R-4

DevOps におけるソフトウェアインスペクション手法

橋浦研究室　　1165133 遠藤 亮河　1175181 橋本 諒介

1. 研究背景

近年のソフトウェア開発では，GitHub[1]等のソースコードホスティングシステムを用いたDevOps[2]が注目されている．DevOps は，開発の各工程を可能な限り自動化，監視をすることで変更から本番環境までの時間を短縮し，品質を確保することができると考えられている。ソフトウェアテストを通過した成果物の欠陥報告や議論は Issue 管理機能によって行われ，問題がないと判断した成果物は本番環境へ自動リリースされる。

ソフトウェアの欠陥検出をするためにはソフトウェアインスペクションが有効であると知られている。インスペクションとは，ソフトウェアの中間成果物，成果物を対象として，対象の作成担当者以外が成果物を目視で確認することによって欠陥を発見，指摘する活動である[3]．しかしながら，このような作業を開発経験が浅い者が実施することは難しいという問題がある．

2. 研究目的

前述の問題が発生する原因として，GitHub 等に搭載されている Issue 管理機能の入力欄がテキストによる自由記述となっていることにあることに着目した。このため，開発経験が浅い者によるインスペクションを行う際に以下の２つの問題が発生する．

P1 インスペクション対象のどのような点に着目すれば良いかわからない

P2 指摘にどのような内容を含めれば良いかわからない

これら問題により，インスペクション時に指摘内容の思考時間が増えたり，曖昧な記述をしてしまうなど有効な指摘ができないことがある。本研究では，これらの問題を解決するために，インスペクション時に必要となる指摘項目をテンプレート化すると共に，GitHub に適した形で実装する方法を提案する．

3. 提案手法

前述で述べた P1，P2 を解決するため，本研究では IEEE 1028[4]からインスペクションを行う上で必要となる項目の抽出を，また IEEE1044[5]から欠陥指摘する上で必要な分類項目の抽出を行った．さらに，DevOps で GitHub が用いられることを前提に，GitHub の Issue に拡張可能なシステムの実装を行った．

4. 実現手法

前述で抽出した情報を GitHub の Issue で活用する方法として，以下の 2 つのフォームを作成した．

4.1. シナリオフォーム

2 章で述べた P1 を解決するため，本研究では，UBR(ユーセージベースドリーディング)[6]を用いる．開発者が機能の追加や変更を行う場合，プルリクエストの機能を用いた変更提案をする．その際に図 1 に示すシナリオテンプレートの各項目に沿った記述を添付することで，開発経験が浅い者でも機能の追加や修正に関する内容を理解できる．

4.2. 欠陥指摘フォーム

2 章で述べた P2 を解決するため，インスペクション担当者は 4.1 のプルリクエストに添付されたシナリオを参考にインスペクションを行い，図 2 に示す欠陥指摘フォームを用いて結果を報告する．

図 1 シナリオフォーム

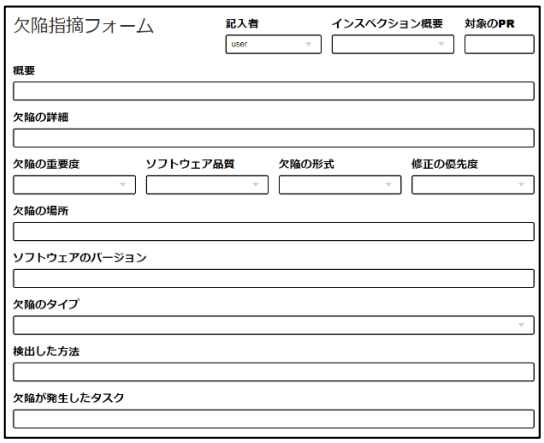

図 2 欠陥指摘フォーム

5. 実験

提案手法の有効性を確認するために実験を行った．実験は実際に動作する簡単なシステムに対して被験者がインスペクションを行い，提案手法によってどの程度欠陥の指摘が行えるかどうかを確かめるものである．

インスペクション対象となるシステムは，著者らが開発した施設予約システムとした．まず，システムに関する要求仕様書を準備し，次にシステムに要求仕様書に反するような欠陥を組み込んでおく．欠陥の組み込みに際しては欠陥の偏りを防ぐために Yuepu と Sampath [7]が提案している Web 欠陥分類リストを用い，対象のシステムに合計 20 個の欠陥の組み込みを行った．実験の評価はこれらの欠陥に対して，適切に指摘できた数や誤った指摘の数を集計することによって行う．実験の具体的な流れは以下の通りである．

1. 事前学習（15 分）
・ レビューの重要性と品質について基礎的な内容の説明
2. 実験説明（15 分）
2.1.要求仕様書を配布
2.2a.（ツールありの場合）
2.2a.1.ツールマニュアル配布
2.2a.2.ツールの使用方法を説明
2.2b.（ツールなしの場合）
2.2b.1.欠陥指摘シートを配布
2.3. 説明内容に関する質疑応答
3. 実験（60 分～90 分）
・ 対象のシステムに対するレビューを実施

実験の被験者は日本工業大学工学部情報工学科・情報メディア工学科に所属する 12 名とし，無作為に半数をツールあり（実験群），残りの半数をツールなし（統制群）に割り振った．

6. 実験結果と考察

実験結果の概要を表 1 に示す．表中の値はそれぞれ 6 人の被験者の平均値である．ツールありの方が適切な欠陥指摘数が減少しているように見えるが，これらの結果に対してマンホイットニーの U 検定を行った結果（表 2），有意差は認められなかった．

一方で，不適切な欠陥指摘数はレビューに要する時間が増加しているにも関わらず，発生件数が半減している．このことに着目し，時間あたりの不適切な欠陥指摘数を求め（表 3），これらに対してマンホイットニーの U 検定を行った結果，有意差が認められた．

表 1 実験結果の概要

	平均値	
	ツール有	ツール無
適切な欠陥指摘数	4.17	5.83
不適切な欠陥指摘数	1.00	2.17
実験時間(min)	88.3	71.7
不適切な欠陥指摘効率	0.01	0.03

表 2 適切な欠陥指摘数の検定結果

検定統計量	期待値	分散	検定統計量	有意確率 p
6.50	18.00	47.64	1.875	0.061

表 3 不適切な欠陥指摘効率の検定結果

検定統計量	期待値	分散	検定統計量	有意確率 p
4.00	18.00	36.14	2.832	0.020

表 2 の検定結果から，本ツールの効果は「レビュー時に不適切な指摘が少なくなる」ことがわかった．従って，本ツールを使用することで，不必要で曖昧な欠陥指摘から欠陥の再現をすることが少なくなる．これにより，重要な欠陥指摘に対しての検証時間に多くのリソースを費やすことが可能となった．

7. まとめと今後の課題

本研究では，DevOps で利用される GitHub を用いたソフトウェアインスペクション手法を提案し，実験によって本手法が不適切な指摘効率の減少に効果があることを明らかにした．今後は有効な指摘数を増やすための提案手法の改良および，ソフトウェア開発 PBL に対する適用などさらに実践的な環境での評価を実施していきたい．

文　献

[1] GitHub, Inc. ”GitHub, ”(accessed : 2020-11-30).
[2] Benjamin Benni, Philippe Colle, Guilhem Molines, Sébastien Mosser , Anne-Marie Pinna-Déry , "Teaching DevOps at the Graduate Level: A report from Polytech Nice Sophia, "Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment, Lecture Notes in Computer Science, Vol. 11350, pp. 60-72, Springer, Mar. 2018.
[3] K Wiegers, ”ピアレビュー 高品質ソフトウェア開発のために,” 日経 BP, Feb. 2004.
[4] IEEE , “IEEE 1028-2008 - IEEE Standard for Software Reviews and Audits, ” Jun. 2008.
[5] IEEE, “IEEE 1044-2009 - IEEE Standard Classification for Software Anomalies, ” Nor. 2009.
[6] 児玉公信, "UML モデリング入門, " 日経 BP, Apr. 2008.
[7] G. Yuepu, S. Sampath , "Web Application Fault Classification-An Exploratory Study , "In Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement (ESEM 08). pp.303-305, Jan. 2008.