自動車開発における ソフトウェアテスト完了判断と 市場故障発生予測のモデル に対する検証

～オーカマモデル～

2021-02-26

ソフトウェア品質管理研究会SQiP
研究コース1　ソフトウェアプロセス評価・改善　分科会

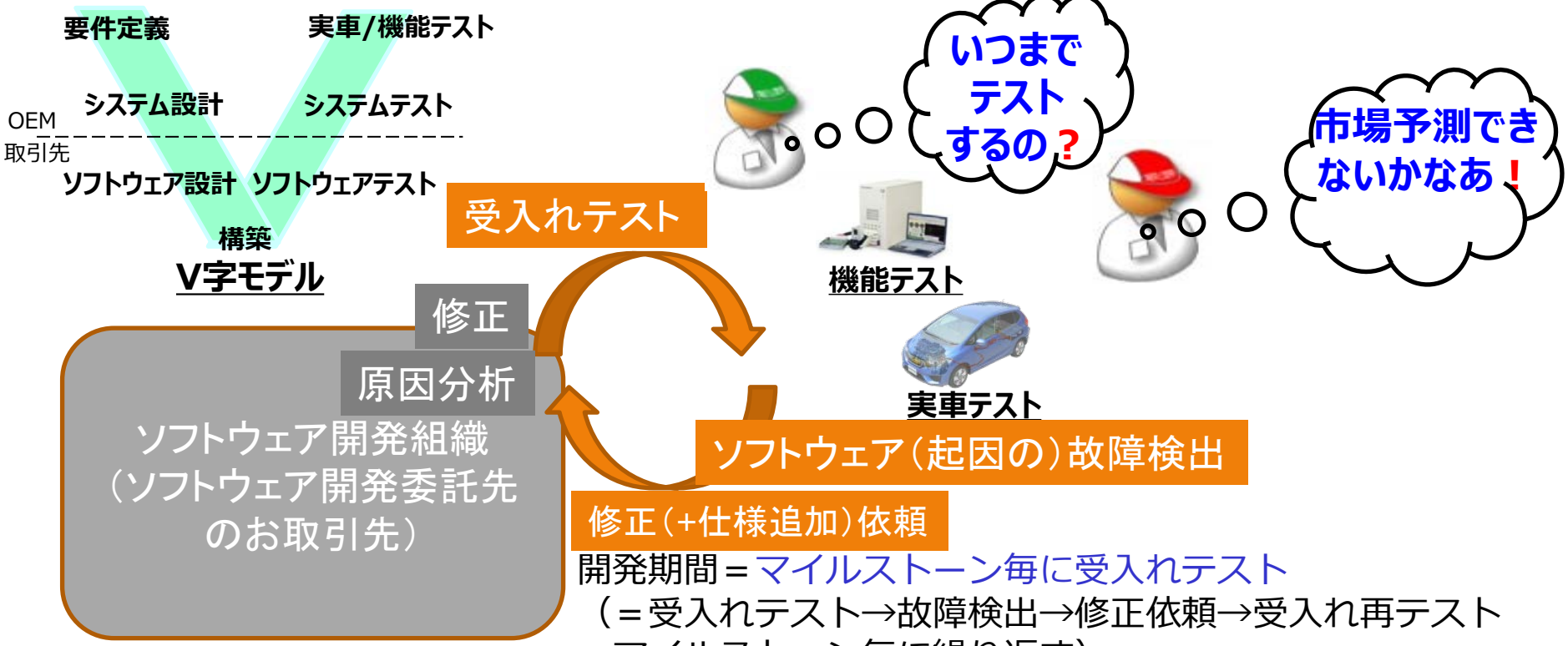
研究員：大釜俊洋（本田技研工業株式会社）

主査：山田淳（株式会社東芝）
副主査：田中桂三（オムロン株式会社）
アドバイザー：中森博晃（パナソニック株式会社）

1. 研究の背景（現状）と問題

自動車開発における、ソフトウェア開発委託取引先から受け入れてテストを実施する際に問題がある：

- ソフトウェア故障の残存発生数の予想が困難
- テスト完了の判断方法・完了判断の定量的な基準が不明確
- ソフトウェアリリース後,残存故障の市場発生予測が不十分

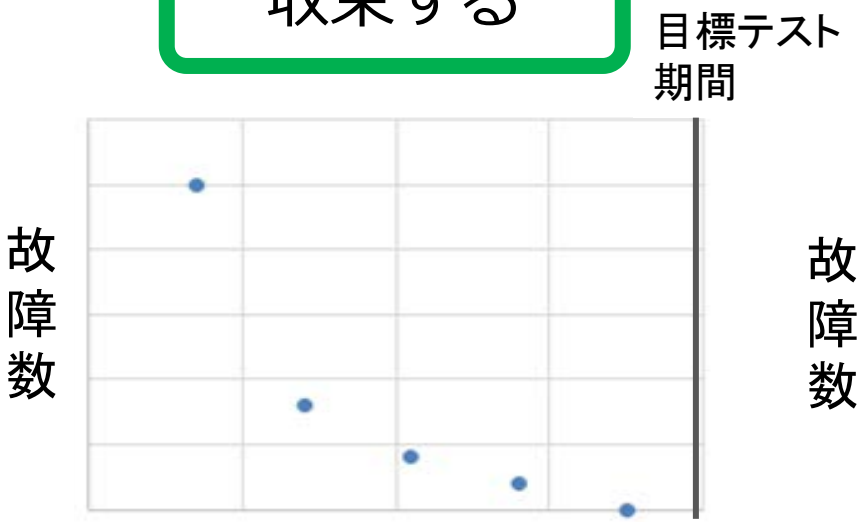


開発期間＝マイルストーン毎に受入れテスト
（＝受入れテスト→故障検出→修正依頼→受入れ再テスト
：マイルストーン毎に繰り返す）

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

テストでの故障発生の傾向

収束する



開発期間（マイルストーン毎に受入れテスト）

図1-1

①検出した故障の件数が収束して信頼度成長曲線も適用しやすい場合

収束しない!?

開発期間（マイルストーン毎に受入れテスト）

図1-2

②残存故障の検出が困難で故障件数がなかなか収束しない場合

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

2. 問題解決のアプローチ

1. テスト完了を数値で定量的に判断するために,受入れテスト中に算出される故障の発生率（k：故障率）に着目し,ソフトウェア信頼度成長モデルと組合せることで,市場での故障発生を予測しテスト完了の判断基準を明確にする.

2. テスト工程中の故障が収束していない場合,収束するための原因分析.

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

3. 提案するオーカマモデルの全体像

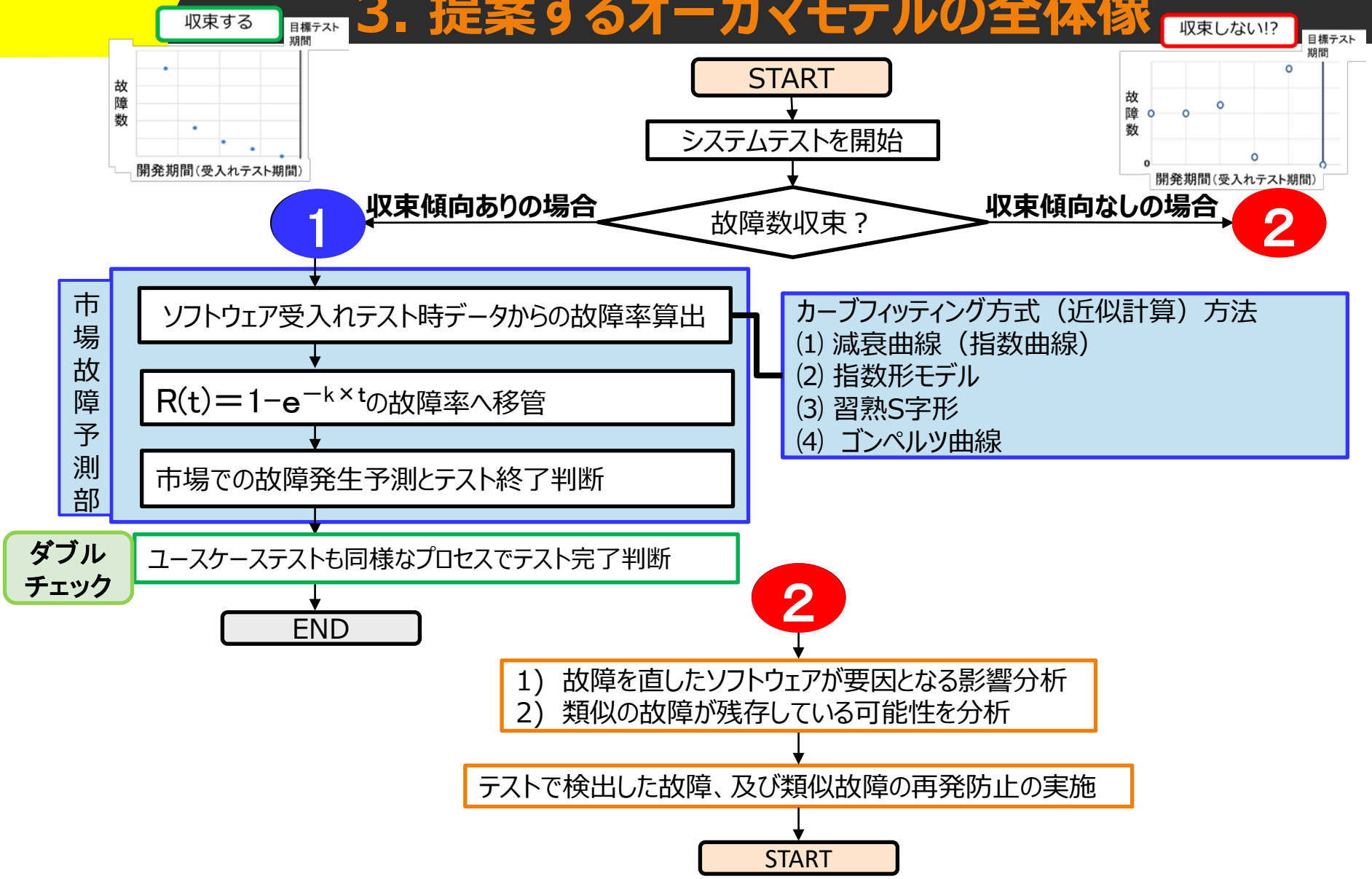
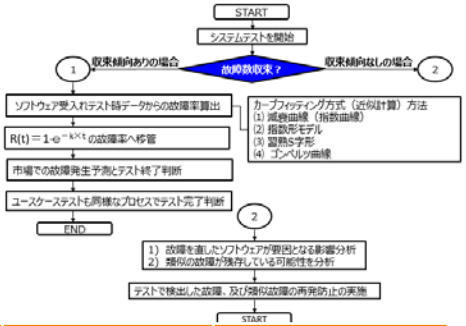


図3-1　ソフトウェアテスト完了判断と市場故障発生予測のモデルで用いる方法（オーカマモデル全体）の手順

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

3.1テスト工程の故障収束判断と故障数の経時変化をモニター（検証事例）



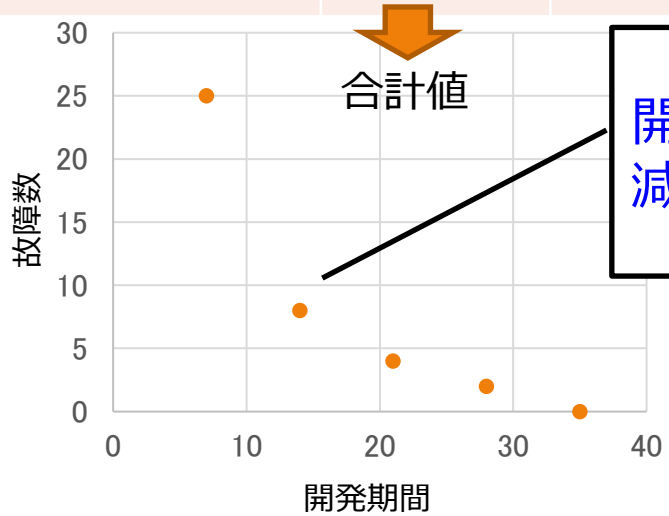
◆収束傾向？

開発中の典型的な値を検証に用いた.

開発期間	各マイルストーン期間	マイルストーン回数	テストケース	レビュー項目数	市場保証期間	市場生産台数	保証件数
35か月	7か月	5回	100	30	10年	100万台	1件未満

（要求仕様の追加仕様もマイルストーン毎には減少する）

	マイルストーン	1回目 7月	2回目 14月	3回目 21月	4回目 28月	5回目 35月
動的テスト	テスト時故障数	10	3	2	1	0
静的テスト	レビュー指摘数	15	5	2	1	0



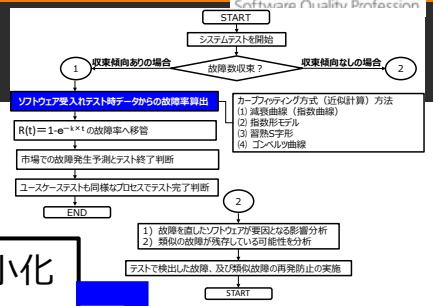
開発マイルストーン毎の故障数は、減衰曲線を描き、収束傾向にある

1

図3-2　開発マイルストーン毎の故障数

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

3.2 減衰曲線（指数関数）とソフトウェア信頼度成長モデルで故障累積数をカーブフィッティングした推定曲線から故障率を算出



◆故障率kを計算 1

1．実故障と指数減衰曲線推定（最小二乗法で補正前処理）の差分の二乗和を最小化
2．最小二乗法で補正前処理したカーブの傾き（故障率(t)）の時間変化

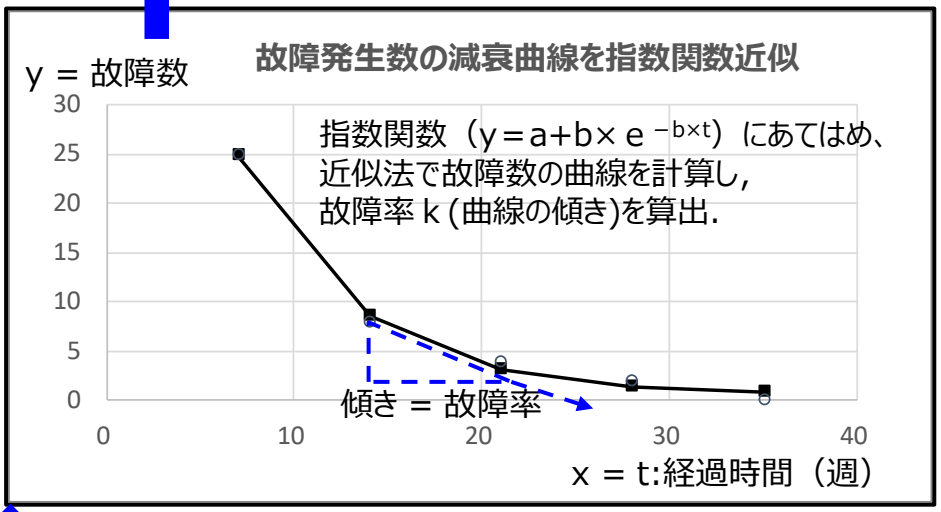


図3-3a 故障発生数の減衰傾向を指数関数の減衰曲線で近似

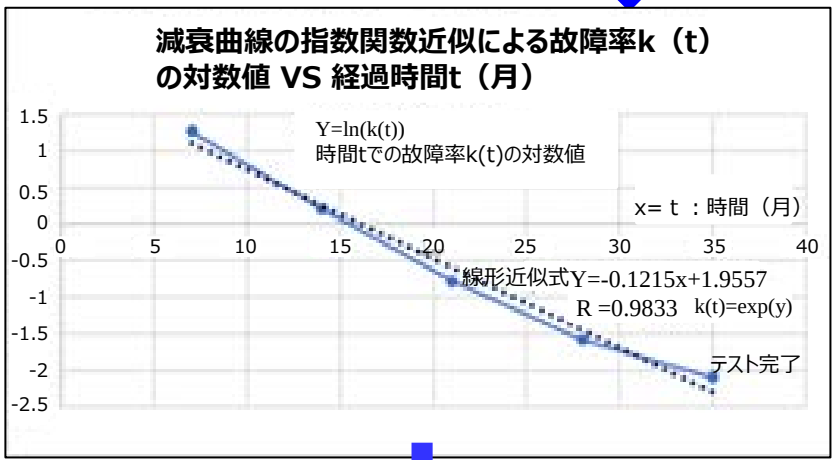


図3-3b 対数変換して線形近似

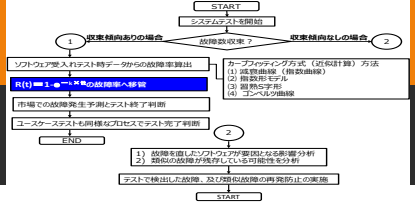
各ソフトウェア信頼度成長モデルから故障率とMTBF算出

Mean Time Between Failure =1/故障率

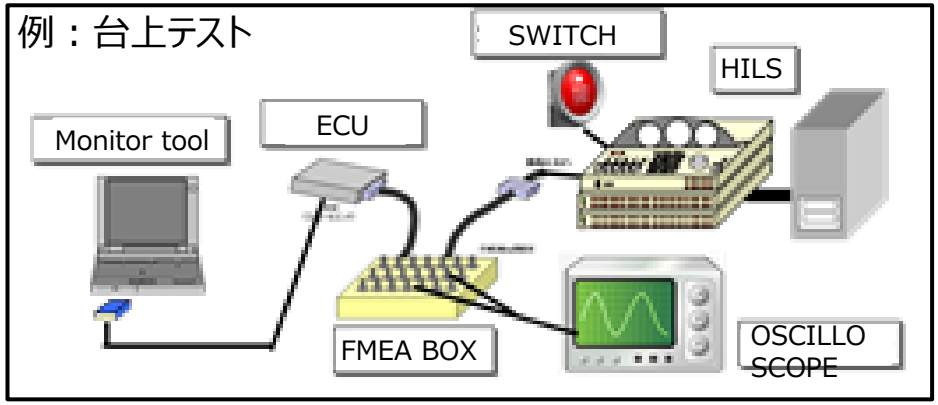
		故障率（件/月）	MTBF(月)
(1) 減衰曲線（指数曲線）	F(t)＝a+b× e ⁻ᵇˣ ᵗ	0.089(36か月)	11.2
(2),(3),(4)の故障率k(t(i))は時間t(i-1（35か月）)からt(i（36か月）)の間の単位時間の傾き			
(2) 指数形モデル	F(t)＝a×(1- e ⁻ᵇˣ ᵗ)	0.0390	25.7
(3) 習熟S字形	F(t)＝a×(1- e ⁻ᵇˣ ᵗ)/(1+c× e ⁻ᵇˣ ᵗ)	0.0390	25.7
(4) ゴンペルツ曲線	F(t)＝a×exp(-b× e ⁻ᶜˣ ᵗ)	0.00711	140.7

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

3.3 R(t)＝1- e $^{-k\times t}$ の故障率へ移管



開発期間



市場

ソフトウェア開発中の減衰曲線を想定した故障曲線

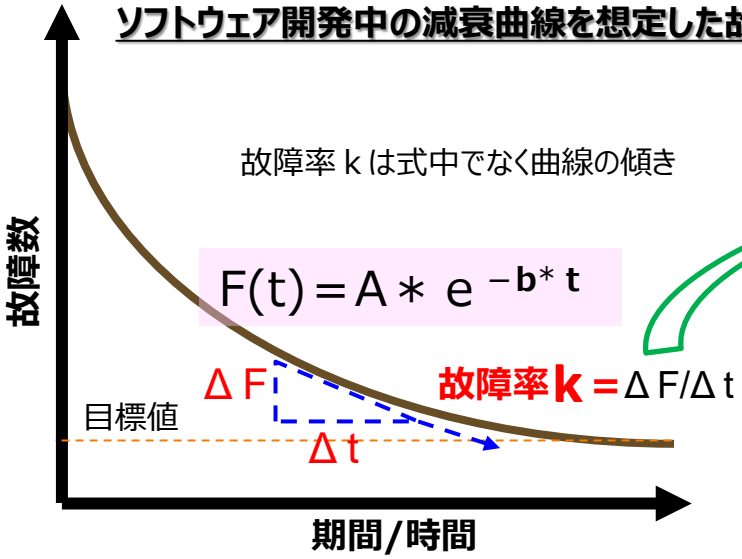
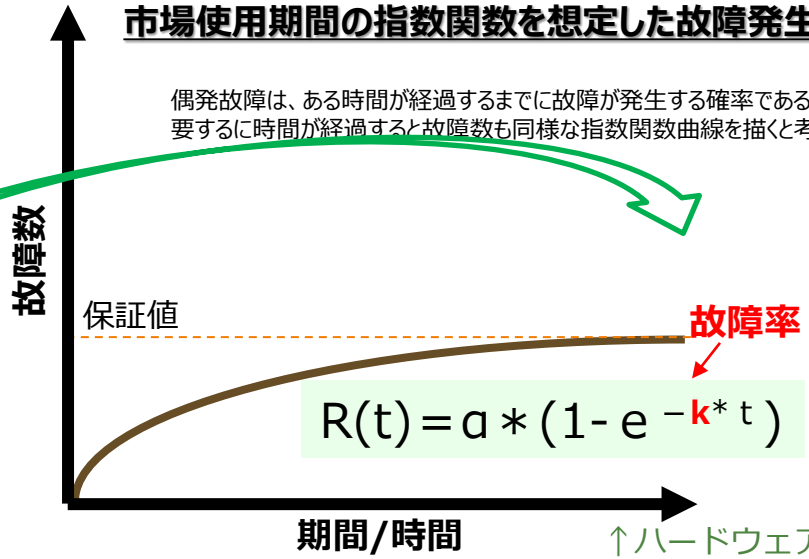


図3-4a　ソフトウェア開発期間（開発期間）

市場使用期間の指数関数を想定した故障発生曲線

偶発故障は、ある時間が経過するまでに故障が発生する確率である。
要するに時間が経過すると故障数も同様な指数関数曲線を描くと考える



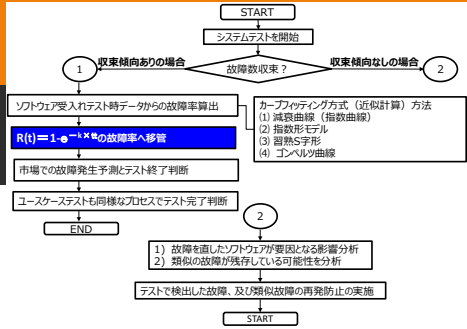
↑ハードウェア信頼性
予測計算式を応用

図3-4b　市場使用期間

市場での故障発生を予測しテスト完了の判断基準を明確にする.

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

3.4 市場での故障発生予測の計算



◆市場での故障発生予測

算出した故障率を用いて, $R(t)=\alpha\times(1-e^{-k\times t})$の故障率kに置き換え,市場での故障発生予測をする.

表3-1 算出法別の故障率k，MTBFと市場故障数予測一覧

$R(t)=\alpha\times(1-e^{-k\times t})$

	故障率k（件/月）	MTBF（月）	その時点でのMTBF＞テスト完了後の市場保証期間10年完了時(35+10 x 12=155月)となるまでの，テスト開始からの経過時間（年）	100万台中10年間の市場故障予測（台）
⑴ 減衰曲線（指数曲線）	0.089	11.2	4.6	999,977
⑵ 指数形モデル	0.0390	25.7	3.9	990,710
⑶ 習熟S字形	0.0390	25.7	3.9	990,710
⑷ ゴンペルツ曲線	0.00711	140.7	3.0	573,951

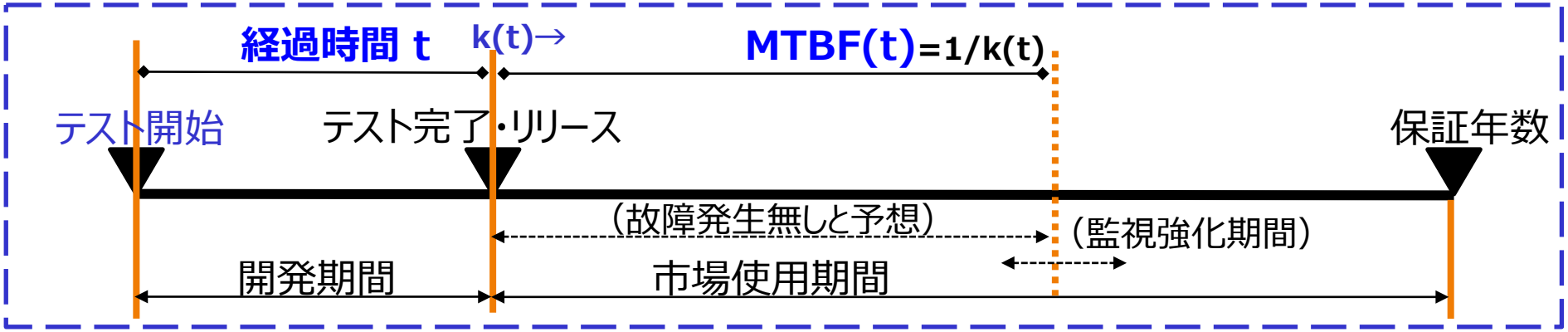
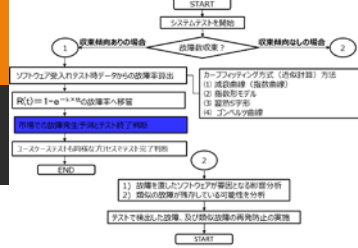


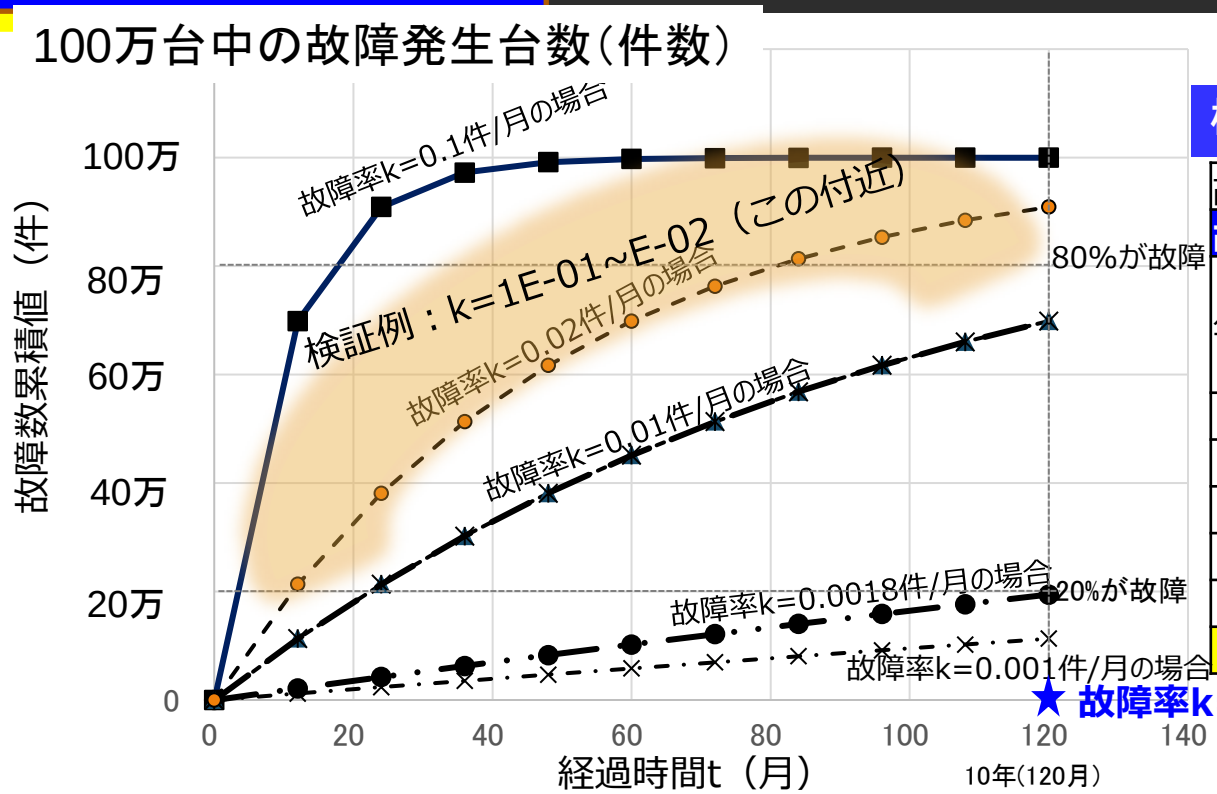
図3-5 テスト完了直後のk(t)，MTBF(t)と保証年数

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

3.4 市場での故障発生予測の計算



◆市場での故障発生予測



検証例：故障率k=1E-01~E-02

Gap⇒リスク

故障率k(件/月)

故障率k=0.000008（8 E-06）

年数	故障なし台数（無故障台数）$=a\times e^{-k\times t}$	故障台数比率(%)
0	1000000	0.00
2	999808	0.02
4	999616	0.04
6	999424	0.06
8	999232	0.08
10	999040	0.10

★ **故障率k=1E-09の場合は保証期間に故障なし**

図3-6 故障率k（件/月）の違いによる市場使用期間中の故障数の累積値予測（故障あり台数$R(t)=a(1-e^{-k\times t})$）

- ■故障率k=0.000008(8E-06)件/月以下程度なら保証値0.1%以下に故障数を収められる.
- ■故障率k=1E-09件/月以下程度なら保証値0%以下に故障数を収められる.

理想的な故障率kとのGapから，市場リスクが判断できると考える

ソフトウェア故障は本来台数とは無関係だが，十分にテストされたソフトウェアでも，市場で長時間使用される環境下では，テストケースで網羅されていない条件が揃う可能性が増しソフトウェア故障が確率的に発生する場合がある.このため$R(t)=a\times(1-e^{-k\times t})$による市場での故障累積数予測と設定保証値との比較は，リスクを表現する手段の一つとして考えるのがよい.

3.5 考察：　市場故障予測に関する考察

１）故障率の算出

指数減衰曲線推定（最小二乗法で補正前処理）

最小二乗法で補正前処理したカーブの傾き（故障率(t)）の時間変化

減衰曲線の指数関数近似を適用
$y = \ln(k(t))$

故障率をより的確に算出できる可能性あり

図3-7　故障率の算出（指数減衰曲線推定の場合）

⇒　複数の算出法（指数減衰曲線+信頼度成長モデル）を組合せて一覧し，比較又は選択して用いるのが良いと考える.

２）市場故障予測の保証値(＝1件)未満には達しなかった.

⇒　故障率のGapから定量的な市場リスク指標　⇒　テスト完了判断材料

⇒　故障台数累積予測式$R(t)=a\times(1-e^{-k\times t})$によるソフトウェアの市場故障数予測値は，市場リスクを表現する手段の一つとして考えるのがよい

さらに下記の対策が考えられる.

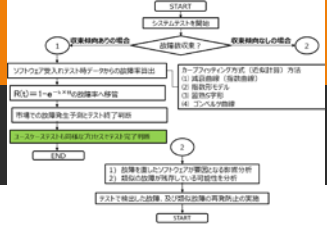
(a)保証値を達成する故障率になるまで**継続してテストを実施**し,検出故障を確実に対策する

(b)テスト終了後に**MTBF**の期間毎に市場監視し，テスト長期化を避ける

(c)ソフトウェア開発組織内の**データを蓄積,活用**することで推定精度を上げる

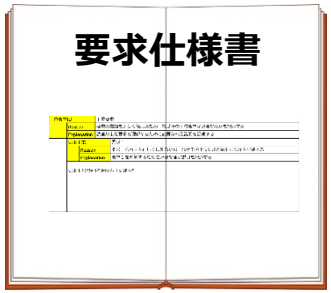
Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

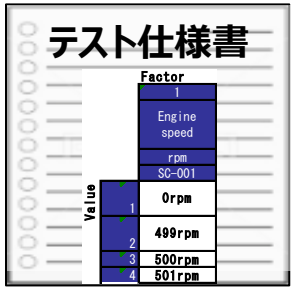
3.6 市場使われ勝手を想定したユースケーステスト



◆第2段階のテスト

ダブル
チェック





前項までのテスト

＋

ユースケーステスト

図3-8　機能・動作検証中心のテストに加えてユースケーステストも実施し故障発生予測を適用

ユースケーステストにおける故障数を図3-９に示す．
*1ユースケース故障数曲線から減衰曲線の指数関数近似の適用の可能性を示唆しており，
$R(t)=a*(1-e^{-k*t})$　を用いることで，
市場故障発生数予測の可能性に期待できる．

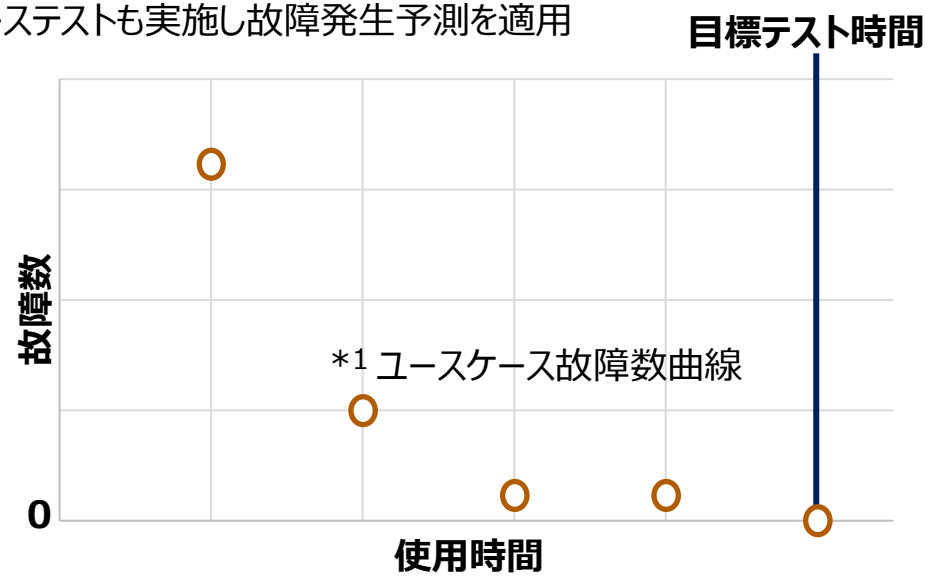


図3-９　実車Aを用いたユースケーステスト結果

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

3.7 故障数が収束していないケース

◆収束傾向なしの場合 2

テストを実施し故障数が収束していないケースを図３-10に示す

◦**ソフトウェアに欠陥が残存している可能性が高い**
→ソフトウェア故障発生の可能性も高い

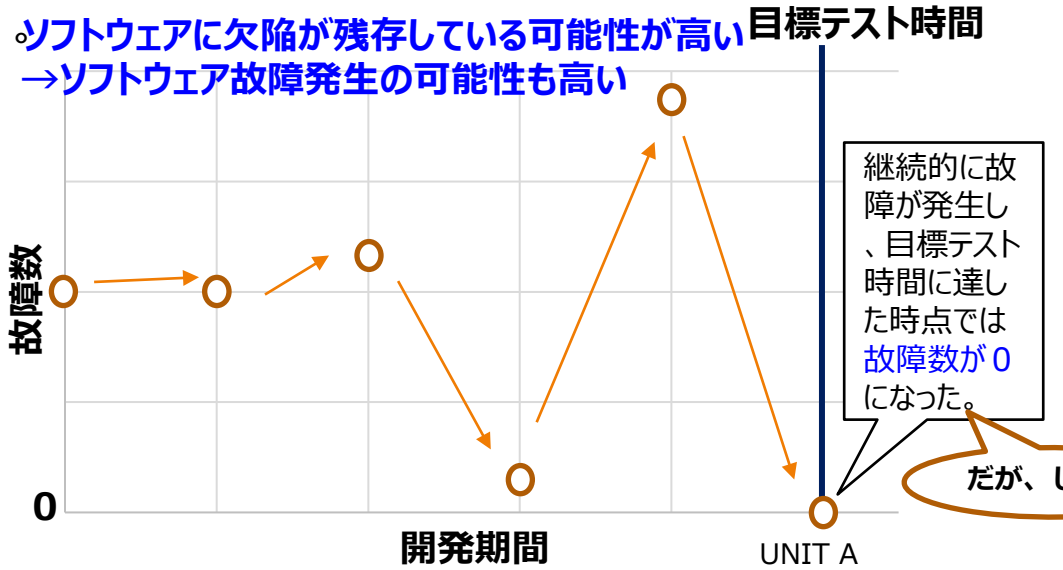


図3-10　UNIT A　故障数曲線

UNIT A
ROM使用量：796K
RAM使用量：35K

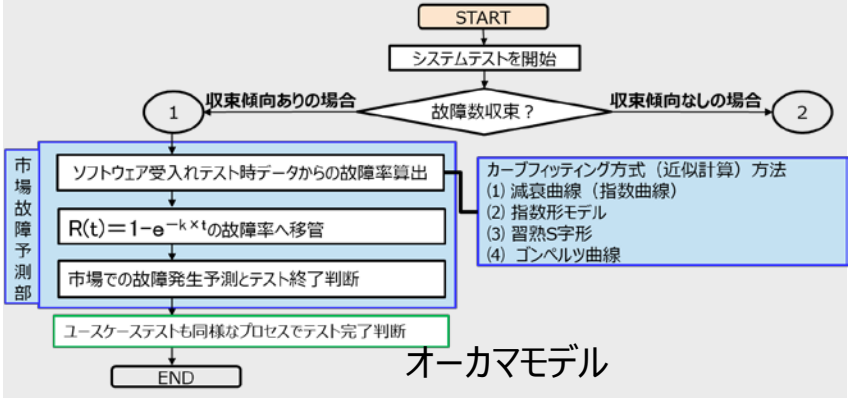
改善策

児島だよチーム
“ＡＲＣ
メソッド”

あつ森チーム
“あつ森法”

1.故障の原因を分析し再発防止を実施
2.故障を直したソフトウェアの影響分析を実施

原因コード
要求仕様書の確認不足
要求仕様書の理解不足
関連機能の確認不足
関連機能の理解不足
テスト仕様書の確認不足
テスト仕様書の理解不足
テスト方法習熟不足
テスト仕様書の検討不足



オーカマモデル

改善策を講じ、システムテストを開始し、故障数収束を確認する

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

4.結論

ソフトウェア信頼度成長モデルから故障率を算出し,$R(t) = 1 - e^{-k \times t}$ に置換するオーカマモデルを用いることで,市場故障発生予測と関連付けて，より精度良くテスト完了・追加の判断へとつながることが期待できる.

また,MTBFより算出した平均故障時間より，定期的な市場監視を推奨できる.

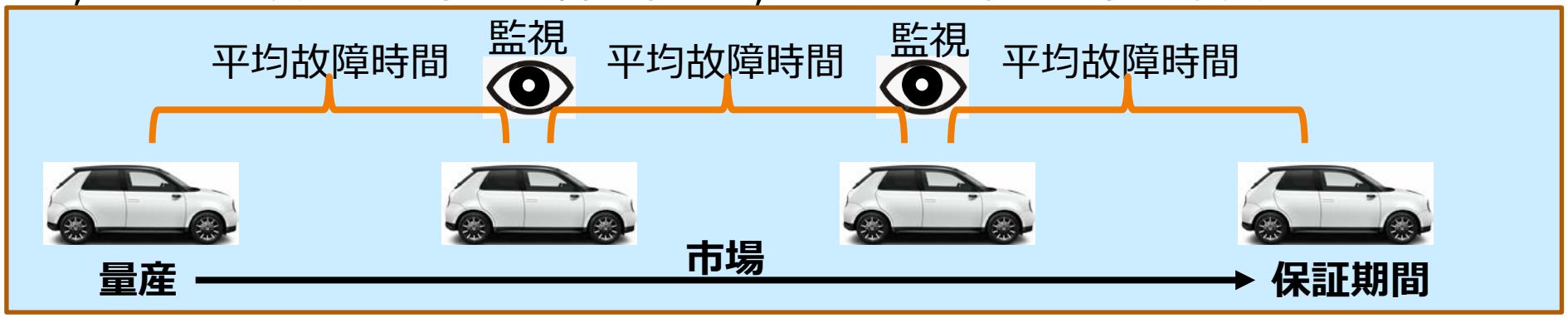


図4-1　テスト＋定期的市場監視

5.今後の課題と展開～おわりに

1）故障率の精度向上,市場故障発生予測の的確性を求めるために,ソフトウェア信頼度成長モデルの更なる追究.

2）原因分析と対策及びテストケース内容との関連性,またソフトウェア故障区分別の潜在故障予想及びテスト完了判断指標の策定.

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

ご清聴ありがとうございました。

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

APPENDIX

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

表２　テスト故障原因分析

原因コード	原因コードの推定材料
要求仕様書の確認不足	①　Input情報の入手 　　１．確認したInput情報バージョン 　　２．確認したInput情報の網羅性 　　３．確認したInput情報の伝達 ②　要求仕様書の一致性
要求仕様書の理解不足	①　要求仕様書の合意
関連機能の確認不足	①　利害関係者の期待事項 ②　システム要求、ソフトウェア要求の構造化 ③　システム要求、ソフトウェア要求の影響 ④　システム要求、ソフトウェア要求の構造化の一致性
関連機能の理解不足	①　システム要求、ソフトウェア要求の構造化の合意
テスト仕様書の確認不足	①　テスト戦略 ②　テスト設計書 　　１．テスト仕様書 　　２．テスト手順書 　　３．テストケース
テスト仕様書の理解不足	①　テスト設計書の合意
テスト方法習熟不足	①　テスト担当者がテスト習熟しなかったことに気付いたテスト手順書があるケース
テスト仕様書の検討不足	①　上記に該当しなかった場合は、『テスト仕様書の検討不足』が原因コード

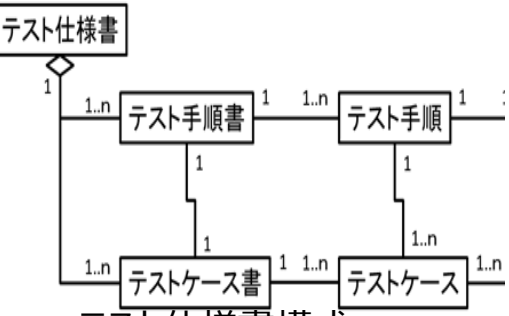
Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

補足

テスト仕様書の概要

テストケース書は、要求仕様書からテスト対象の機能、シナリオ、期待値（判断基準）、因子、水準を抽出し作成される

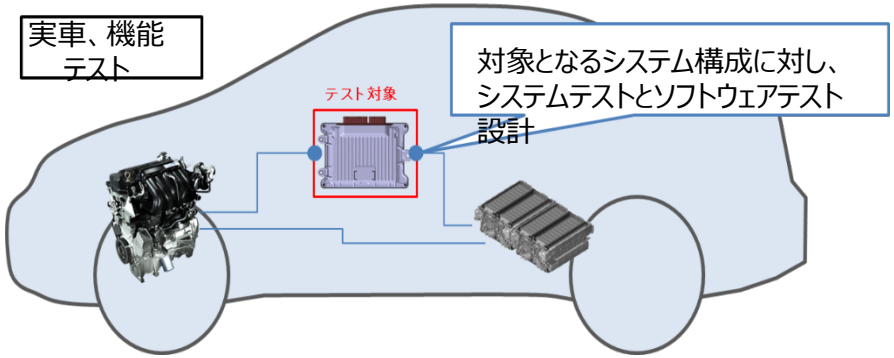
テスト仕様書は、「テスト手順書」と「テストケース書」で構成



テスト仕様書構成

No	確認内容(シナリオの内容)	期待値(判断基準)	因子	水準	1	2	3	4	5	6	7	8	9	10	11	12
1	Start Stop switchを押してすぐ離した(長押ししていない)場合のSTS信号は入力される。	STS信号は、0v-バッテリ電圧、730ms±20ms以外のこと	電圧	0v												
				5.9v(SPEC外)	○	○	○	○								
テスト区分				6v					○	○	○	○				
				12v(SPEC外)									○	○	○	○
			ON時間	709m(SPEC外)	○				○				○			
Autoモード				710ms		○				○				○		
				750ms			○				○				○	
				751ms(SPEC外)				○				○				○

テストケース書例



対象機能環境下におけるテストとテスト対象システム例

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

従来の代表的SRGMとそのY方程式

※**SRGM**: software reliability growth model

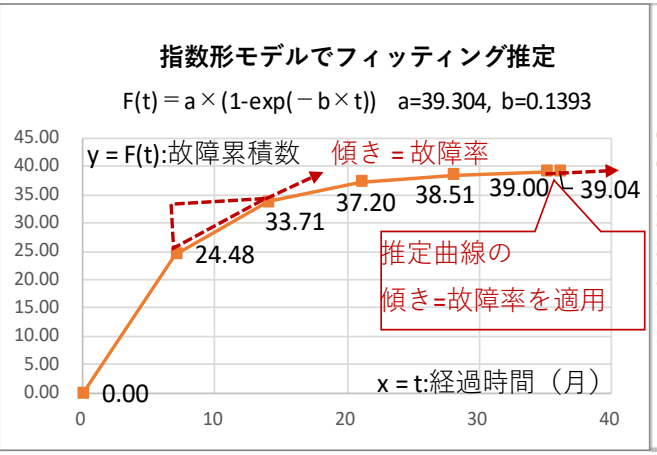
モデル（曲線）	一般式	統合モデルの微分方程式のパラメータ 1), 2)				一次微分 y'	Y方程式
		α	β	γ	δ		
指数形	$y = N(1 - e^{-bt})$	Nb	b	1	0	$N b e^{-bt}$	$Y_3 + b Y_1 - \ln(N b) = 0$
ゴンペルツ	$y = N \exp(-k e^{-bt})$ (注1)	kb	b	0	0	$k b e^{-bt} y$	$Y_3 - Y_2 + b Y_1 - \ln(k b) = 0$
ロジスティック	$y = \dfrac{N}{1 + \psi e^{-bt}}$	$\dfrac{b\psi}{N}$	b	-1	0	$\dfrac{b \psi e^{-bt}}{N} \cdot y^2$	$Y_3 - 2 Y_2 + b Y_1 - \ln\left(\dfrac{b\psi}{N}\right) = 0$
習熟S字形	$y = N \dfrac{1 - e^{-bt}}{1 + \psi e^{-bt}}$ (注2)	$\dfrac{b\psi^2}{N(\psi + 1)}$	b	-1	$\dfrac{N}{\psi}$	$\dfrac{N b (\psi + 1) e^{-bt}}{(1 + \psi e^{-bt})^2}$	—
遅延S字形	$y = N\{1 - (1 + b t) e^{-bt}\}$	$\sim 2.2 N^{0.46} b$	$\sim 0.78b$	~ 0.46	0	$N b^2 t e^{-bt}$	$Y_3 - 0.54 Y_2 + 0.78 b Y_1 - \ln(0.78 N b) = 0$ （近似式）
超指数形	$y = \sum_{i=1}^{n} N_i (1 - e^{-b_i t})$	$\sim \dfrac{N^r \beta}{\gamma}$	—	>1	0	$\sum_{i=1}^{n} N_i b_i e^{-b_i t}$	（Y2の係数が正となるY方程式。ただしY3＝１とする）
ワイブル過程	$y = \lambda t^n$	$n \lambda^{\frac{1}{n}}$	0	$\dfrac{1}{n}$	0	$\lambda n \left(\dfrac{y}{\lambda}\right)^{\frac{n-1}{n}}$	$Y_3 + \dfrac{1-n}{n} Y_2 - \ln(n) - \dfrac{1}{n} \ln(\lambda) = 0$

古山恒夫　ソフトウェア信頼度成長モデルに関する統合モデルの解析的パラメータ推定法　情報処理学会論文誌　,1996

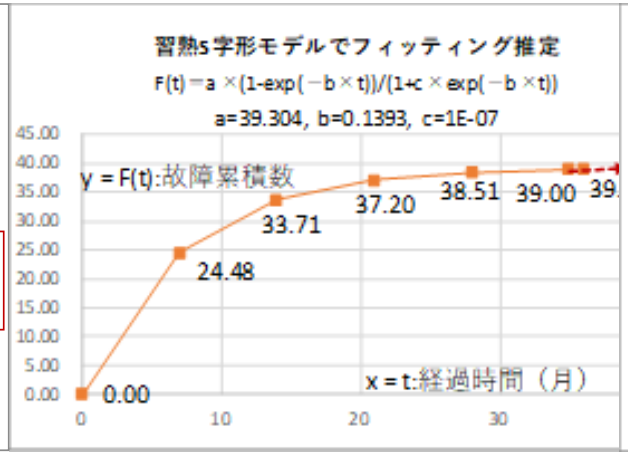
Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

3.2 補足 ソフトウェア信頼度成長モデル故障で累積数曲線を カーブフィッティングした推定曲線から故障率を算出

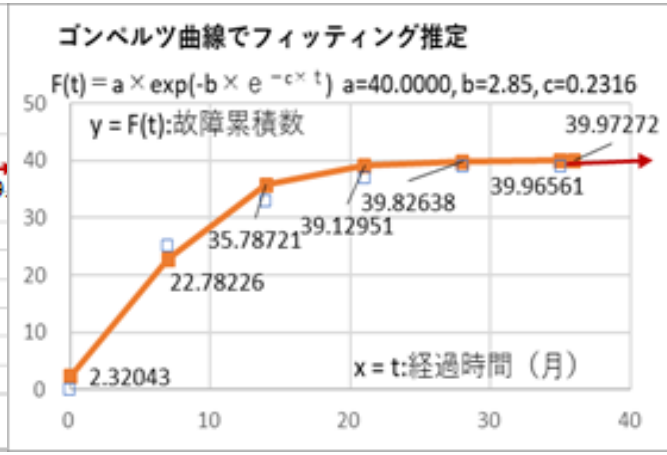
指数形モデル



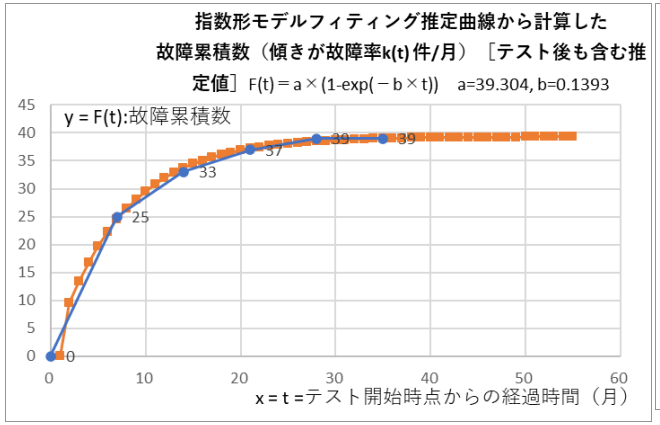
習熟S字形モデル

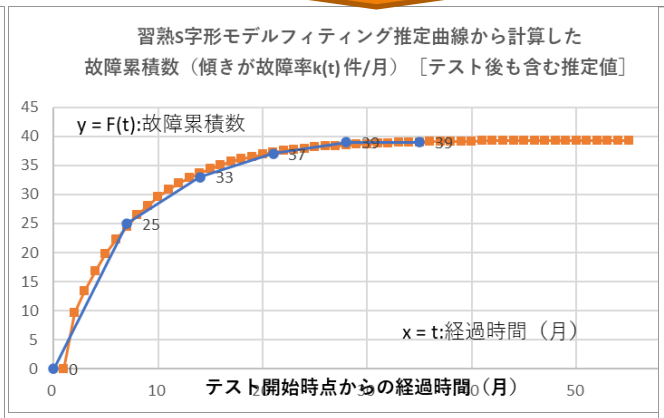


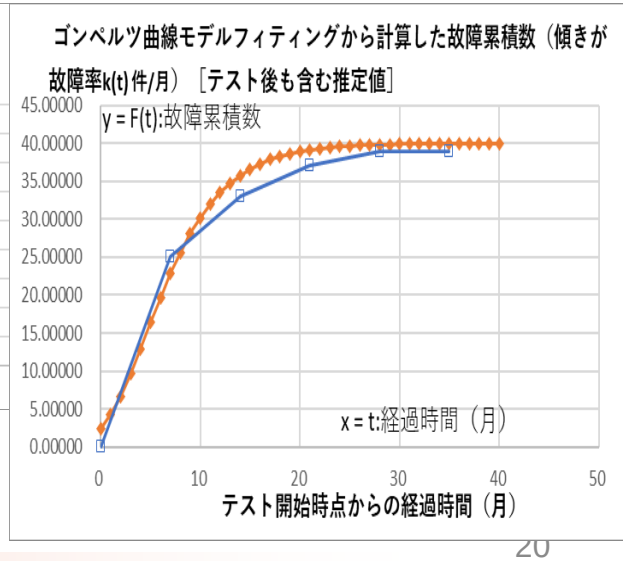
ゴンペルツ曲線



テスト後も含めて推定



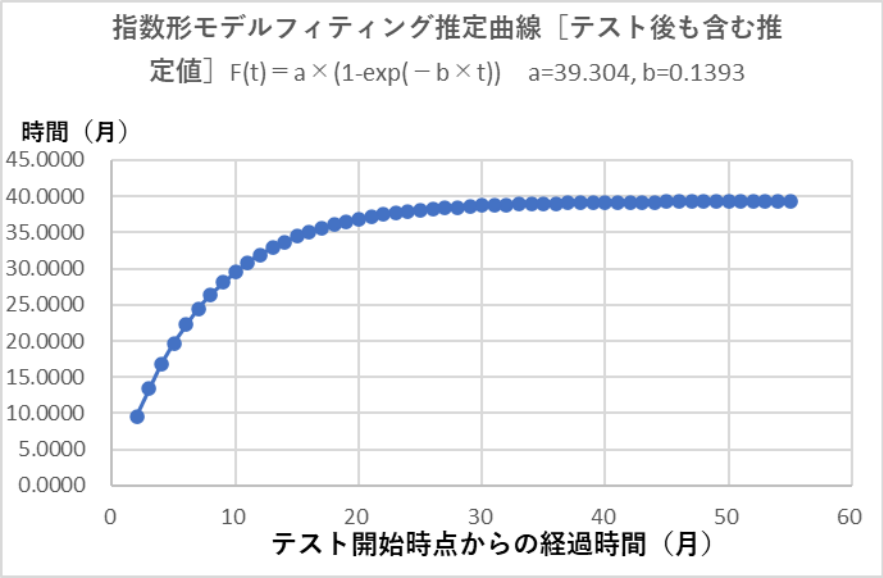




Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

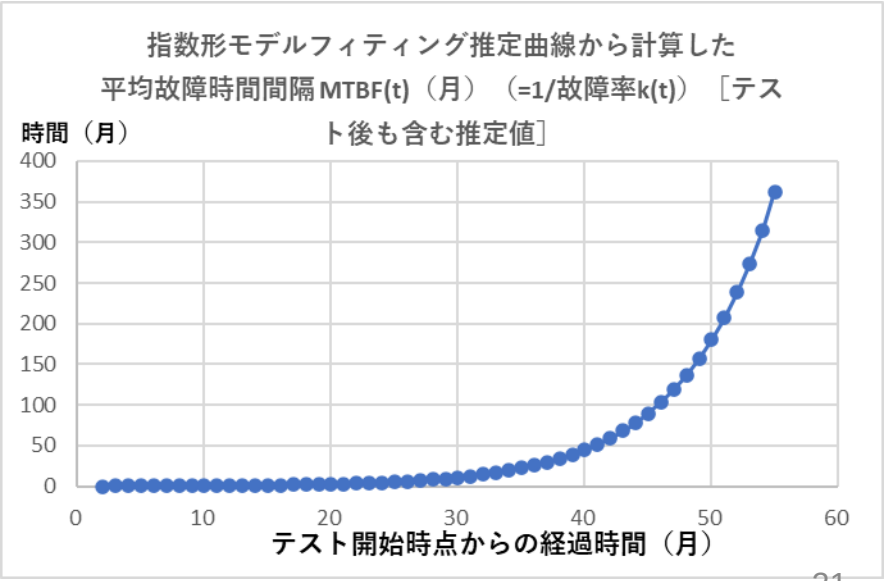
信頼度成長モデルから故障率算出を通して推定MTBFを算出

信頼度成長モデルでフィッティング



単位時間当たりの傾きから故障率を算出

単位時間当たりの傾きから故障率の逆数から経過時間tでのMTBF(t)を計算

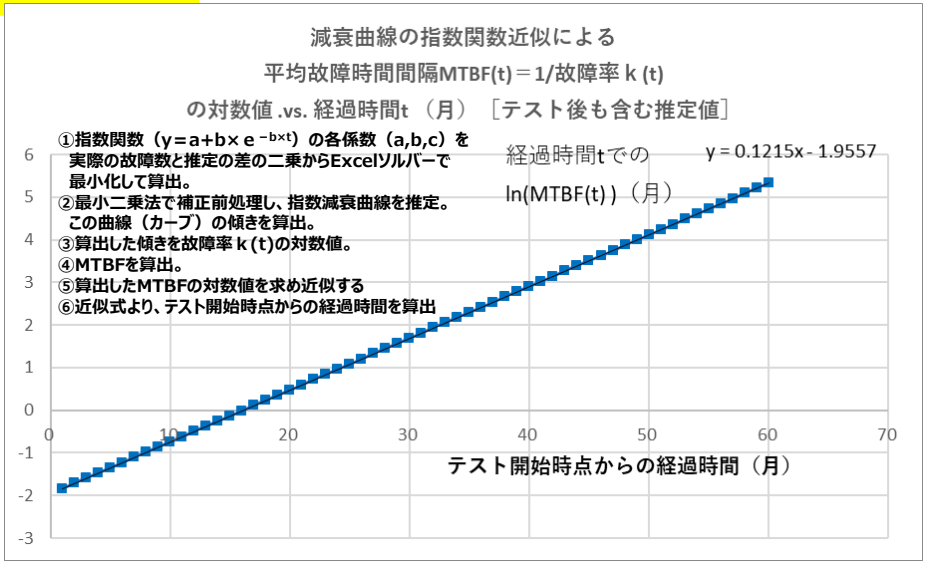


Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

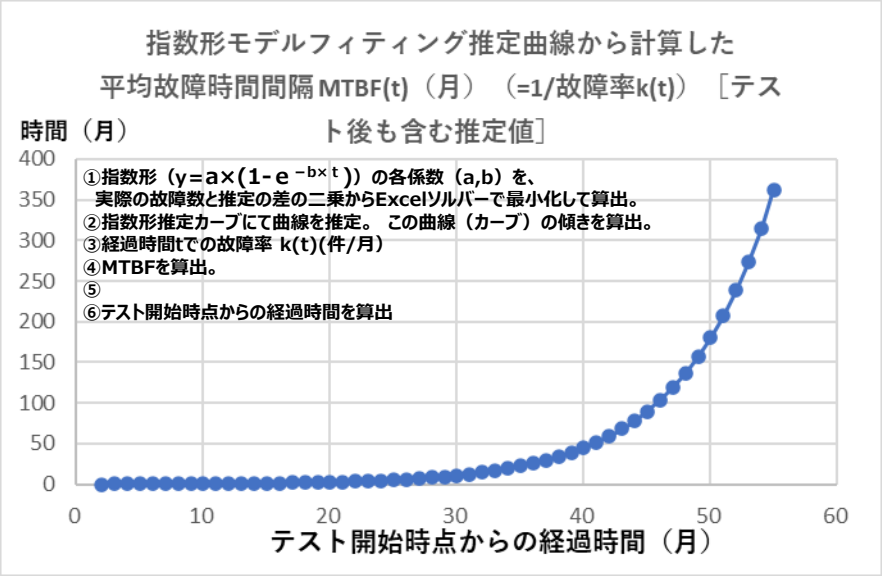
3.4 補足　故障率の各算出法から推定したMTBF

MTBF = 開発時間 / 故障回数 (故障率 = *1 / MTBF*)

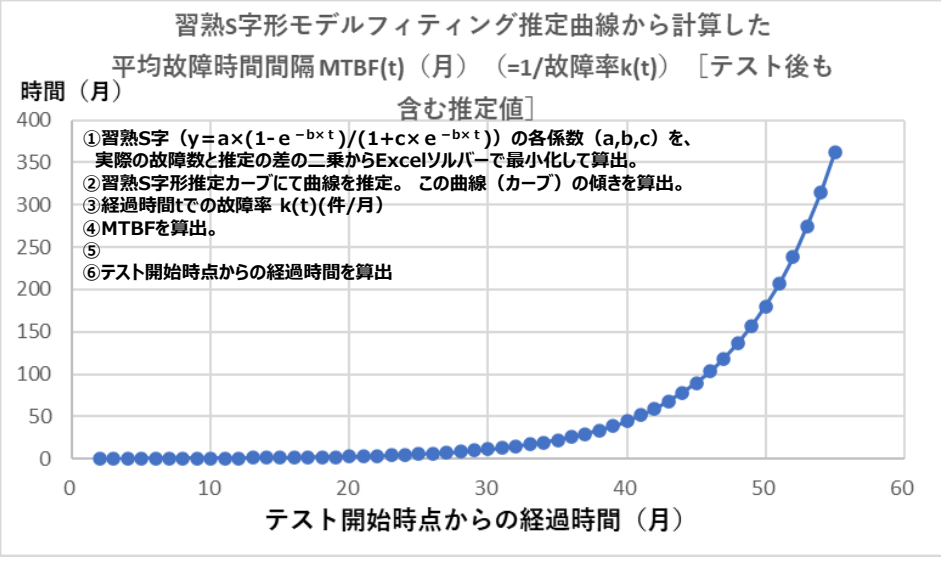
(1)減衰曲線の指数関数近似



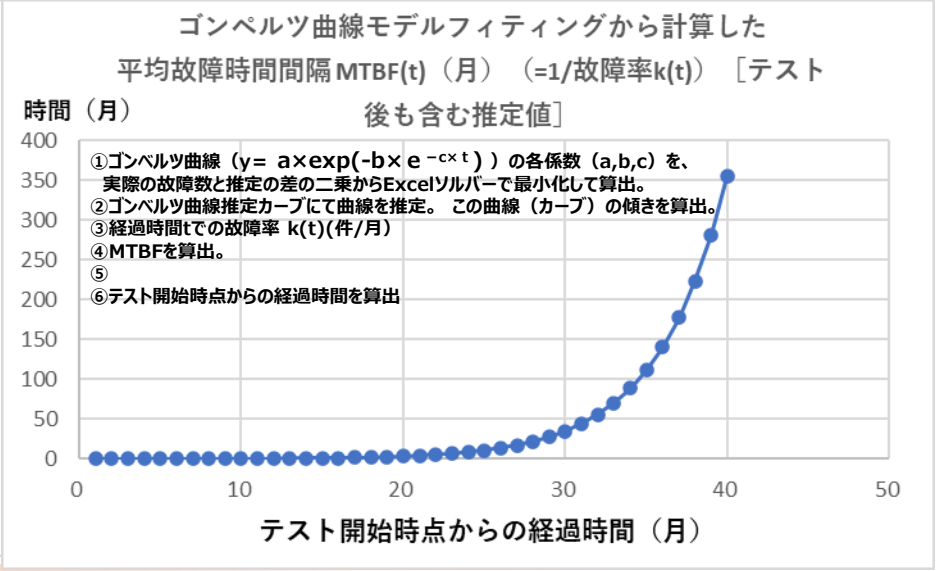
(2)指数形モデルフィッティング



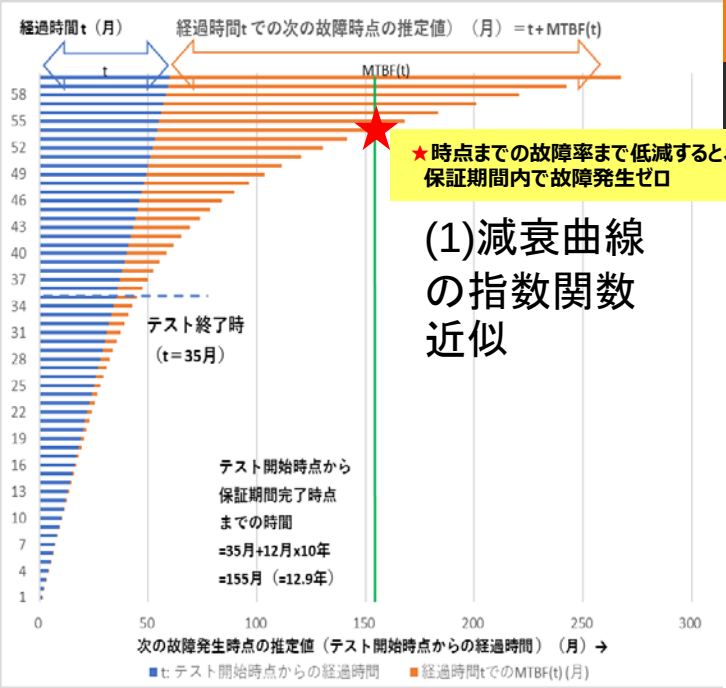
(3)習熟S字形モデルフィッティング

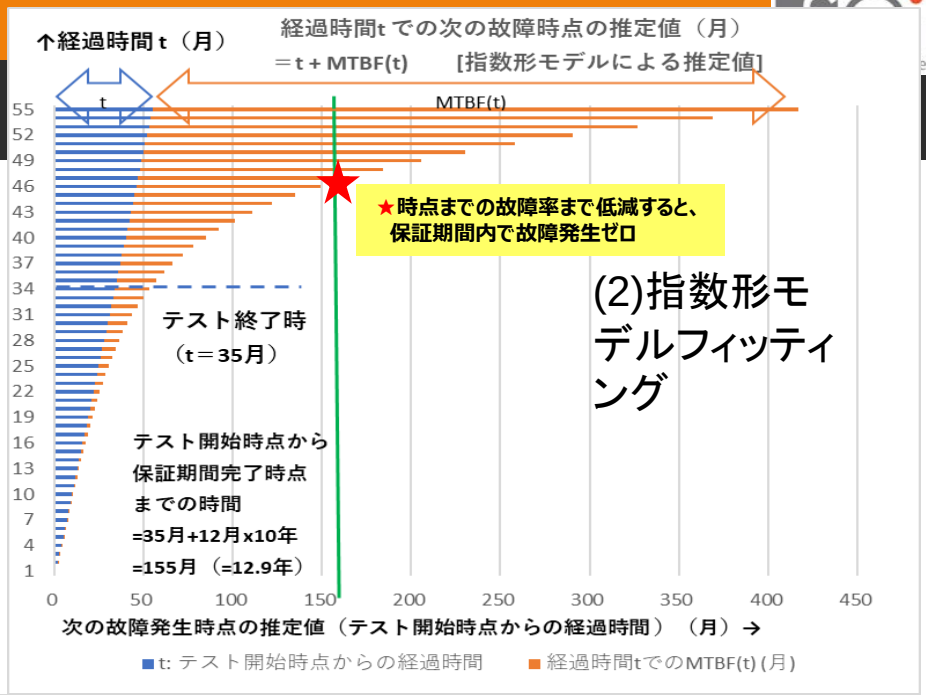


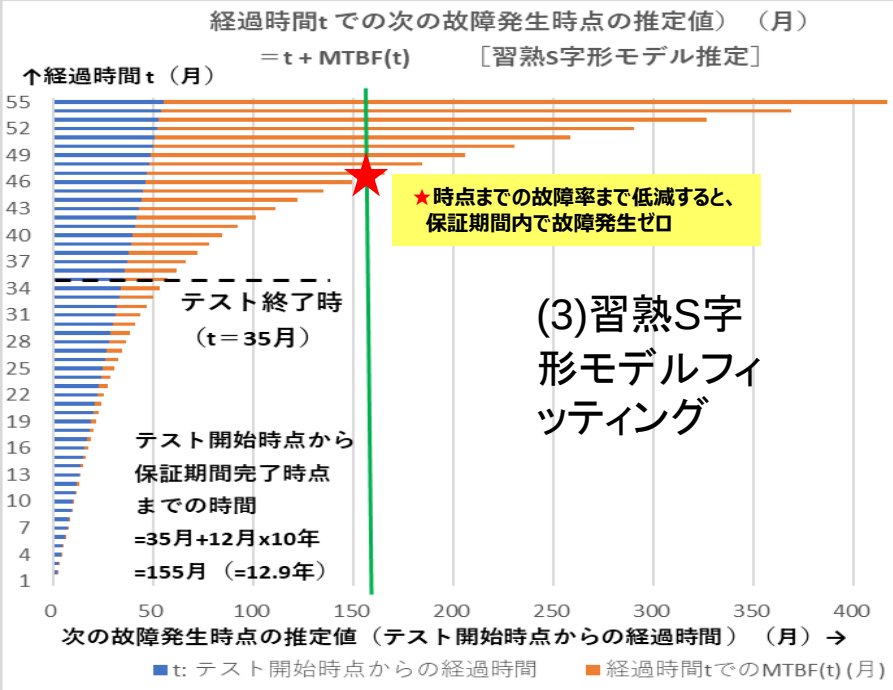
(4)ゴンペルツ曲線モデルフィッティング

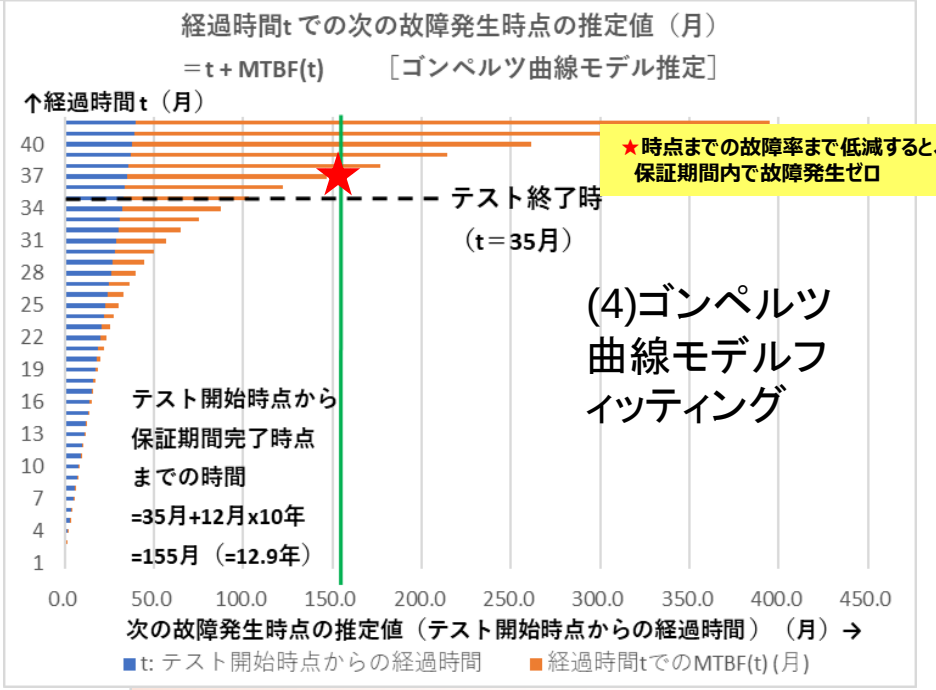


Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved









Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

1）故障率の算出解説

指数減衰曲線推定（最小二乗法で補正前処理）

最小二乗法で補正前処理したカーブの傾き（故障率(t)）の時間変化

減衰曲線の指数関数近似を適用　y ＝ ln(k(t))

故障率をより的確に算出できる可能性あり

①指数関数（y＝a+b×ｅ$^{-b\times t}$）の各係数（a,b,c）を実際の故障数と推定の差の二乗からExcelソルバーで最小化して算出。
②最小二乗法で補正前処理し、指数減衰曲線を推定。 この曲線（カーブ）の傾きを算出。
③算出した傾きを故障率ｋ(t)の対数値。
④MTBFを算出。
⑤算出したMTBFの対数値を求め近似する
⑥近似式より、テスト開始時点からの経過時間を算出

⑥

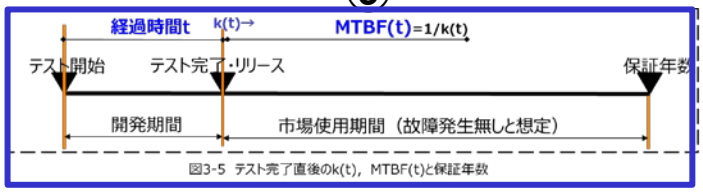


図3-5 テスト完了直後のk(t)，MTBF(t)と保証年数

			①	①	②	③	④	⑤
減衰曲線の指数関数近似による故障率ｋ(t)			y＝A+B*exp(-C*t)					
開発期間	35か月	A=	0.579655789					
マイルストーン間隔	7か月	B=	73.98748062					
		C=	0.158909309		最小二乗法で補正前処理したカーブの傾き（故障率(t)）の時間変化	傾き=故障率k(t)の対数値		
t(月)		故障数	指数減衰曲線推定（最小二乗法で補正前処理）	実際と推定の差の二乗	**経過時間tにおける故障率k(t)/月**	Ln(k(t))	MTBF＝1/k(月)	
0						#NUM!		
7		25	24.90529089	0.008969816	3.5578986985	1.269170117	0.2810647758	-1.2691701174
14		8	8.577448363	0.333446612	1.2253497662	0.203226327	0.8160935168	-0.2032263266
21		4	3.209173407	0.6254067	0.4584533438	-0.779896751	2.1812470417	0.7798967507
28		2	1.444189701	0.308925088	0.2063128145	-1.578361745	4.8470086679	1.5783617452
35		0	0.863897614	0.746319087	0.1234139448	-2.092211169	8.1028120569	2.0922111689
				2.02306730				
				↑Excelソルバーで最小化				

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

補足
追検証
データ

UNIT Bの累積故障曲線を示す。 UNIT Bは、現在量産しており、ROM使用量は1477K,RAM使用量は194Ｋである。

実開発のUnit Bの開発変遷と市場保証期間と市場生産台数を以下に記す。開発期間を15か月、各マイルストーン期間を15か月、テストケース3762件、生じた故障数を111件。全故障に対して対策を講じた。 また、市場保証期間を10年

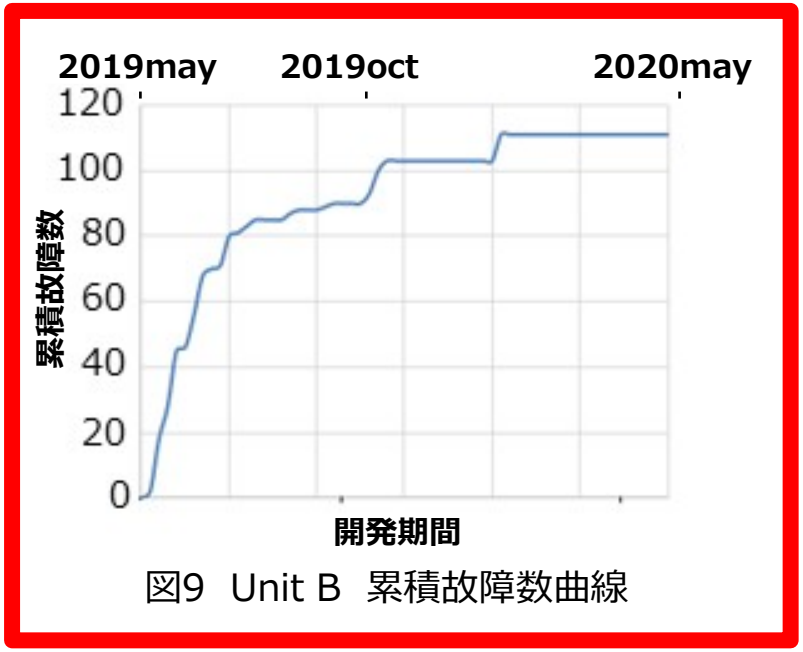


図9 Unit B 累積故障数曲線

市場故障予測分析の結果：
UNIT-Bの算出法別の故障率k，MTBFと市場故障数予測一覧

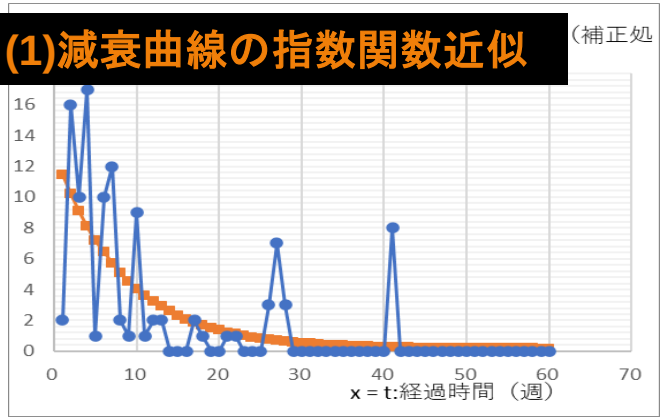
故障率の算出方法	テスト完了直後の故障率k(件/月)	テスト完了直後から市場保証期間10年間・1000万台中の市場故障数予測(台)	同左の1000万台中の無故障数予測(台)	テスト完了直後の平均故障時間(月) MTBF(Mean Time Between Failure)=1/k	その時点でのMTBF＞テスト完了後の市場保証期間10年完了時(15+10 x 12=135月=580週)となるまでの，テスト開始からの経過時間
(1)減衰曲線の指数関数近似	0.08件/週=0.344件/月	10,000,000	0	0.24年=2.9月=12.6週	2.1年=25.6月=111週(テスト後1.0年=11.8月=51週)
(2)指数形モデルフィッティング	0.033件/週=0.144件/月	10,000,000	0	0.58年=7.0月=30.2週	1.8年=21月=91週(テスト後0.6年=7.2月=31週)
(3)習熟S字形モデルフィッティング	0.033件/週=0.144件/月	10,000,000	0	0.58年=7.0月=30.2週	1.8年=21月=91週(テスト後0.6年=7.2月=31週)
(4)ゴンペルツ曲線モデルフィッティング	0.015件/週=0.064件/月	9,995,539	4,461	1.3年=15.6月=67.4週	1.5年=18月=78週(テスト後0.3年=4.2月=18週)

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

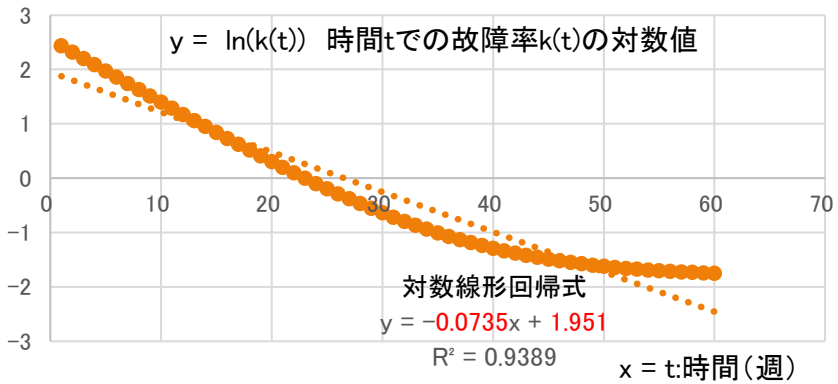
補足　追検証データ
UNIT B

ソフトウェア信頼度成長モデル故障で累積数曲線をカーブフィッティング

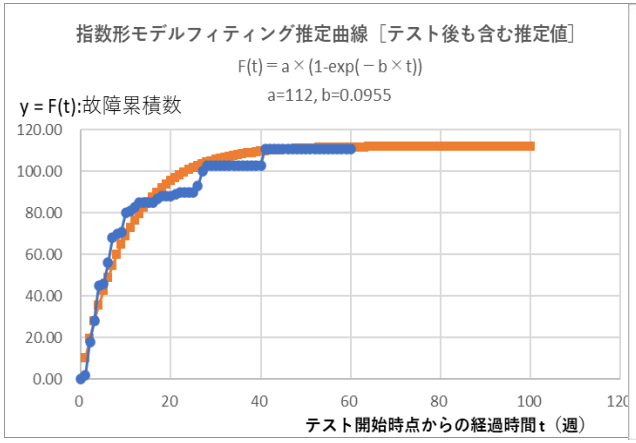
(1)減衰曲線の指数関数近似



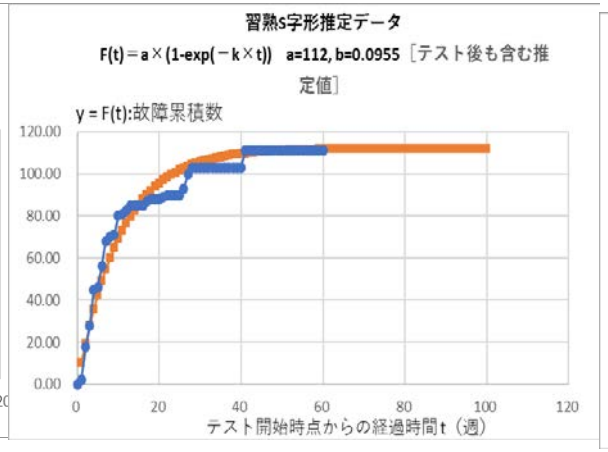
最小二乗近似の減衰曲線の指数関数近似による経過時間tでの故障率k(t)（対数値）.vs. 経過時間t（週）



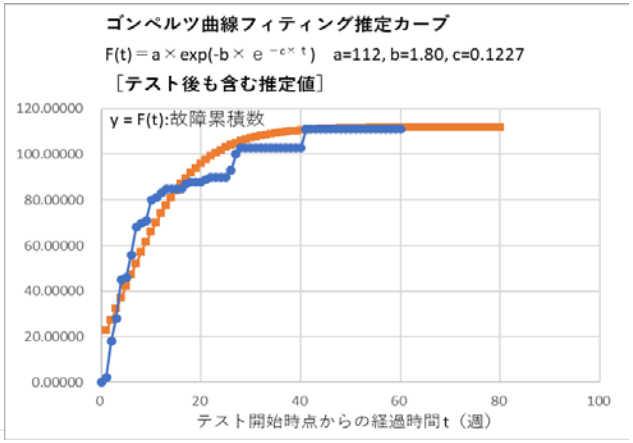
(2)指数形モデル



(3)習熟S字形モデル



(4)ゴンペルツ曲線モデル

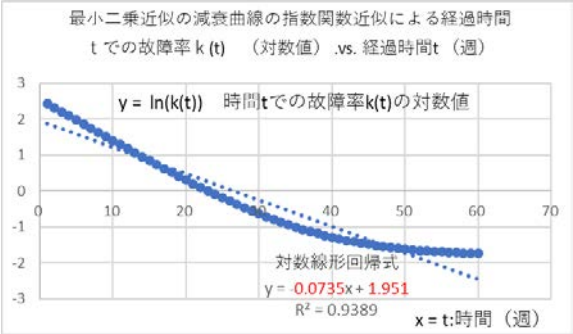


Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

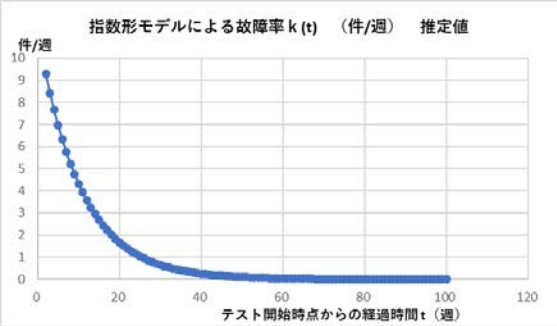
補足　追検証データ
UNIT B

ソフトウェア信頼度成長モデル故障で累積数曲線をカーブフィッティングした推定曲線から故障率を算出

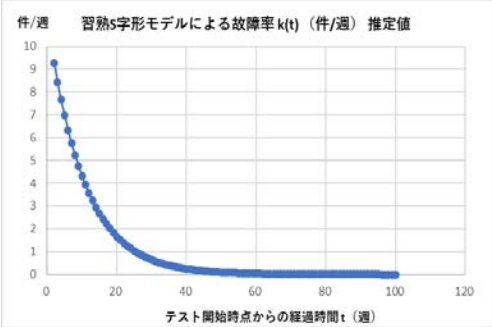
(1)減衰曲線の指数関数近似



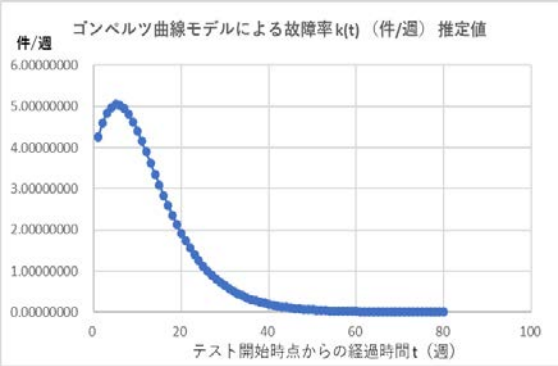
(2)指数形モデル



(3)習熟S字形モデル



(4)ゴンペルツ曲線モデル

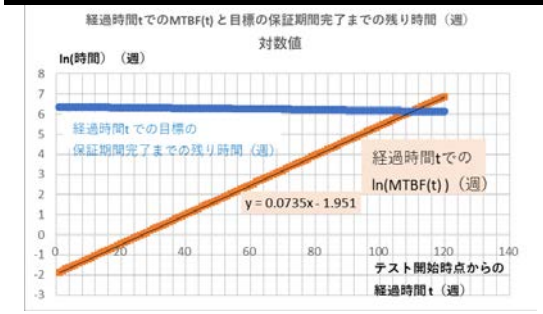


Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

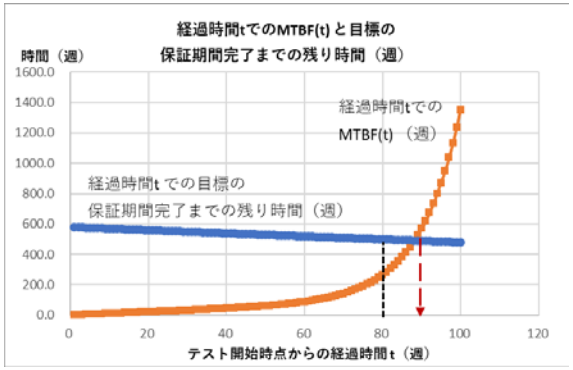
補足　追検証データ
UNIT B

経過時間tでのMTBF(t) と目標の保証期間完了までの残り時間（週）

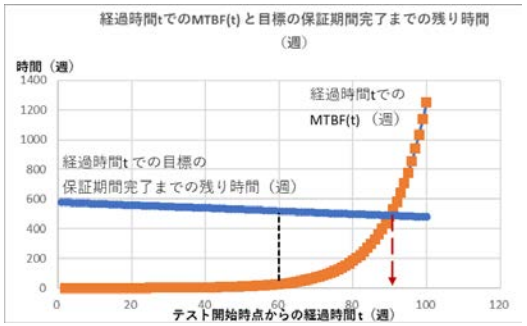
(1)減衰曲線の指数関数近似



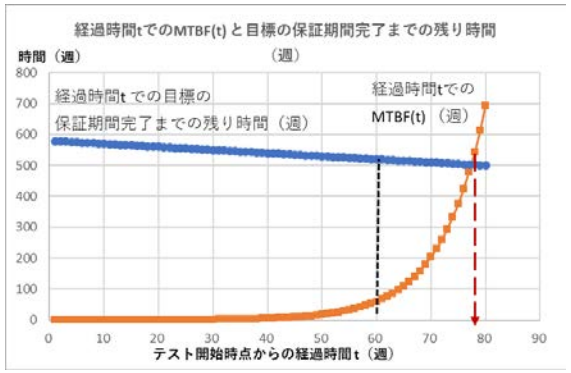
(2)指数形モデル



(3)習熟S字形モデル



(4)ゴンペルツ曲線モデル

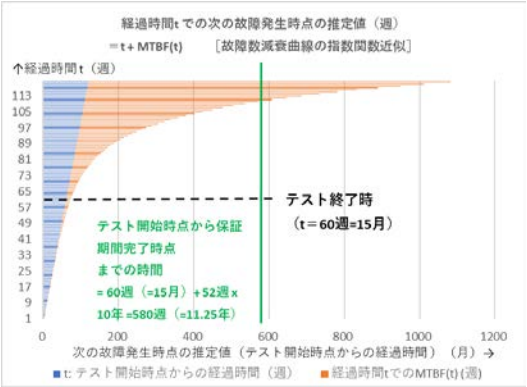


Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

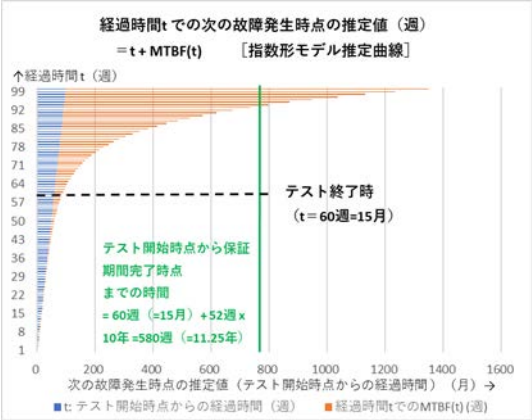
補足　追検証データ
UNIT B

経過時間t での次の故障発生時点の推定値（週）=t + MTBF(t)

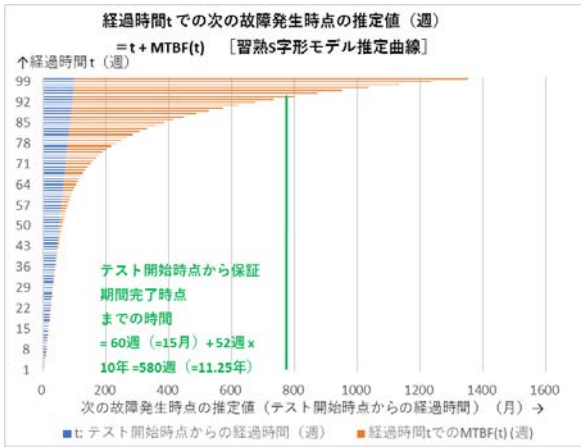
(1)減衰曲線の指数関数近似



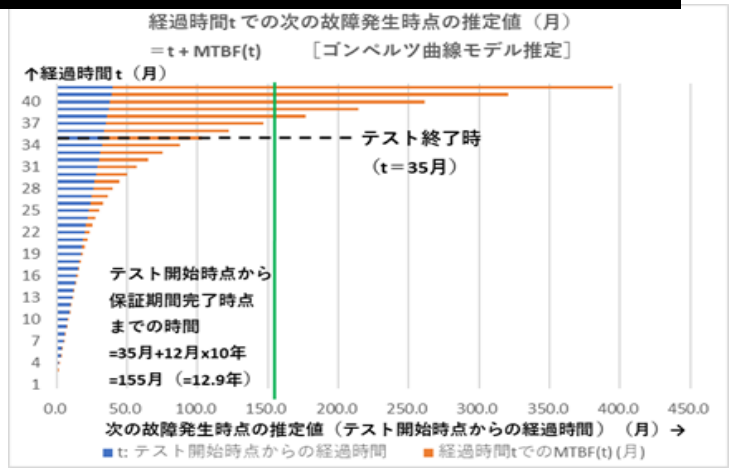
(2)指数形モデルフィッティング



(3)習熟S字形モデルフィッティング



(4)ゴンペルツ曲線モデルフィッティング



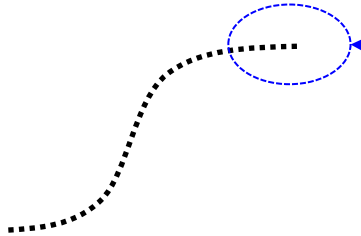
Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

3.5 考察　補足 故障率に関する考察

1）カーブフィッティング方式（近似計算）方法によって算出した故障率は,
(1) 減衰曲線（指数曲線）≠ (2) 指数形モデル ≠ (3) 習熟S字形 ≠ (4) ゴンペルツ曲線
（小数点以下1位から3位までの値）

複数の算出法を組合せて一覧し，比較又は選択して用いるのが良いと考える.

2) ソフトウェア信頼度成長モデルに関して，故障数が,指数関数的な減衰を示す場合，S字形モデル（習熟S字形,ゴンペルツ曲線）も指数に非常に近い計算結果となった．習熟S字形とゴンペルツ曲線は2点の変曲点を有する特性だが，時間t(i)での故障率k(t(i))は時間t(i-1（35か月）)からt(i（36か月）)の間の単位時間の傾きのためと考える.



時間t(i)での故障率k(t(i))は時間t(i-1（35か月）)から
t(i（36か月）)の間の単位時間の傾き

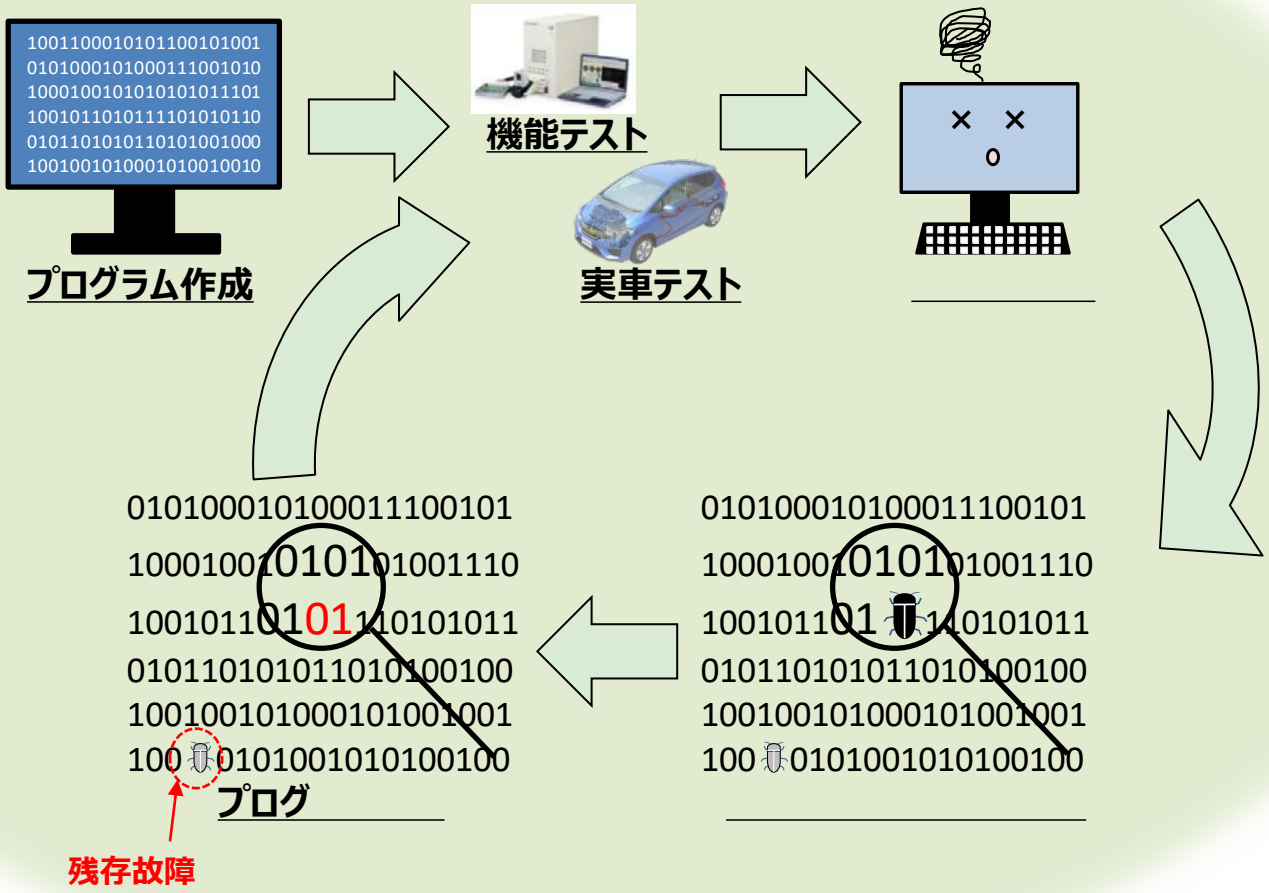
習熟S字形,ゴンペルツ曲線の概念図

ソフトウェアがリリースされる毎の
受入れテストへの有効性

ソフトウェア開発組織内（取引先様）
のテストへの有効性も期待

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

2.1 課題



故障を直しても他の故障は残存する可能性があり。

いつまでテストを実施すればソフトウェアの市場品質を保証できるか？

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

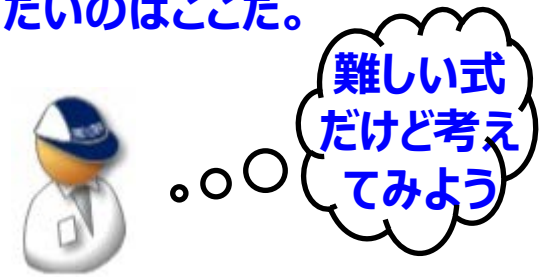
ソフトウェア受入れテスト時データからの故障率算出

「やったー、故障が出なくなったからテストは終了だね」、「故障をグラフ化することで品質確認できるかな」と検討できる信頼度成長モデルの例

モデル（曲線）	一般式
指数形	$Y=N(1-e^{-bt})$
ゴンベルツ曲線	$Y=N\ \exp(-k\ e^{-bt})$
習熟S字形	$Y=N(1-e^{-bt})/(1+\psi e^{-bt})$

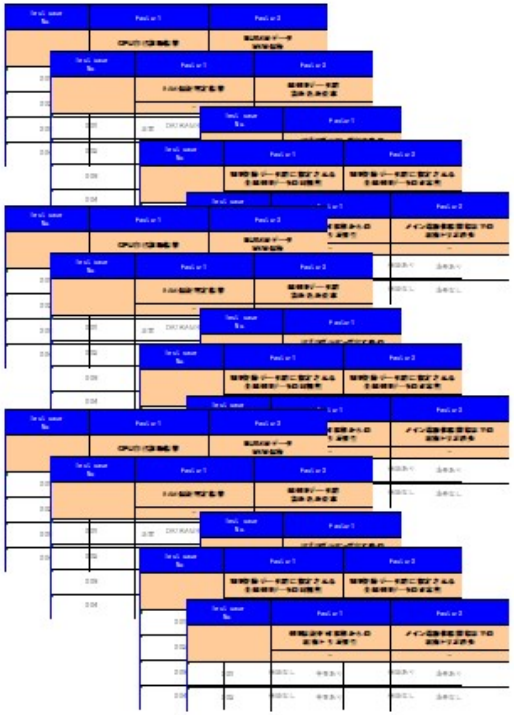
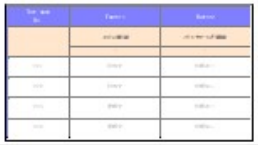
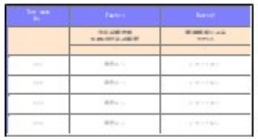
古山恒夫　ソフトウェア信頼度成長モデルに関する統合モデルの解析的パラメータ推定法　情報処理学会論文誌　,1996

テスト終了判断！
残存故障による故障が、市場で生じるのか！
故障が生じる可能性があるならその頻度は？
知りたいのはここだ。



Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved

SW Variant違いによるテスト

	共通テスト仕様	Variant違いによる テスト仕様
Variant1		
Variant2		
Variant3		
Variant4		
Variant5		
	共通テストは代表機種（ソフトウェア）でテスト	Variant違いは、各々のソフトウェアでテスト

Copyright (C) 2021 ソフトウェア品質管理研究会 All Rights Reserved