ArgoCon

North America

#argocon

ArgoCon
North America

Untangling the threads

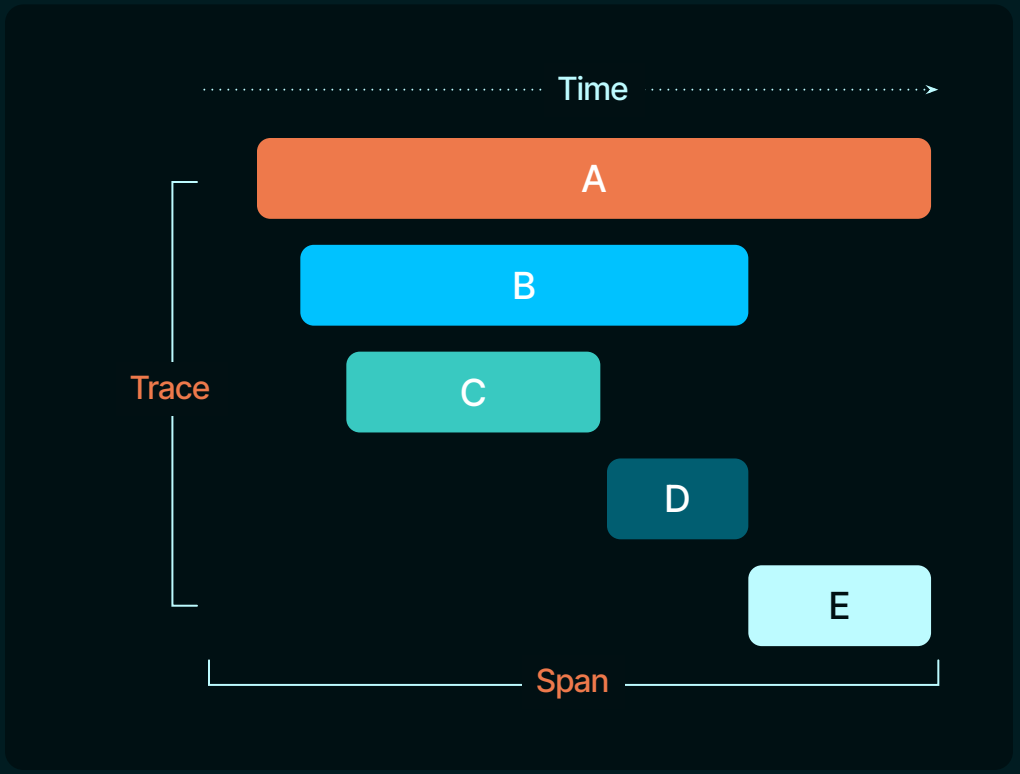
November 12, 2024.
Salt Lake City

Alan Clucas
Staff Software Engineer
pipekit

JM (Jason Meridth)
Senior Software Engineer
GitHub

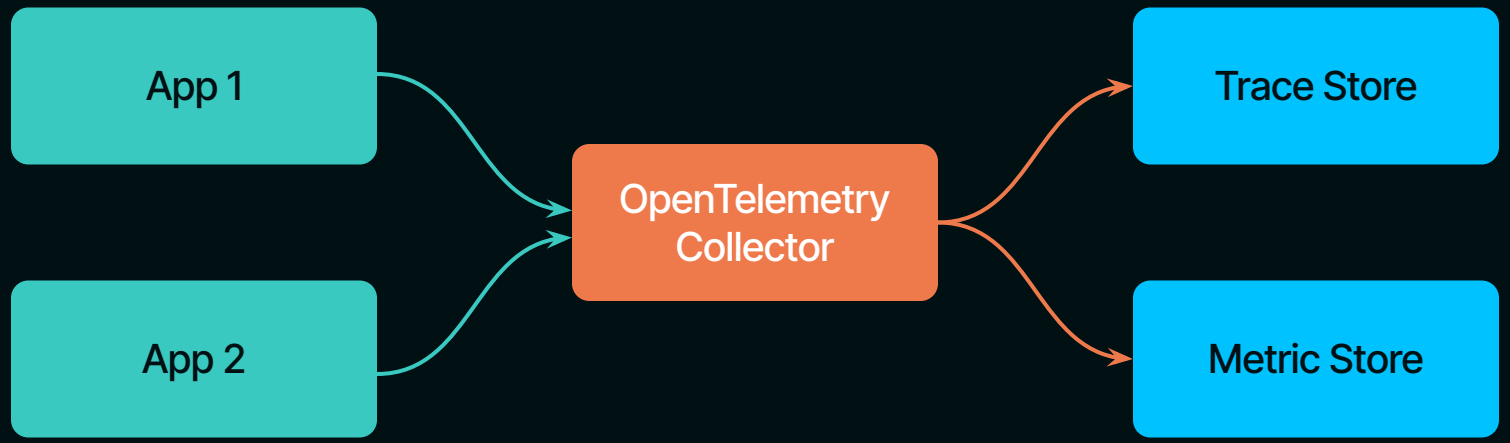
What is a trace?

Traces consist of spans
Spans start and stop in time
Spans can have child spans
Spans can have attributes
Events are timestamps inside a span



OpenTelemetry collector

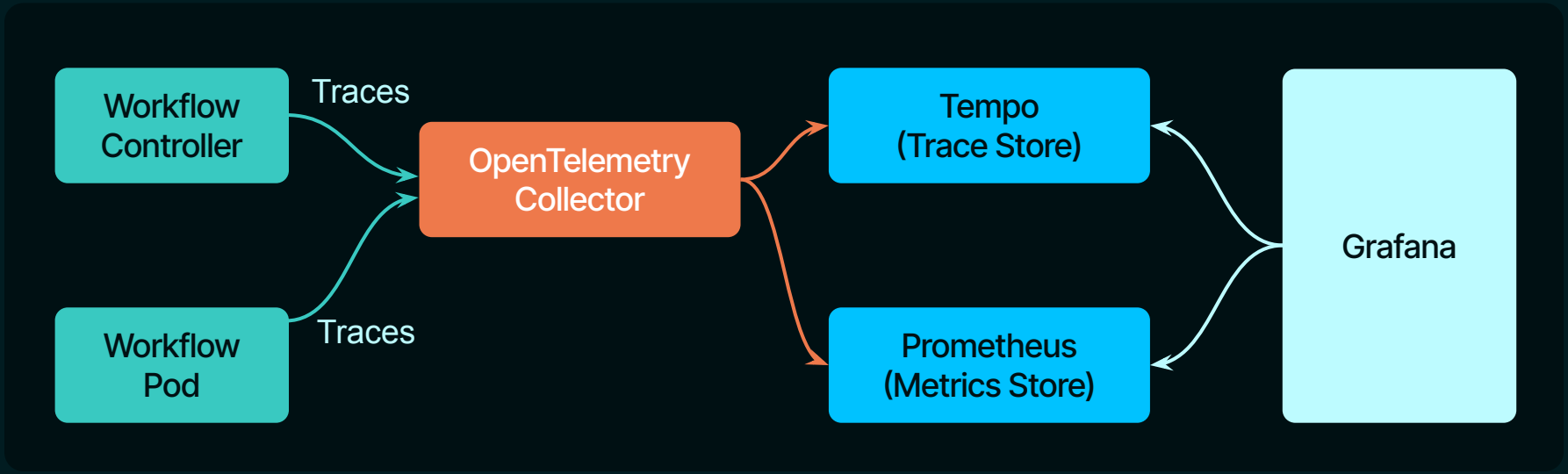
OpenTelemetry recommended architecture



Architecture details

Specifics:

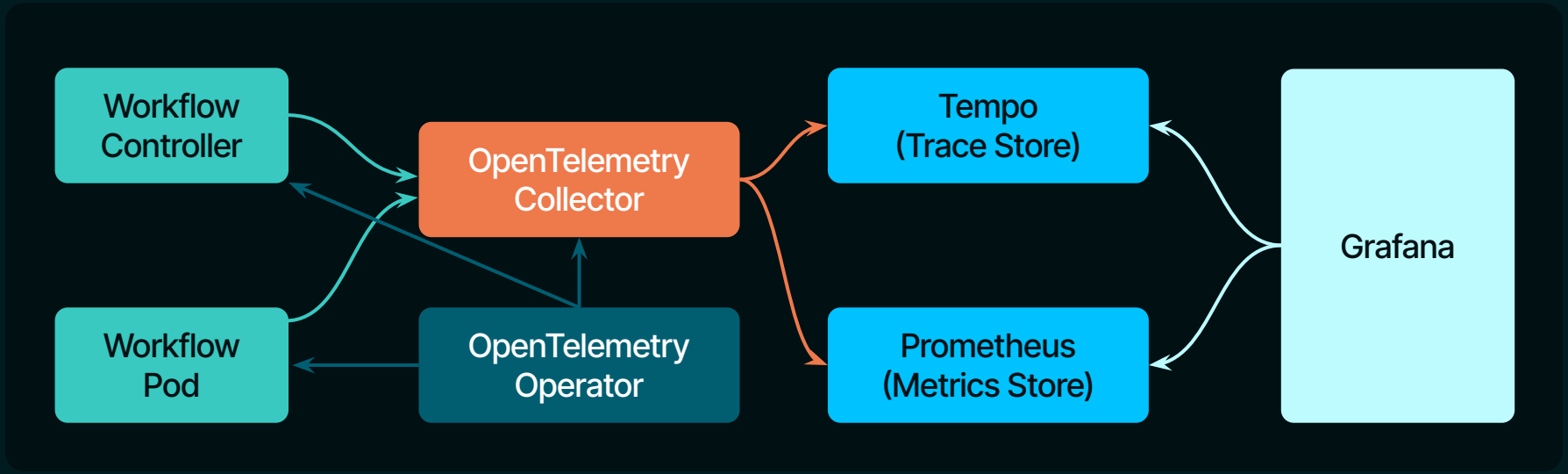
- Sending traces into the collector
- Storing traces and metrics in Grafana stack
- Visualising with Grafana



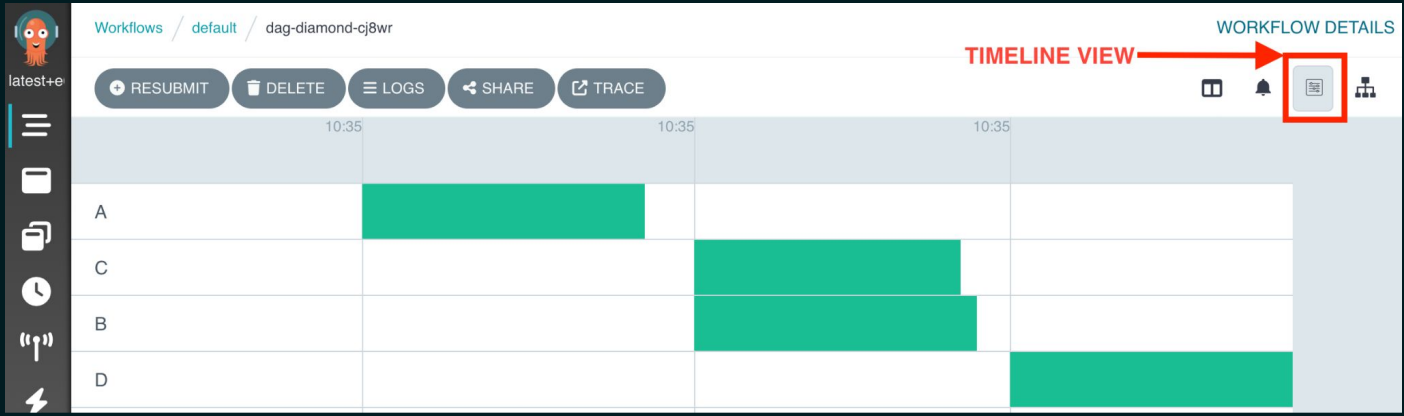
OpenTelemetry operator

Operator is

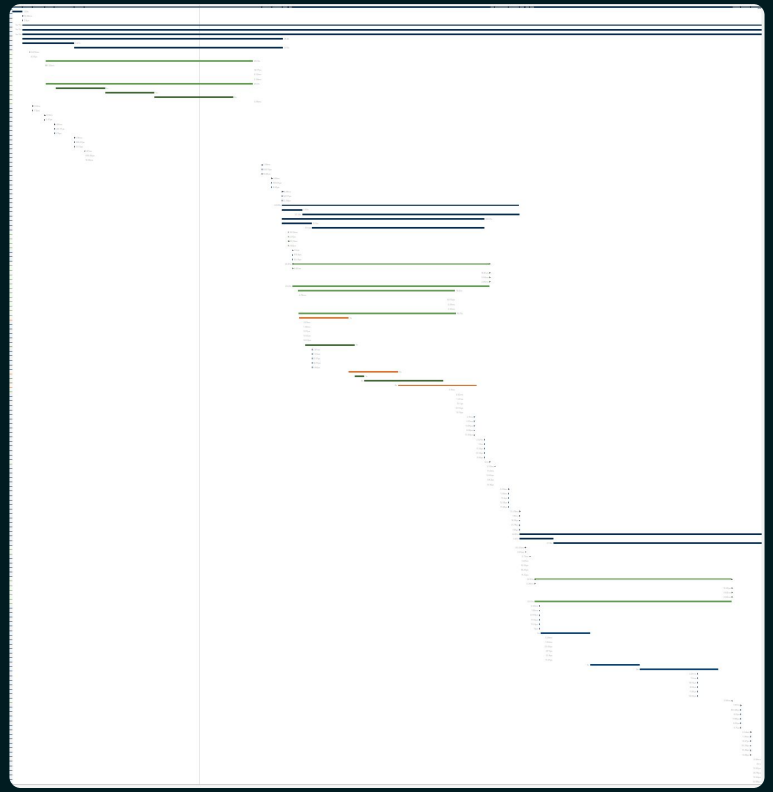
- Managing collector
- And controlling delivery



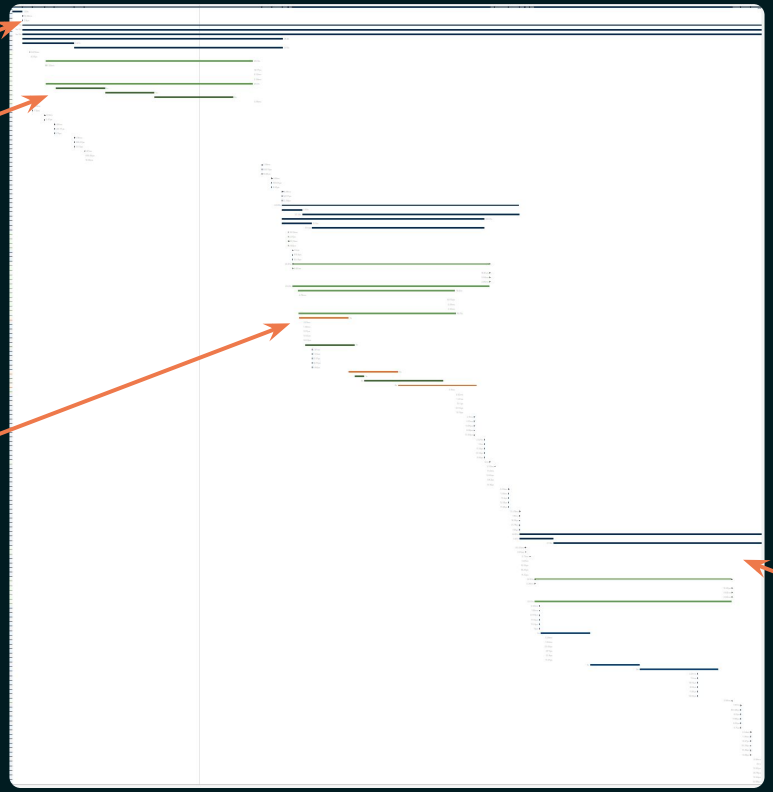
Demo DAG Diamond



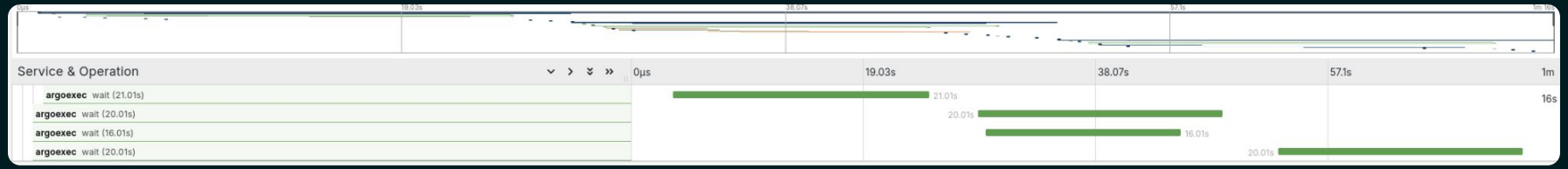
A full trace of the same



A full trace of the same



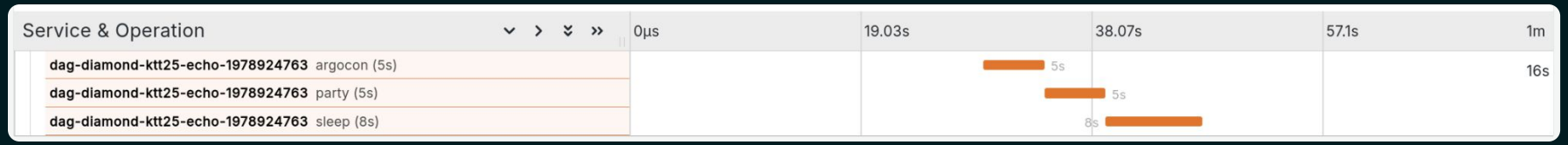
Just the nodes



Tracing inside the pods

Example of tracing in a pod

```
export TRACEPARENT="${ARGO_OTEL_traceparent}"
export OTEL_EXPORTER_OTLP_PROTOCOL=grpc
/otel-cli exec --name argocon sleep 5
/otel-cli exec --name party sleep $((1 + $RANDOM % 5))
/otel-cli exec --name sleep sleep 8
```



Inside the pods

dag-diamond-ktt25-echo-1978924763 argocon (5s)

dag-diamond-ktt25-echo-1978924763 party (5s)

dag-diamond-ktt25-echo-1978924763 sleep (8s)

5s 16s

5s

party

Service: dag-diamond-ktt25-echo-1978924763 | Duration: 5s | Start Time: 34.2s (11:12:57.598) | Kind: client

Status: unset | Library Name: github.com/equinix-labs/otel-cli

Library Version: 0.4.5 0d4b8a9c49f60a6fc25ed22863259ff573332060 2024-04-01T20:56:07Z

Span Attributes

otel	"true"
process.command	"sleep"
process.command_args	["sleep", "5"]
process.owner	"root"
process.parent_pid	42
process.pid	35

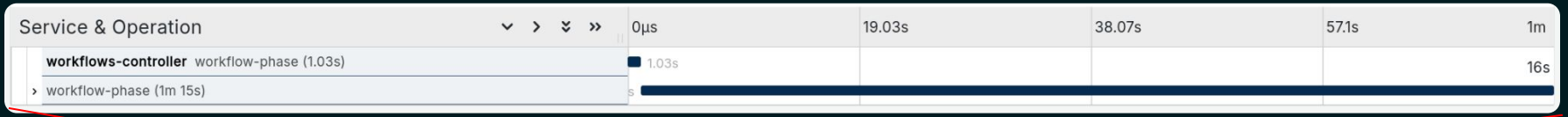
Resource Attributes: k8s.container.name = main | k8s.namespace.name = default | k8s.node.name = k3d-otel-agent-1...

SpanID: 9cc1872397f70408

8s

Level of detail is useful and probably 🤑

Pending workflows



Pending workflows

workflows-controller workflow-phase (1.03s)

1.03s 16s

workflow-phase

Service: workflows-controller | Duration: 1.03s | Start Time: 297.25µs (11:12:23.402) | Kind: internal
Status: unset | Library Name: workflows-controller | Trace State: workflow=default/dag-diamond-ktt25

› Span Attributes: otel = true | phase = Pending

› Resource Attributes: k8s.container.name = workflow-controller | k8s.deployment.name = workflow-controller | k8s.nam...

SpanID: 92daac15eaf8bfda

› workflow-phase (1m 15s)

s

workflow-phase

Service: workflows-controller | Duration: 1m 15s | Start Time: 1.03s (11:12:24.433) | Child Count: 9
Kind: internal | Status: unset | Library Name: workflows-controller
Trace State: workflow=default/dag-diamond-ktt25

› Span Attributes: otel = true | phase = Running

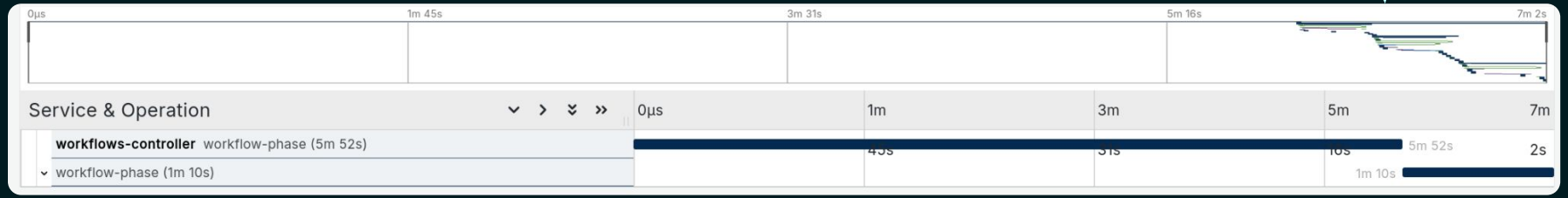
› Resource Attributes: k8s.container.name = workflow-controller | k8s.deployment.name = workflow-controller | k8s.nam...

SpanID: a8223338cab03064

Pending workflows

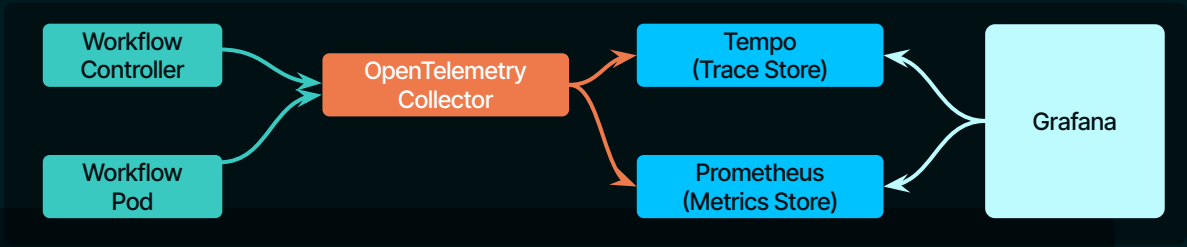
This one takes 7 minutes?

All the action



Span metrics

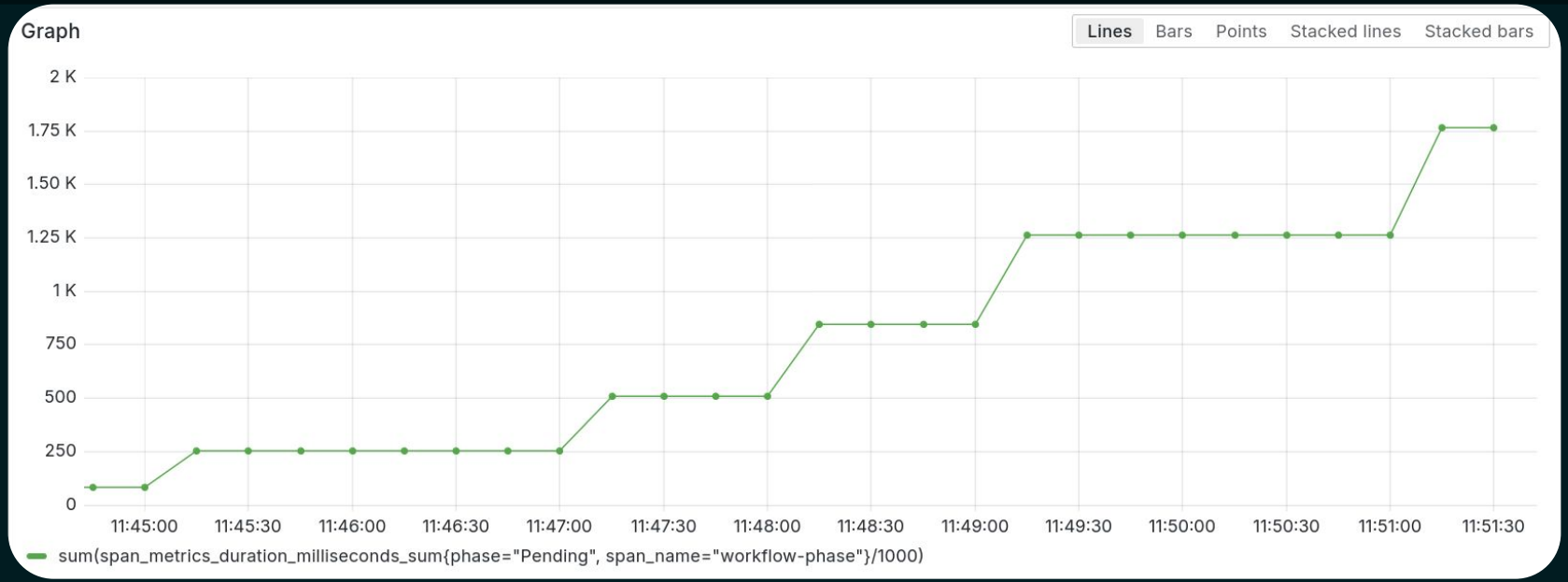
Connectors are specialized components that bridge the different pipelines within the OpenTelemetry Collector.



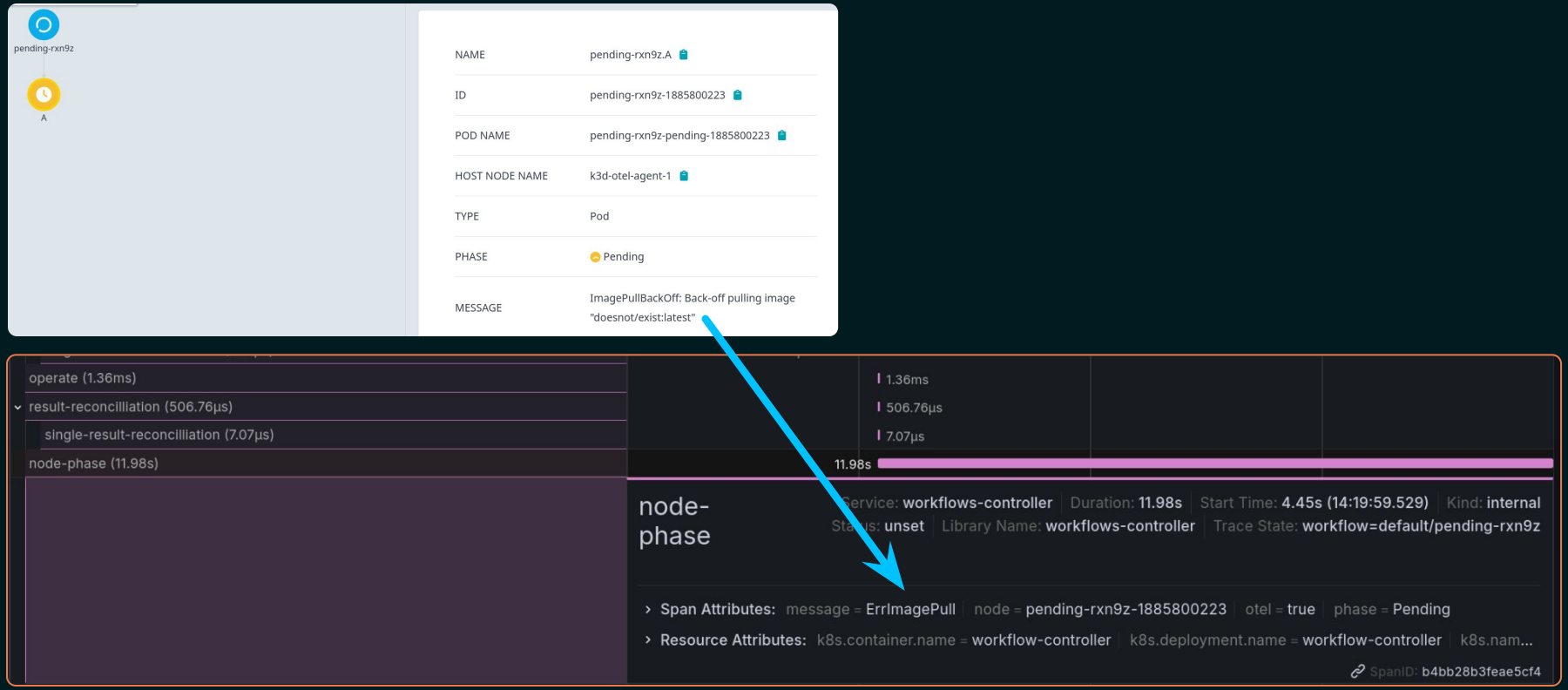

```
connectors:
  spanmetrics:
    namespace: span.metrics
    dimensions:
      - name: phase
```

How much are we limited by parallelism

```
sum(span_metrics_duration_milliseconds_sum{phase="Pending", span_name="workflow-phase"}/1000)
```



Node Pending

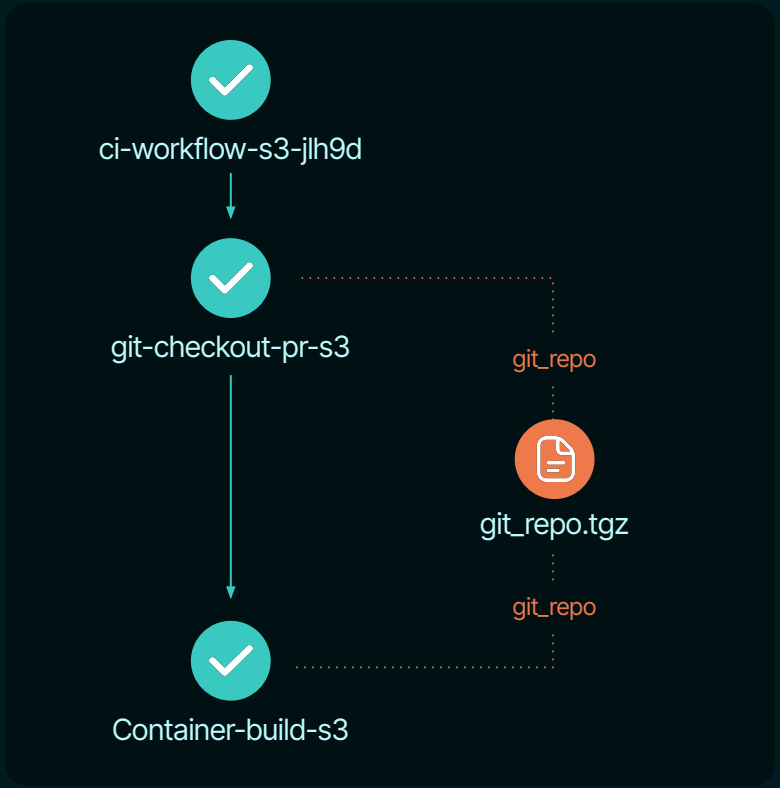


CI Example

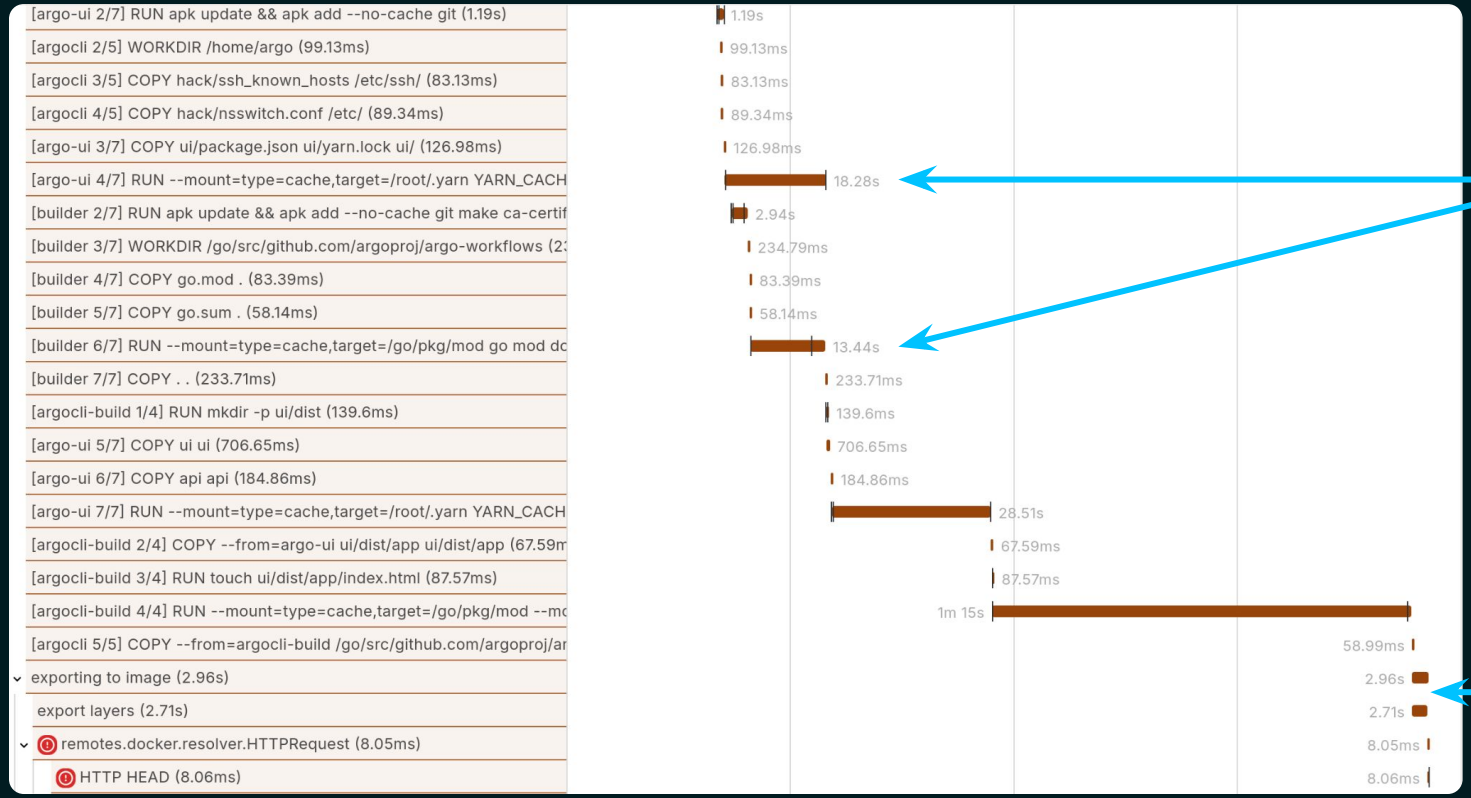
Two steps

1. **Checkout Repo**
   - Store this as an artifact
2. **Build docker image using BuildKit**
   - Repo comes in as an artifact
   - BuildKit has tracing support

Here we are building workflows CLI+UI
This is a multistage Dockerfile



Buildkit output



Parallel build steps

Export

Buildkit output

[builder 6/7] RUN --mount=type=cache,target=/go/pkg/mod go mod dc

13.44s

[builder 6/7] RUN --mount=type=cache,target=/go/pkg/mod go mod download

Service: ci-workflow-s3-7hfds-container-build-2620049006
Duration: 13.44s
Start Time: 32.94s (08:58:58.656) | Kind: internal
Status: unset
Library Name: go.opentelemetry.io/otel/sdk/tracer

Span Attributes

otel	"true"
vertex	"sha256:637780026229faf7a9a7465b8cc522370120c2c6b23e0d8cd09384a203afc1ae"

Resource Attributes: k8s.container.name = main | k8s.namespace.name = default | k8s.node.na...

Events (4)

- 32.94s: message = ExecOp started
- 32.99s: message = Container created
- 32.99s: message = Container started
- 43.79s: message = Container exited | exit.code = 0

Log timestamps are relative to the start time of the full trace.

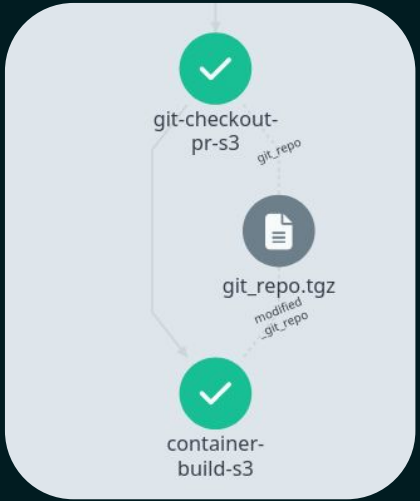
SpanID: e2f82857639010ee

This will get expensive

Artifact transfer time

argoexec load artifacts (13.64µs)	13.64µs
argoexec save artifacts (266.17ms)	266.17ms
argoexec load artifacts (824.49ms)	824.49ms

The artifact here is the git repo.
Curious that it takes 3 times as long to load as it does to save.



Artifacts again

20.57s

load artifacts

Service: argoexec | Duration: 20.57s | Start Time: 47.91s (09:34:34.281)
Kind: internal | Status: unset | Library Name: argoexec
Trace State: workflow=default/artifact-passing-7qhzf

› Span Attributes: otel = true

› Resource Attributes: k8s.container.name = wait | k8s.namespace.name = default | k8s.node.nam...

⌄ Events (2)

› 47.91s: message = download artifact | file = foo

› 51.76s: message = download artifact | file = bar

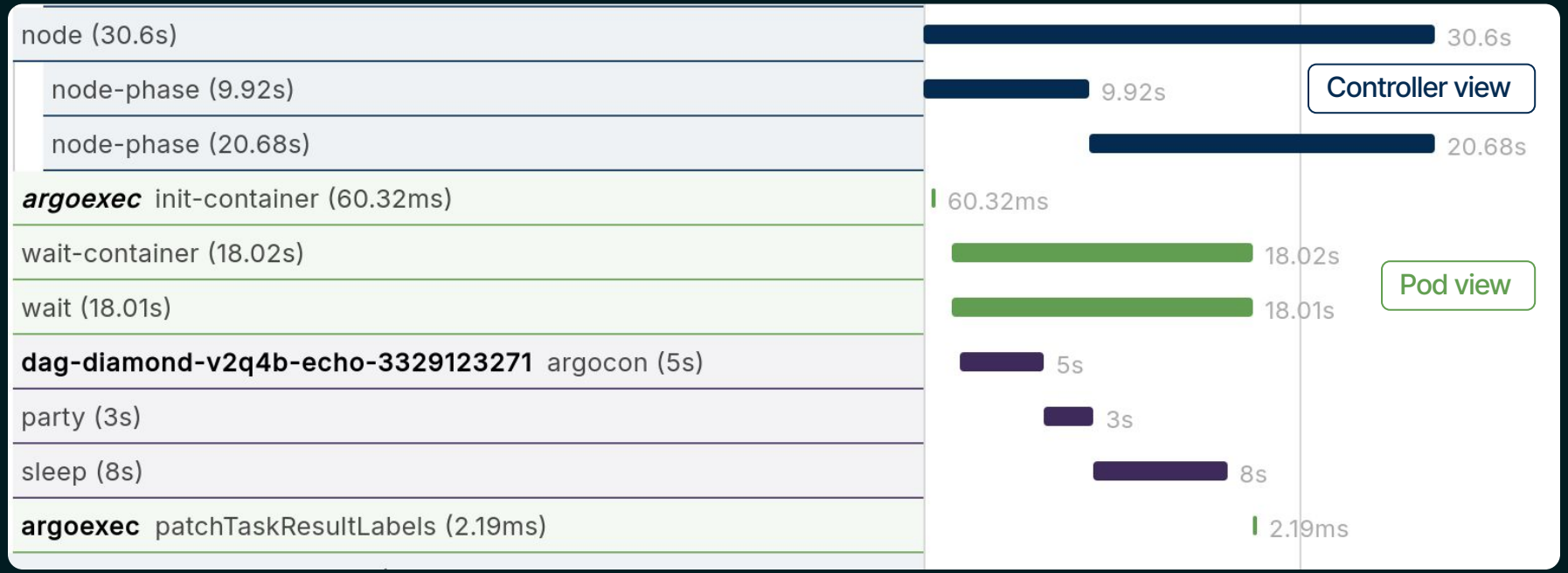
Log timestamps are relative to the start time of the full trace.

SpanID: 23e82e3d80762573

Events: timestamp + data within a span

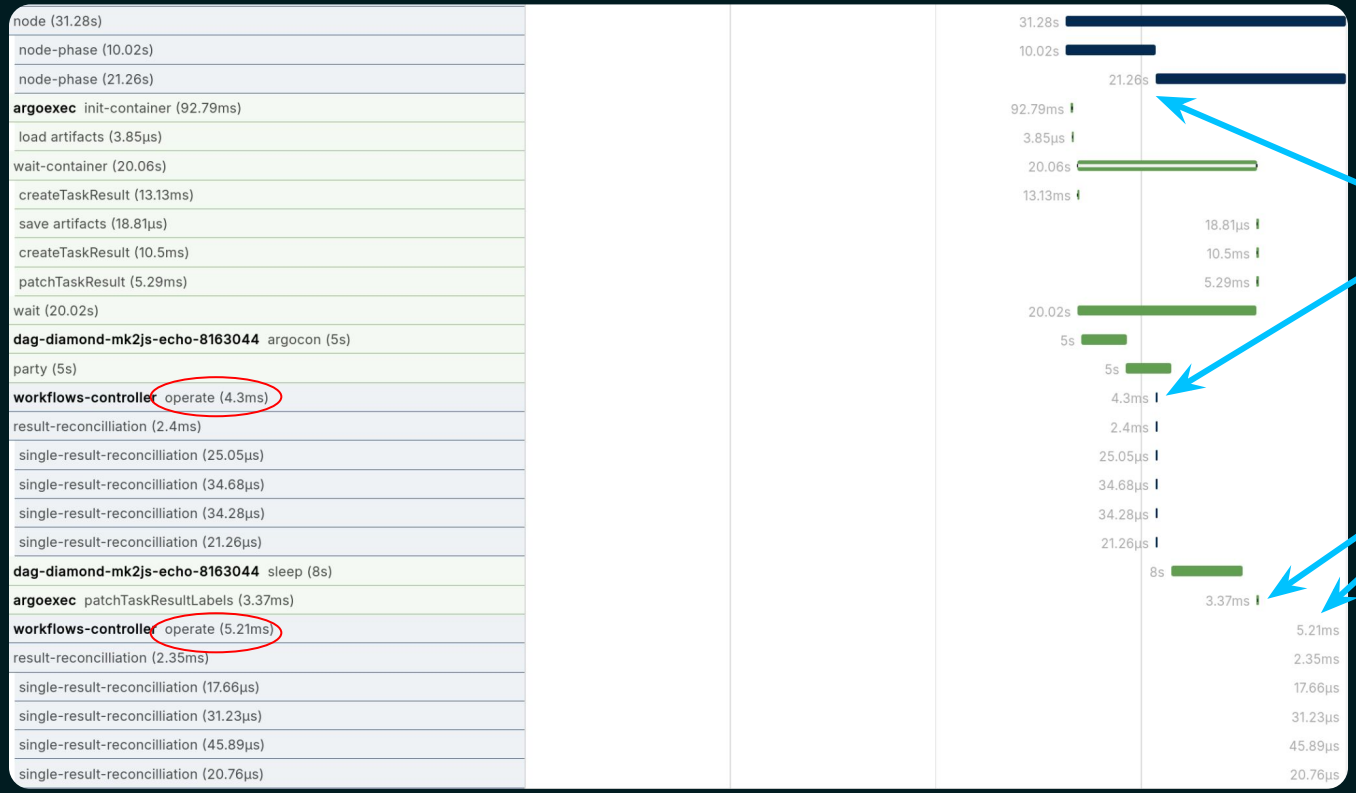
As a developer I'd like to know more...

Going back to the initial DAG Diamond...



ArgoCon North America

As a developer I'd like to know more...



Reconcile

Why this delay

What next

GitHub

https://github.com/pipekit/argo-workflows/tree/tracing-base
https://github.com/Joibel/otel-deploy

Aiming for release in 3.7
How do we control traces - can be super noisy
Which things get traced?
Turn it on for this workflow only.
Re-submit with tracing on?
Standards compliance - semconv

- https://opentelemetry.io/docs/specs/semconv/attributes-registry/cicd/

Issues during implementation

OpenTelemetry operator doesn't annotate initContainers

Issue #3308

OpenTelemetry SDK expects spans to start and end in the same running binary

Not necessarily the case with the controller restarting

About Pipekit

pipekit

Scale Argo & Kubernetes with Pipekit

- Direct support from 40% of the active Argo Workflows maintainers in the world
- Save engineering time and up to 60% on compute costs
- Add 3 Argo maintainers and 7 Argo contributors to your team
- Serving startups & Fortune 500 enterprises since 2021:

Enterprise Support for Argo:
Ideal for Platform Eng teams scaling with Argo

Control Plane for Argo Workflows:
Ideal for data teams, granular RBAC, and multi-cluster architectures

#argocon

ArgoCon
North America

Find us at the Argo stand in the Project Pavillion

Free Argo/Infrastructure Help & Advice:

- Booth T33
- Regular Office Hours @pipekit.io/office-hours