

Interactive Code Review for Systematic Change

Tianyi Zhang, Myoungkyu Song, Joseph Pinedo, Miryung Kim
2015 IEEE/ACM 37th IEEE International Conference on
Engineering

背景と目的

- ソフトウェア開発でコードレビューはコストの高い作業
 - コードの変更を追うのは大変
 - バグの見落とし
 - 変数名等の変更の漏れ
 - スキルの高い開発者に負担が集中する
- CRITICS(コード変更チェックツール)を開発したので実際に使えるのか試してみる

アプローチ

- Salesforce.com社の開発者6人に使用した感想を聞く
- テキサス大学の学生12人にCRITICSとEclipseDiffの両方を使って比較した結果と感想を聞く

CRITICS

- コード変更チェックツール

The screenshot displays the Eclipse IDE interface with several panels illustrating different aspects of code change checking:

- Compare ('NEW_WIN3213516' - 'OLD_WIN3213515'):** Shows a side-by-side comparison of Java source code. A red annotation states: "① A user selects a sub-region of a diff patch, using Eclipse compare view." The code in the 'NEW_WIN3213516' pane includes a `case OS.VK_RETURN:` block with `Event event = new Event ();`, `event.item = focusItem;`, `sendEvent(focusItem.expanded ? COLLAPSE : EXPAND, event);`, and `refreshItem(focusItem);`.
- Diff Template View (New Rev.):** Displays an Abstract Syntax Tree (AST) diagram. A red annotation states: "② A side-by-side view of AST edits and context." The diagram shows a `METHOD` node containing a `SWITCH_STATEMENT`, which branches into `ASSIGNMENT`, `METHOD_INVOCATION`, and `METHOD_INVOCATION`. A `VARIABLE_DECLARATION_STATEMENT` is also shown.
- Diff Template View (Old Rev.):** Displays another AST diagram, similar to the one in the 'New Rev.' view, showing the state of the code before the changes.
- Matching Result:** Shows the locations of systematic changes matching the abstract diff template. A red annotation states: "③ Locations of systematic changes matching the abstract diff template". The matching locations are listed as `[Eclipse SWT/win32/org/eclipse/swt/widgets][Widgets.java][keyPressedEvent]`.
- Change Anomalies:** Shows locations of change anomalies. A red annotation states: "④ Locations of change anomalies". The anomalies are listed as `[Eclipse SWT/win32/org/eclipse/swt/widgets][Widgets.java][buttonUpEvent]` and `[Eclipse SWT/win32/org/eclipse/swt/widgets][Widgets.java][buttonReleaseEvent]`.
- Diff Details:** Shows the details of the diff. A red annotation states: "⑤ Matching diffs in the found location". The details include the `keyDownEvent` method signature and the `sendEvent` method call.
- Diff Template:** Shows the textual representation of the abstract diff template. A red annotation states: "⑥ Textual representation of the abstract diff template". The template includes the `keyDownEvent` method signature and the `sendEvent` method call.

CRITICS

- コード変更チェックツール
 - ① パッチの差分を個別に選択できる
 - ② コードを構文木表示して操作出来る
 - ③ 指定にマッチする箇所を特定して表示
 - ④ 例外箇所を特定して表示
 - ⑤ コード内の差分の表示
 - ⑥ コードのテンプレートの表示

開発者の視点

- Salesforce.com社の開発者に使用した感想を聞く
 - 4つのパッチで試用してみる
 - 質問内容
 - コードレビューでどんな問題に直面しているか？
 - どんな場面でCRITICSは役に立つと思うか？
 - CRITICSをどう思う？

開発者の視点

- 対象とする4つのパッチ

No	Commit Description	Changed LOC	Num of Changed Files
1	Refactor test cases by moving bean maps to respective utils classes	743	22
2	Refactoring the API to get versioned field values by passing the version context as a parameter	943	34
3	Refactor tests by using try-with-resources statements to ensure the resource object is released after the program	484	10
4	Update common search tests by getting versioned test data	2224	12

TABLE II: Diff patches from Salesforce.com used for the study

開発者の感想

- コードレビューでどんな問題に直面しているか？
 - いつもはCOLLABORATORというツールを使っている
 - このツールはアップロードしたパッチしかチェック出来ないのが不便
 - 違いの少ない変更やピンポイントな間違いを見つける能力に欠ける

開発者の感想

- どんな場面でCRITICSは役に立つと思うか？
 - 熟練開発者に割り当てていたコードレビューを新規開発者に割り当てて熟練開発者の負担を軽減になる
 - チーム全体のコードレビューへの知識も深まる

開発者の感想

- CRITICSをどう思う？
 - 2,30分でサーチが終わった
 - 時間の節約になる
 - ビジュアル化されるのが嬉しい概ね好印象
- 実際の開発現場で使える基準だった

比較実験

- 学生12人にCRITICSとEclipseDiffの両方を使って比較した結果と感想を聞く
 - EclipseDiffはデフォルトで使えるので選出
 - シンプル(190行)と複雑(680行)の変更をチェック
- よく似た変更の区別
- 矛盾する変更の発見
- 失ったアップデートの発見
- タスク完了タイム
- 感想

対象パッチと問題

	Versions	Change Description	Similar Change	Inconsistent Change	Missing Update	Size(LOC)
Patch1 (Simple)	JDT 9800 vs. JDT 9801	initiate a variable in a for loop instead of using a hashmap	getTrailingComments(ASTNode) getLeadingComments(ASTNode) getExtendedEnd(ASTNode)	getExtendedStartPosition(ASTNode)	getComments(ASTNode) getCommentsRange(ASTNode)	190
	Q1. Given the change in the method <code>getTrailingComments</code> , what other methods containing similar changes can you find? Count the number. A. 0 B. 1 C. 2 D. 3 or more Answer: C. <code>getLeadingComments</code> and <code>getExtendedEnd</code> .					
	Q2. Which of the following methods contains inconsistent changes compared with the change in <code>getTrailingComments</code> ? A. <code>storeTrailingComments</code> B. <code>getExtendedEnd</code> C. <code>getLeadingComments</code> D. <code>getExtendedStartPosition</code> Answer: It uses a wrong expression, <code>i<=this.leadingPtr</code> instead of <code>range==null && i<=this.trailingPtr</code> .					
	Q3. How many methods share context similar to the change in <code>getTrailingComment</code> but missed the similar update? A. 0 B. 1 C. 2 D. 3 or more Answer: C. <code>getComments</code> and <code>getCommentsRange</code> .					
	Versions	Change Description	Similar Change	Inconsistent Change	Missing Update	Size(LOC)
Patch2 (Complex)	JDT 10610 vs. JDT 10611	extract the logic of unicode traitement to a method	getNextChar() getNextCharAsDigit() getNextToken() ... 9 locations in total	getNextCharAsJavaIdentifierPart()	jumpOverMethodBody() getNextChar(char, char) getNextToken() ... 11 locations in total, all located in another file	680
	Q1. Given the change in the method <code>getNextChar</code> , what other methods containing similar changes can you find? A. 0 B. 1-5 C. 6-9 D. 10 or more Answer: C. <code>getNextCharAsDigit()</code> and <code>getNextToken()</code> , ... 8 methods in total					
	Q2. Which of the following methods contains inconsistent change compared with the change in <code>getNextChar</code> ? A. <code>getNextCharAsDigit</code> B. <code>getNextCharAsJavaIdentifierPart</code> C. <code>jumpOverMethodBody</code> D. <code>jumpOverUnicodeWhiteSpace</code> Answer: It invokes a wrong method, <code>jumpOverMethodBody</code> instead of <code>getNextUnicodeChar</code> .					
	Q3. How many methods share context similar to the change in <code>getNextChar</code> but missed the similar update? A. 0 B. 1-5 C. 6-9 D. 10 or more Answer: D. <code>jumpOverMethodBody</code> , <code>getNextToken</code> ... 11 methods in total.					

TABLE III: The description of two patches and corresponding questions for code review tasks.

結果

- よく似た変更の区別
 - CRITICS : 10/12人が正確に答えた
 - EclipseDiff : 5/12人
 - EclipseDiffは選んだキーワードに依存してチェックしていた
- 矛盾する変更の発見
 - CRITICS : 12/12人
 - EclipseDiff : 7/12人
 - 複雑でもシンプルでもCRITICSは高いスコアだった

結果

- 失ったアップデートの発見
 - CRITICS : 11/12人
 - EclipseDiff : 4/12人
 - 複雑でもシンプルでもCRITICSは高いスコアだった
- タスク完了タイム
 - CRITICS vs EclipseDiffで31.9%早かった

感想

- コードレビューが早くなると思う
- UIが直感的でない
- ヒントを出して欲しい, 今は使ってる人が考えて使ってる