Your Very Nice Project Title

Mark Zhang, William Zheng
University of Illinois at Urbana-Champaign
{zz91,xinzez2}@illinois.edu

Abstract

Cloud application systems running on top of by container orchestration systems such as Kubernetes are often managed by domain-specific controller code called operator to automate laborious manually-done operations. The correctness and the reliability of such code is critical to productionized application yet operators are highly complex and difficult to ensure correctness. We conduct empirical study on 52 popular Kubernetes operator from a software engineering perspective to characterize their complexity. TODO

1 Introduction

With the prevalence of cloud-native software systems, Kubernetes[1] has become one of the popular choices for orchestrating productionized container applications. Developers extend Kubernetes APIs and functionalities by writing specialized programs called operators, which automate the laborious task of deploying and managing services and applications. Since operators are responsible for various highly complex operations such as service scaling, backup, and migration in production environment, their correctness is paramount for system reliability. To understand the reliability of operators and how it could be improved, we must first develop a deep understanding of the complexity of operators and operators fail.

The significance of reliability of Kubernetes operators stems from challenges inherent in their design and execution within production environments. Operators undertake diverse and mission-critical responsibilities, including but not limited to dynamic scaling of services, data backup, and seamless application migration. As a result, any failures within operator functionalities can cause cascading repercussions, ranging from service disruptions to potential data loss, ultimately undermining the stability and availability of the entire system.

2 Methods

To understanding the reliability implications in Kubernetes operators, we collect various metrics from 52 popular open-sourced operators, who are responsible for managing diverse system including but not limited to message queue, database system, distributed synchronization manager. We synthesize findings and implications in section3.

3 Results

3.1 What are the challenges in writing reliable operator?

3.2 What are the developers' challenges?

Operators are responsible for operating a very specifc managed system and it is operators' job to reconcile the managed system to the desired state given users' expectation. Modern software systems can be gigantuous with tens of thousands lines of code and complicated states management required. This engenders burdens for operator developers to understand the managed system thoroughly, increasing the likelyhood for them to make mistakes in developing the operator.

In Figure1, we collect and visualize the involvement of operator developers in contributing to the managed system using GitHub repository contributor data. While the absence of operator develoipers' contribution to the managed system does not necessarily imply their incompetence in understanding the managed system, it is a still a reflective metric on the average familiarity of operator developers on the managed system. In our dataset, only 19% of the operator projects have more than half of the developers being contributing to the managed system. The proportion of developers who are also developing the managed system drops as the operator project gets larger. In the two largest operator project in our dataset, with regard to the size of developers, only 10% of the developers are contributors to both the managed system and its operator.

4 Related Work

There are previous efforts in understanding device driver in modern system[2]. The operator shares a very similar figure with the devce driver. Both of them impose the challenges of reliably bridging the gap between managed system and the underlying operating system interfaces, while the underlying system for device driver is operating system and the underlying system for the operator is Kubernetes. Though both modern operating system and cloud management system are complicated, interacting with cloud management system faces even more problems in handling distributed asynchronous issues. The cloud environment raises new classes of reliability challenges like staleness and faults when interacting with cloud management system[3] ,which requires extra attention from operator developers to avoid bugs. This work focuses on both common reliability problem faced by

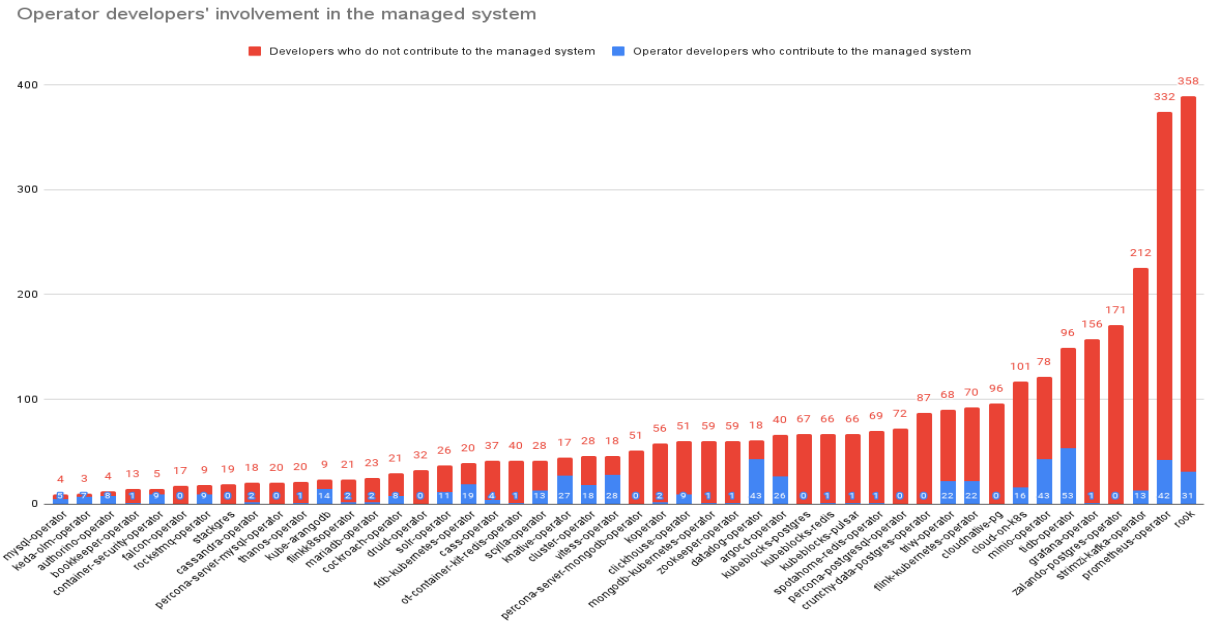

Figure 1. Operator developers' involvement in the managed system developement

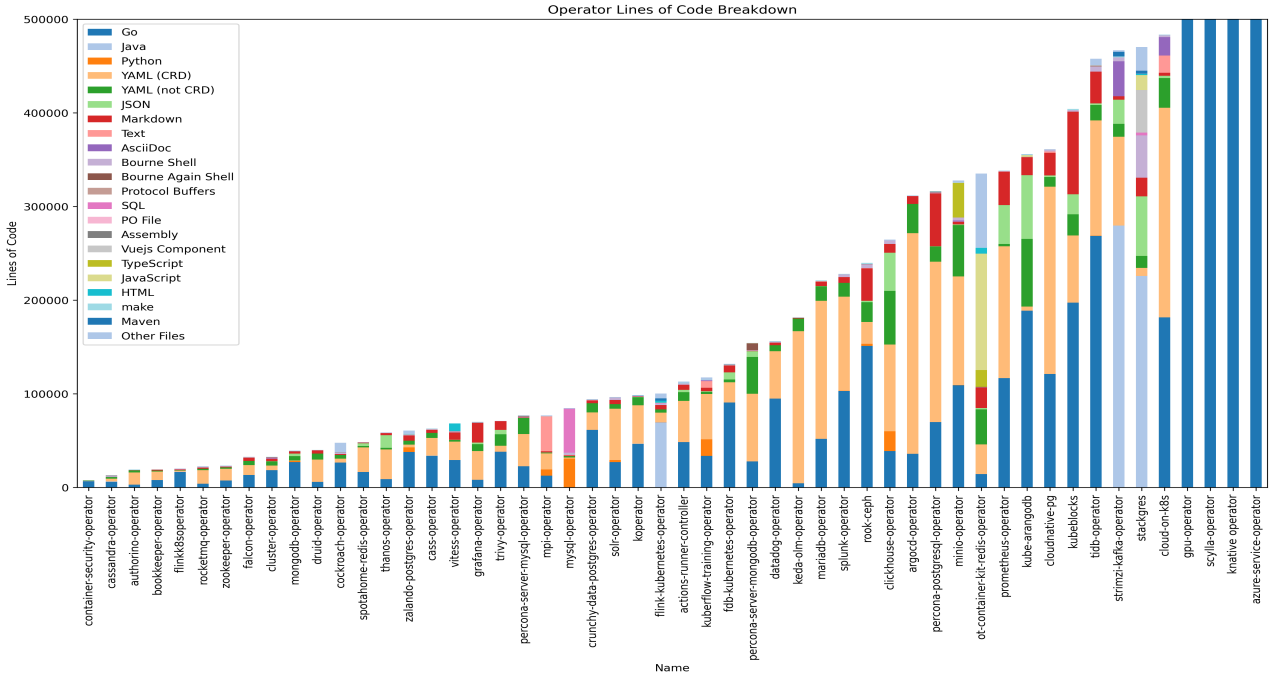

Figure 2. LOC and programming language distribution over operators

operator and device driver, and the newly raised reliability problem in the cloud environment.

5 Conclusion

This project is awesome.

6 Metadata

The presentation of the project can be found at:

https://zoom/cloud/link/

The code/data of the project can be found at:

https://github.com/you/repo

References

[1] Kubernetes. https://kubernetes.io/, 2024.

[2] Kadav, A., and Swift, M. M. Understanding modern device drivers. In *Proceedings of the Seventeenth International Conference on Architectural Support for Programming Languages and Operating Systems* (New York, NY, USA, 2012), ASPLOS XVII, Association for Computing Machinery, p. 87–98.

[3] Sun, X., Luo, W., Gu, J. T., Ganesan, A., Alagappan, R., Gasch, M., Suresh, L., and Xu, T. Automatic reliability testing for cluster management controllers. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)* (Carlsbad, CA, July 2022), USENIX Association, pp. 143–159.