

先進ソフト開発プロジェクトへのEPMの適用と分析について

奈良先端科学技術大学院大学
産学官連携研究員
松村知子

2005年6月23日
エンピリカルソフトウェア工学研究会(東京・田町)

1

本活動への期待

- 従来のEPM適用方法
 - 障害データ項目・各ツール運用方法は各社の既存開発プロセスに依存
 - プロジェクト終了後の事後分析と評価
- 先進プロジェクトにおけるEPM適用の方針
 - 分析を目的としたデータ項目の収集
 - 分析目的にあったツール等の運用方法の導入
 - プロジェクトの進行に並行したデータ分析
 - プロジェクトへの分析結果のリアルタイムフィードバック

分析に有用なデータ収集が可能になり、結果をリアルタイムに活用することによって、データ分析による開発プロセス改善効果を逐次評価できる

EPM導入手順

何を目的とするか
何を知りたいか

何を集めるか

どのように集めるか

構成管理ツール
障害管理ツール

分析案の作成・提示

↓

収集データ項目の決定

↓

ツール運用方法の決定

↓

データ収集環境・
管理方法・分析方法...

参考資料・情報
決定方法

IEEE std.982*
PSP**
EASE分析案

分析案
各社障害票
IEEE std.1044***

分析案
各社構成管理方法
ケース別対処方法検討

*IEEE Std.982-1988 IEEE Standard Dictionary of Measures to Produce Reliable Software
 **Personal Software Process: W.S. Humphrey, 'A Discipline for Software Engineering', Addison-Wesley, 1995
 ***IEEE Std.1044-1995 IEEE Guide to Classification for Software Anomalies

分析案の作成

- IEEE Standard 982.1-1988:ソフトウェア信頼性の尺度の適用
- PSP:レビュー結果発見される欠陥(defect)の分析の適用
- EASE分析案(暫定)

4

IEEE ソフトウェア信頼性の尺度(1)

- 39尺度を実績に応じてレベル0~3に分類
- レベル2以上で、一般的なレビュー議事録・障害管理データから測定可能な5尺度を選択
- Fault Density
 - 残バグ数の予測
 - 十分なテストが行われたかの検証
 - Fault-Density(バグ密度)の標準の確立
- Defect Density
 - 新規開発および大規模な修正に対する設計・コードレビュー結果を定量的に分析し、レビュープロセスの精査の指標とする

5

IEEE ソフトウェア信頼性の尺度(2)

- Man hours per major defect detected
 - 修正すべき主要な欠陥の発見にかかる工数から設計、レビュープロセスの効率を評価する
- Mean-time-to-failure (MTTF)
 - ほとんどのソフトウェア信頼性の基準として要求されるパラメータで、(i-1)番目とi番目の故障間の時間(CPU/時計)である
- Mean time to discover the next K faults
 - 次のK個のFaultが見つかるまでの時間平均を測定し、要求される信頼性レベルに達するまでの時間指標とする

6

PSP 欠陥摘出率・暫定的尺度

- 欠陥摘出率
 - レビュー品質を計測する尺度
 - 開発完了後、欠陥発生が収束した時点で計測
- 暫定的尺度
 - 開発中にレビューの質を測定する尺度。レビュー対象もしくはレビュープロセスの情報から得られる定量的データ。
 - 暫定的尺度例
 - KLOCあたりの発見欠陥数
 - レビュー時間あたりの発見欠陥数
 - 時間当たりのレビュー量 (LOC等)

7

EASE 障害対応コストの予測

- 各障害(故障・バグ)に対して、修正したCVS履歴上の情報・修正ファイルの情報を用いて特徴づけし、新たに発生する障害のコスト予測モデルを作成し、進捗・スケジュールの遅延予測・調整に用いる

8

EASE 障害原因予測

- 障害が混入した工程・成果物のバージョンから、障害の原因を予測し、障害混入(主に、デグレージョン)予防に用いる
- コードレビューや回帰テストの指標となる

9

EASE 障害発見平均時間

- 障害が混入してから発見されるまでの時間を、障害の種類毎に計測し、問題プロセスを検出して早期発見の改善を行う

10

EASE 障害対応進捗管理

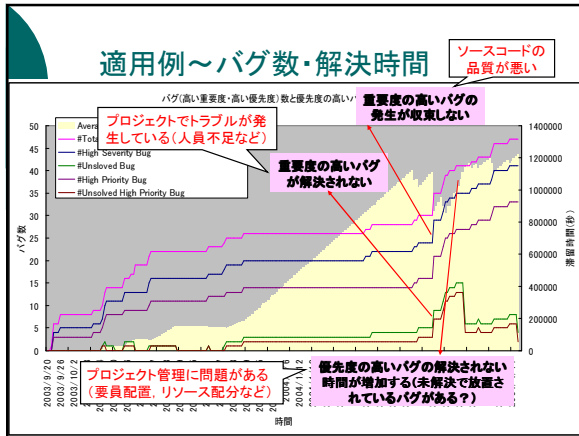
- 障害が発見されてからの対応状況を時系列でモニターし、障害対応・プロジェクト管理等の問題を検出する
- Prof.Basiliとの共同作成GQMモデル適用
 - Business Goal
 - EPMの収集データを用いて、プロダクトの品質報告を支援する
 - Measurement Goal
 - バグに関するGNATSデータを分析することによって、企業における特定のプロジェクトにおいて、プロジェクトマネージャの視点から、プロダクトの品質を評価する

11

障害対応進捗管理 GQMモデル

- Model
 - 重要度の高いバグ数の急激に増加している(累積報告数のグラフの傾きが急である)場合、ソースコードの品質が悪い
 - 解決していない重要度の高いバグ数が増加し続ける(未解決バグ数のグラフの傾きがいつまでもたっても0より大)場合、プロジェクトでトラブルが発生している(人員不足など)
 - 優先度の高いバグの平均滞留日数(平均解決日数)が増加し続ける(優先度の高いバグが未解決のままになる)場合、プロジェクト管理に問題がある(要員配置、リソース配分など)

12

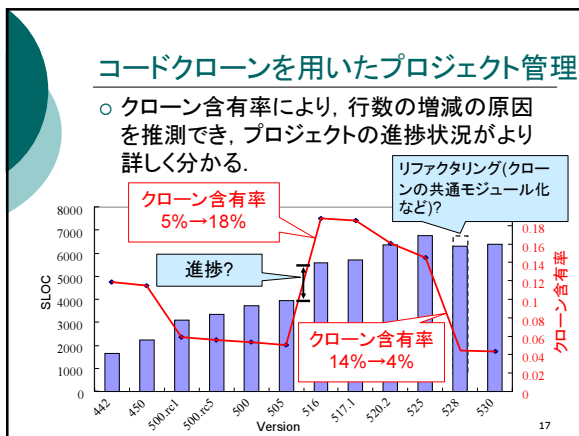
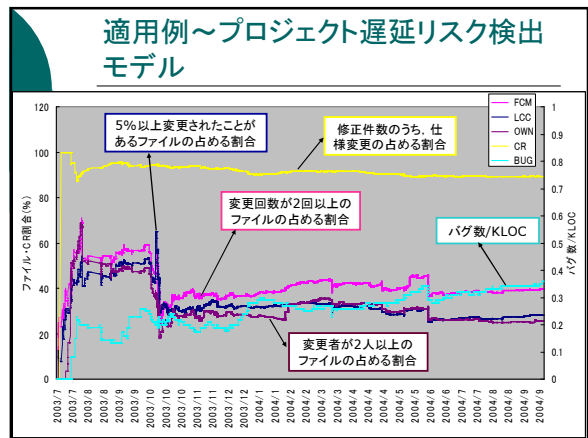


プロジェクト遅延リスク検出モデル

- 開発担当者へのアンケートに基づくプロジェクト遅延の3つの要因を抽出
 - 不安定な仕様/不完全な設計/劣悪なソースコードの品質
- Prof. Basiliとの共同作成GQMモデル適用
 - Measurement Goal: 構成管理データ(CVS)と障害管理データ(GNATS)のデータからファイル変更パターンを解析し、企業における特定のプロジェクトにおいて、プロジェクトマネージャの視点から、不安定な要求・不完全な設計・劣悪なソースコード品質について評価する

具体的モデルの検証と改善

- モデルの検証
 - (変更回数 ≥ 2) and (変更行数/ファイルサイズ $> 5\%$) and (仕様変更による変更 $> 60\%$) の場合, 要求が不安定である
 - (変更回数 ≥ 2) and ((変更行数/ファイルサイズ $> 5\%$) or (25%より多くのファイルの変更者数 ≥ 2)) の場合, 設計が不完全である
 - (変更回数 ≥ 2) and (不具合による変更数/KLOC > 10) の場合, コード品質が悪い
- メトリクス・閾値の改善・測定方法の工夫



Logical Couplingを用いた複雑化分析

- Logical Coupling: あるコード片の変更に対して、別のコード片も同時に変更しなければならないといった論理的結合関係
- ファイル単位のLogical Couplingの推移を監視することで、システム構造の複雑化やその原因の推定、再設計の必要性を判定する

バグ修正仕様変更

ファイル

同時変更関係

収集データ項目

- 収集可能データ
 - CVS(構成管理システム)レポジトリ
 - ソースコードを含む
 - 障害管理データ
 - 各社オリジナル障害票データ(単体試験まで)
 - EASE推奨GNATS標準データ(結合試験以降)
 - 問題記述票
 - レビュー議事録(欠陥記録)
 - メール
 - 社間メールのみ
 - メール記述内容を含む

19

EASE推奨GNATS標準データ

- 収集必須データ
 - 対応状況, **起票分類**, 責任者, 発見日, テスト対象, 発見工程, **発見バージョン**, 問題内容, **再現度**, **重要度**, **優先度**, 問題原因[詳細], 混入工程(**混入バージョン**), 問題を本来発見すべき工程, **修正工数**, 修正内容, 完了日(自動入力)
- 提案項目から変更・削除された項目
 - レビュー関連データ削除: レビューによって発見された欠陥は, 問題記述票で対応
 - 再現率: 再現度に変更(%の記入は困難)
 - 重要度: 5→3段階に変更

20

GNATS項目議論内容

- 起票分類: 仕様変更の取り扱い
 - 仕様変更は, 基本的に無いという前提で, もし発生しても, GNATSでは取り扱わない
- 発見バージョン: 何をどう記述するか?
 - 運用ルールと合わせて検討・提案
- 混入バージョン: 把握が困難なため, 必須項目から削除
 - 重要な項目なので, 別途検出方法を検討*
- 修正工数
 - 時間管理を行っていない企業があり, 記入が困難

*Jacek Silwerski, Thomas Zimmermann and Andres Zeller, "When Do Changes Induce Fixes?", Proc. International Workshop on Mining Software Repositories(MSR), May 2005, Saint Louis, Missouri, USA.

22

各ツール運用方法

- 運用方法策定目的
 - 構成管理と障害管理, (メール情報)を関連付けた分析を行う
 - 粒度のそろったデータを収集する(各社における既存の運用方法が異なる)
- 運用方法検討対象ツール
 - 構成管理ツール(CVS)
 - 障害管理ツール(GNATS)
 - メール

22

CVS運用原則(1)

- コミットを変更要求単位(障害票単位)に行う
 - 複数のバグを1度に修正してコミットしない
 - 1つの変更要求に対するコミットは, 複数ファイルを変更する場合も1コミットとする
 - コミット時にコメントを必ず付加する(フォーマットを統一)

コメントフォーマット

```

CVS: -----
CVS: 管理種別: 登録
CVS: 障害番号: 3
CVS: 開発者ID: e-iwamura
CVS: 修正内容: 位置データ用のバッファ領域を
CVS: 50→100に拡張
CVS: -----
    
```

障害票に登録された変更 / 登録なしの変更

障害票番号

23

CVS運用原則(2)

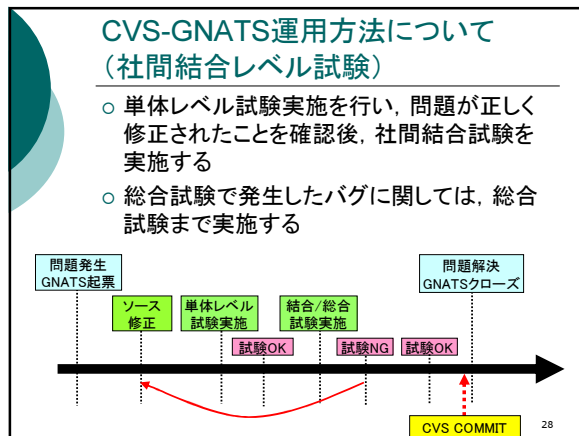
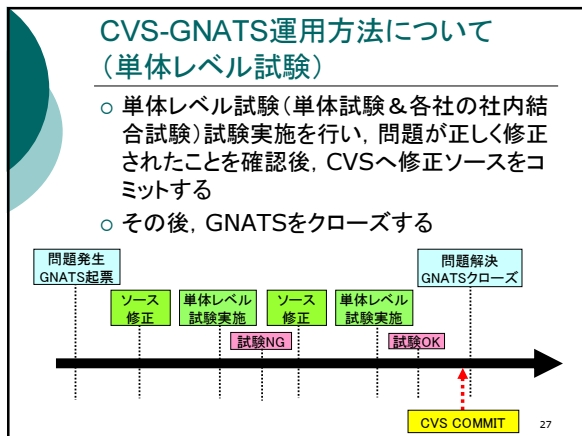
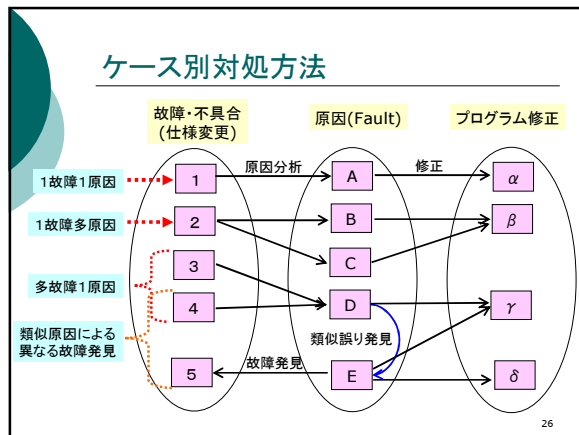
- ブランチを作らない
- 変更のないファイルを更新(上書き)しない
- テスト用オブジェクトの構築
 - CVS TAG機能を使用して, テスト用オブジェクトを識別できるようにする
 - 障害発生時は, TAG番号を「発見バージョン」に記入する

24

GNATS運用原則

- 本プロジェクトにおける問題管理は、「原因 (Fault)の分析」に焦点を当てる
 - 原因単位で分析可能な運用を行う
 - 原因間に何らかの関連性がある場合、その関連を抽出できるようにする(論理的結合関係 [Logical Coupling]の抽出を可能にする)
- 可能な限り、開発作業を阻害しない運用とする
 - 不自然な作業の発生を要求しない(問題票の再起票・分割等)

25



メール運用原則

- 社間結合試験以降の障害対応状況の補完情報として参照する
- 障害毎に区別できるメールタイトルとする
 - GNATS(障害)番号があれば、それをタイトル内に記す
- 返信はReply Allとし、同一案件は同一タイトルのチェーンメールとなるようにする
 - 別の話題になった場合は、速やかにタイトルを変え、別スレッドを立てる

29

今後の予定

- 各社からの問題記述票(レビュー議事録)の分析(5月中旬～)
- 各社内製造・単体試験データの収集と分析・進捗会議での報告(7月中旬～)
- 社間結合試験へ向けてのデータ収集・分析環境の整備(～8月下旬)
- 社間結合試験工程でのデータの収集と分析・進捗会議での報告(8月下旬～)

30

