

OSS開発におけるテストコードの保守頻度 が与える欠陥混入への影響

坂口英司, 伊原彰紀, 松本健一(奈良先端科学技術大学院大学)

目的

見逃し欠陥をリリース前に検出する技術の開発

リリース後に検出された 欠陥のこと



次 欠陥予測:見逃し欠陥の位置を特定する

テスト自動生成:見逃し欠陥を検出する テストケースの生成

- ソフトウェアは日常生活と強く結びついており、不具合が発生すると多く の人に影響を与える.
 - 例:カンタス航空の旅客機が飛行操縦システムの不具合により急降 下し,多数の負傷者が出た.
- ソフトウェアによっては、製品出荷後の欠陥修正は容易ではない。
 - 例:トヨタ自動車は、制御システムの不具合が原因で30万台以上の 自動車のリコールを発表した.

開発者にとってもユーザにとっても、 見逃し欠陥の検出は重要である.

- 本研究の貢献:
 - 見落とされやすい欠陥の特徴を理解することができる

既存の欠陥予測技術との違い

テスト工程で全欠陥を検出することは不可能である. その理由として,

- 1. 開発工数が有限であり、全ての実行パターンを検証できない
- 2. 人手でテストコードを書くため、検出漏れが生じる

が挙げられる. 既存の欠陥予測技術は1を解決することが目的であったが, 本研究では2を解決することを目的としている. ヒトが見落としやすい欠陥 の特徴を明らかにすることで、より高品質なソフトウェア開発を目指すこと ができる.

保守頻度の実態調査

実態調査:

テストコードを持つプロダクトコードからどのくらいの欠陥が検出されるか.

調査対象:

Apache HTTP Client プロジェクト

- 調査手法:
 - 1. Java 言語を用いて開発を行っている,調査対象プロジェクトのGit リ ポジトリをクローンする
 - 2. 任意の期間のコミットログを取得し、プロダクトコードとテストコードの コミット履歴を取得する.
 - 3. テストコードの import 文を基にして、テスト対象のプロダクトコードを 特定する.
 - 4. 各プロダクトコードに ID を割り当てる(各テストコードには, テスト対象 コードと同一のIDを割り当てる).
 - 5. SZZ アルゴリズムを用いて、欠陥が混入したプロダクトコードと欠陥が 混入したコミットを明らかにする.
 - 6. 各プロダクトコード/テストコードのコミット履歴と欠陥の混入情報をま とめ、可視化する.

結果と今後の方針

- 調査結果:
 - ソフトウェアから検出された欠陥のうち、26.6%が見逃し欠陥(テスト 対象のプロダクトコードから検出された欠陥)である(図1).
- 今後の方針:
 - 見逃し欠陥の特徴抽出
 - 見逃し欠陥予測モデルの構築
 - 見逃し欠陥検出に特化したテストケース自動生成技術の開発

図1. ソースコードが更新され、欠陥が混入する様子 (左側がプロジェクト全体,右側が調査対象の図を拡大したもの) 知見1: 欠陥が集中的に混入して テストコードを持つが、欠陥が いる時期がある 混入したプロダクトコード (調査対象) テストコードを持ち、欠陥が_ 知見2:プロダクトコードが頻繁に 混入しなかったプロダクトコード 変更されている 知見3:テストコードの更新頻度が テストコードを持たず、欠陥が 混入したプロダクトコード 低いと、頻繁に欠陥が混入する 各点は, プロダクトコードの変更(緑) テストコードの変更(青) 欠陥の混入(赤) テストコードを持たず、欠陥が を表す 混入しなかったプロダクトコード

謝辞:本研究の一部は,頭脳循環を加速する戦略的国際研究ネットワーク推進プログラムによる助成を受けた.



NARA INSTITUTE of SCIENCE and TECHNOLOGY