

研究進捗状況

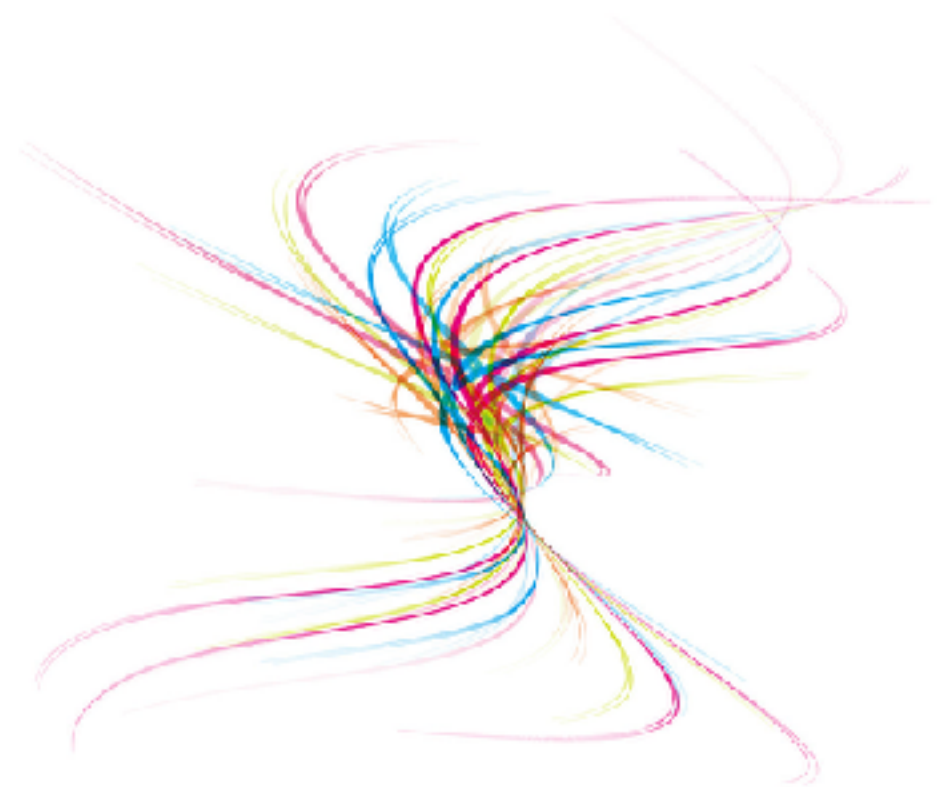
2018年5月21日（月）

発表者 吉富 楓雅

目次

- 研究のイメージ
- 分析手法
- 分析結果
- 今後の予定

研究のイメージ



ソフトウェアテストとは

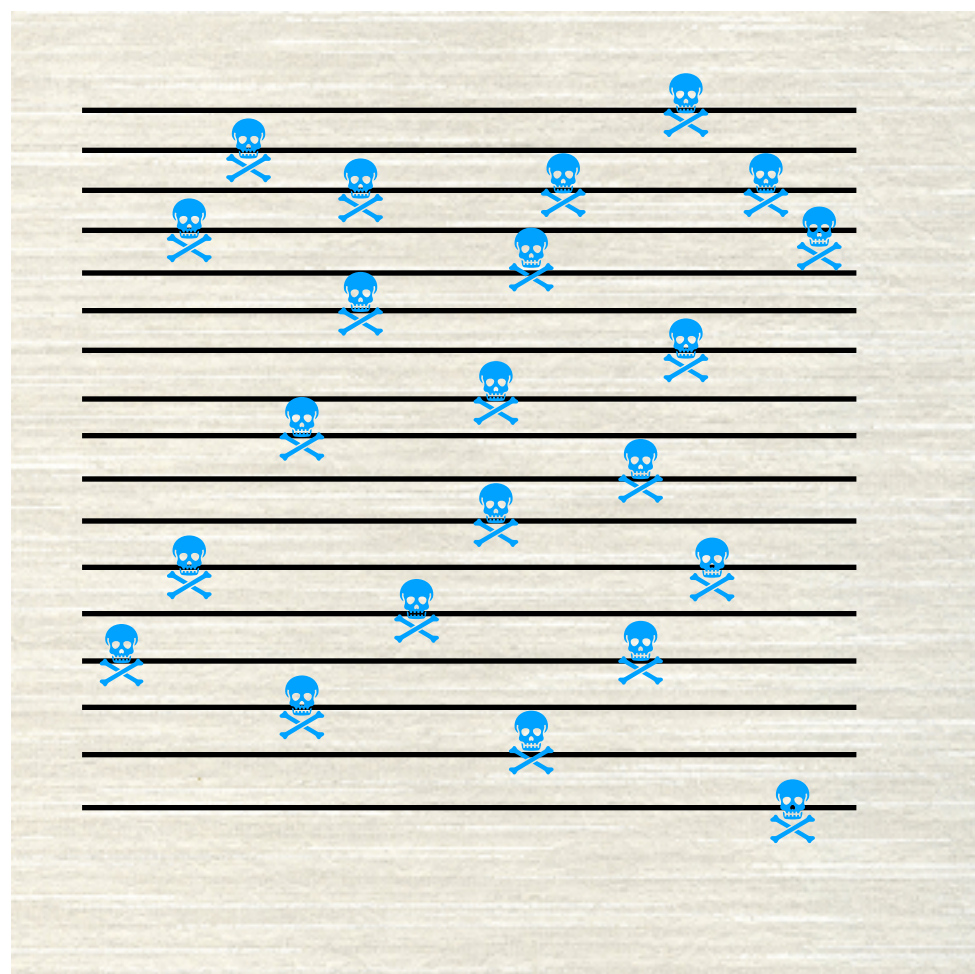
- ソフトウェアテストとは

- *完成したソフトウェアが正しく動作するか, 欠陥が混入していないかを確認するための作業
- *オープンソースソフトウェア開発における品質保証活動は, 公開されたソースコードを不特定多数の開発者が検証, 報告し, 欠陥を修正する形態が主流
- *テスト活動に関心のある開発者の不足やソフトウェアテストを熟知している開発者が不足[1][2]

- [1] Beller, M., Gousios, G., Panichella, A. and Zaidman, A.: When, How, and Why Developers (Do Not) Test in Their IDEs, *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2015*, pp. 179–190 (2015).
- [2] Tomotaka Minami :Master's Thesis, Department of Information Science, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-MT1551105, March 16, 2017.

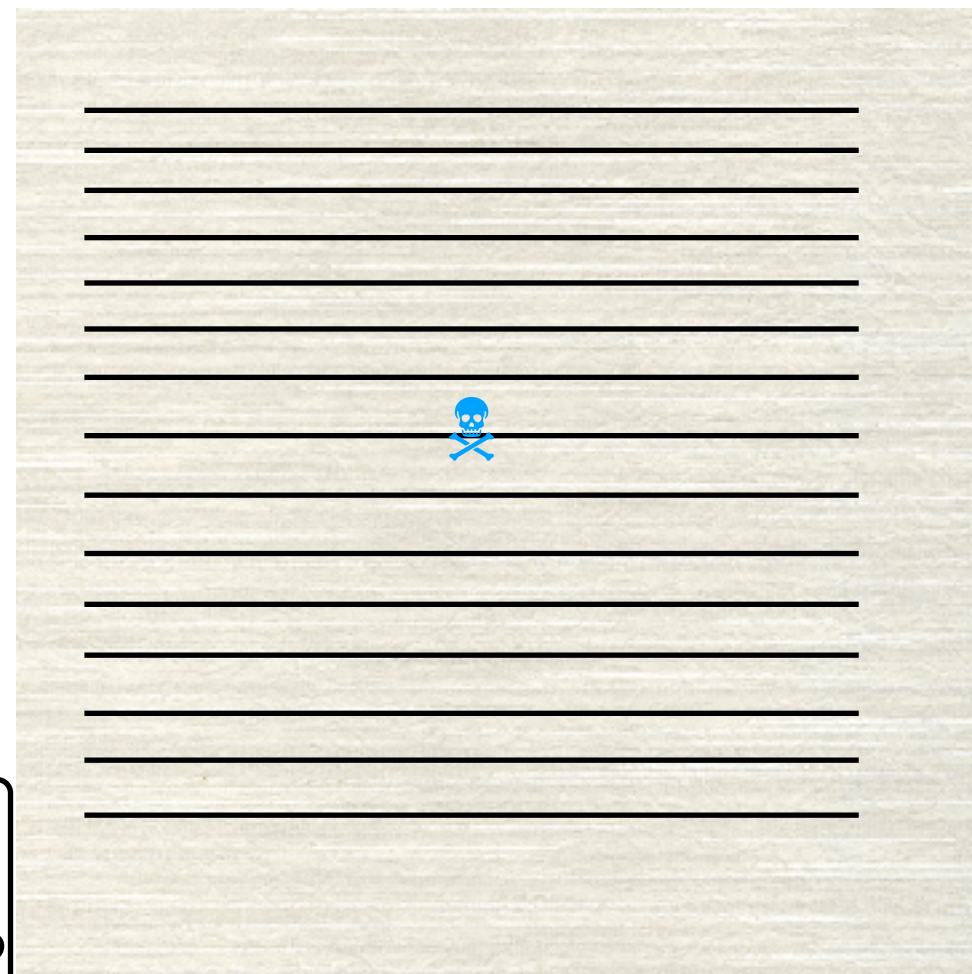
テスト時の欠陥数

プログラム生成時



1行に1.5個

テスト担当者に渡った時



99%は
発見される

100行に1個程度

テストに対しての周囲の期待



社員Aさん

プログラミングテキストに
完全にテストする
方法が書いてありました。
全パス網羅だったら
完璧ですね。

部下に完全にテストする
ように言いました。
外部に対して保証が
できますからね。



プロジェクト
マネージャー
Bさん



顧客

ソフトウェアテストを請け
負う会社のパンフレットに
全てのコードをテストする
と宣伝していた。

テスト担当者の現実



テスト担当者Cさん

完璧にやろうと思っても
バグがどんどん出てくるし。
しかも周りが完全完全って騒ぐから
めっちゃストレス溜まるんやけど。
どないなっとなねん！！！！

テストに対する周囲の過度な期待によるストレスと、
テストがもれなく記述できているかということに対しての不安

テストコードの発達

- 様々なオープンソースプロジェクトの開発が進むなか、多くのプロジェクトでは、大量のテストコードが開発中にプロダクトコードとともに進化している[3]
- プロダクトコードとテストコードの比が1:1とも言われている[4]
- つまり、プロダクトコードが開発されるにつれ、テストコードにもバグが混入する可能性が高まる

- [3] A. Zaidman, B. Van Rompaey, A. van Deursen, and S. Demeyer, “Studying the co-evolution of production and test code in open source and industrial developer test processes through repository mining,” *Empirical Software Engineering*, 16(3), pp. 325-364, June 2011.

- [4] A. van Deursen, L. Moonen, A. van den Bergh, and G. Kok, “Refactoring test code,” 2nd Int. Conf. on Extreme Programming and Flexible Processes in Software Engineering (XP 2001), 2001.

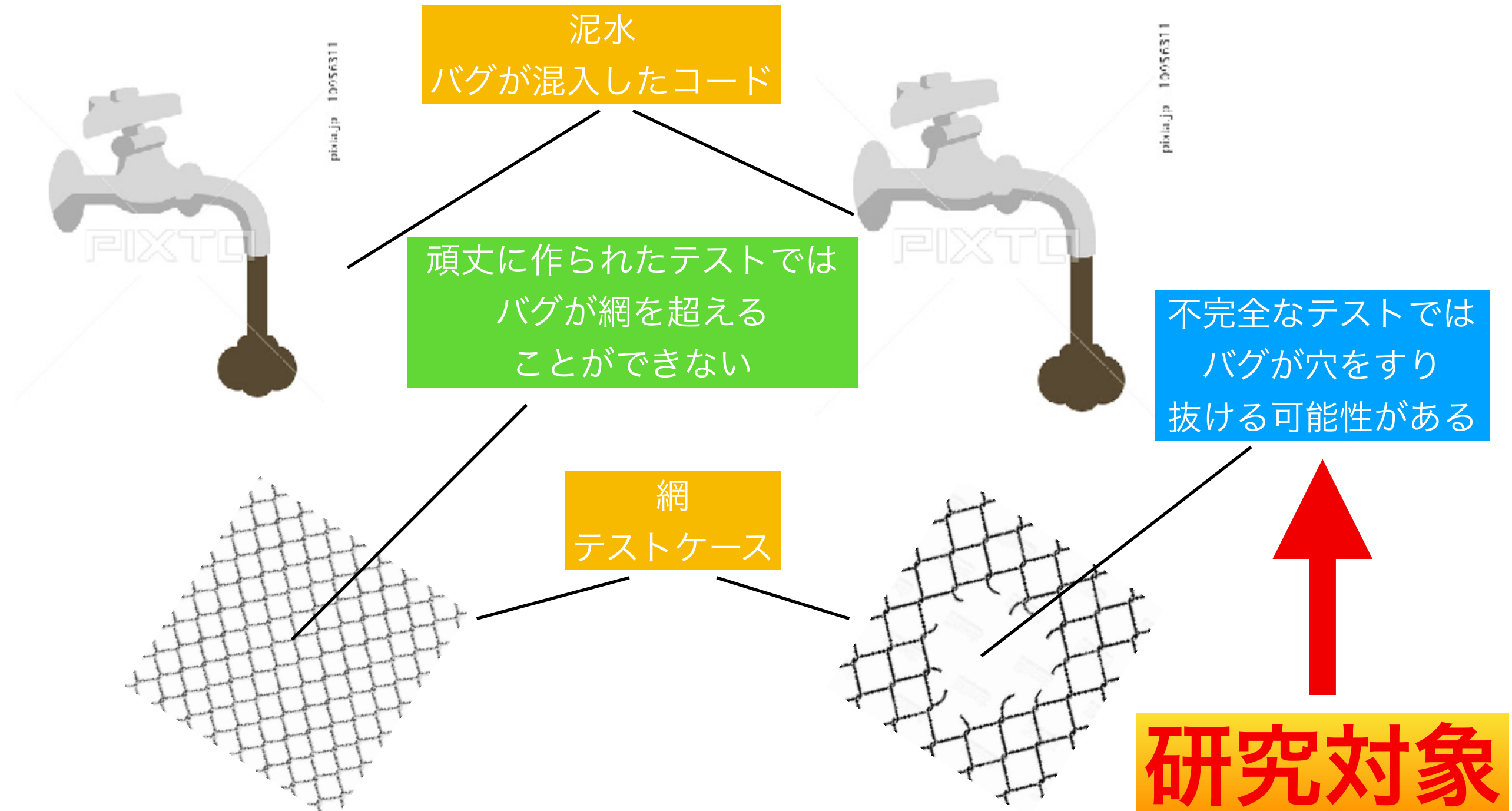
テストはどうやって保証するか？

- プロダクトコードに対して、テストコードは一般的に1～2倍の量を記述する
- テストが適切であるかを調べるためには、この2倍の量のテストを行わなければならないのか？
 - * 現場において、そのようなことは不可能

最悪のケース

- テストコードの誤りによってプロダクトコード内のバグの見逃しが起こること
 - * 企業においても、プロダクトコードに対して書かれた膨大な量のテストコードを誰がテストするのかについて疑問視されている

研究のイメージ



研究の目的

- 本研究では, テストケースの誤りによってプロダクトコードのバグが見逃されている事例が存在するのかを検証
- どのような事例の時に発生しやすいのかについて分析を行う

研究による展望

- 企業などでソフトウェアテストをする際に,テストにおける最悪の失敗を事前に知っておくことが可能となる
 - * テスター自身が安心してテストを作成することができる
と予想される

分析手法(5月13日地点)



データのサンプリング

- 分析対象



- * Mybatis-3

- ▶ **MyBatis** :カスタム SQL, ストアドプロシージャ, 高度なマッピング処理に対応した永続化フレームワーク

- 150のテストデータをサンプリング

- * 条件: バグIDに紐づけられており, プロダクトコード・テストコード共に修正あるいはコードの追加が見られるもの

分析方法

- Mybatis-3のバグIDに対応しているコミットを検出
- 全てのコミットの中からバグIDに対応しているもののみを抽出,githubで分析
- このうちテストコードに変更・修正が見られるものを抽出対象とする

分析方法

- 目視で確認する

- * 分析項目

- ▶ バグID

- ▶ コミットID

- ▶ テストコード・プロダクトコードのどちらから先に変更されたのか？

- ▶ テストコードの誤りによってプロダクトコードの誤りが見逃されている可能性があるか？

分析結果(5月13日地点)

RESULTS





プロジェクトコードとテストコードのバグ修正検証 2 — 自分の Mac に保存済み

ホーム 挿入 ページ レイアウト 数式 データ 検閲 表示 開発 テーブル

文字列の繰り返し セルを結合して中央揃え

標準

条件付き書式 テーブルとして書式設定 セルのスタイル

挿入 削除 書式

並べ替え/フィルター

N37

	I	J	K	L	M	N	O	P
1								
2								
3								
4								
5	ファイルパス	ファイル変更はあるか?	テスト変換はあるか?	変換点	コードのどちらから先にできたよってバグが見逃されて	コメント		バグレポート
6	src/test/java/org/apache/ibatis/submitted/foreach/ForEachTest.java	○	○	コードの追加	テストのメッセージの不具合と 考えられる	恐らくない	このケースは追加のみで、テストケースを単に追加しただけ だと考えられる	エラープロンプトはあまり明確で メッセージはあまり明確ではなく、
7	src/test/java/org/apache/ibatis/submitted/keygen/Jdbc3KeyGeneratorTest.java	○	○	修正とコードの追加	テストからテストが生まれた (?)	ない	このケースはテストケースを単に追加しただけだと考えられ る	keyPropertyが見つからない 発生させる
8	src/test/java/org/apache/ibatis/reflection/MetaObjectTest.java	○	○	修正とコードの追加	?	?	?	?
9	src/test/java/org/apache/ibatis/submitted/cache/CacheTest.java	○	○	コードの追加	単にテスト内の変更を行ってい るだけ	ない	このケースはテストケースを単に追加しただけだと考えられ る	Add properties attrib '@CacheNamespa
10	src/test/java/org/apache/ibatis/submitted/blocking_cache/BlockingCacheTest.i	○	○	修正とコードの追加	eclipse / sonarlintにおける 問題を解決したものであるの	ない	eclipse / sonarlintにおける問題を解決したものである と考えられる	Update logging issues eclipse/sonarlint「ec
11	src/test/java/org/apache/ibatis/submitted/enum_interface_type_handler/Color	○	○	テストはコードの追 加	恐らくコード変更によってテスト 変更が行われたと考えられる	?	型の引数に異常が見られたことによる変更と考えられる	バージョンアップに伴ってハンド ルの処理要求が発生した時に
12	src/test/java/org/apache/ibatis/mapping/BoundSqlTest.java	○	○	修正とコードの追加	テストのメッセージの不具合と 考えられる	恐らくない	このケースは追加のみで、テストケースを単に追加しただけ だと考えられる	エラープロンプトはあまり明確 ん!
13	src/test/java/org/apache/ibatis/submitted/sqlprovider/OurSqlBuilder.java	○	○	修正とコードの追加	?	?	?	?
14	src/test/java/org/apache/ibatis/submitted/cache/CacheTest.java	○	○	修正とコードの追加	単にテスト内の変更を行ってい るだけ	ない	このケースは追加のみで、属性を単に追加しただけだと考 えられる	@CacheNamespaceRef を追加する
15	src/test/java/org/apache/ibatis/binding/MapperMethodParamTest.java	○	○	テストはコードの追 加	単にテスト内の変更を行ってい るだけ	パラメータの誤りによっ て、テストに不具合が生	"size"パラメータに不具合がある。	パラメータ名が "size"の場合、 マップされます
16	src/test/java/org/apache/ibatis/submitted/nestedresulthandler/NestedResultHa	○	○	importの削除	?	?	?	3.2.x release: NestedResultSetHandler
17	src/test/java/org/apache/ibatis/type/Ty	○	○	テストはコードの追 加	単にテスト内の変更を行ってい るだけ	恐らくない	型ハンドラを使用しているが、リストアップが不足しているも のと考えられる	Allow registering a type for a common interface

Mybatis-3

コマンド

125%

分析結果

- テストコードの誤りによってバグコードが見逃されているものは確認できなかった
 - * コードと開発者同士の会話しか見ていなかったため
- 今回のバグレポートは、プログラムの仕様に対してのコード変更が多かった
 - * 例：エラープロンプトの明確性・バージョンアップによる修正・Mybatisのプログラムで使用できない項目の削除など

今後の予定



今後の分析方法について

- REOPENされているものを探す



分析項目の追加

- 分析項目の追加
 - * テストの種類
 - * テスト欠陥の種類

15種類のバグ分類

- A. アルゴリズムの不首尾 (an algorithm awry)
- B. 不手際または台無し (a blunder or botch)
- C. 一貫性または明快さのための美文書化 (a clean-up for consistency or clarity)
- D. データ構造の崩壊 (a data structure debacle)
- E. 効率の増強 (an efficiency enhancement)
- F. 忘れられた機能 (a forgotten function)
- G. 一般化または能力の成長 (a generalization or growth of ability)
- I. 対話性の改善 (an interactive improvement)
- L. 言語の責任 (a language liability)
- M. モジュール間の不適切な組合せ (a mismatch between modules)
- P. 移植性の増進 (a promotion of portability)
- Q. 品質の追求 (a quest for quality)
- R. 堅牢性の強化 (a reinforcement of robustness)
- S. 想定外のシナリオ (a surprising scenario)
- T. ちょっとした誤植 (a trivial typo)