社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－ＩＴＳ－28 (5)
2007／3／16

# オープンソースソフトウェアに対する故障強度を考慮した信頼性評価法

福永康裕† 田村慶信†† 山田茂††† 鷲野翔一††††

†鳥取環境大学大学院 環境情報学部研究科 E-mail: w053004n@kankyo-u.ac.jp

††広島工業大学情報学部 E-mail: tam@cc.it-hiroshima.ac.jp

†††鳥取大学工学部 E-mail: yamada@sse.tottori-u.ac.jp

††††鳥取環境大学 環境情報学部 E-mail: washino@kankyo-u.ac.jp

近年，ネットワーク環境を利用して開発されたオープンソースソフトウェア(open source software，以下OSS と略す)は，世界中の誰もが開発に参加でき，ソースコードが公開され，誰でも自由に改変することが可能なソフトウェアであることから，組込みシステムやサーバ用途として広く採用されている．

本論文では，従来から提案されている確率微分方程式に基づく信頼性評価法を改良し，OSS への適用可能性について考察する．

# A Reliability Assessment Method Considering Failure Intensity for Open Source Software

Yasuhiro Fukunaga† Yoshinobu Tamura†† Shigeru Yamada††† Shouiti Washino††††

†Graduate School of Environmental and Information Studies, Tottori University of Environmental Studiesn E-mail: w053004n@kankyo-u.ac.jp

††Faculty of Applied Information Science, Hiroshima Institute of Technology E-mail: tam@cc.it-hiroshima.ac.jp

†††Faculty of Engineering, Tottori University E-mail: yamada@sse.tottori-u.ac.jp

††††Faculty of Environmental and Information Studies, Tottori University of Environmental Studies E-mail: washino@kankyo-u.ac.jp

Recent software development environment has been changing into new development paradigms such as concurrent distributed development environment and so-called open source project by using network computing technologies.

In this paper, we focus on the reliability assessment method based on stochastic differential equation. Moreover, we analyze actual software fault count data to show numerical examples of software reliability assessment for the open source project. Also, we compare our software reliability assessment method with several conventional software reliability assessment methods in terms of goodness-of-fit for actual data.

## 1. はじめに

現在，ソフトウェアは複雑化・大規模化し，信頼性評価が重要な課題となっている．今までに，多くの高信頼性ソフトウェアの実現とその信頼性評価に関する研究が行われてきた．ソフトウェアの信頼性評価は，故障の発生現象を不確定事象としてとらえ確率・統計論的に取り扱う方法がとられている[1]．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．中でも，非同次ポアソン過程(nonhomogeneous Poisson process，以下NHPPと略す)モデルは単純性，利便性，総合的観点から，実際のソフトウェア信頼性評価に広く応用されてきた[1]．

従来のソフトウェア開発はホスト集中型の開発環境であった．しかしながら，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[2-5]．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[6]．特に，ネットワーク環境を利用して開発されたオープンソースソフトウェア[7] (open source software，以下OSS と略す) は，世界中の誰もが開発に参加でき，ソースコードが公開され，誰でも自由に改変することが可能なソフトウェアであることから，組込みシステムやサーバ用途として広く採用されている．一方，OSS の利用に関しては，未だに多くの不安が残されている．特に，システム導入後のサポートおよび品質上の問題といった利用者側の一般的な不安である．特に，サポートや品質上の問題については，OSS の普及を妨げる大きな要因として考えられている．

今までに，同一企業内の分散開発環境で開発されたソフトウェアやホスト集中型の開発環境で開発されたソフトウェアなど，一般的な企業組織で開発されているソフトウェアを対象にした信頼性評価モデルは数多く提案されていた．中でも，代表的な指数形ソフトウェア信頼度成長モデル[8]や遅延S字形ソフトウェア信頼度成長モデル[9]は，実際のソフトウェア信頼性評価に多く適用されてきた．これらのモデルは発見フォールト数が有限であると仮定していたが，機能が頻繁に拡張されるという特徴を持つOSSの適用には非現実的であると考えられる．OSS開発の最も大きな特徴は，フォールトの修正と機能の改変・追加が頻繁に行われ，内部構造が大きく変化することが予想される．これに伴い，機能が改変・追加が行われるたびに，新たなフォールトが作りこまれることが考えられる．さらに，OSSでは機能が次々と拡張され続けることにより，コンポーネントの数もそれに応じて多くなるため，コンポーネント間の相互作用が一般の企業で開発されているようなソフトウェアよりも大きくなると考えられる．したがっ

て，OSSの信頼性を評価する場合，コンポーネント間の相互作用を考慮することが重要であると考える．これまでに，コンポーネント間の相互作用を考慮した信頼性評価法として，AHP，または，ニューラルネットワークで各コンポーネントの重要度を算出することにより，コンポーネントの相互作用を考慮した信頼性評価法が提案されている[10]．しかしながら，AHPまたは，ニューラルネットワークを用いて各コンポーネントの重要度を推定する場合，規模の小さなコンポーネントでは，発見フォールト数が少ないため正確にコンポーネントの重要度を推定することが困難である．なお，AHPとは，Analytic Hierarchy Processの略で，意思決定手法の一つであり，人間の意思決定問題を定量化するものである[11]．

本論文では，従来から提案されている動的解析に基づく信頼性評価法に改良し，そのモデルに対してOSSへの適用可能性について考察する．具体的には，確率微分方程式に基づく信頼性評価法[12]に対して改良を加える．

従来から提案されている確率微分方程式に基づく信頼性評価法は，ソフトウェア内に残存するフォールトが有限であり，フォールト発見率が常に不規則な状態であると仮定してしている．しかしながら，機能が次々と追加されるという特徴をもつOSSにおいて，フォールトが有限であると仮定するのは非現実的であると考えられる．

本論文では，残存フォールト数が無限個存在し，故障強度が時間とともに変化し，常に不規則な状態であると仮定した信頼性評価法に改良する[1]．さらに本論文で改良した信頼性評価法を用いて，実際のOSSの信頼性評価を行い，ソフトウェア信頼度，瞬間MTBF（mean time between failures，以下MTBFと略す），累積MTBFなどの信頼性評価尺度を導出する．さらに，従来から提案されているSRGMとの適合性比較を行う．

## 2. 確率微分方程式に基づく信頼性評価

時刻 $t=0$ でOSSがリリースされ，任意の時刻 $t$ におけるソフトウェア内の発見フォールト数を $\{S(t), t\geq 0\}$ とする．$S(t)$ の時間変化率は，$dS(t)/dt$ は，ソフトウェア内の発見フォールト数の影響を受けながら，時々刻々と変化すると考える．このような性質から，以下のような常微分方程式によって記述されるものと仮定する．

$$\frac{dS(t)}{dt}=\lambda(t)S(t), \qquad (1)$$

ここで，$\lambda(t)$ は時刻 $t$ におけるソフトウェア故障強度を表す．

OSSのフォールト発見は，ユーザからのフォールト報告によって行なわれることが多い．OSSのフォールト報告では，フォールトが発見されると同時にバグトラッキングシステムにフォールト情報が登録されるというわけではなく，それぞれのユーザのフォールト情報の登録にはばらつきがある場合が多い．このように，バグトラッキングシステム上へのフォールトの登録を考えた場合，OSSのフォールト発見現象は，常に不規則な状態であると考えられる．こうした不規則性を，標準化されたGauss型白色雑音によって近似的に表現す

ることを考える．さらに，OSSは常にフォールト修正やメジャーバージョンアップが繰り返されていることから，OSSの内部構造もそれに応じて変化する．すなわち，時間の変化とともに故障強度が変動するものと考える．

上記のことから，故障強度 $\lambda(t)$ に不規則性を導入すると，式(1)は，

$$\frac{dS(t)}{dt}=\{\lambda(t)+\sigma\gamma(t)\}S(t), \quad (2)$$

となる．ここで，$\sigma(>0)$ は定数パラメータであり，$\gamma(t)$ は解過程のMarkov 性を保証するために標準化されたGauss 型白色雑音である．さらに，$\lambda(t)$ は時刻 $t$ における故障強度関数を表す．式(2) を以下のIto 型の確率微分方程式に拡張して考えると

$$dS(t)=\{\lambda(t)+\frac{1}{2}\sigma^2\}S(t)dt+\sigma S(t)dw(t), \quad (3)$$

となる．式(3)の確率微分方程式を初期条件 $S(0)=v$ の下で Ito の公式を用いて変換すると，

$$S(t)=v\cdot\exp(\int_0^t\lambda(s)ds+\sigma W(t)), \quad (4)$$

となる．ここで，$v$ は以前のバージョンまでに発見されたフォールト数を表す．式(4)で定義された $w(t)$ の性質は，次の通りである．

(1) $w(t)$ はGauss 過程である．

(2) $w(t)$ の平均および分散は，それぞれ

$$E[w(t)]=0,\ \mathrm{var}[W(t)]=\sigma^2t,$$

により与えられる．

(3) $w(t)$ は定常独立増分をもつ．

(4) $\Pr[w(0)=0]=1$

本論文では，故障強度関数を以下のように定義する．

$$\int_0^t\lambda(t)ds=(1-\exp[-\sigma t]), \quad (5)$$

ここで，$\sigma$ は故障強度の加速係数を表す．式(4)から

$$\lim_{t\to\infty}S(t)=\infty, \quad (6)$$

となり，時刻 $t$ を∞と考えた場合，発見フォールト数は無限となっていることが分かる．OSSについて考えた場合，人気のあるOSSは，フォールトの修正，機能の改変・拡張などが頻繁に行われている．したがって，フォールトの修正やバージョンアップが行われ，新しいフォールトが作り込まれることが考えられる．よって，OSSの場合，発見されるフォールトが有限であると仮定するより，無限であると仮定した方が現実的であると考えられる．

なお，式(4)のモデルに含まれる未知パラメータ α 及び σ は最尤法を用いて推定することができる．

## 3. ソフトウェア信頼性評価尺度

式(4)から，ソフトウェア信頼性評価尺度として総期待発見フォールト数，瞬間MTBF，累積MTBFを導出する．

### 3.1 総期待発見フォールト数

任意の時刻 $t$ における発見フォールト数の期待値 $E[S(t)]$ は，Wiener過程 $W(t)$ の密度関数が

$$f(W(t))=\frac{1}{\sqrt{2\pi t}}\exp\{-\frac{W(t)^2}{2t}\}, \quad (7)$$

であることから,

$$\mathrm{E}[S(t)]=v\cdot\exp(\int_0^t\lambda(s)ds+\frac{\sigma^2}{2}t), \quad (8)$$

となる.

### 3.2 MTBF

ソフトウェアが故障を起こさない状況で動作し続けることができる時間を平均化した尺度である. したがって, MTBFが大きな値を取れば, それだけソフトウェア故障が起きにくくなり, ソフトウェア信頼性が向上したと判断できることになる. 瞬間MTBF は, 任意の時刻 $t$ における瞬間的なフォールト発見間隔の平均を意味する. 累積MTBFは, リリース時点から考えたときの発見フォールト1個当りに要する発見時間の平均を意味する.

**・瞬間MTBF**

瞬間MTBF は,

$$MTBF_I(t)=\mathrm{E}[\frac{1}{dS(t)/dt}], \quad (9)$$

により表すことができる. ここでは, 計算の簡略化のために,

$$MTBF_I(t)=\frac{dt}{E[dS(t)]}, \quad (10)$$

で近似的に計算すると, 瞬間MTBFは

$$MTBF_I(t)=1/v\,(\lambda(t)+\frac{1}{2}\sigma)\cdot\exp(\int_0^t\lambda(s)ds+\frac{\sigma^2}{2}t), \quad (11)$$

となる.

**・累積MTBF**

累積MTBFは

$$MTBF_C(t)=\mathrm{E}[\frac{t}{S(t)}], \quad (12)$$

により表すことができる. 計算の簡略化のために,

$$MTBF_C(t)=\frac{t}{\mathrm{E}[S(t)]}, \quad (13)$$

で, 近似的に計算する. よって, 累積MTBFは

$$MTBF_C(t)=t/v\,(\lambda(t)-\frac{1}{2}\sigma)\cdot\exp(\int_0^t\lambda(s)ds+\frac{\sigma^2}{2}t), \quad (14)$$

となる.

## 4. 数値例

本論文では, 改良した信頼性評価法を用いて実際のOSSの信頼性評価を行い, ソフトウェア信頼度, 瞬間MTBF, 累積MTBFなどの信頼性評価尺度を導出する. さらに, 従来から提案されているSRGMと適合性比較も行う.

### 4.1 確率微分方程式に基づく信頼性評価法

確率微分方程式に基づく信頼性評価法でThunderbird[13]の信頼性評価を行なった. 本論文では, Thunderbird 0.9までの結果に基づいてThunderbird 1.0以降の信頼性評価結果を行なった. 確率微分方程式に基づく信頼性評価法により推定された式(4)における累積フォールト数のサンプルパスを図1に示す. 図1から, 時間の経過とともに, 故障強度が不規則に変動する様子が確認できる. 式(8)の累積フォールト数のサンプルパスの推定値の期待値を図2に示す. さらに, 瞬間MTBFを図3, 累積MTBFを図4に示す.

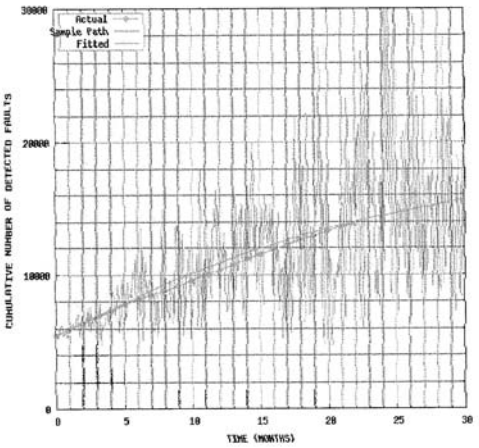


図1　累積フォールト数のサンプルパスの推定値

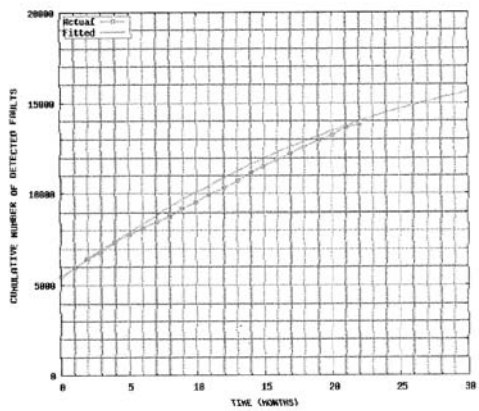


図2　総期待発見フォールト数

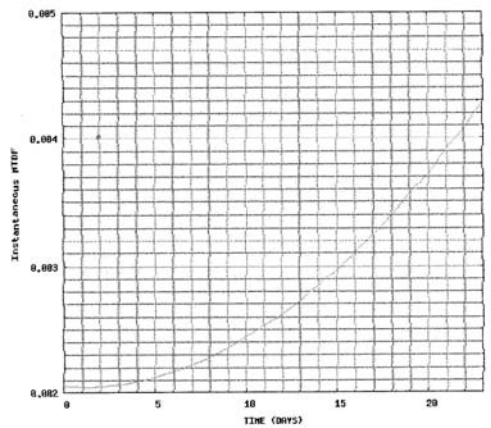


図3　推定された瞬間MTBF

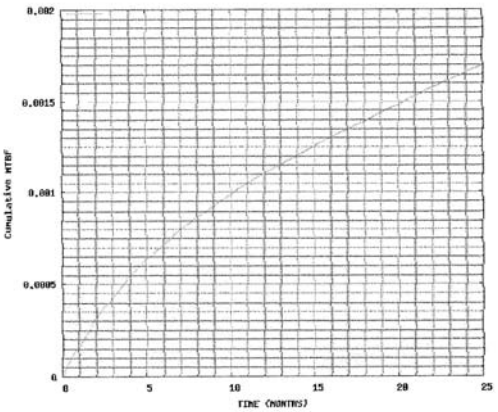


図4　推定された累積MTBF

図2の総期待発見フォールト数の推定値より，テスト開始から30ヶ月目では，約16,000個のフォールトが発見されることが分かる．さらに，図3および図4の結果から，瞬間MTBFおよび累積MTBFは時間が過ぎるにつれて，少しずつ大きくなることが確認できる．

## 4.2　適合性評価

確率微分方程式に基づく信頼性評価法と，従来から提案されている信頼性評価法に対する適合性比較を行った．適合性を比較するために，ThunderbirdのBugzillaに登録されているフォールト発見数データを使用した．本論文では，適合性比較を行なうために予測相対誤差を用いた．予測相対誤差は，任意の時刻 $t_k$ において推定したときの，フォールト報告終了時刻 $t_K$ までに発見される累積発見フォールト数の予測値と実測値との相対誤差である[1]．従来から提案されている信頼性評価法として指数形ソフトウェア信頼度成長モデル，遅延S字形ソフトウェア信頼度成長モデル，対数型ポアソン実行時間モデルを用いた．また，従来から提案されているOSSに対

する信頼性評価法として，ニュ ーラルネットワークを用いた信頼性評価法を適用した．Thunderbird の適合性比較の例を図5に示す．

図5の NN は，ニューラルネットワークを用いた信頼性評価法，SDE は確率微分方程式に基づく信頼性評価法，EXP は，指数形ソフトウェア信頼度成長モデル，Ds は遅延S字形ソフトウェア信頼度成長モデル，IS は習熟S字形ソフトウェア信頼度成長モデル，L は対数型ポアソン実行時間モデルを表している．

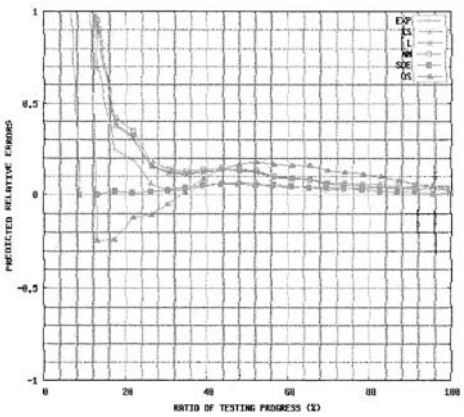


図5　Thunderbird を用いた適合性比較の例

図5より，確率微分方程式に基づく信頼性評価法は，進捗率約10%で推定値が安定していることが確認できる．その他のモデルは進捗率約30%で推定値が安定していると確認できる．これにより，従来の信頼性評価法と比較して実際の OSS に対して，本手法は従来の信頼性評価法よりも適合性が高いことが確認できる．

## 5. おわりに

本論文では，従来から提案されている信頼性評価法に改良し，それらのモデルに対して OSS への適用可能性について考察した．また，それらの信頼性評価法を適用した実際の OSS に対する信頼性評価を行った．さらに，従来から提案されているソフトウェア信頼性評価法と適合性比較も行った．適合性比較の結果から，OSS の信頼性評価について考えた場合，確率微分方程式に基づく信頼性評価法が従来から提案されている信頼性評価法よりも適合性が高いという結果を得た．

確率微分方程式に基づく SRGM で OSS の信頼性評価を行う場合，最低1回以上のバージョンアップを行わなければならないといった制約がある．しかし，OSS を運用し始めた初期段階は活動状況が活発でないことから，発見されるフォールト数も極めて少ないため，従来から提案されているモデルでも OSS を運用し始めた初期段階では信頼性評価を行うことは困難である．

近年，OSS が注目されている．しかしながら，OSS の普及を妨げる大きな要因として，サポートや品質上の問題が挙げられている．したがって，OSS の信頼性に関するなんらかの指標を提示することが必要であると考える．OSS は活動状況が活発になるに従い，大規模な機能改変を行うメジャーバージョンアップやフォールトの修正を行うマイナーバージョンアップが頻繁に行われるようになる．それに伴い発見されるフォールト数も多くなることから，OSS の信頼性がある程度向上した段階でメジャ

ーバージョンアップを行うことにより，ソフトウェア故障の原因解析やデバッグ作業の負担の軽減され，本論文で提案された信頼性評価法により，OSSの品質に関する可視化された尺度を提供することが可能となるものと考える．

## 謝辞

本研究の一部は，文部科学省科学研究費基盤研究(C)（課題番号18510124）および若手研究(B)（課題番号17700039）の援助を受けたことを付記する．

## 参考文献

[1] 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.

[2] L. T. Vaughn："Client/Server System Design and Implementation", McGraw-Hill, New York, 1994.

[3] 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).

[4] R. S. *Pressman*, *Software* Engineering:"A Practitioner's Approach (4th ed.)", McGraw-Hill, New York, 1982.

[5] A. Umar:"Distributed Computing and Client-Server Systems", Prentice Hall, New Jersey, 1993.

[6] 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.

[7] The Open Source Initiative: http://www.opensource.org

[8] A.L. Goel and K. Okumoto, "Time-dependent errordetection rate model for software reliability and other performance measures," IEEE Trans. Reliab., vol.R-28, no.3, pp.206-211, Aug. 1979.

[9] S. Yamada, M. Ohba, and S. Osaki, "S-shaped reliability growth modeling for software error detection," IEEE Trans. Reliab., vol.R-32, no.5, Dec. 1983.

[10] 福永康裕，田村慶信，山田茂，“OSSに対する開発者およびユーザ指向の信頼性評価法に関する一考察”，電気・情報関連学会中国支部第57回連合大会講演論文集，岡山理科大学，p. 157-158.，2006年10月21日.

[11] 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(1998).

[12] S. Yamada, M. Kimura, H. Tanaka, and S. Osaki, "Software reliability measurement and assessment with stochastic differentialequations," IEICE Trans. Fundamentals, vol. E77-A, no.1, pp. 109-116, Jan.1994.

[13] Mozilla Foundation, Mozilla Thunderbird, http://www.mozilla.org/products/thunder bird/