Impact of Smart Contract Code Refactoring on Ethereum Gas Price

Aurel Ikama[1]
[1] *Dep. of Computer and Software Engineering, Polytechnique Montréal, Montréal*

***Abstract*—TODO**
***Keywords*—** Anti-patterns, Fault-proneness, SZZ algorithm

I. Introduction

[Aurel: Définitions : Smart contract Refactoring de code Gas price Ethereum]

[Aurel: Orgainisation du papier]

Refactoring is the process of modifying and improving a code without changing its external behavior [1] [2] [3]. From this definition follows a practice that can be found in software engineering depending on the programming language.[x] utilise des projets Java, [x] quant à lui utilise le javascript ou encore [x] avec ses travaux sur le refactoring sur les logiciels mobiles. Par ailleurs, la motivation derrière le refactoring est beaucoup étudier dans la littérature, [x] et [x] se basent sur des sondage auprès des développeurs pour comprendre les motivations, [x] complête ses précédents papiers en ajoutant en plus d'une enquête, une analyse de 124 projets dans Github. Plusieurs études sont fait pour étudier l'impacte du refactoring sur la qualité du logiciel. *[Aurel: Ajoute des exemples de travaux]*. Dans ce papier, nous nous intéressant de l'impacte des refactorings sur la performance d'un code en Solidity.*[Aurel: Des travaux sur le refactoring en solodity]*

The remainder of the paper is organized as follows. Section II describes our study method. We report our case study results in Section III followed by discussion of the findings in Section IV. Next, we describe the threats to validity in Section V and related work in Section VI. Finally, we conclude the paper in Section VII.

II. Study Design

RQ1: What Ethereum smart contract code refactoring catalog is known to date?

RQ2: Does code refactoring change the gas price?

RQ3: What kinds of refactoring impact the gas price?

A. Data collection

Smart Bugs : https://github.com/smartbugs/smartbugs-wild/tree/master/contracts

B. Data Processing

In this sub-section, we describe the details of the data collection and analysis approach followed to answer our different research questions. This approach is depicted in Figure 1. In the following, we elaborate on each data processing step.

1) *Collecter et trier les Smart contract:*
2) *Mesurer le prix du gaz de chaque Smart contrat:* https://github.com/paperSubmition2020/GasmetReplicationPackage
3) *Inserer des modifications (aléatoirement - manuellement):*
4) *Documenter les modifications:*
5) *Mesurer le prix du gaz des smart contrats modifés:*

C. Replication Package

III. Case Study Results

RQ1: What Ethereum smart contract code refactoring catalog is known to date?

RQ2: Does code refactoring change the gas price?

RQ3: What kinds of refactoring impact the gas price?

IV. Discussion

V. Threats to validity

This section discusses threats to the validity of our study following the guidelines for case study research.

A. Threats to construct validity

B. Threats to internal validity

C. Threats to External validity

D. Threats to reliability validity

VI. Related Work

A. Replication studies in software engineering

B. Studies on the limitations of SZZ

C. Studies on the impact of Antipatterns

VII. Conclusion

[4]

References

[1] M. Fowler, *Refactoring: improving the design of existing code*. Addison-Wesley Professional, 2018.
[2] D. Silva, N. Tsantalis, and M. T. Valente, "Why we refactor? confessions of github contributors," in *Proceedings of the 2016 24th acm sigsoft international symposium on foundations of software engineering*, 2016, pp. 858–870.
[3] W. F. Opdyke, "Refactoring object-oriented frameworks," 1992.
[4] D. Jhon, "test," in *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2021, pp. 1082–1086.

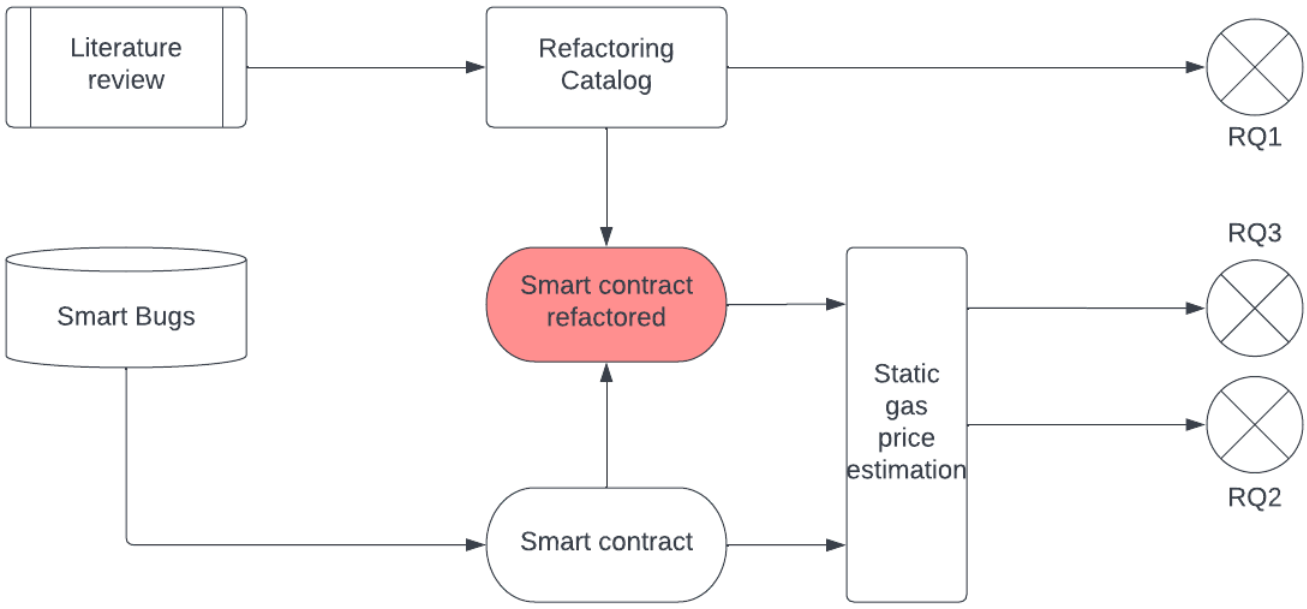

Figure 1: Overview of the study method