法により推定された重みパラメータ p_i との重み付き平均により表されるものとし、

$$C = \frac{\sum_{i=1}^n p_i \cdot c_i}{\sum_{i=1}^n p_i} = \sum_{i=1}^n p_i \cdot c_i, \quad (9)$$

により定義する。ここで、 n はソフトウェアのコンポーネント数を表す。さらに、 p_i は各コンポーネントに対する重みパラメータを表し、システム全体に対する各コンポーネントの重要度を表す。また、 c_i は i 番目のコンポーネントに対する習熟S字形SRGMに含まれる習熟係数を表す。

式(8)の平均値関数をもつNHPPモデルから、ソフトウェア信頼性評価のための定量的尺度を導出できる。

例えば、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 $\{N_c(\infty) - N_c(t)\}$ の期待値を考えることにより

$$N_c(t) \equiv E[N_c(\infty) - N_c(t)] = a - H_c(t), \quad (10)$$

により与えられる。

4 数値例

4.1 各コンポーネントに対する信頼性評価

Xfce のフォールトデータを適用した数値例を示す。本研究で用いたデータは、6つのコンポーネントから構成された Xfce デスクトップ環境におけるバグトラッキングシステムから採取されたものである。

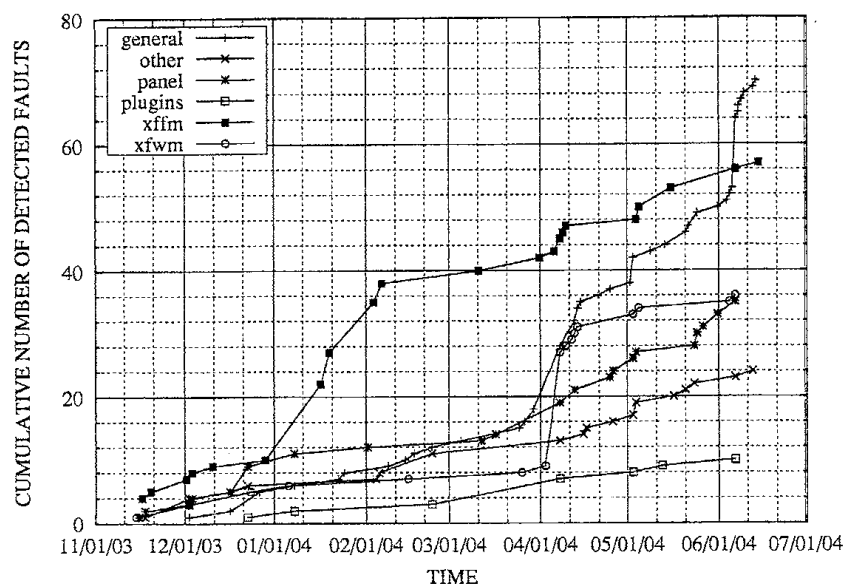


図 1：各コンポーネントの累積フォールト発見数データ。

各コンポーネントの累積フォールト発見数データを図 1 に示す。さらに、システム全体としての累積フォールト発見数データを図 2 に示す。各コンポーネントに対するモデルの適用基準としては MSE を採用した。各コンポーネントに対する MSE の推定結果を表 1 に示す。

表 1：各コンポーネントに対する MSE の推定結果。

	Exponential SRGM	Inflection S-shaped SRGM
general	—	12.055*
other	2.4387	2.4272*
panel	—	3.9734*
plugins	—	4.4011*
xffm	29.546	22.429*
xfwm	29.189	18.740*

(* means the selected model)

次に、2.2 の AHP に基づく各コンポーネントに対する重みパラメータ $p_i (i = 1, 2, \dots, n)$ の推定結果を表 2 に示す。特に、評価基準としては、各コンポーネントに対するフォールトの重要度を上げた。表 2 から、xffm コンポーネントに対する重要度が最も大きいことが分かる。一方、plugins コンポーネントに対する重要度は最小であることが確認できる。

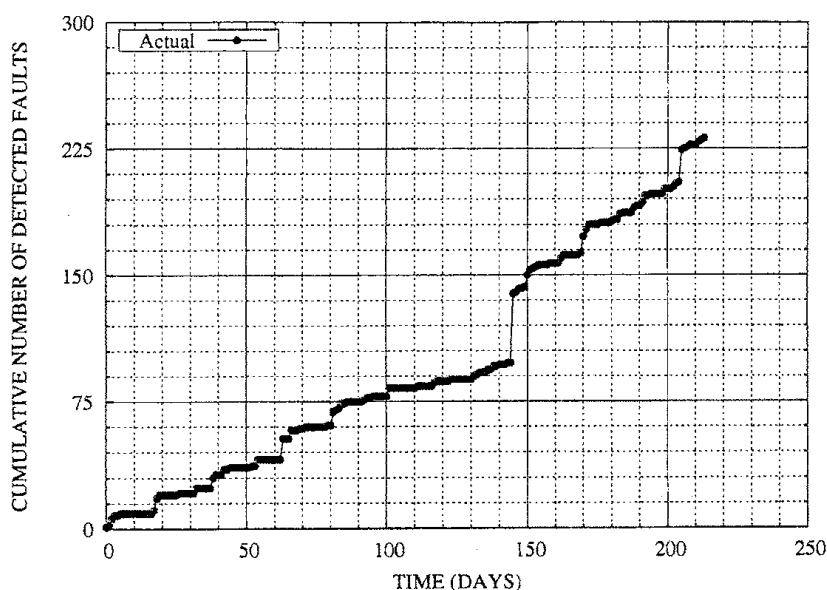


図 2：システム全体の累積フォールト発見数データ。

表 2：AHP に基づく各コンポーネントに対する重み係数の推定結果。

Component	Weight parameter p_i
general	0.20314
other	0.07975
panel	0.06936
plugins	0.01543
xffm	0.43629
xfwm	0.19604

4.2 システム全体に対する信頼性評価

次に、各コンポーネントに対して適用された SRGM に含まれる未知パラメータを最尤法により推定された結果を踏まえて、Xfce デスクトップ環境の信頼性評価の一例を示す。式 (8) における累積フォールト発見数の期待値の推定値 $\hat{H}_c(t)$ および式 (10) における残存フォールト数の期待値の推定値 $\hat{N}_c(t)$ を図 3 および図 4 に示す。

5 おわりに

本論文では、オープン・ソース・プロジェクトの下で分散共同開発されている UNIX 系 OS 上で動作するデスクトップ環境の 1 つである Xfce に対する信頼性評価法について議論した。特に、意思決定手法の 1 つである AHP 手法に基づき各コンポーネントに対する重み係数を推定し、習熟 S 字形 SRGM に基づいた各コンポーネント間の相互作用を包括した信頼性評価法を提案した。また、実際の Xfce に基づいたフォールト発見数データに対する数値例を示した。さらに、式 (8) に対する累積フォールト発見数と残存フォールト数の各期待値、ソフトウェア信頼度、瞬間 MTBF および累積 MTBF を導出した。

本論文では、各コンポーネントに対する重要度を表す重みパラメータ $p_i (i = 1, 2, \dots, n)$ に対して AHP 手法を適用した。AHP では、評価基準と代替案との間は独立であるべきと定めているが、現実には評価基準と代替案との間に依存関係があり、その依存関係を組み込んだ形で評価可能な ANP (Analytic Network Process) に基づい

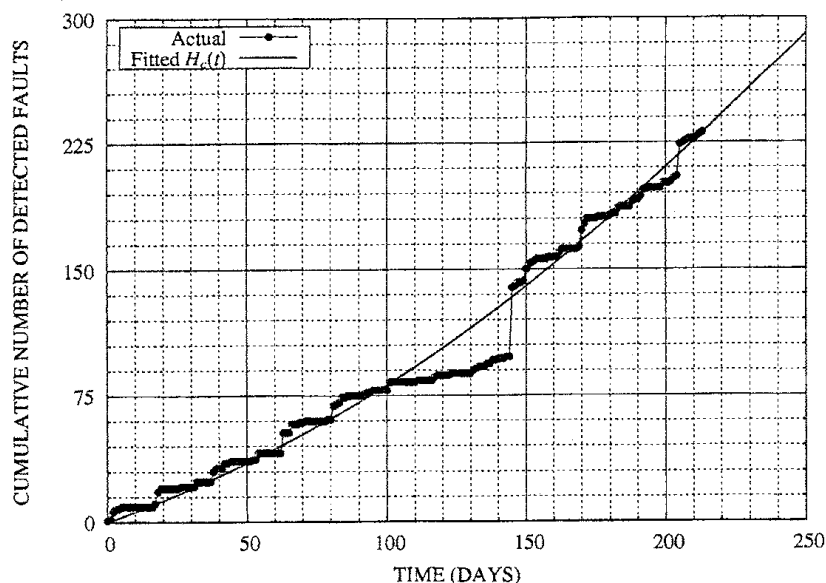


図 3：推定された累積フォールト発見数の期待値， $\widehat{H}_c(t)$ 。

た重み係数の推定を行う必要がある。今後は、ANPに基づく信頼性評価法とともに、取り上げる評価基準についてもフォールトレベルだけではなく、その他の評価基準についても考慮する必要がある。

従来のSRGMは、同一企業内において開発されたソフトウェアシステムのテスト工程に対する信頼性予測のために使用されてきた。一方、本論文で対象としているXfceは、デスクトップ上で様々なアプリケーションソフトウェアが動作する環境であるために、従来のような単体で動作するソフトウェアシステムとは環境が大きく異なることが考えられる。今後は、こうしたアプリケーションソフトウェアのデスクトップ環境に及ぼす影響などについても考慮する必要がある。さらに、本論文では、ソフトウェア内における検出可能フォールト数が有限と仮定したモデルを適用したが、検出可能フォールト数が無限であると仮定されたモデルによる考察も必要であると考えられる。

謝辞

本研究の一部は、文部科学省科学研究費基盤研究(C)(2)（課題番号15510129）の援助を受けたことを付記する。

参考文献

- [1] A. Umar, *Distributed Computing and Client-Server Systems*, Prentice Hall, New Jersey, 1993.
- [2] 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, 東京, 1997.
- [3] 赤羽 豊和, クライアント/サーバ・システムのテスト技法, ソフト・リサーチ・センター, 東京, 1998.
- [4] Xfce - Desktop Environment, <http://www.xfce.org/>.
- [5] 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, 東京, 1994.
- [6] 田村 慶信, 木村 光宏, 山田 茂, “分散開発環境に対するソフトウェア信頼度成長モデル: 確率微分方程式アプローチとその推定,” 日本応用数理学会論文誌, vol. 11, no. 3, pp. 121-132, 2001年9月.

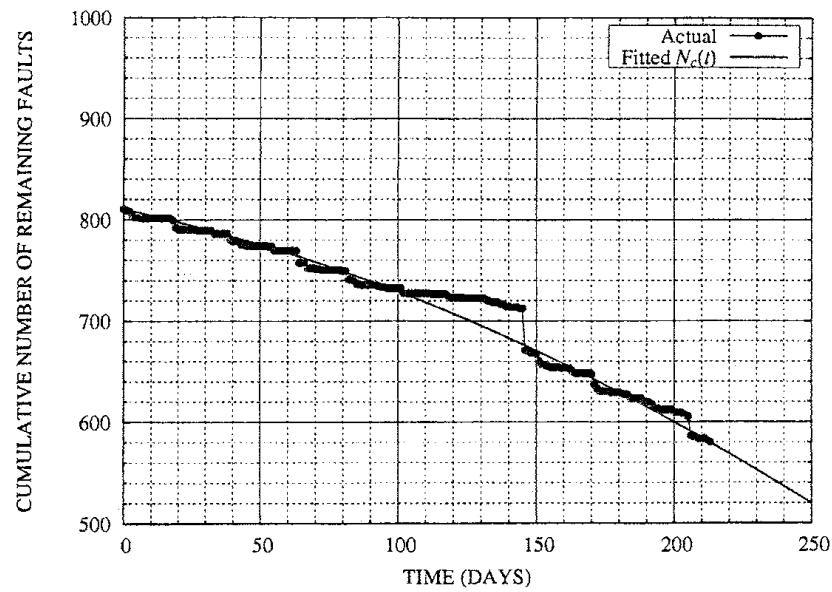


図 4: 推定された残存フォールト数の期待値, $\widehat{N}_c(t)$.

- [7] M. Uchida, Y. Tamura, and S. Yamada, "Software reliability analysis and optimal release problem based on a flexible stochastic differential equation model in distributed development environment," *Proc. 8th ISSAT International Conference on Reliability and Quality in Design*, Honolulu, Hawaii, U.S.A., pp. 12-16, August 7-9, 2003.
- [8] 加藤 豊, 小沢 正典, OR の基礎－ AHP から最適化まで－, 実教出版株式会社, 東京, 1998.
- [9] 木下 栄蔵, 入門 AHP, 日科技連出版社, 東京, 2000.