Managing Environments with GitOps

Andrew Morgan
INDEPENDENT
@mogronalol

Overview

Problems with traditional deployment strategies

Jenkins X environment repositories and GitOps

A deep dive into jx deployments

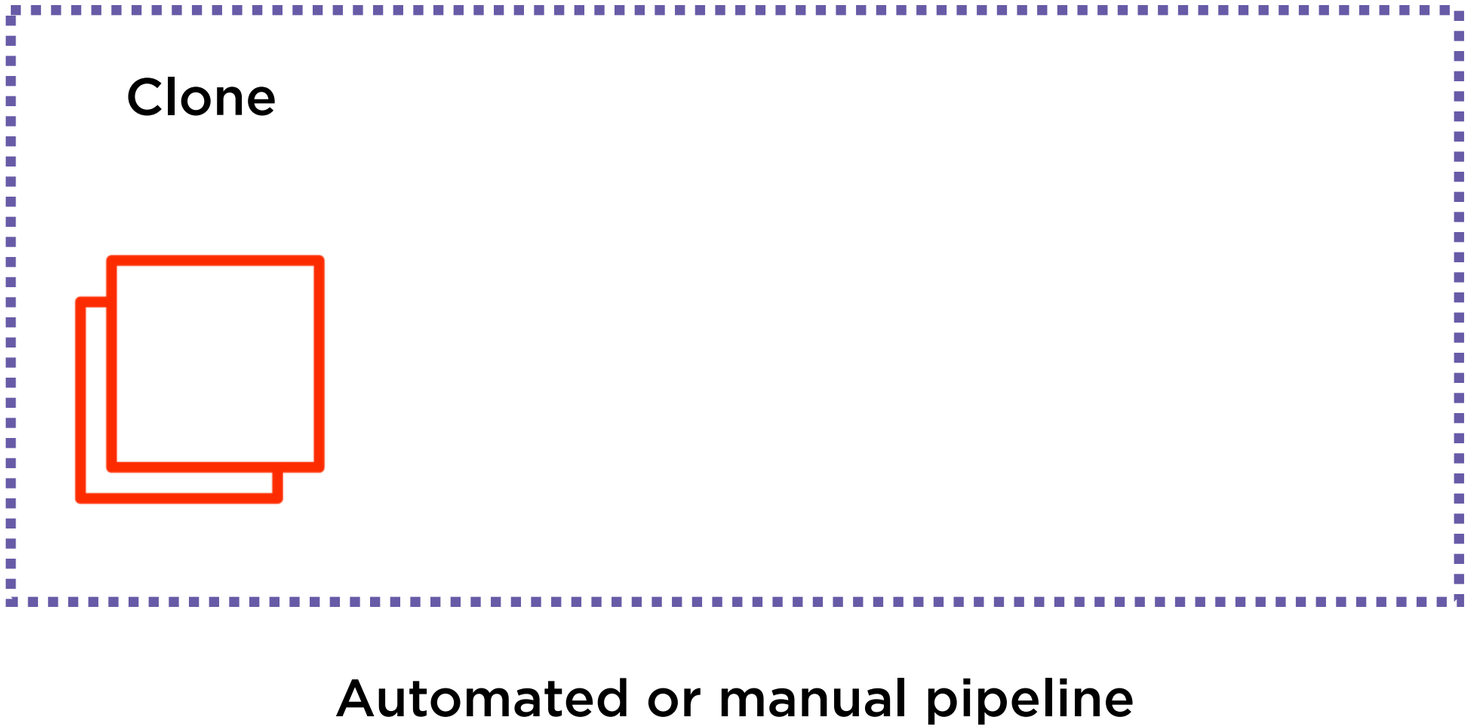
Creating new environments

Environment promotion

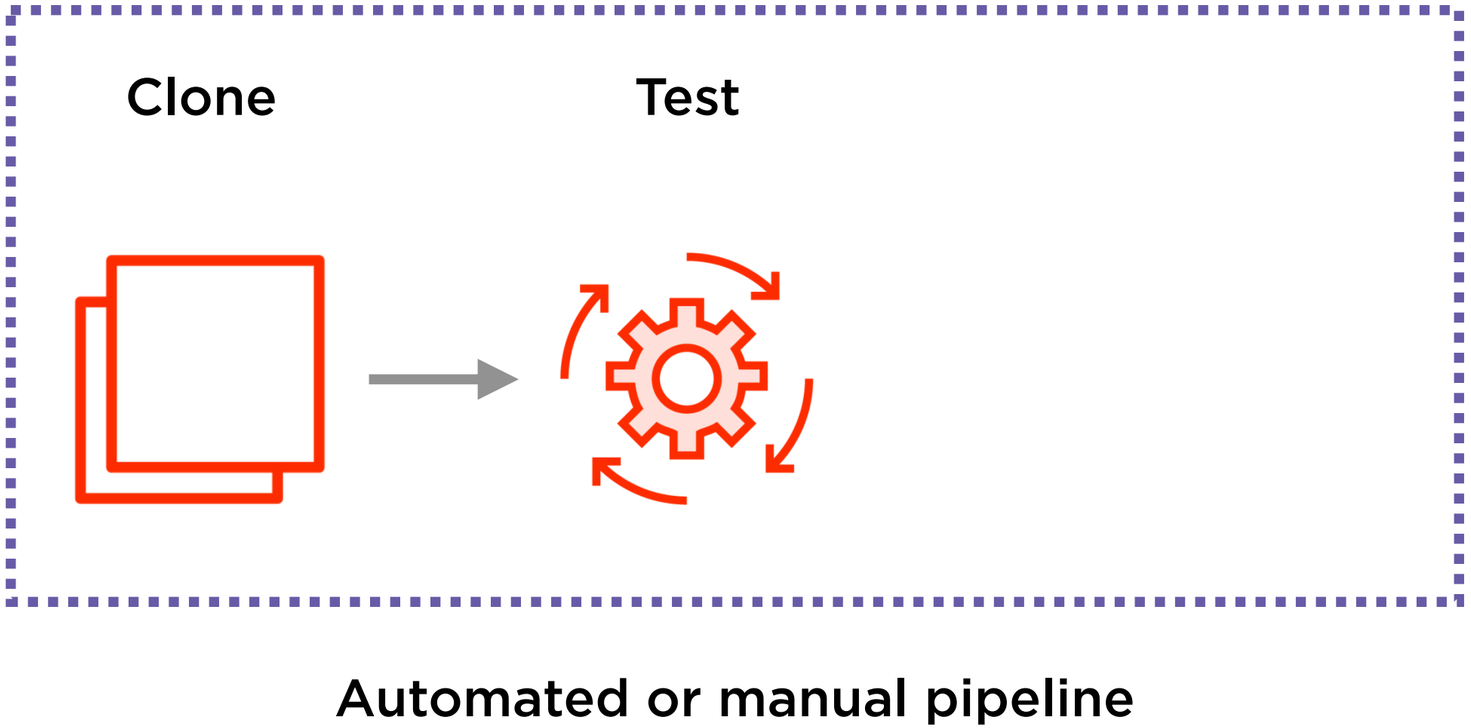
Manual Deployment Strategy Overview

Automated or manual pipeline

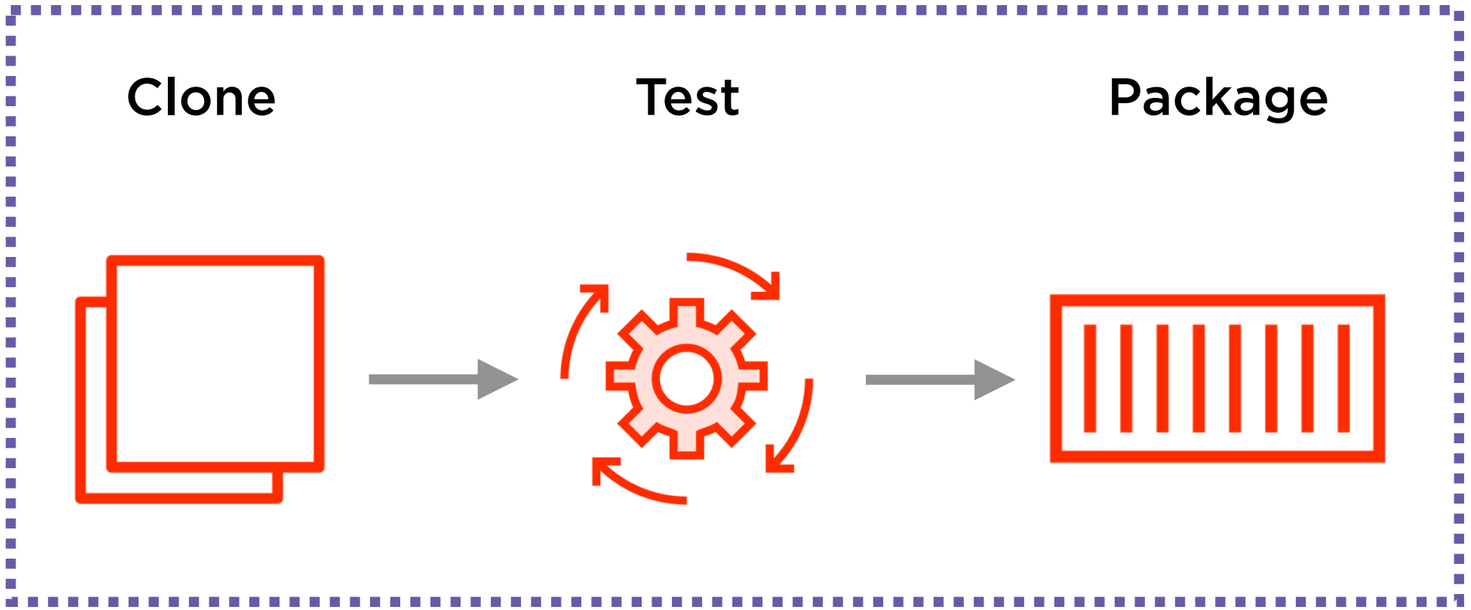
Manual Deployment Strategy Overview



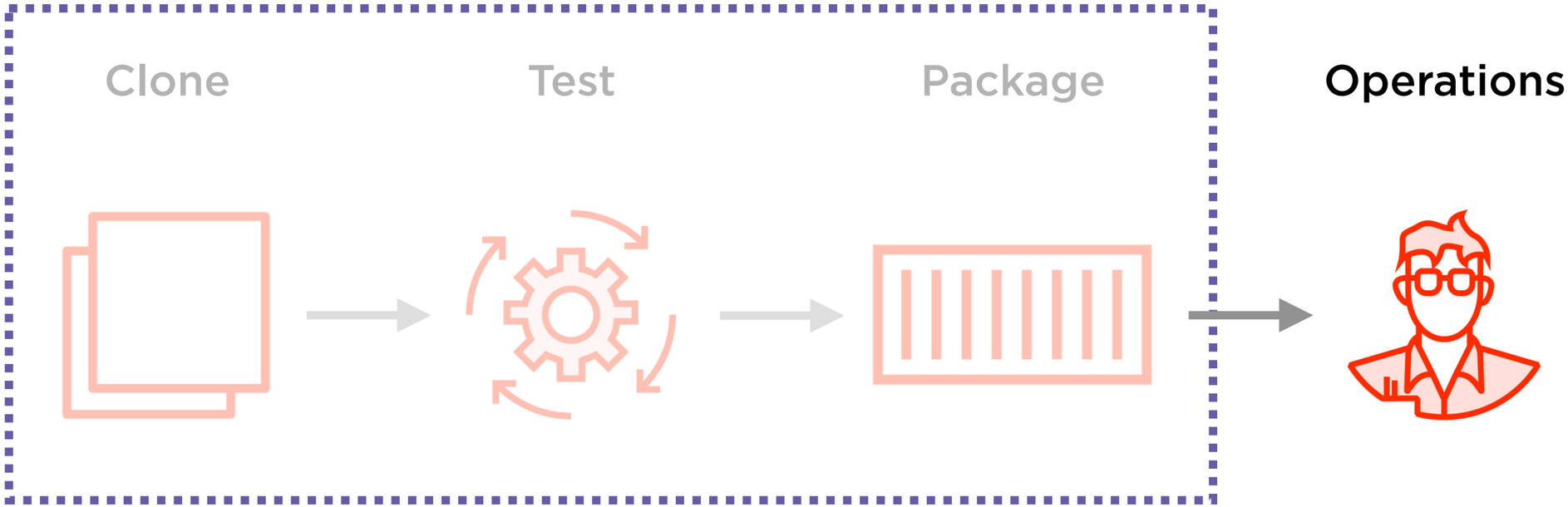
Manual Deployment Strategy Overview



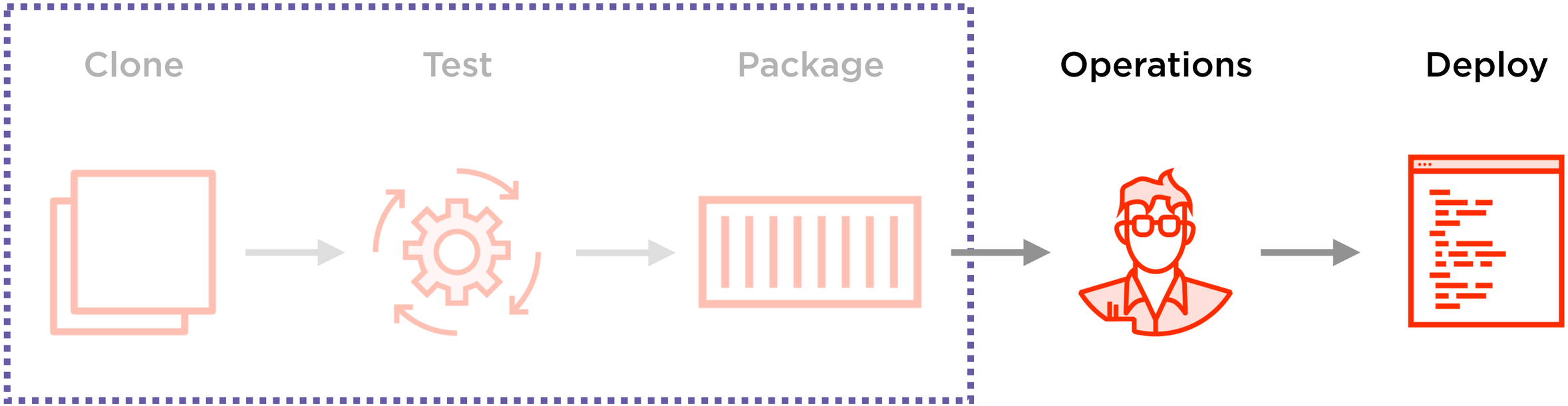
Manual Deployment Strategy Overview



Manual Deployment Strategy Overview



Manual Deployment Strategy Overview

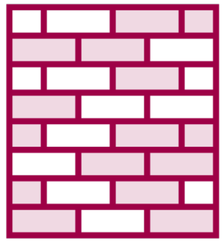


Problems with Manual Deployments

Manual deployments mean human error

Problems with Manual Deployments

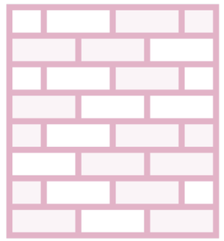
Manual deployments mean human error



Creating a silo between developers and operations

Problems with Manual Deployments

Manual deployments mean human error



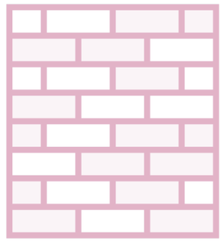
Creating a silo between developers and operations

Not observable

Problems with Manual Deployments

Manual deployments mean human error

Cannot easily recreate an environment or rollback



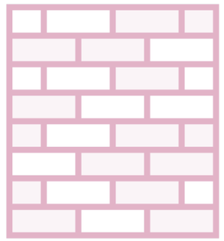
Creating a silo between developers and operations

Not observable

Problems with Manual Deployments

Manual deployments mean human error

Cannot easily recreate an environment or rollback



Creating a silo between developers and operations

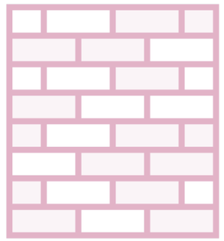
End-to-end testing

Not observable

Problems with Manual Deployments

Manual deployments mean human error

Cannot easily recreate an environment or rollback



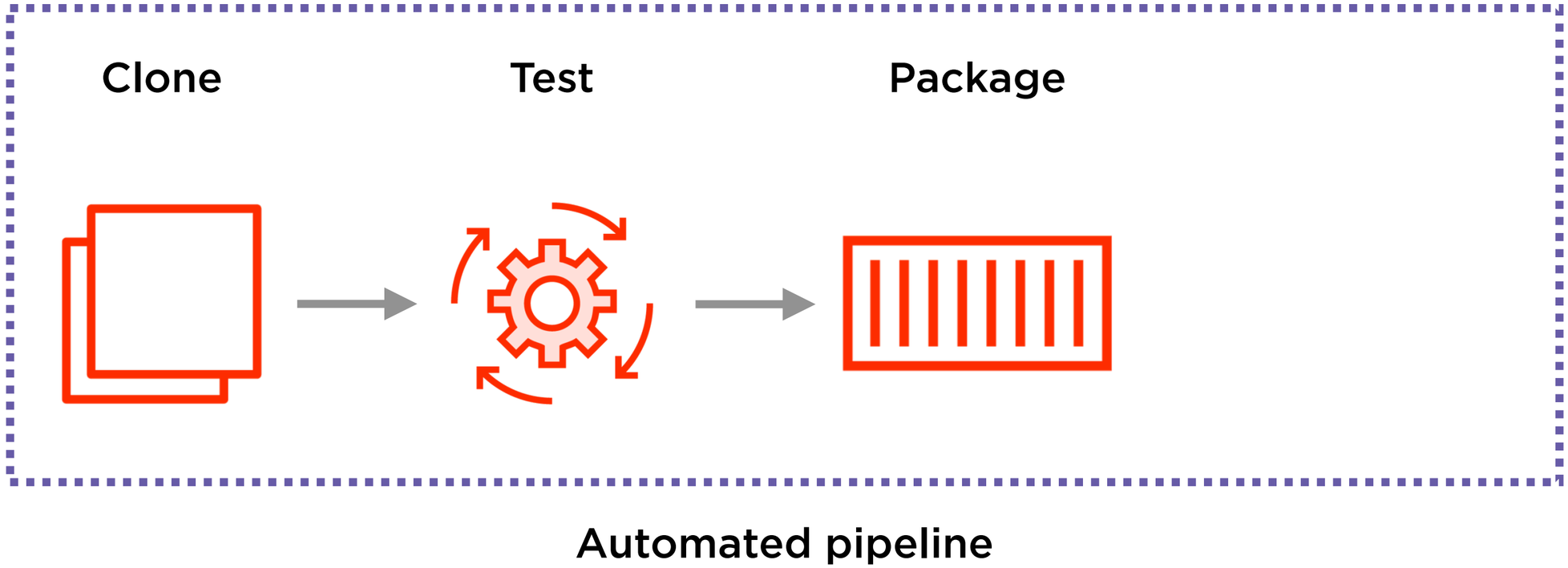
Creating a silo between developers and operations

End-to-end testing

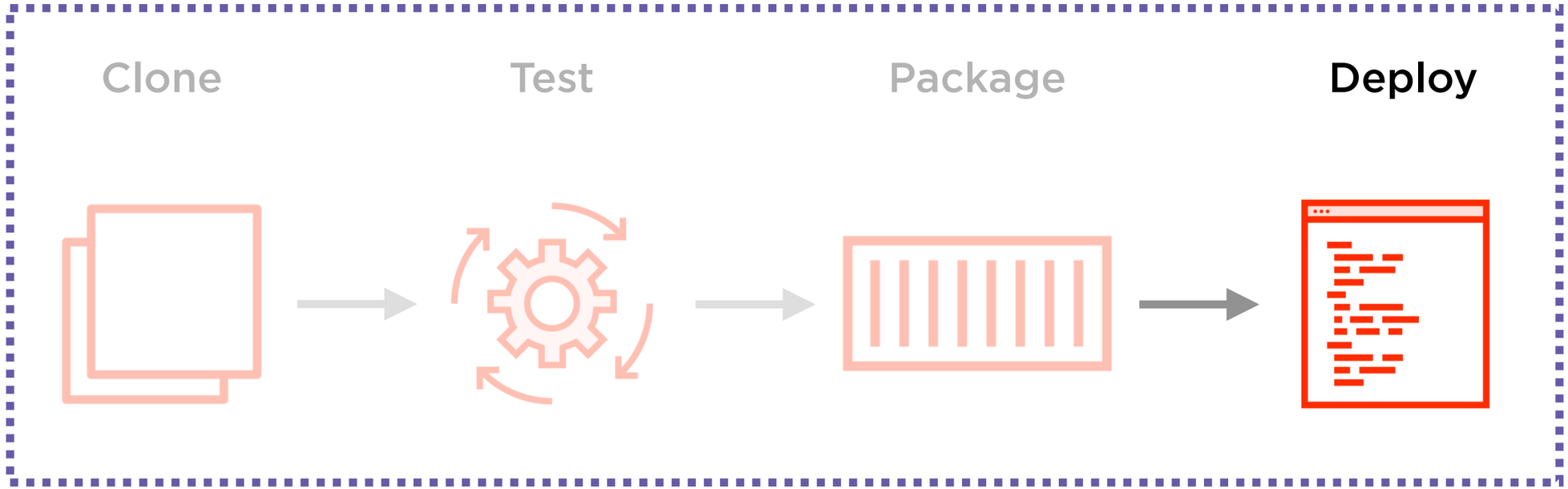
Not observable

May not be declarative or idempotent

Automated Deployment Strategy Overview

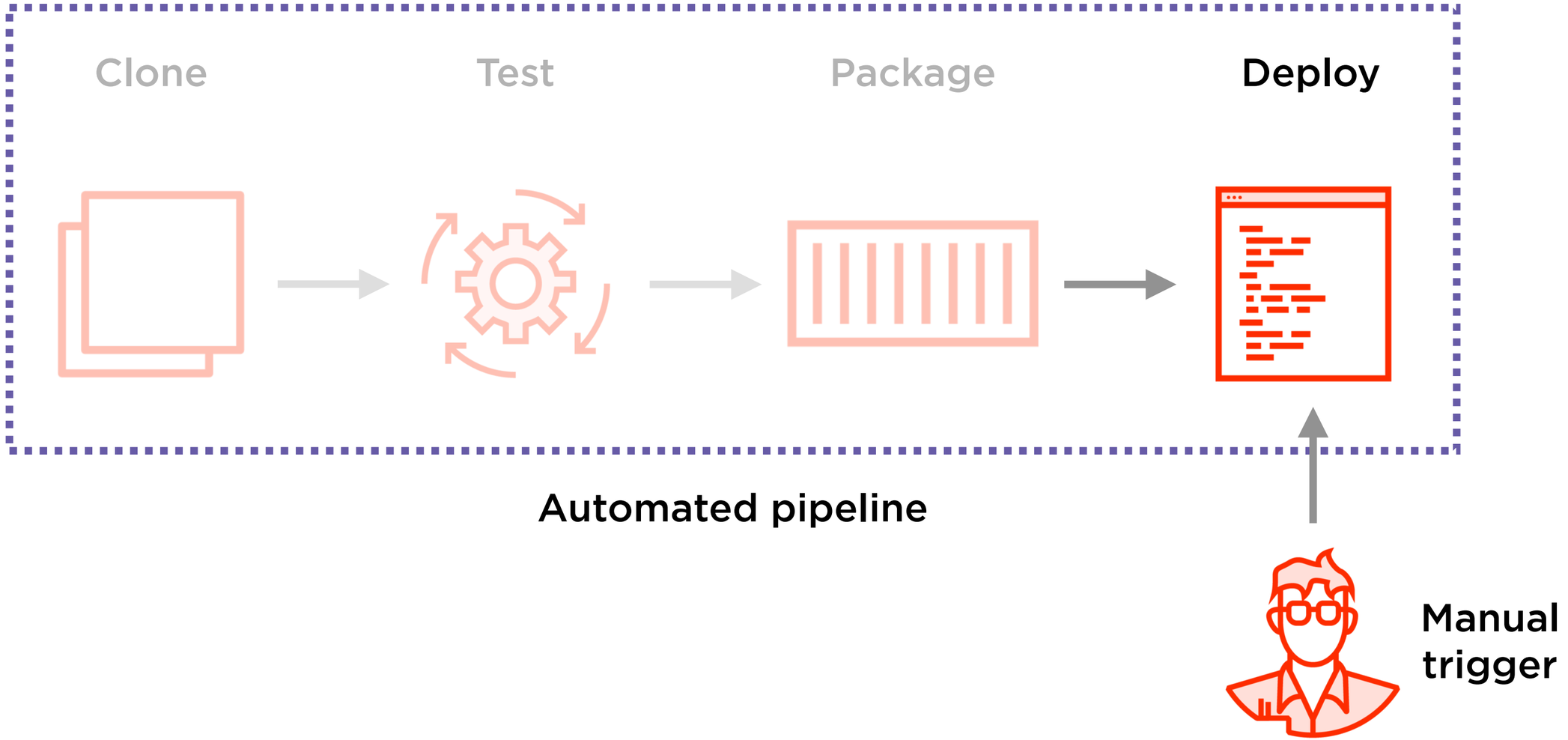


Automated Deployment Strategy Overview



Automated pipeline

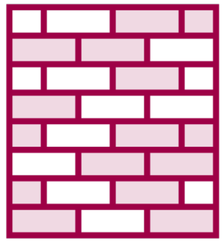
Automated Deployment Strategy Overview



Problems with Automated Deployments

Manual deployments mean human error

Cannot easily recreate an environment or rollback



Creating a silo between developers and operations

End-to-end testing

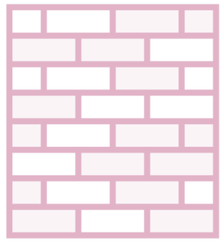
Not observable

May not be declarative or idempotent

Problems with Automated Deployments

Automated, so no mistakes

Cannot easily recreate an environment or rollback



Creating a silo between developers and operations

End-to-end testing

Not observable

May not be declarative or idempotent

Problems with Automated Deployments

Automated, so no mistakes

Cannot easily recreate an environment or rollback

Semi-silo between developers and operations

End-to-end testing

Not observable

May not be declarative or idempotent

Problems with Automated Deployments

Automated, so no mistakes

Cannot easily recreate an environment or rollback

Semi-silo between developers and operations

End-to-end testing

Semi-observable

May not be declarative or idempotent

Problems with Automated Deployments

Automated, so no mistakes

Cannot easily recreate an environment or rollback

Semi-silo between developers and operations

End-to-end testing

Semi-observable

May not be declarative or idempotent

Problems with Automated Deployments

Automated, so no mistakes

Cannot easily recreate an environment or rollback

Semi-silo between developers and operations

End-to-end testing

Semi-observable

May not be declarative or idempotent

Problems with Automated Deployments

Automated, so no mistakes

Cannot easily recreate an environment or rollback

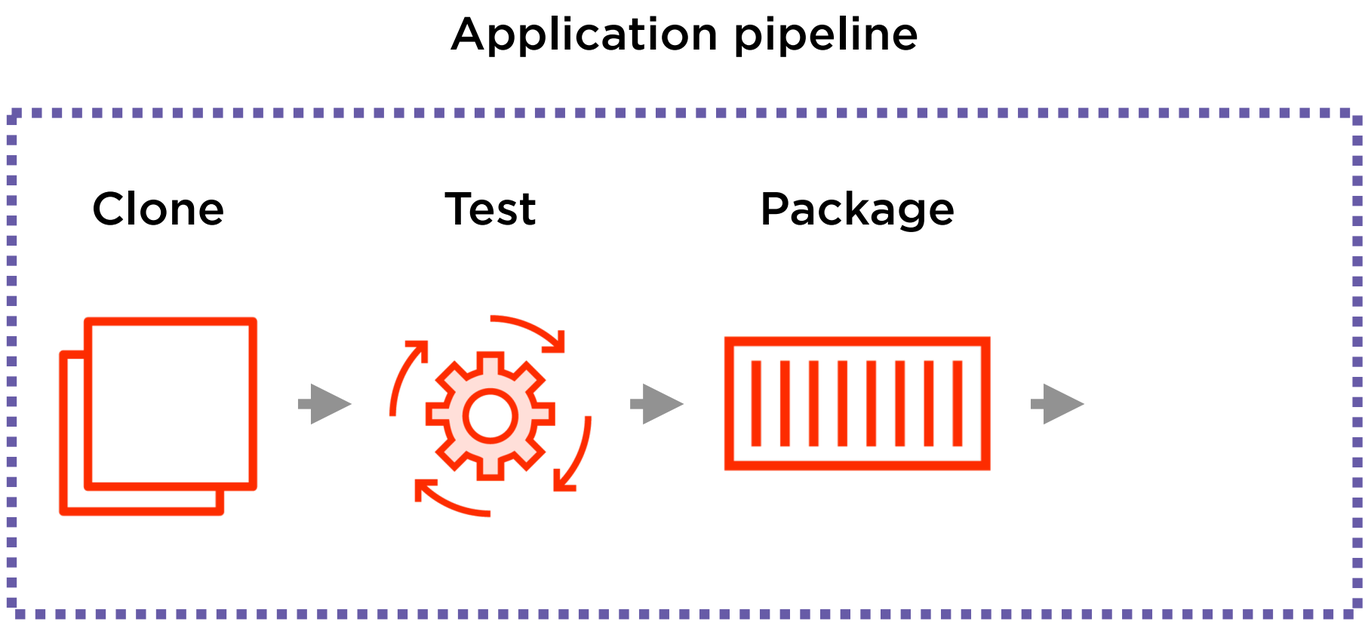
Semi-silo between developers and operations

End-to-end testing

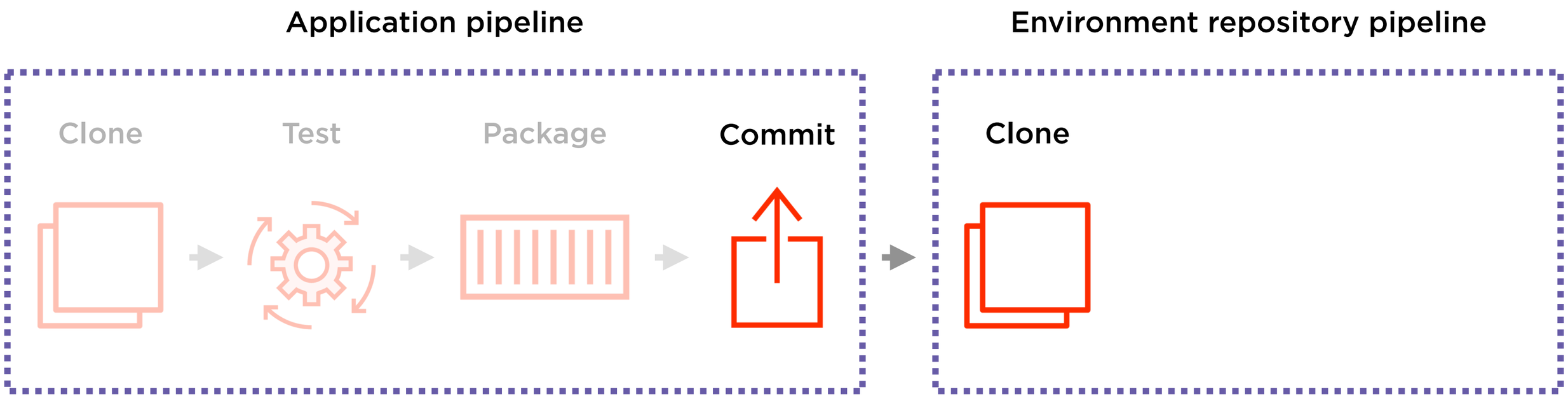
Semi-observable

May not be declarative or idempotent

GitOps Deployment Strategy Overview



GitOps Deployment Strategy Overview



Jenkins X Environment Repository

requirements.yaml

```
dependencies:
- alias: user
  name: user-service
  repository: http://chartmuseum:8080
  version: 2.3.118
- alias: billing
  name: billing-service
  repository: http://chartmuseum:8080
  version: 2.3.118
```

Jenkins X Environment Repository

requirements.yaml

```
dependencies:
- alias: user
  name: user-service
  repository: http://chartmuseum:8080
  version: 2.3.118
- alias: billing
  name: billing-service
  repository: http://chartmuseum:8080
  version: 2.3.118
```

Jenkins X Environment Repository

requirements.yaml

```
dependencies:
- alias: user
  name: user-service
  repository: http://chartmuseum:8080
  version: 2.3.118
- alias: billing
  name: billing-service
  repository: http://chartmuseum:8080
  version: 2.3.118
```

Jenkins X Environment Repository

requirements.yaml

```
dependencies:
- alias: user
  name: user-service
  repository: http://chartmuseum:8080
  version: 2.3.118
- alias: billing
  name: billing-service
  repository: http://chartmuseum:8080
  version: 2.3.118
```

Jenkins X Environment Repository

requirements.yaml

```
dependencies:
- alias: user
  name: user-service
  repository: http://chartmuseum:8080
  version: 2.3.118
- alias: billing
  name: billing-service
  repository: http://chartmuseum:8080
  version: 2.3.118
```

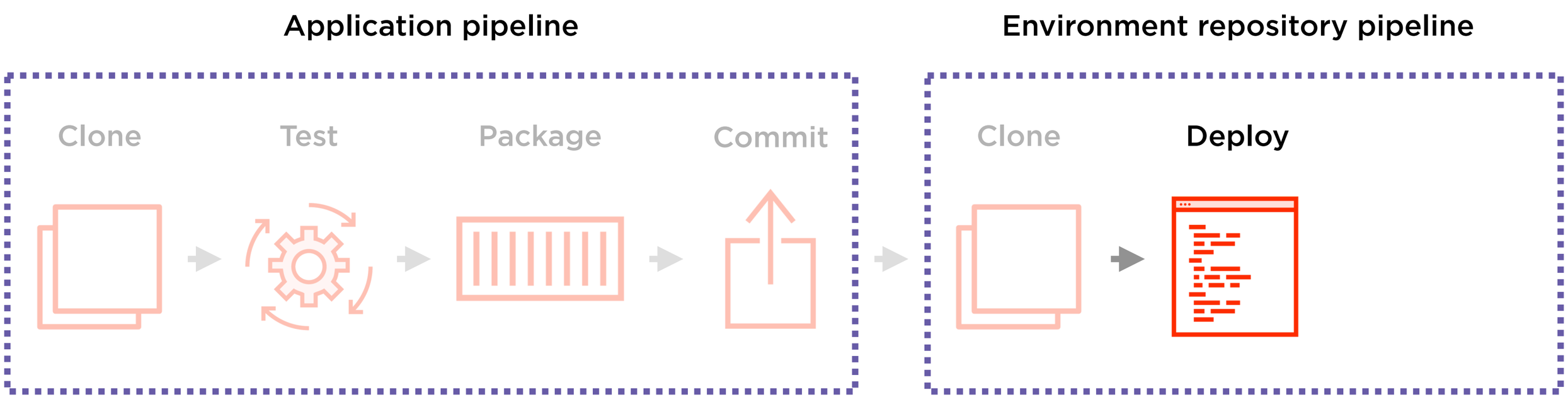
Desired state of an environment

Declarative list of services using Helm

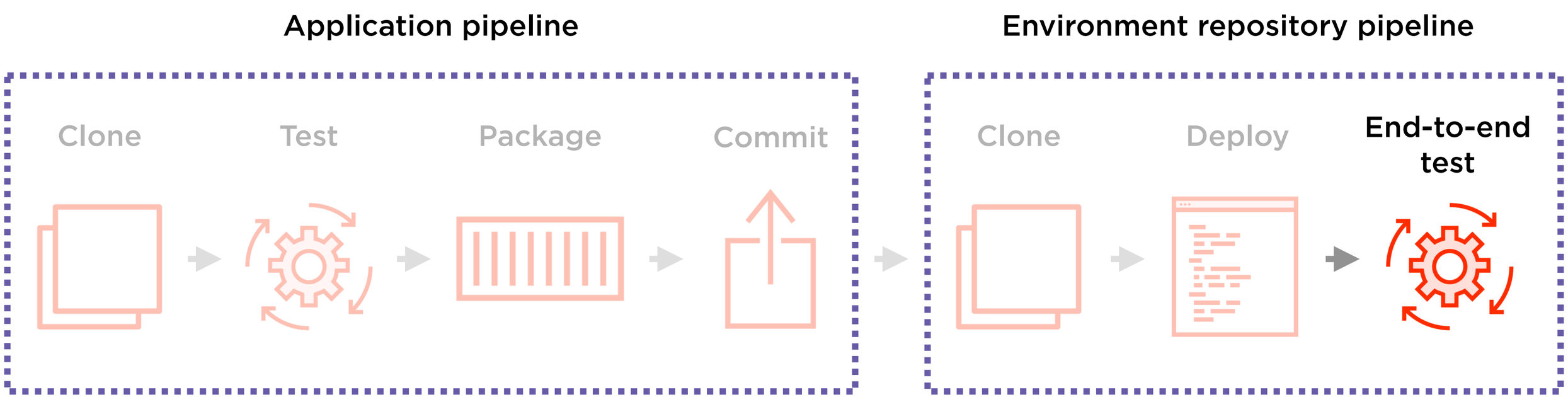
Each commit triggers a pipeline which deploys everything

Kubernetes deployments are declarative and idempotent

GitOps Deployment Strategy Overview



GitOps Deployment Strategy Overview



Advantages of Environment Repository

Automated

Cannot easily recreate an environment or rollback

Creating a silo between developers and operations

End-to-end testing

Semi-observable

May not be declarative or idempotent

Advantages of Environment Repository

Automated

Cannot easily recreate an environment or rollback

Developers have full control of packaging and deployments

End-to-end testing

Semi-observable

May not be declarative or idempotent

Advantages of Environment Repository

Automated

Developers have full control of packaging and deployments

Full observability through Git

Cannot easily recreate an environment or rollback

End-to-end testing

May not be declarative or idempotent

Advantages of Environment Repository

Automated

Native Git can easily rollback, deploy groups, create new environments, etc.

Developers have full control of packaging and deployments

End-to-end testing

Full observability through Git

May not be declarative or idempotent

Advantages of Environment Repository

Automated

Native Git can easily rollback, deploy groups, create new environments, etc.

Developers have full control of packaging and deployments

Each deployment can end-to-end test all services

Full observability through Git

May not be declarative or idempotent

Advantages of Environment Repository

Automated

Native Git can easily rollback, deploy groups, create new environments, etc.

Developers have full control of packaging and deployments

Each deployment can end-to-end test all services

Full observability through Git

Natively declarative and idempotent deployments

GitOps

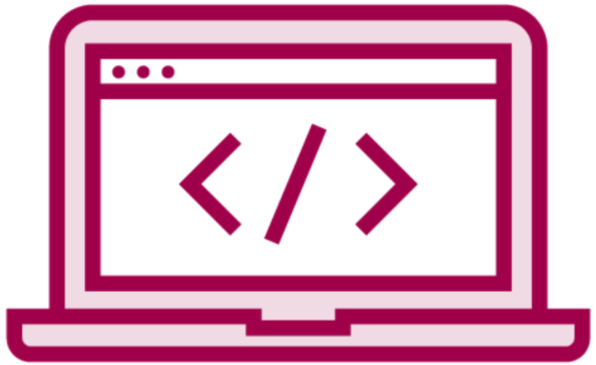
Where Git is the single source of truth for an environment, meaning all changes are observable and declarative

Demo

Deploy an application to staging

Analysis of staging repository

Using JX to view the application and staging pipelines



jx create env

Automated creation of a new environment

Creates Git repository and sets up pipelines

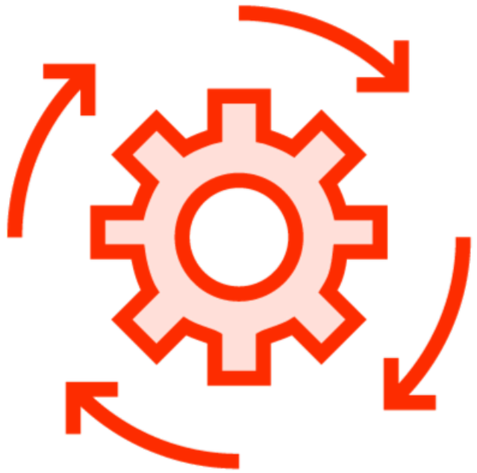
Creates namespace within Kubernetes

Environment Creation Strategy Advantages

Can be done by developers

Environment Creation Strategy Advantages

Can be done by developers



Fast and fully automated process

Environment Creation Strategy Advantages

Can be done by developers

Fast and fully automated process

Completely observable through Git

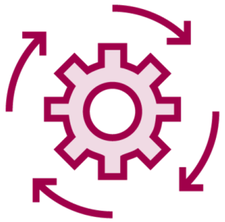
Demo

Create a new environment

Analysis of new repository

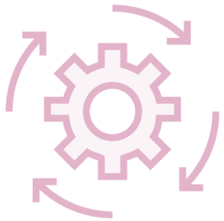
Deploy an application to our environment

Environment Promotion Strategies



Auto : All new releases are automatically promoted

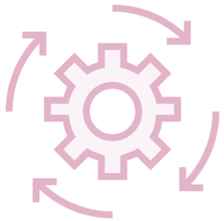
Environment Promotion Strategies



Auto : All new releases are automatically promoted

Manual : Releases are promoted manually

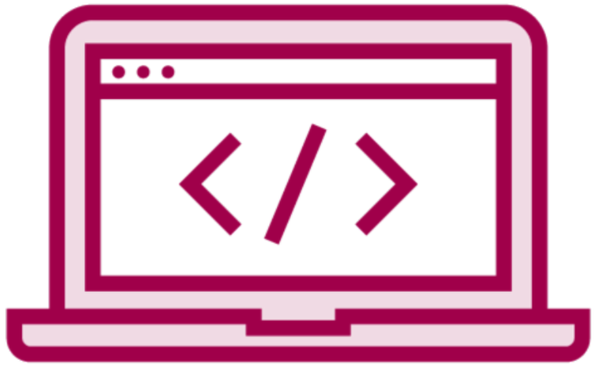
Environment Promotion Strategies



Auto : All new releases are automatically promoted

Manual : Releases are promoted manually

Never : Releases are never promoted to the environment



jx promote

Deploy a service to an environment

Orchestrated via Git commits behind the scenes

Useful for manual releases to production

Demo

Look at our environment's promotion strategies

Promote an application to production

Summary

Jenkins X adopts GitOps, where Git is the single-source-of-truth for our environments

Deployments are declarative commits to these repositories

We automatically deploy to auto environments

We manually deploy to manual environments