

ソフトウェアメトリクスと データ分析の基礎

コンピュータソフトウェア Vol.28 No.3(2011)

阿萬 裕久 野中 誠 水野 修

60171050 松野祐介

はじめに

◆ ソフトウェアメトリクス

- ◆ ソフトウェアやその開発過程・環境を測定対象とした定量的尺度
- ◆ 開発現場で積極的に活用されているとはいえない
- ◆ メトリクスに関する基礎知識やデータ分析で注意すべき事項を中心として解説

メトリクスとその分類

◆ プロダクトメトリクス

◆ プロセスマトリクス

◆ リソースメトリクス

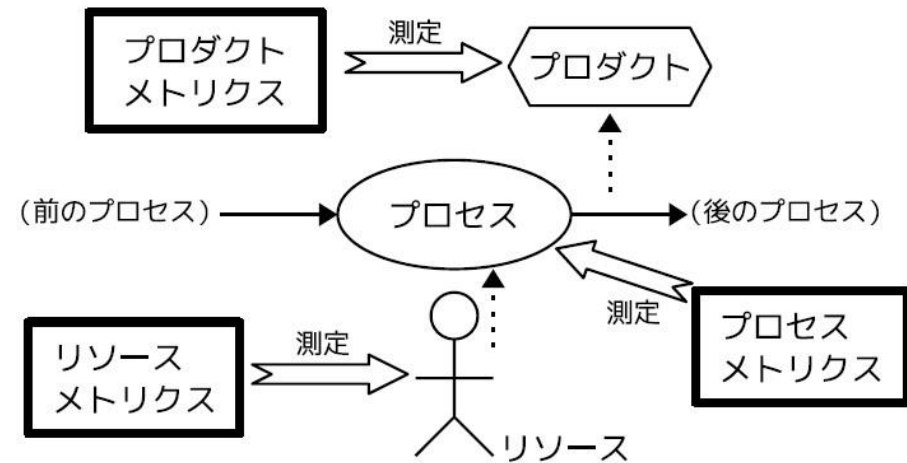


図 1 プロダクトメトリクス, プロセスマトリクス及びリソースメトリクスにおける測定の位置付け

◆ 重要な属性

◆ 尺度の種類

◆ 現場での制御可能性

プロダクトメトリクス

◆ソフトウェア開発のプロダクトを測定対象としたメトリクス

◆メトリクス例

プロダクト	ユースケース数, ファンクションポイント, LOC, サイクロマティック数 (制御フローの複雑さ), Halstead メトリクス, パブリックインスタンスメソッドの数, インスタンス変数の数, 特殊化指数[48], CK メトリクス (WMC: 複雑さで重み付けされたメソッド数, CBO: オブジェクト間の結合度, RFC: クラスに対する応答集合の大きさ, DIT: 継承木の深さ, NOC: 子クラスの数, LCOM: クラスにおける凝集度の欠如; バリエーションがいくつかある[15]), メソッド呼出しの数, クラス内に属性等の型として登場するクラス数・種類数 (クラス間結合度)[46], モジュールのファンイン・ファンアウト数, 情報フローの複雑さ (Henry-Kafura メトリクス), コードクローンの個数・最大長・割合[10],[57], コメント文記述の密度・頻度[5],[7], テストケース数, テストケースの網羅率, 欠陥密度
-------	--

◆得られた情報を解析することで, 品質の評価・予測を行う

プロセスメトリクス

◆ソフトウェアサイクルの各フェーズにおける活動を測定対象としたメトリクス

◆メトリクス例

プロセス	工数, 工期, 生産性, 顧客との対話時間, インタビュー実施回数, 開発開始後の仕様変更回数・変更率, ソースコードの変更回数・変更率, プロトタイピング実施の有無, 静的解析ツールの使用・不使用, コードレビュー実施の有無, レビュー効率, 欠陥除去率
------	--

◆活動の定量的な把握・分析を通じて状況判断や評価, 見積もりに役立てられる

リソースメトリクス

◆ソフトウェア開発作業において投入・消費される資源を測定対象としたメトリクス

◆メトリクス例

- ◆開発要員数
- ◆開発要員の作業単価
- ◆開発要員のスキルレベル
- ◆開発用マシンのスペック
- ◆ネットワーク環境の品質及びコスト
- ◆コミュニケーションツールの品質及びコスト

尺度の種類1

◆メトリクスによって得られた値がこういった尺度の下で得られたものであるのかを確認する必要がある

→誤った演算・処理をしないため

表 2 主な尺度とメトリクスの例

尺度	表現可能な情報	測定値に許容される処理 (各行, それより上の処理を含む)	メトリクス例
名義尺度	対象間での 等価性	頻度の計上 (頻度, 最頻値)	使用言語を “C: 1, C++: 2, Java: 3, COBOL: 4, VB: 5, その他: 6” で表現
順序尺度	対象間での 順序性	大小比較, 順位付け (中央値, 順位相関, パーセンタイル)	ソフトウェア故障に対してその深刻度を “致命的: 4, 重大: 3, 普通: 2, 軽微: 1” で表現
間隔尺度	対象間での差	加算, 減算 (平均, 標準偏差, 積率相関)	——
比率尺度	対象間での比率	乗算, 除算 (幾何平均, 変動係数)	コードの長さを物理行数で表現

尺度の種類2

◆ 名義尺度

- ◆ 対象物に数値を名前として割り当てるもの

- 値の大きさに意味は無い

- 例：学籍番号

◆ 順序尺度

- ◆ 対象物の間で順序関係が成立するように数値を割り当てるもの

- 違いの大きさは反映されない

- 例：故障の深刻度“致命的：4，重大：3，普通：2，軽微：1”

尺度の種類3

◆間隔尺度

- ◆順序尺度の性質を満たし、なおかつ値の差にも意味のあるもの
→値の比には意味が無く、除算は伴う処理は難しい
例：摂氏や華氏による温度測定

◆比率尺度

- ◆間隔尺度の性質を満たし、なおかつ値の比にも意味のあるもの
→通常の四則演算が可能で、どのような数値モデルにも利用可

現場での制御可能性

- ◆メトリクスで得られる情報が“観測可能なだけ”か“制御可能でもある”のか
- ◆メトリクスを品質管理に活用しようとする場合考慮しなければならない
- ◆制御できない量との相関関係を知見として与えられても活用できない場合がある

対象属性の明確化

◆測定の際，測定対象物のどの属性を測定しようとしているのかを明確にしなければならない

◆例：モジュールの品質を測定したい

→欠陥密度 = $\frac{\text{検出された欠陥数}}{\text{モジュール規模}}$ という形で表す

◆GQMアプローチが有効

◆目的(Goal)→質問(Question)→メトリクス(Metric)

数理モデル1

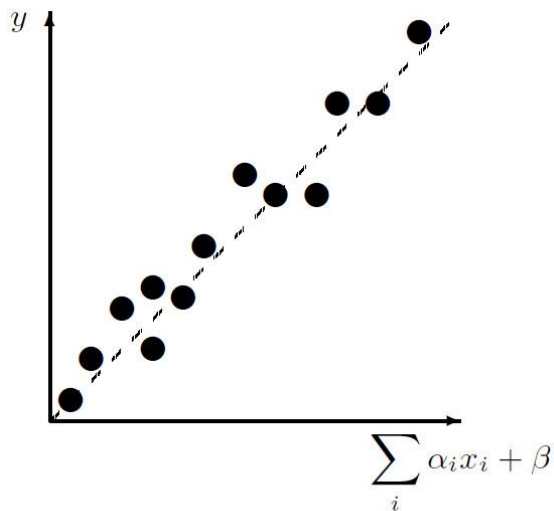


図 2 重回帰モデルのイメージ図

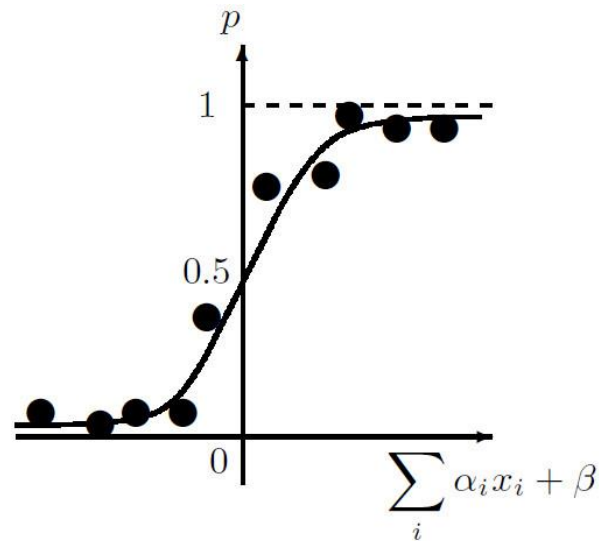


図 3 ロジスティック回帰モデルのイメージ図

◆ 重回帰モデル

◆ N個の変数 x_i の値を使って別の変数 y の値を予測する

◆ $y = \sum_i^N \alpha_i x_i + \beta$

◆ ロジスティック回帰モデル

◆ $\log \frac{p}{1-p} = \sum_i^N \alpha_i x_i + \beta$, $p = 1/(1 + e^{-z})$, $z = \sum_i^N \alpha_i x_i + \beta$

数理モデル2

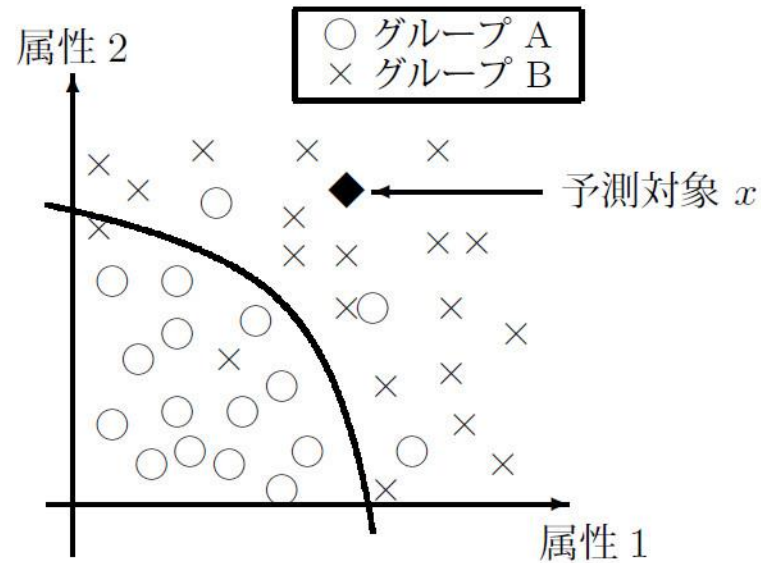


図 4 判別分析モデルのイメージ図

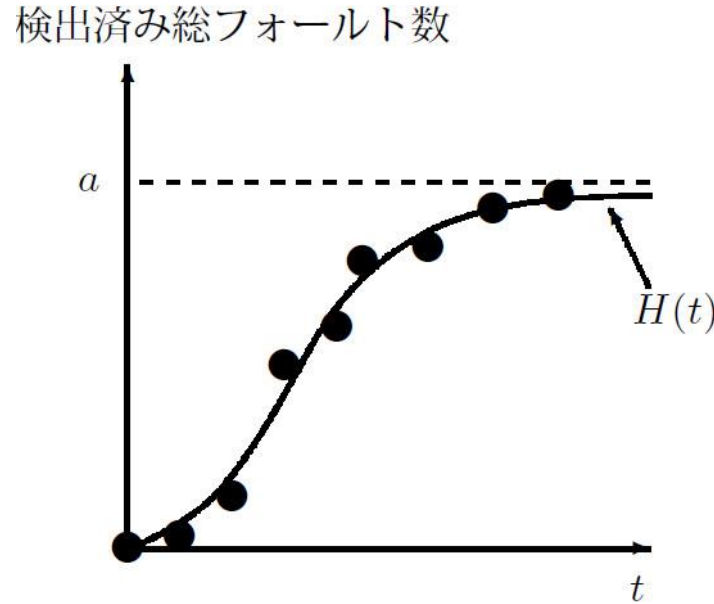


図 5 成長曲線モデルのイメージ図

指数型ソフトウェア信頼度成長モデル

$$H(t)=a(1-e^{-bt})$$

t:時刻

a:総フォールト数

b:一個当たりのフォールト検出率

H(t):tの時点での検出済みフォールト数

◆判別分析モデル

◆N次元空間上でデータが属するグループを判別する

◆成長曲線モデル

◆フォールトの検出数などを時間の経過を考慮してモデル化する

数理モデル3

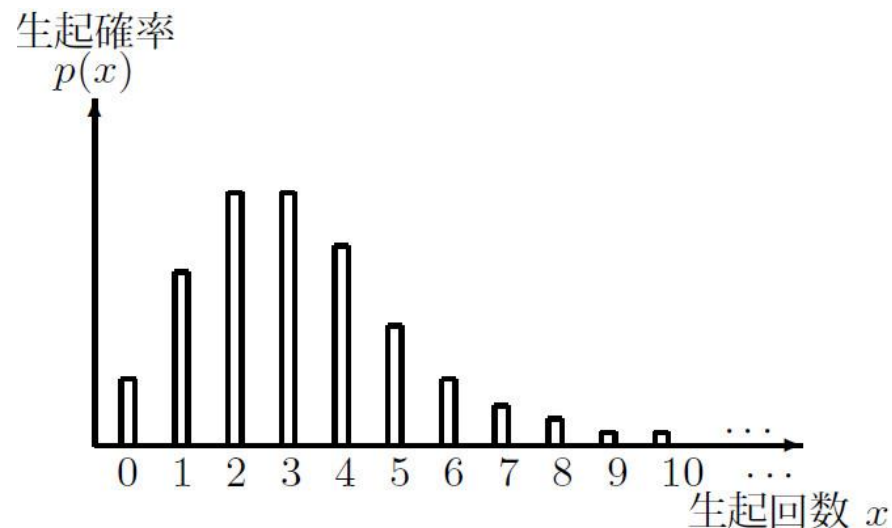


図 6 ポアソン分布の例

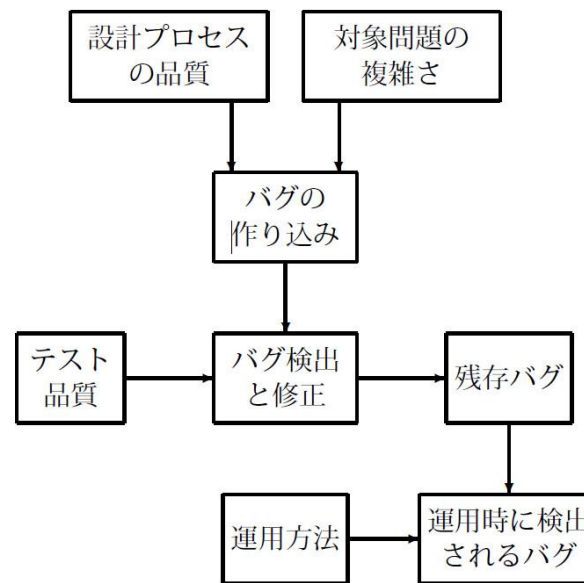


図 7 ベイジアンネットワークの例
(文献 [24] より抜粋; 著者らで表現を一部変更)

◆ 計数データの解析モデル

◆ 計数データを対象とした統計解析モデルの活用(ex:ポアソン回帰モデル)

◆ ベイズ統計モデル

◆ 推定したいパラメータが何らかの確立分布(事前分布)に従うと仮定して, 特定のインスタンスに関するパラメータの確立分布(事後分布)を推定するモデル

関連ツール

表 4 メトリクス測定ツール (一部)

名称	概要	メトリクス			
Eclipse Metrics Plugin [23]	統合開発環境 Eclipse [22] のプラグイン。開発中のコードを測定し、測定値を適正範囲の観点から可視化。	LOC, サイクロマティック数, フィールド数, CK メトリクス (一部) 等。	DependencyFinder [83]	Java のソースコード向けの測定アプリケーション (入力にはコンパイル済みのバイトコードを使用)。クラス間、メソッド間での依存関係を可視化。	メソッド呼出し、他クラスの使用といった各種の依存関係の件数等。
CCCC [47]	C 及び C++ ソースコード向けの測定アプリケーション。ソースファイルを測定し、測定値を適正範囲の観点から可視化。	LOC, サイクロマティック数, コメント行数, ファンイン・ファンアウト数, Henry-Kafra メトリクス等。	CCFinderX [42]	C, C++, C#, Java, VisualBasic, COBOL のソースコード向けのコードクローン [43] 検出アプリケーション。指定されたフォルダ内のソースファイルを測定し、コードクローンの位置情報と関連する測定値の一覧表を生成。	ソースファイルにおけるコードクローンの占有率, コードクローンの長さや個数等。
SourceMonitor [17]	C, C++, C#, Java, VisualBasic 等のソースコード向け測定アプリケーション。指定フォルダ内のソースファイルを測定し、測定値の一覧表を生成。	LOC, 文の数, コメント密度, 実行パス数, ネスト数, メソッド呼出し数等。	EPM [21]	開発プロジェクトにおけるデータの自動収集と分析を支援するツール。ソースコードだけでなく、障害検出やメール投稿といったアクティビティについてもデータ収集が可能。	LOC, 障害検出数, メール投稿数等。

◆測定ツール

◆作業や処理が煩雑であったり、データ量が膨大であったりするため

◆データ解析ツール

◆複雑な統計処理や大量のデータを整理・解析するため

まとめ

◆ 以下の内容を通じてソフトウェアメトリクス基礎について触れた

- ◆ メトリクスの種類
- ◆ 尺度の種類
- ◆ 現場での制御可能性
- ◆ 対象属性の明確化
- ◆ 数理モデル
- ◆ 関連ツール

◆ 目的や対象, 属性を明確にし, 適切にデータを分析することが効果的なメトリクスの活用につながる