

ソフトウェアメトリ クスの基礎. 6

60151054

吉行 勇人

2012年11月30日

第7章

欠陥に学ぶ

質問！

1. 製品を出荷してから半年間、欠陥が全く報告されていません。よろこんでいいのでしょうか？
2. 上司から、出荷までに99%のバグを発見、修正するように指示がありました。
どう答えますか？

なぜ欠陥を測るのか

ユーザにとって

欠陥はできるだけ少ないほうがいい

しかし

技術者にとって

欠陥は発見されたほうが嬉しい



欠陥はスケジュールやソフトウェア品質において定量的で役に立つメトリクスである

障害と故障

「障害」はシステム内に現存する欠陥のこと

「故障」はシステムを実行したときに、要求されたパフォーマンスを発揮できないこと

7章で学ぶ欠陥メトリクスは「障害」を測定する

次章で学ぶ信頼性メトリクスでは「故障」を測定する

障害と故障（補足）

英語の使い分け

欠陥 defect

障害 fault

故障 failure

バグ bug

欠陥の重要度

欠陥の中には、致命的なものから、取るに足らないものまで幅がある

プロジェクトマネージャーはプロジェクトと製品のパフォーマンスに本当に影響を与える欠陥に着目する必要がある

欠陥密度

一般的によくつかわれる欠陥に関するメトリクスは、欠陥密度である

欠陥密度は、KLOCまたはFPあたりの欠陥の数で測定する

欠陥発見率

欠陥には特有の法則性、ふるまい、パターンがある

欠陥の発見率も特有の法則性がある

欠陥発見率の法則性

欠陥発見率はレイリー曲線に従うことが多い



欠陥と工数の関係

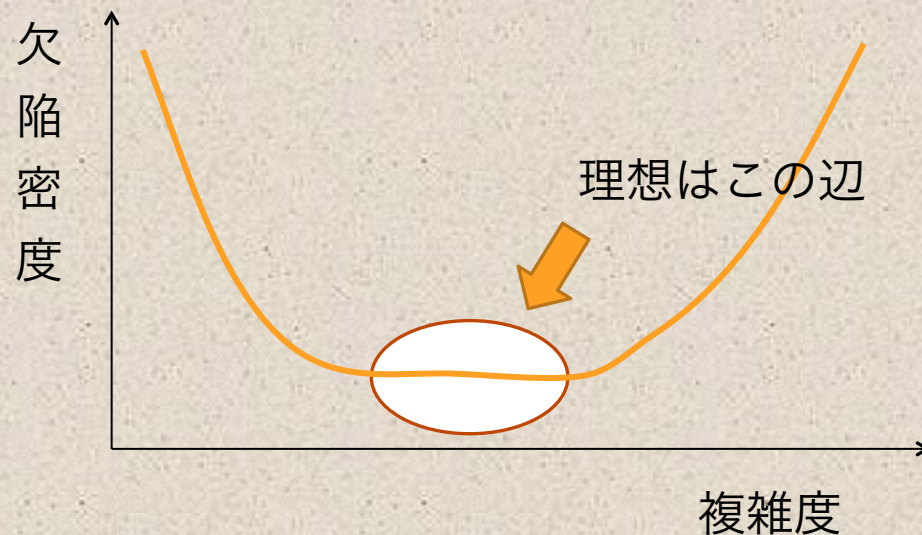
欠陥と工数は一般に直線関係にある（正の相関）

これは人々は欠陥の原因となる誤りをほぼ一定の割合で犯していると言える

これをゼロにはできないが、スキルの向上やスケジュール、プロジェクト要員の調整等で傾きを緩やかにすることはできる

欠陥密度と モジュール複雑度の関係

複雑度と欠陥密度の関係は
一般にバスタブ曲線と呼ばれる



バスタブ曲線について

複雑度が高いとき欠陥密度が高くなるのはわかるが、

複雑度が小さい時欠陥密度が大きいのはなぜか？



スタートアップ欠陥

どんなモジュールにも、ある程度存在する初期欠陥

複雑度が小さい→規模が小さいプログラムだと欠陥
密度は大きくなる

欠陥数の予測

欠陥の予測に用いる技法とモデルの目的は、
総欠陥数の予測と
時系列における欠陥分布の推定である

- 動的モデル：発見された欠陥の統計分布に基づいて欠陥数を予測する
- 静的モデル：プロジェクトとプロセスの属性を用いて欠陥数を予測する

動的欠陥モデル

動的モデルを適用するには、ある程度の欠陥データが必要である

レイリーモデル、指数モデル、S字モデルがある

レイリーモデルは開発サイクル全体をモデル化
指数モデル、S字モデルはテストやシステム導入
プロセスをモデル化している

分布関数

確率密度関数 $f(t)$

時刻 t における欠陥発見数

累積分布関数 $F(t)$

時刻 t までに発見された欠陥数の合計

レイリーモデル

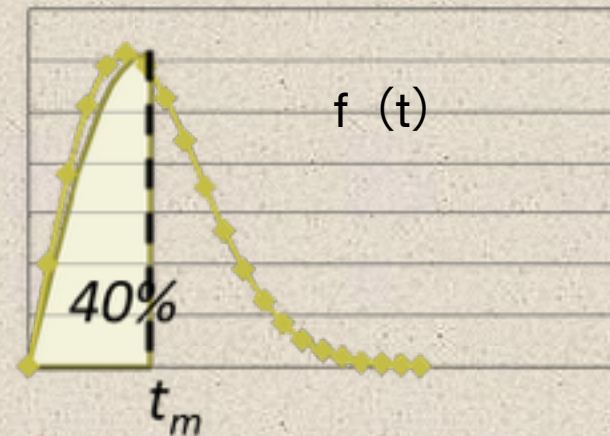
確率密度関数: $f(t) = K \left[(1/tm)^2 t e^{-(1/2tm^2)t^2} \right]$

累積分布関数: $F(t) = K \left[1 - e^{-(1/2tm^2)t^2} \right]$

K: 総欠陥数

t: 時刻

t_m : $f(t)$ が最大となる時刻
 t_m の時点で40%の欠陥が発見される



レイリーモデルの例

週	1	2	3	4	5	6	7	8	9
欠陥	20	41	48	52	62	59	52	44	33

※最初の40%の欠陥が t_m までに発見される

第5週までに発見された欠陥の総数が223

$$K = 223 * (100/40) = 557, t_m = 5$$

$$f(t) = 557(t/25)e^{-t^2/50} = 22.3te^{-t^2/50}$$

$$F(t) = 557(1 - e^{-t^2/50})$$

この式から第12週目の時点で発見された累積欠陥の数を予測する

$F(12) \approx 525$ 12週の時点で約95%の欠陥が発見されている？

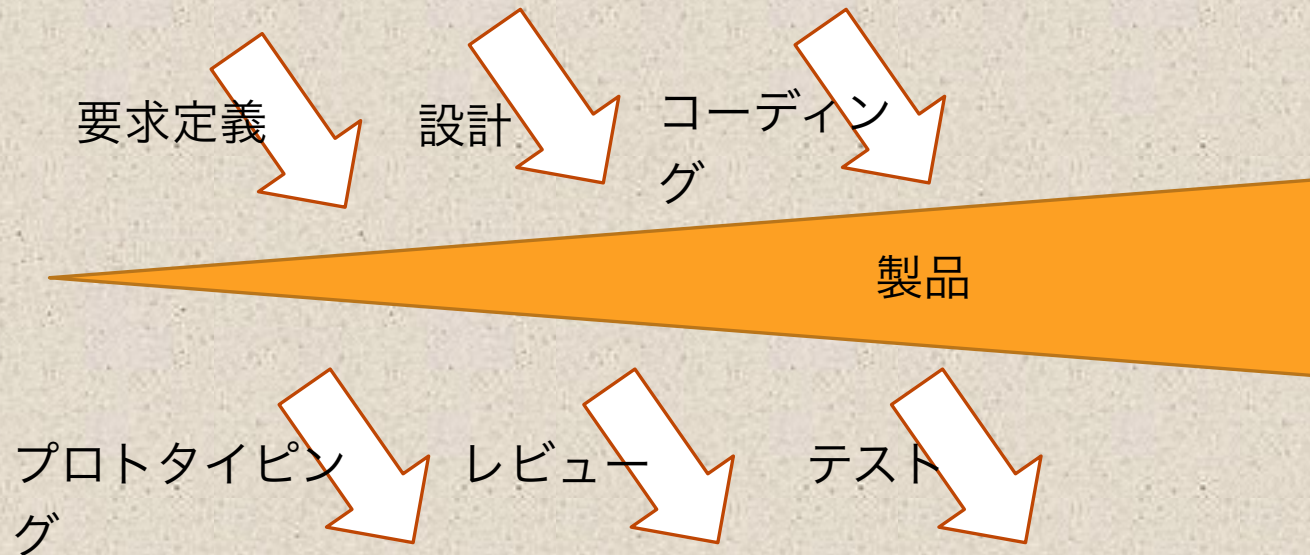
静的欠陥モデル

静的欠陥モデルは過去のプロジェクトデータに基づいたモデルであって、現在のプロジェクトのデータによるものではない

よって、プロジェクト開始前でも適用できる

欠陥の作り込みと除去

欠陥はプロジェクトの至るところで作りこまれ、テストなどを通して除去される



静的欠陥モデルのツール

COQUALMO

要求定義、設計、コーディングにおける欠陥
予測モデル

欠陥作り込み要因と欠陥除去要因に対する評
価を入力として、欠陥密度の予測値を得る

欠陥除去率

ある段階での欠陥がどれくらい除去できたか

全体の欠陥除去率： $E / (E + D)$

E ：製品を出荷するまでの欠陥発見数

D ：出荷後の欠陥発見数

工程別の欠陥除去率： $E_i / (E_i + E_{i+1})$

E_i ：ある欠陥除去工程での欠陥発見数

E_{i+1} ：次の欠陥除去工程での欠陥発見数

欠陥除去率の ベンチマークデータ

過去のデータから（アメリカ）、一般的に、多くのプロジェクトは出荷前の時点で75%～85%の欠陥しか除去できていない

どんなに熟練しても95%程度が限度で、それ以上の欠陥除去には莫大なコストがかかる

一般的な情報システムには95%で十分である

95%以降が難しい

9週の時点で446個

12週の時点で525個

15週の時点で550個



79個見つかった

25個見つかった

同じ3週間でも見つかる欠陥の数が違う

15週の時点で、約99%は達成しているが、
さらなるコストをかけてまでやるべきなのか？

工程ごとの欠陥除去の コスト効率

欠陥の除去が遅れるほど、その除去コストは
高くなる

要求/設計における修正コスト：1倍とすると

コーディングにおける修正コスト：5倍

テスト工程における修正コスト：25倍

出荷後における修正コスト：250倍

顧客から報告される欠陥

顧客から報告される欠陥はユーザ数と正の相関がある

つまり、ソフトウェアは使われれば使われるほど欠陥が多く見つかる

逆に言えば

ソフトウェアがバグだらけなら、使うユーザが少なくなり、欠陥も対して見つからない

質問 1 の答え

製品を出荷してから半年間、欠陥が全く報告されていません。よろこんでいいのでしょうか？



まず確認すべきことは、

そのシステムを使っている人がいるかどうか

報告される欠陥がすくない理由は、誰も使っていないからかもしれない・・・

多くの人が使っていることが確認できて、それでもなお欠陥が報告されていないのであれば、欠陥除去率の高い、いいプロジェクトであったと言える

質問 2 の答え

上司から、出荷までに 99% のバグを発見、修正するように指示がありました。



99% のバグの発見と修正はほぼ不可能であり、とてもコストがかかる

欠陥ベンチマークのデータについて上司と議論し、目標を適切なものとするように提案するべき

まとめ

欠陥除去率：出荷前、または開発の各工程で除去した欠陥の割合

欠陥密度：残存しているKLOCあたりの欠陥数

欠陥には特有のパターンと、法則性がある

欠陥はプロジェクトの至るところで作りこまれ、テストなどを通して除去される

欠陥数や欠陥発見率を予測する方法