ArgoCon

CI/CD for Data Pipelines with Argo Workflows

J.P. Zivalich, Pipekit

#argocon2022 @pipekitio

pipekit

AGENDA

- Intro & goals for the talk
- Background: What is CI/CD for data pipelines?
- Problems CI/CD for data pipelines solves
- Solutions using Argo Events and Argo Workflows
- Demo & How-to
- Next steps
- Q&A

ArgoCon

#argocon2022

Intro & goals for the talk

ArgoCon

#argocon2022

GOALS FOR THE TALK

- Show a development setup of CI/CD for Data Pipelines using Argo Workflows
- Understand how to test WorkflowTemplates
- Learn some strategies for versioning WorkflowTemplates

ABOUT ME

J.P. Zivalich

CTO & Founder @ Pipekit

Contributer to the Argo Workflows project,
ex-Docker Hub maintainer

Focus: Data engineering architecture,
product development

Background: What is CI/CD for data pipelines?

ArgoCon

#argocon2022

BACKGROUND: CI/CD FOR DATA PIPELINES

Data engineering commonly pushes to prod without testing

- People push changes to their data transforms and hope for the best
- Ideally should validate these transforms when making changes
- Rollbacks are currently difficult - no concept of versioning for pipeline components

Forward-thinkers are applying CI/CD concepts to data pipelines

- Factoring out transforms and other critical pieces into components
- Running tests on these components during pull requests and other events
- Versioning these components using semantic versioning

PROBLEMS IT SOLVES

Wasted cloud spend

- Data pipelines can run anywhere from several minutes up to over a month
- Hardware intensive compute instances can be very expensive
- A failure occuring late in the data pipeline can result in lots of wasted resources

Data scientist time

- Learn earlier in the development cycle if bugs are being introduced

BACKGROUND: WORKFLOW TEMPLATES

WorkflowTemplates and ClusterWorkflowTemplates are Argo Workflows's native reuse component

Why factor items into WorkflowTemplates?

- Factoring to WorkflowTemplates you get…
  - DRY-code
  - Easier troubleshooting
- The following are good candidates to make components
  - Data transforms
  - Set-up/tear-down of K8s resources like Dask/Spark deployments
  - Utilities like cloning git repositories, etc.

TESTING WORKFLOW TEMPLATES

To test, use the inputs/outputs of WorkflowTemplates

WorkflowTemplates are essentially functions

- Each can take inputs and generate outputs
- For easy testing/components, ensure tested WorkflowTemplates are pure functions
  - The function return values are the same for identical arguments
  - The function has no side effects
- This allows us to compare inputs and actual outputs vs expected outputs

VERSIONING WORKFLOW TEMPLATES

Semantic versioning: Great to have, not in vanilla Argo

- Semantic versioning allows for structured promotion of components and easy rollbacks

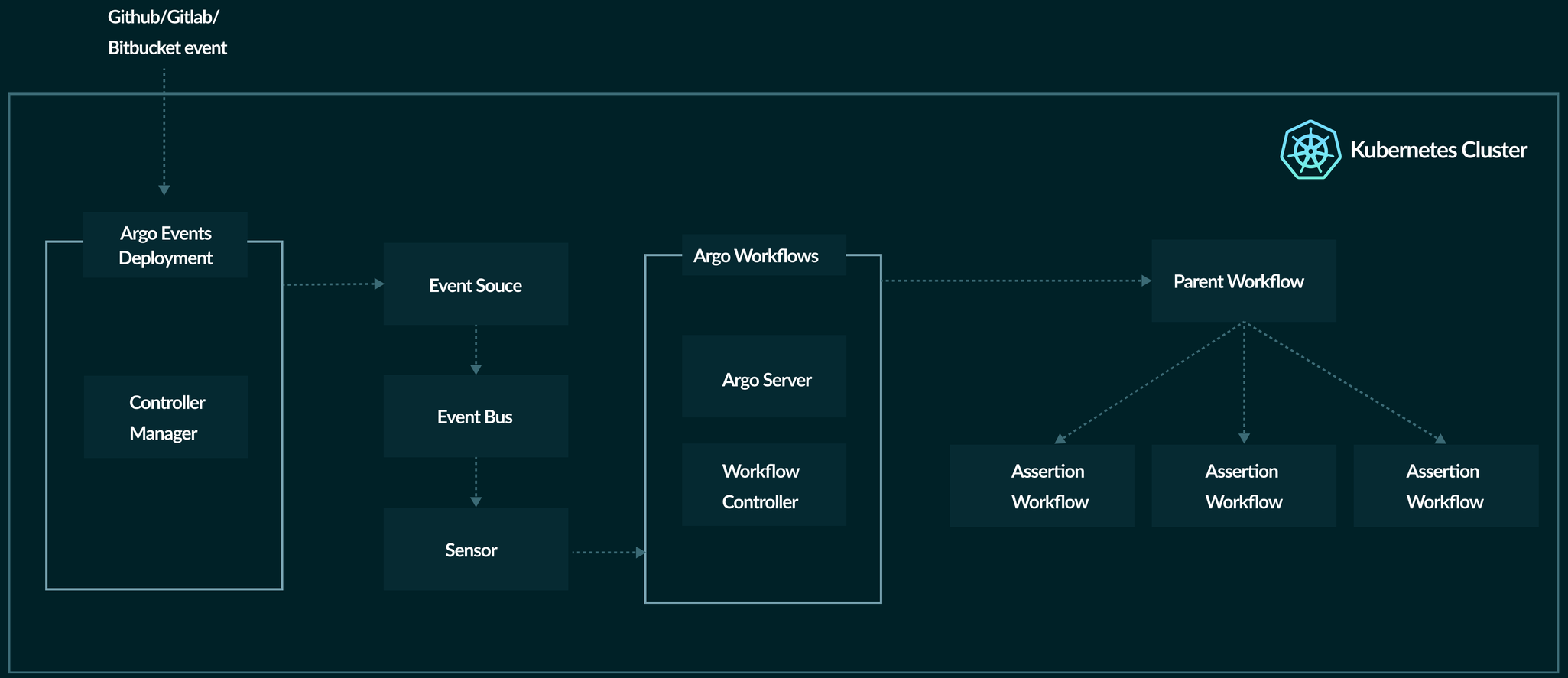
- We've seen two ways of implementing it for WorkflowTemplates in Argo Workflows:
    1. Appending the version to the name with dashes, ie template-12-3-9
    2. Adding a label or annotation denoting the version

Solution

ArgoCon

#argocon2022

CI/CD FOR DATA ARCHITECTURE WITH ARGO EVENTS AND WORKFLOWS



ArgoCon

#argocon2022

ARGO EVENTS

One-time configuration of event source and event bus. Update the sensor as needed with test cases.

- **Event Source**:
  - Configure your version control provider's webhook

- **Sensor:**
  - Map the version control provider's webhook to an Argo Workflow
  - Process the webhook and extract information such as the PR title, sha, etc.

Apply the WorkflowTemplate(s) from a given PR, write assertions, and validate them using a Workflow of Workflows pattern

- Use a Workflow to call WorkflowTemplates that clone the repository and apply the WorkflowTemplates to be tested
- To run newly created/updated WorkflowTemplates, use a Workflow of Workflows pattern
  - WorkflowTemplates are incorporated into a given Workflow at compile time
- Specify the test cases using **withItems** and loop over the given Workflow of Workflows

Demo & How-to

ArgoCon

#argocon2022

DEMO & HOW-TO

In this demo, we will...

- Create an Argo Event source and sensor that reads Github PRs and runs Argo Workflows
- Test changes made to a WorkflowTemplate in the PR

Prerequisites:

- K8s cluster runing locally (K3d, Docker Desktop, Minikube, etc.)
- Github repository
- Argo Workflows v3.3 installed

View repo on GitHub: https://github.com/pipekit/talk-demos/tree/main/argocon-demos/2022-ci-cd-for-data

ArgoCon

#argocon2022

DEMO & HOW-TO

Sample WorkflowTemplate:

```yaml
---
apiVersion: argoproj.io/v1alpha1
kind: WorkflowTemplate
metadata:
  name: doubler
spec:
  entrypoint: double
  templates:
    - name: double
      inputs:
        parameters:
          - name: input
      script:
        command:
          - python
        image: "python:alpine3.6"
        # a simple transform that doubles an input
        source: "print(int(\"{{inputs.parameters.input}}\") * 2)"

```

Assertion WorkflowTemplate:

```yaml
    - name: run-doubler-and-assert
      steps:
      # invoke the doubler template with a given input
      - - name: run-doubler
          templateRef:
            name: doubler
            template: double
          arguments:
            parameters:
              - name: input
                value: "{{workflow.parameters.input}}"
      - - name: assert
          template: assert-equal
          arguments:
            parameters:
              - name: input
                value: "{{workflow.parameters.input}}"
              - name: actual
                value: "{{steps.run-doubler.outputs.result}}"
              - name: expected
                value: "{{workflow.parameters.expected}}"
    - name: assert-equal
      inputs:
        parameters:
        - name: input
        - name: actual
        - name: expected
      script:
        image: python:alpine3.6
        command: [python]
        # ensure the actual value matched the expected value
        source: |
          assert {{inputs.parameters.actual}} == {{inputs.parameters.expected}},
          f"for {{inputs.parameters.input}}, expected {{inputs.parameters.expected}}, got {{inputs.parameters.actual
```

DEMO & HOW-TO

Test cases:

```yaml
    - name: assert-doubler
      steps:
      - - name: checkout-pr
          templateRef:
            name: git-checkout-pr
            template: git-checkout-pr
          arguments:
            parameters:
              - name: app_repo
                value: talk-demos
              - name: git_branch
                value: "{{workflow.parameters.short-sha}}"
      - - name: get-template
          template: get-template
      - - name: apply-template
          template: apply-template
          arguments:
            parameters:
              - name: workflowtemplate
                value: "{{steps.get-template.outputs.result}}"
      - - name: run-doubler-workflow
          template: run-doubler-workflow
          arguments:
            parameters:
            - name: input
              value: "{{item.input}}"
            - name: expected
              value: "{{item.expected}}"
          withItems: # the test cases to be ran
            - { input: '2', expected: '4' }
            - { input: '4', expected: '8' }
            - { input: '0', expected: '0' }
```

Workflow of Workflows:

```yaml
    - name: run-doubler-workflow
      inputs:
        parameters:
          - name: input
          - name: expected
      resource:
        action: create
        successCondition: status.phase == Succeeded
        failureCondition: status.phase in (Failed, Error)
        manifest: |
          apiVersion: argoproj.io/v1alpha1
          kind: Workflow
          metadata:
            generateName: assert-doubler-
          spec:
            arguments:
              parameters:
                - name: input
                  value: "{{inputs.parameters.input}}"
                - name: expected
                  value: "{{inputs.parameters.expected}}"
            # invoke the assertion template with given arguments
            workflowTemplateRef:
              name: assert-doubler
```

ArgoCon

#argocon2022

DEMO & HOW-TO

The output of the parent Workflow and children in the UI:

ArgoCon

#argocon2022

Next steps

ArgoCon

#argocon2022

NEXT STEPS

What does this unlock for us?

- Audit trail of workflows run with different component versions
- Assertions that correspond to cases found in production runs
- Clear guidelines for promoting WorkflowTemplates to higher environments

Resources & References

ArgoCon

#argocon2022

RESOURCES & REFERENCES

- Visit our repo: https://github.com/pipekit/talk-demos/tree/main/argocon-demos/2022-ci-cd-for-data
- Visit our blog: https://pipekit.io/blog/
- Follow @pipekitio on Twitter

ArgoCon

#argocon2022

ArgoCon