ソフトウェアのインクリメンタル開発における信頼性評価モデルに関する研究

椎名, 紘伸 / SHIINA, Hironobu

(出版者 / Publisher)
法政大学大学院理工学・工学研究科
(雑誌名 / Journal or Publication Title)
法政大学大学院紀要. 理工学・工学研究科編 / 法政大学大学院紀要. 理工学・工学研究科編
(巻 / Volume)
55
(開始ページ / Start Page)
1
(終了ページ / End Page)
6
(発行年 / Year)
2014-03-24
(URL)
https://doi.org/10.15002/00010355

ソフトウェアのインクリメンタル開発における信頼性評価モデルに関する研究

A STUDY ON SOFTWARE RELIABILITY ASSESSMENT MODELS FOR INCREMENTAL DEVELOPMENT ENVIRONMENT

椎名紘伸
Hironobu SHIINA
指導教員　木村光宏

法政大学大学院工学研究科システム工学専攻修士課程

In the literature of software reliability growth modeling, almost of all models have been constructed based on the so-called waterfall software development scheme. This thesis proposes a new methodology for the purpose of software reliability assessment in the agile/incremental software development processes. After the model description, we analyze some real datasets by using the proposed model. We show that our model can present good prediction results for the achieved reliability in the final phase of the incremental software development.

Key Words: Software reliability, Software metrics data, Incremental software development

1. はじめに

ソフトウェア内に潜在するフォールト数の数を高い精度で推定したいという目的のため, 従来から数多くのソフトウェア信頼性モデルが開発されてきた. 中でも, ソフトウェア開発の最終段階であるテスト工程において採取される, ソフトウェアテスト時間と累積発見フォールト数のデータを用いて, 時系列解析的にソフトウェア内の累積フォールト数などの信頼性を評価するモデル, つまりソフトウェア信頼性成長モデルは近年, よく知られたものとなりつつある. 一方ソフトウェア開発手法, 特にソフトウェア開発プロセスの改善も努力的に行われている. それらは古典的なウォーターフォール型の開発プロセスを脱却し, 開発途中での仕様の変更など, ウォーターフォール型開発プロセスでは対応が困難である要求に応えようとしているものである. それらの中でアジャイル (agile; 機敏な) 開発手法は近年注目されており, 比較的小規模なソフトウェアの開発に用いられてきている. 本研究では, アジャイル開発の 1 つとして知られる, インクリメンタル開発に着目し, そこから得られるデータに基づいて, ソフトウェアの信頼性を評価するモデルについて検討する. 特に, このプロセスは, ウォーターフォール型プロセスに比べて, 長期に渡る時系列データが得られないという特徴がある. これは時系列が短いデータが数本得られるというものになっており, 従来のソフトウェア信頼度成長モデルがうまく適用できない場合もありうる. 本研究ではこの点を回避し, また, ソフトウェア開発の上流工程から別途得られる情報を組み入れる手法について検討する [1].

2. ソフトウェア開発プロセス

(1) インクリメンタル開発

インクリメンタル開発とは, システム全体を部分に分割してそれらを段階的に開発し, コアとなる部分に対して機能拡張を行う形で全体に統合していく開発方法である (図 1 参照). 分割した部分のそれぞれをインクリメントという. ウォーターフォールモデルでは, ソフトウェアの開発工程を工程ごとに明確に区別している. そのため, 開発の進捗管理がしやすく大規模システムの開発においても, 信頼性の高いシステムを構築できるメリットがある. しかしながら, 一方でデメリットもある. ウォーターフォールモデルは, コーティングを始めるまでにすべての要求と設計が確定していることを求めている. しかし, 要求事項や詳細の設計をすべて明確にすることは簡単ではなくまた, システムへの要求事項も時々刻々と変化しているのが普通である. ウォーターフォールモデルの欠点を克服するため, 特に仕様変更への柔軟性が高いとされるインクリメンタル開発手法が提案され, 実践されて来ている [2].

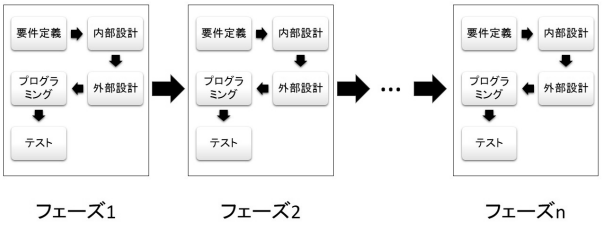

図 1　インクリメンタル開発の概要.

3. モデリング

(1) NHPP モデル

代表的なソフトウェア信頼性モデルとして NHPP モデルについて述べる [3, 4]. いま，計数過程 $\{N(t), t \geq 0\}$ を

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

$$\mathrm{E}[N(t)] = H(t), \quad \mathrm{Var}[N(t)] = H(t), \tag{2}$$

と表し，$H(t)$ が t の一次関数とはならないとき，これは非同次ポアソン過程（nonhomogeneous Poisson process）と呼ばれる．ここで，$H(t)$ は平均値関数と呼ばれる，$H(0) = 0$ となる単調増加関数であり，この過程の時間経過に対する平均的振舞いを記述する．このとき，テスト時刻 t における累積発見フォールト数を $N(t)$ と置くことにより，ソフトウェアの信頼性モデルとしてこの枠組みを採用することができる．実際のモデル化においては，観測データをよく表す平均値関数 $H(t)$ を適切に選ぶ必要があるが，いま仮に，遅延 S 字形ソフトウェア信頼度成長モデルを用いると，インクリメンタル開発におけるモデル化の例として，以下の定式化が行える．すなわち，各フェーズ i $(i = 1, 2, \cdots, I)$ の潜在フォールト数の期待値を a_i, フォールト発見率を b_i, テスト時刻を t とすると, 平均値関数 $H_i(t)$ を

$$H_i(t) = a_i[1 - (1 + b_i t)e^{-b_i t}], \tag{3}$$

とできる．未知パラメータ, a_i および b_i の推定については，本研究ではメトリクスデータに付随して得られる時系列データを用いて，最尤法や最小二乗法により a_i ,b_i を推定する．

(2) 信頼性評価尺度

NHPP の平均値を用いて与えられるソフトウェアの信頼性評価尺度について述べておく．

2.1 ソフトウェア信頼度

テストが時刻 t まで進行しているときに, 時間区間 $(t, t + x]$ においてソフトウェア故障の発生しない確率は

$$R(x|t) = \exp[-\{H(t + x) - H(t)\}], \tag{4}$$

となる．式 (4) は, ソフトウェア信頼度 (software reliability) と呼ばれる．

2.2 MTBF

時間区間 $(t, t + x]$ においてソフトウェア故障が発生する確率を求めると，それは

$$\begin{aligned} F(x|t) &= 1 - R(x|t) \\ &= 1 - \exp[-\{H(t + x) - H(t)\}], \end{aligned} \tag{5}$$

により与えられるが，平均値関数が $H(\infty) < \infty$ である場合は，$F(x|t)$ は通常の分布関数の条件 $F(\infty|t) = 1$ を満足していない. したがって，たとえば既に述べた遅延 S 字形モデルを採用する場合，ソフトウェア故障発生時間間隔の平均値は通常の MTBF の定義通りには求められない．そのため，実務的には NHPP の強度関数の逆数

$$MTBF_I(t) = \frac{1}{h(t)}, \tag{6}$$

や，テスト時間を NHPP の平均値関数で割った

$$MTBF_C(t) = \frac{t}{H(t)}, \tag{7}$$

を定義し, それぞれを平均ソフトウェア故障時間間隔あるいは平均フォールト発見時間間隔いわゆる MTBF の代替的尺度として考えることができる．これらはテスト時刻 t における瞬間 MTBF(instantaneous MTBF) および累積 MTBF(cumulative MTBF) と呼ばれている [3]. 本研究では, 実務でもよく用いられる累積 MTBF を採用し, 実測データに基づくソフトウェアの信頼性評価 (予測) について考察する.

4. メトリクスデータ

通常のソフトウェア開発では, ソフトウェア製品や開発プロセスを特徴付けるメトリクスデータとしてテストケース数や開発規模などが観測され, これらはフォールト検出事象に大きく影響を与えることが知られている．本研究の目的の一つは，第 1 フェーズから第 $i-1$ フェーズまでのメトリクスデータを用いて，最終（出荷直前）のフェーズである第 I フェーズのプログラム内に含まれたソフトウェアフォールト数を精度よく見積りたいというものである．以下に，文献 [1] に基づいて，あるソフトウェア開発プロジェクトにおいて観測されたメトリクスデータおよび時系列データの意味付けを示す.

(1) 変量の意味づけ

本研究では，既存の論文 [1] で使われたソフトウェア開発プロジェクトのデータを用いて，構築されたモデルの予測精度を評価する．以下に，各データの意味付けも合わせて列挙する. 本研究では 2 つのプロジェクトのメトリクスデータを扱ったがここでは, その内の 1 つのメトリクスデータのみを紹介する.

1.1 メトリクスデータ

各フェーズ $i(=1,2,\cdots,I)$ に対して品質に関連したメトリクス QM, 開発規模に関連したメトリクス SM が観測されている．表 1 および表 2 に本研究で分析するプロジェクトについての観測データを示す.

表 1　メトリクスデータ [1](前半).

INC	No. days	No. faults	MD	UT	RV
i	t_i	x_i	$y_{i,1}$	$y_{i,2}$	$y_{i,3}$
1	5	6	3	28	18
2	7	14	2	25	25
3	4	3	4	6	7
4	4	4	1	19	10
5	3	2	2	24	16
6	5	8	3	26	19

表 2　メトリクスデータ [1](後半).

INC	No. test cases	Size	Effort
i	$y_{i,4}$	$y_{i,5}$	$y_{i,6}$
1	1259	24.1	78
2	686	22.3	118
3	664	8.6	53
4	797	6.6	70
5	673	10.2	67
6	1422	12.8	86

ここで, 記号は以下の通りである.

i　INC
ソフトウェア開発プロジェクトにおけるフェーズ数.

t_i　No. of days
第 i フェーズにおけるテストにかかった日数.

x_i　No. of faults
第 i フェーズの最終的な累積ソフトウェアフォールト数.

$y_{i,1}$　MD: No. of modules in development (0〜100)
第 i フェーズにおいて, 開発されたモジュールの数.

$y_{i,2}$　UT: No. faults detected in unit test
第 i フェーズの結合テストにおいて検出されたソフトウェアフォールト数.

$y_{i,3}$　RV: No. of reviews (日/KLOC)
第 i フェーズの結合テストの前に行われるレビューの数.

$y_{i,4}$　No. of test cases
第 i フェーズのシステムテストで実施されたテストケース数.

$y_{i,5}$　Size
第 i フェーズの開発規模の大きさ．KLOC で表される (1 KLOC=1000 Lines of Code).

$y_{i,6}$　Effort
第 i フェーズの延べのテスターの数.

1.2 時系列データ

提供されているデータは, 上述のメトリクスデータだけではなく, 時系列データも与えられている．これについて述べる．データは各フェーズ $i(=1,2,\cdots,I)$ に対してのテスト実施日および発見フォールト数からなり，その総和がフェーズ i における x_i に一致する．表 3 および表 4 に本研究で分析する 2 つのプロジェクトについての観測データを示す.

表 3　ProjectA の時系列データ [1](前半).

	INC.1		INC.2		INC.3	
ST 期間	10/27	2	11/24	4	12/22	1
	10/28	0	11/25	1	12/23	0
	10/29	1	11/26	1	12/24	0
	10/30	2	11/27	3	12/25	2
	10/31	1	11/28	2	12/26	0
			11/29	1		
			11/30	2		

表 4　ProjectA の時系列データ [1](後半).

	INC.4		INC.5		INC.6	
ST 期間	1/28	1	2/26	0	3/22	1
	1/29	3	2/27	2	3/23	3
	1/30	0	2/28	0	3/24	1
	1/31	0			3/25	1
					3/26	2
					3/27	0

5. モデルの適用法

本研究では，以下の手順に従って，インクリメンタル開発の最終フェーズ（その番号を I で表す）における累積

MTBF を推定する方法を提案する．実際の詳細については次節に述べることとし，ここでは概略を説明しておく．

Step 1： フェーズ番号を $i=1$ とする
Step 2： 第 i フェーズでの時系列データに対して，NHPP モデルを適用する．平均値関数 $H(t)$ については，指数形あるいは遅延 S 字形を用いることとなるが，その選択については時系列データの時間変化の振舞いをグラフに描き，その大まかな形状から選択する．このような平均値関数の選択は，ウォーターフォール型開発での NHPP モデルの適用においても通常なされる方法である．その結果として，NHPP の適用により，そのフェーズの平均値関数に含まれるパラメータ，a_i および b_i が推定される．このときの推定法は，本研究での試みとして，最尤法と最小二乗法のそれぞれを試すこととする．
Step 3： $i=i+1$ として，もし $i<I-1$ ならば Step 2 に戻る．

以上により，信頼性予測の対象であるフェーズ I に対して，$I-1$ までの時系列解析が終了している．続いて以下に従う．

Step 4: ここまでに得られた (a_i,b_i) $(i=1,2,\ldots I-1)$ の推定値を用いて，以下の構造を仮定しメトリクス情報を利用して，(a_i,b_i) の値を修正した上で，更に第 I フェーズの予測に用いるためのパラメータ，(a_I,b_I) を推定する．

つまり，各フェーズで得られた (a_i,b_i) の推定値は，前者はそのフェーズでの潜在フォールトの数，後者はフォールトの発見率に対応する量である．それらはそのフェーズでの開発作業によって値の大小が影響されると考えるわけである．したがって，定数パラメータ α_0〜α_3，および β_0〜β_3 を導入して

$$\log a_i = \log\alpha_0 + \alpha_1 y_{i,k} + \alpha_2 y_{i,l} + \alpha_3 y_{i,m}, \quad (8)$$
$$\log b_i = \log\beta_0 + \beta_1 y_{i,k} + \beta_2 y_{i,l} + \beta_3 y_{i,m} \quad (i=1,2,\ldots,I-1), \quad (9)$$

の関係があると仮定する．この構造は先行研究 [1] において採用されたものを修正して用いている．式 (8) および式 (9) は重回帰モデルであり，目的変数と説明変数の候補から，もっとも当てはまりの良い説明変数 y_* を 3 つ選択する．その後，最小二乗法によってこれら重回帰モデルの未知パラメータを推定する．

Step 5： 推定された未知パラメータ，$\hat{\alpha}_0$〜$\hat{\alpha}_3$，および $\hat{\beta}_0$〜$\hat{\beta}_3$ を用いて，フェーズ 1 からフェーズ $I-1$ の平均値関数のパラメータ a_i および b_i を修正することができる．さらに，当初の目的であるフェーズ I の NHPP の平均値関数に含まれるパラメータ a_I および b_I を，フェーズ I の時系列が得られるより前に推定することが可能となる．つまり，

$$\log a_i = \log\hat{\alpha}_0 + \hat{\alpha}_1 y_{i,k} + \hat{\alpha}_2 y_{i,l} + \hat{\alpha}_3 y_{i,m}, \quad (10)$$
$$\log b_i = \log\hat{\beta}_0 + \hat{\beta}_1 y_{i,k} + \hat{\beta}_2 y_{i,l} + \hat{\beta}_3 y_{i,m} \quad (i=1,2,\ldots,I), \quad (11)$$

により，左辺の a_i，b_i はそれぞれ更新かつ推定される．

これにより，最終フェーズでの平均値関数の振舞いを規定するパラメータが予測可能となることから，第 $I-1$ フェーズが終了し，第 I フェーズのコーディング・レビュー過程が終了した時点において，つまり最終テストをする前の段階で，最終テストでどのようなフォールトの発見過程となるかを予測することができる．これは従来の単純な時系列解析モデルをそのままインクリメンタル開発に適用しただけでは不可能な，新しい推定法と言える．次が最終ステップとなる．

Step 6： a_I および b_I を用いて，フェーズ I での平均値関数 $H_I(t)$ が得られる．これを用いて，累積 MTBF を描画し，最終フェーズ I における信頼性評価・予測として用いる．

以上が本研究で開発した，インクリメンタル開発向けの信頼性予測手順である．

6. データ解析例

構築した信頼性評価モデルを実測値に当てはめモデルの評価を行う．第 1 フェーズから第 $I-1$ フェーズまでのメトリクスデータを使い最小二乗法, 最尤法により NHPP モデルの未知パラメータを推定する．その後, メトリクス情報を用いたモデルに適用し, 第 I フェーズのメトリクスデータを用いて推定を行う．さらに MTBF の推定値と MTBF の実測値との比較を行い評価を行う．ここでは, 最小二乗法, 最尤法ともに ProjectA を例に解析を行う手順を示す.

(1) 最小二乗法

はじめに, 最小二乗法を用いた信頼性評価モデルの説明をする.
Step 1：
フェーズ番号を $i=1$ とする
Step 2： a_i, b_i の推定
各フェーズ i $(i=1,2,\cdots,I)$ のフォールト数の期待値を a_i, フォールト発見率を b_i, 時刻を t $(=0,1,\cdots,T)$, 時刻

t までに発見された累積ソフトウェアフォールト数を $x_{i,t}$ とする．各フェーズごとに

$$H_i(t) = a_i[1 - (1 + b_i t)e^{-b_i t}], \tag{12}$$

$$sse_i = \sum_{t=0}^{T}(x_{i,t} - H_i(t))^2, \tag{13}$$

を設定し，最小二乗法により第 $I-1$ フェーズまでのフォールト数の期待値 a_i，フォールト発見率 b_i を推定する．ここで，a_i，b_i ともに対数をとる．a_i，b_i は表 5 のようになる．

表 5　最小二乗法により推定した $\hat{a}_i$，$\hat{b}_i$．

i	$\hat{a}_i$	$\hat{b}_i$
1	8.589	0.474
2	15.019	0.527
3	5.995	0.351
4	4.421	1.322
5	3.288	0.763

Step 3：
$i = i + 1$ として，もし $i < I - 1$ ならば Step 2 に戻る．

Step 4： **未知パラメータの推定**
すべてのメトリクスデータを使うのではなく，使用する変数の選択をする．選択方法として $\log \hat{a}_i$，$\log \hat{b}_i$ を目的変数とし，説明変数 $y_{i,1}$〜$y_{i,6}$ から寄与率の高い変数の組み合わせを選択する．本研究では寄与率の高い 4 組の組み合わせを使用する．寄与率の高い組み合わせは表 6，表 7 となる．

表 6　$\log \hat{a}_i$ の寄与率．

$y_{i,k}$, $y_{i,l}$, $y_{i,m}$	寄与率
$y_{i,1}$, $y_{i,2}$, $y_{i,5}$	0.992
$y_{i,2}$, $y_{i,5}$, $y_{i,6}$	0.978
$y_{i,2}$, $y_{i,4}$, $y_{i,6}$	0.961
$y_{i,1}$, $y_{i,3}$, $y_{i,6}$	0.967

表 7　$\log \hat{b}_i$ の寄与率．

$y_{i,k}$, $y_{i,l}$, $y_{i,m}$	寄与率
$y_{i,1}$, $y_{i,4}$, $y_{i,5}$	0.996
$y_{i,1}$, $y_{i,3}$, $y_{i,4}$	0.958
$y_{i,1}$, $y_{i,2}$, $y_{i,5}$	0.955
$y_{i,1}$, $y_{i,4}$, $y_{i,6}$	0.953

$\log \hat{a}_i$, $\log \hat{b}_i$ ともに寄与率の高い 4 組を
最小二乗法

$$sse_a = \sum_{i=1}^{I-1}(\log a_i - (\log \alpha_0 + \alpha_1 y_{i,k} + \alpha_2 y_{i,l} + \alpha_3 y_{i,m}))^2, \tag{14}$$

$$sse_b = \sum_{i=1}^{I-1}(\log b_i - (\log \beta_0 + \beta_1 y_{i,k} + \beta_2 y_{i,l} + \beta_3 y_{i,m}))^2, \tag{15}$$

により，定数パラメータ (α_0, α_1, α_2, α_3, β_0, β_1, β_2, β_3) を推定する．

Step 5： **第 I フェーズの $\hat{a}_I$, $\hat{b}_I$ の推定**
推定した定数パラメータを

$$\log a_i = \log \alpha_0 + \alpha_1 y_{i,k} + \alpha_2 y_{i,l} + \alpha_3 y_{i,m}, \tag{16}$$

$$\log b_i = \log \beta_0 + \beta_1 y_{i,k} + \beta_2 y_{i,l} + \beta_3 y_{i,m}, \tag{17}$$

にあてはめ第 I フェーズのフォールト総数の期待値 $\hat{a}_i$，フォールト発見率 $\hat{b}_i$ を求める．推定した定数パラメータ，第 I フェーズのフォールト総数の期待値 $\hat{a}_I$ フォールト発見率 $\hat{b}_I$ は表 8 になる．

表 8　偏回帰パラメータと $\hat{a}_I$, $\hat{b}_I$.

$\hat{\alpha}_0$	0.373	$\hat{\beta}_0$	1.485
$\hat{\alpha}_1$	0.308	$\hat{\beta}_1$	-0.315
$\hat{\alpha}_2$	-0.048	$\hat{\beta}_2$	0.016
$\hat{\alpha}_3$	0.037	$\hat{\beta}_3$	-0.032
$\hat{a}_I$	8.962	$\hat{b}_I$	0.587

Step 6： **MTBF 実測値との比較**
第 I フェーズの累積 MTBF の推定を行う．これに対応する実測値は $\frac{t}{x}$ で与えられる．第 I フェーズのフォールト総数の期待値 $\hat{a}_I$, フォールト発見率 $\hat{b}_I$ を
指数形 MTBF

$$MTBF = \frac{t}{a_I(1 - e^{-b_I t})}, \tag{18}$$

または
遅延 S 字形 MTBF

$$MTBF = \frac{t}{a_I[1 - (1 + b_I t)e^{-b_I t}]}, \tag{19}$$

に代入し推定し，下側 95% 信頼区間を求める．この信頼区間はポアソン分布に基づいて導出することができる．

例えば, 遅延 S 字形で第 I フェーズの実測値と推定値を比較すると図 2 のようになる．図より，第 I フェーズの時系列データが得られるより前に，MTBF の推定が概ね行えていることが見て取れる．

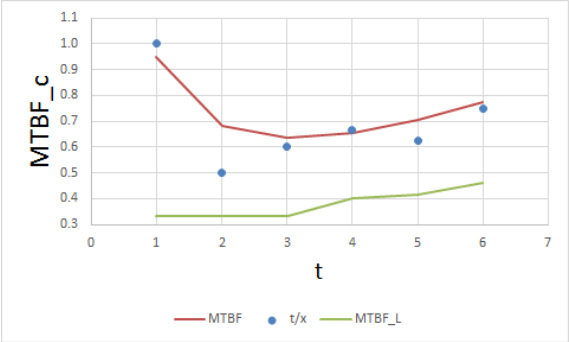

図 2　最小二乗法による遅延 S 字形信頼度成長モデルに基づく予測結果.

最尤法を用いた場合についても同様に推定でき，時系列解析の結果を表 9 に示す．

表 9　最尤法により推定した $\hat{a}_i$, $\hat{b}_i$.

i	$\hat{a}_i$	$\hat{b}_i$
1	6.853	0.723
2	17.050	0.522
3	4.674	0.685
4	4.289	0.674
5	2.488	1.007

前述した Step 3 から Step 6 について，最小二乗法と同じ手順で推定を行い MTBF 実測値との比較をする．例えば, 遅延 S 字形モデルを用いて第 I フェーズの実測値と推定値を比較すると図 3 のようになる．

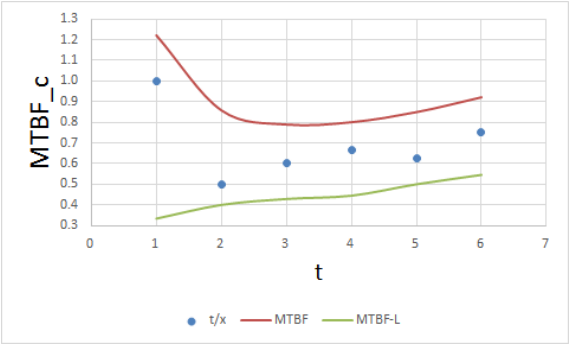

図 3　ProjectA の最尤法による遅延 S 字形モデルに基づく予測結果.

こちらの結果は，累積 MTBF は実測値よりも大きな値として評価される．

7. 結果と考察

ここでは, ProjectA に対する累積 MTBF の第 I フェーズでの実測値と推定値の当てはまりについてのみ述べる．2 つのモデル（指数形・遅延 S 字形）および 2 つの推定方法（最小二乗法・最尤法）により推定された最終フェーズでの累積 MTBF の予測値と実測値との残差平方和は表 10 のようになった．

表 10　ProjectA の各推定方法による残差平方和.

推定方法	残差平方和
最小二乗法・遅延 S 字形	0.044
最小二乗法・指数形	0.665
最尤法・遅延 S 字形	0.311
最尤法・指数形	0.524

結果として，遅延 S 字形モデルを最小二乗法で推定した場合の予測精度が最もよくなった．これは Project B についても同様である．いずれのモデルを最終フェーズにおいて用いるかについては，現在のところこの手法を用いるソフトウェア開発管理者の経験に委ねざるを得ず，これは今後の課題として残される．

8. おわりに

本研究では, ソフトウェアのインクリメンタル開発に適したソフトウェア信頼性成長モデルを考案し, 既存の論文内で使われたソフトウェアメトリクスデータを用いて, 最小二乗法, 最尤法よって構築されたモデルを複数提案した上で, 各モデルの有用性を評価した．2 つのプロジェクトに対し, 有用性を評価したが, どちらのプロジェクトも, 最小二乗法で推定したモデルのほうが精度が高い結果を得ることができた．しかしながら，データが少ないため，変数選択やパラメータ推定をうまく行うことができなかった他のプロジェクトには適用できなかった．今後の課題としては, 他の変数選択法やパラメータ推定を行うなどが挙げられる．

参考文献

[1] T. Fujii, T. Dohi, and T. Fujiwara, “Towards quantitative software reliability assessment in incremental development processes.”, *Proceedings of the 33rd International Conference on Software Engineering*, pp. 41-50. ICSE ’11, (2011).

[2] Mint（経営情報研究会），図解でわかる　ソフトウェア開発のすべて，日本実業出版 (2000).

[3] 山田茂，ソフトウェア信頼性モデル，日科技連出版社 (1994).

[4] 木村光宏，藤原隆次，ソフトウェアの信頼性，日科技連出版社 (2011).