ARGO-9 - Argo-Kubernetes-native Tools to Run Workflow Evaluation

Authors: Maciej Skoczeń, Adrian Beściak, Bartosz Szlachta, Gabriel Kępka
Year: 2023

Table of contents:

1. Introduction

Argo is a tool for managing workflows for applications in a Kubernetes cluster. It allows users to define and run complex tasks in a Kubernetes cluster that may consist of multiple steps or processes.

This tool is part of the Kubernetes ecosystem and leverages Kubernetes mechanisms, such as containers and services, to enable users to define, run, and monitor workflows. Argo can also integrate with other tools and services, such as Git, Slack, and Jira, to facilitate the deployment and management of complex processes.

Argo offers many features, such as automatic task resumption after failure, horizontal and vertical scaling, dynamic node selection, and many others. Thanks to these features, Argo Workflow is often used in environments where managing complex application processes in a Kubernetes cluster is required.

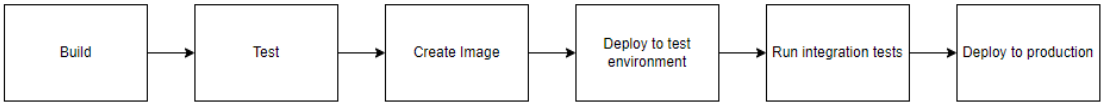
2. Theoretical background/technology stack

Kubernetes: Argo Workflows is built on top of Kubernetes, an open-source container orchestration platform. It leverages Kubernetes' powerful APIs for managing containers, services, and other resources to automate the execution of workflows. Argo follows the GitOps approach, which means that workflows are defined and version-controlled in Git repositories. This makes it easy to manage changes to workflows and ensures that workflows are auditable and reproducible. It uses containerization to package and deploy

workflows. Workflows can be defined as a series of containers that execute specific tasks or steps.

3. Case study concept description

Idea is to use Argo Workflows to create a CI/CD pipeline for our application. The workflow will include steps such as building and testing the application, creating a container image, deploying the application to a test environment, running integration tests and deploying to production.



How to use our technology in the pipeline:

1. Deploy Argo in K8s cluster.
2. Configure Argo to track Git repository that stores the configuration.
3. Argo monitors for any changes and applies them automatically.

This approach helps us to divide CI and CD pipelines. The CI pipeline that may be held by Jenkins is owned by Developer/Devops and the CD pipeline working with Argo is Operations/Devops responsibility.

Planned steps:

1. Fork example open source project repository to create a CI/CD for it.
2. Launch environment by creating a K8 cluster with docker images.
3. Create Argo config for the workflow.
4. Deploy Argo to the cloud, e.g. AWS.
5. Add and test other Argo functionalities such as Argo GUI or deployment notification on e.g. Slack.

4. Solution architecture

TODO

5. Environment configuration description

TODO

6. Installation method

TODO

7. How to reproduce - step by step

TODO

8. Demo deployment steps

TODO

9. Summary – conclusions

TODO

10. References