A Unified Code Review Automation for Large-scale Industry with Diverse Development Environments

Hyungjin Kim
hjkim17.kim@samsung.com
Samsung Research
Seoul, Korea

Yonghwi Kwon
yhwi.kwon@samsung.com
Samsung Research
Seoul, Korea

Hyukin Kwon
hyukin.kwon@samsung.com
Samsung Research
Seoul, Korea

Yeonhee Ryou
yeonhee.ryou@samsung.com
Samsung Research
Seoul, Korea

Sangwoo Joh
sangwoo.joh@samsung.com
Samsung Research
Seoul, Korea

Taeksu Kim
taeksu.kim@samsung.com
Samsung Research
Seoul, Korea

Chul-Joo Kim
chuljoo1.kim@samsung.com
Samsung Research
Seoul, Korea

ABSTRACT

Code Review is an essential activity to ensure the quality of the software in the development process. *Code Review Automation* with various analyses can reduce human efforts of code review activities. However, it is a challenge to automate the code review process for large-scale companies such as Samsung Electronics due to their complex development environments: many kinds of products, various sizes of software, different version control systems, and diverse code review systems. In this work, we show how we automated the code review process for those intricate environments, and share some lessons learned during two years of operation. Our unified code review automation system, **Code Review Bot**, is designed to process review requests holistically regardless of such environments. Our findings provide practical evidence that our system motivates developers in Samsung Electronics to improve code quality.

KEYWORDS

code review, review bot, code review automation, static analysis

ACM Reference Format:
Hyungjin Kim, Yonghwi Kwon, Hyukin Kwon, Yeonhee Ryou, Sangwoo Joh, Taeksu Kim, and Chul-Joo Kim. 2022. A Unified Code Review Automation for Large-scale Industry with Diverse Development Environments. In *44nd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP '22), May 21–29, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3510457.3513062

1 CHALLENGES IN A LARGE-SCALE INDUSTRY

Code Review Automation is a process of generating code reviews from the result of predefined set of tools. It decreases human effort in quality assurance and takes an important role in modern code review activities [2]. Despite the convenience, adopting code review automation to a real-world industry area has hurdles that interfere with the application.

Samsung Electronics manufactures many kinds of products in various domains such as smartphones, wearable devices, laptops, televisions, home appliances, network equipment, or semiconductors. Variety in product lines obviously leads to a huge size of software, and it is usual that a review request contains a large size of code changes.

Samsung Electronics adopts two kinds of source code management systems, *Git* and *Perforce Helix Core*. Moreover, numerous code review systems such as *GitHub*, *Gerrit*, and *Helix Swarm* are used in the field. Developers choose systems depending on the characteristics of their projects such as project size, nature of target platform, and build environments. Thousands of review requests are created in a day from more than six thousand projects. Moreover, 1.4 million lines of code are added while one million lines are removed every day. Mean values are 1,373 lines with a standard deviation of 8.70 for added lines and 757 lines with a standard deviation of 4.75 for removed lines.

Because of enormous code changes, it is hard to automate code review. Scalability, performance, and rapid response are the most important quality attributes what automation tools or Continuous Integration(CI) systems should meet in the large-scale industry [1].

Variations in development process and culture of each team also raise difficulty. A development team defines its own quality assurance policies, and uses various kinds of analysis tools to ensure software quality. For instance, some business divisions melt static analyzers into a CI, but some teams perform the tools in an integration testing phase. The huge size of target software and the scale of each review request also make the automation to be a tough challenge. A scalability problem is one of the most important quality attributes in the industry because an automation system with low performance may be a bottleneck in the development process.

2 CODE REVIEW BOT: AUTOMATIC CODE REVIEW SYSTEM

We developed a unified code review automation system for Samsung Electronics, ***Code Review Bot***, which addresses various analysis tools and code review systems. **Code Review Bot** helps to find a unified code review process no matter which systems and tools are used in a project. It supports review systems such as GitHub, Swarm,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ICSE-SEIP '22, May 21–29, 2022, Pittsburgh, PA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9226-6/22/05…$15.00
https://doi.org/10.1145/3510457.3513062

and Gerrit, which are used in Samsung Electronics. The automation system also provides various services such as code change size measurement in a review request, typo check, potential defect detection, and coding style violation.

When a developer creates a review request using a review system, the review system triggers **Code Review Bot** to start analyses. Receiving the request, each service of **Code Review Bot** inspects code changes from the review request and generates review comments, including its analyses results. The developer confirms comments by herself and checks detailed information from **Code Review Bot**. Reviewers are able to concentrate on understanding semantics of the code change and make review comments without worry of simple errors such as typos, potential defects, or coding style violations.

Regarding the difficulty in the adoption of code review automation to Samsung Electronics, we focused on three quality attributes in the design of **Code Review Bot**: extensibility, scalability, and performance.

- We designed an abstract module for a review system to abstract concrete review systems and encapsulate their details. It provides common interfaces for every review system and lets **Code Review Bot** easily extend to new kinds of systems.
- To address the scalability problem, invocation of analysis services and scheduling among services are controlled by a task worker module. A task worker module controls invocating and scheduling analysis services.
- Moreover, it is designed as parallel processing for the efficiency.

Code Review Bot has been applied in Samsung Electronics for about two years from 2019. We chose a single month (August 2021) and analyzed the numbers of requests from all review systems. Totally 81,712 review requests were created and the average request count in a day is 3,714. About 80% of requests were created from Helix Swarm and Gerrit. In terms of built-in services, *Typo*, *Commit Size* and *Potential Defects* are the most frequently used top three services (86.7%). Some services such as *Coding Style*, *Ground Rule*, *Test Coverage*, and *Architecture Maturity* takes lower portion than top three services because the services can be applied to neither Helix Swarm nor Gerrit.

Code Review Bot has been successfully deployed to whole development teams of Samsung Electronics. More than 60% detected defects were fixed and merged into production code since developers are strongly motivated by our system to improve the code quality. To show usefulness, we conducted a qualitative analysis on code fixes related to code reviews by **Code Review Bot**. We gathered comments pointing out typos or potential defects during a month, August 2021. In total, we collected 52,889 comments: 25,759 from GitHub, 22,663 from Helix Swarm, and 4,467 comments from Gerrit. However, some comments are submitted more than once even for a single review request because developers requested **Code Review Bot** to review the patch repeatedly. We found 73.2%, 45.2%, and 47.8% of duplicated comments from GitHub, Helix Swarm, and Gerrit, respectively. We excluded such duplicated comments for this analysis. Overall, Table 1 shows fix rates of typos and potential defects.

Despite the different characteristics of each review system, **Code Review Bot** motivates developers to improve code quality. GitHub has a higher rate of duplicated comments than other two types of review systems. It implies that developers interact with **Code Review Bot** actively during code review processes. Fix rates of GitHub also support the implication, which are higher than other two systems. On the other hand, Helix Swarm has the largest number of detected typos and potential defects.

Table 1: Fix Rate of Typos ans Potential Defects

	Typos			Potential Defects		
System	Fixed	Found	Fix rate	Fixed	Found	Fix rate
GitHub	793	1, 092	72.62%	5, 571	5, 806	95.95%
Swarm	3, 611	6, 324	57.10%	3, 745	6, 094	61.45%
Gerrit	556	937	59.34%	1, 067	1, 394	76.54%
Total	4, 960	8, 853	59.38%	10, 383	13, 294	78.10%

Code Review Bot is designed to process review requests holistically regardless of environments. **Code Review Bot** provides flexible interface for those review systems, and motivated 15,383 code fixes for a month, on three different systems.

3 FUTURE WORK

We have collected massive code review data while operating **Code Review Bot**, and one can use it for training deep learning models. For example, a large neural model trained with this data can reduce some burden of the code review process[3]. Further, some valuable patches can be automatically extracted from many projects' very long commit history. These can also be delivered to developers to improve the code quality.

Finally, a more rigorous investigation of interactions between **Code Review Bot** and users is required.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers, Yoonki Song, Geunsik Lim, Youil Kim, and Joonbae Park, for their insightful comments and feedback that helped improve the paper. This work was supported in part by Intelligent Dev. Assistant & Quality Tool Development (RAJ0121ZZ-32RF), Samsung Research, Samsung Electronics Co., Ltd.

REFERENCES

[1] Geunsik Lim, MyungJoo Ham, Jijoong Moon, and Wook Song. 2021. LightSys: Lightweight and Efficient CI System for Improving Integration Speed of Software. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. IEEE Press, NY, USA, 1–10. https://doi.org/10.1109/ICSE-SEIP52600.2021.00009

[2] Caitlin Sadowski, Emma Söderberg, Luke Church, Michal Sipko, and Alberto Bacchelli. 2018. Modern Code Review: A Case Study at Google. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice* (Gothenburg, Sweden) *(ICSE-SEIP '18)*. Association for Computing Machinery, New York, NY, USA, 181–190. https://doi.org/10.1145/3183519.3183525

[3] Rosalia Tufano, Luca Pascarella, Michele Tufano, Denys Poshyvanyk, and Gabriele Bavota. 2021. Towards Automating Code Review Activities. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE Press, NY, USA, 163–174. https://doi.org/10.1109/ICSE43902.2021.00027