Logging, Scale, and HA

Agenda

1. Cloud Native Apps
2. Elastic Runtime Architecture
3. High Availability

FUNDAMENTAL CHANGES

Distributed systems are hard to build, test, manage, and scale.

Ephemeral Infrastructure

Virtual machines and containers are temporary.

Immutable Infrastructure

Updates to systems and applications are not done in-place but rather new, updated instances are created instead.

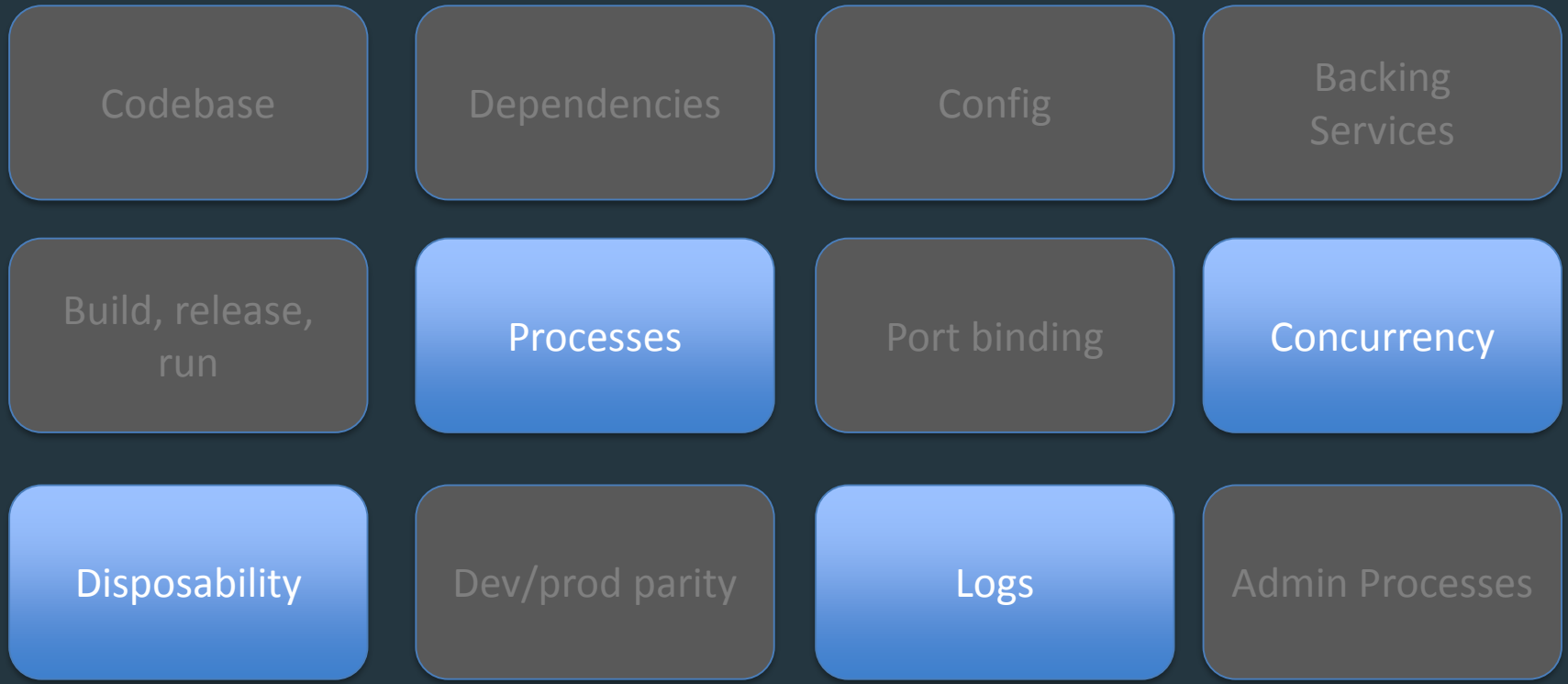
CHANGES IN APP DESIGN

The 12 Factor App

Methodology for building web apps suitable for running on cloud platforms.

http://12factor.net/

The 12 Factors

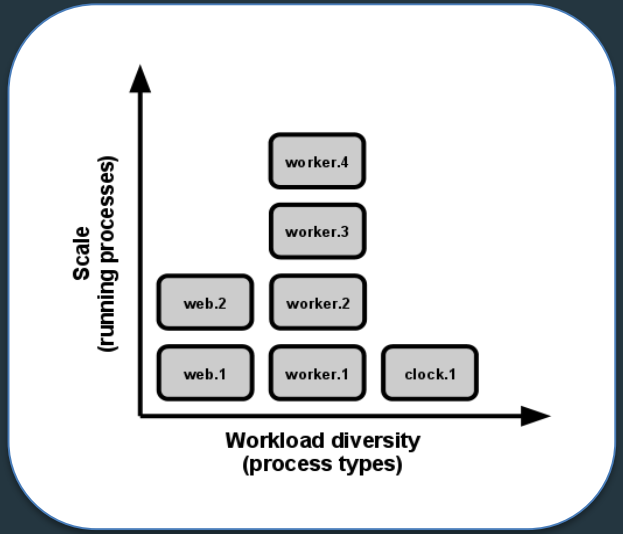


Processes

Execute the app as one or more stateless processes.

http://12factor.net/processes

Scale out via the process model.



http://12factor.net/concurrency

Disposability

Maximize robustness with fast startup and graceful shutdown.

http://12factor.net/disposability

Logs

Treat logs as event streams.

http://12factor.net/logs

Agenda

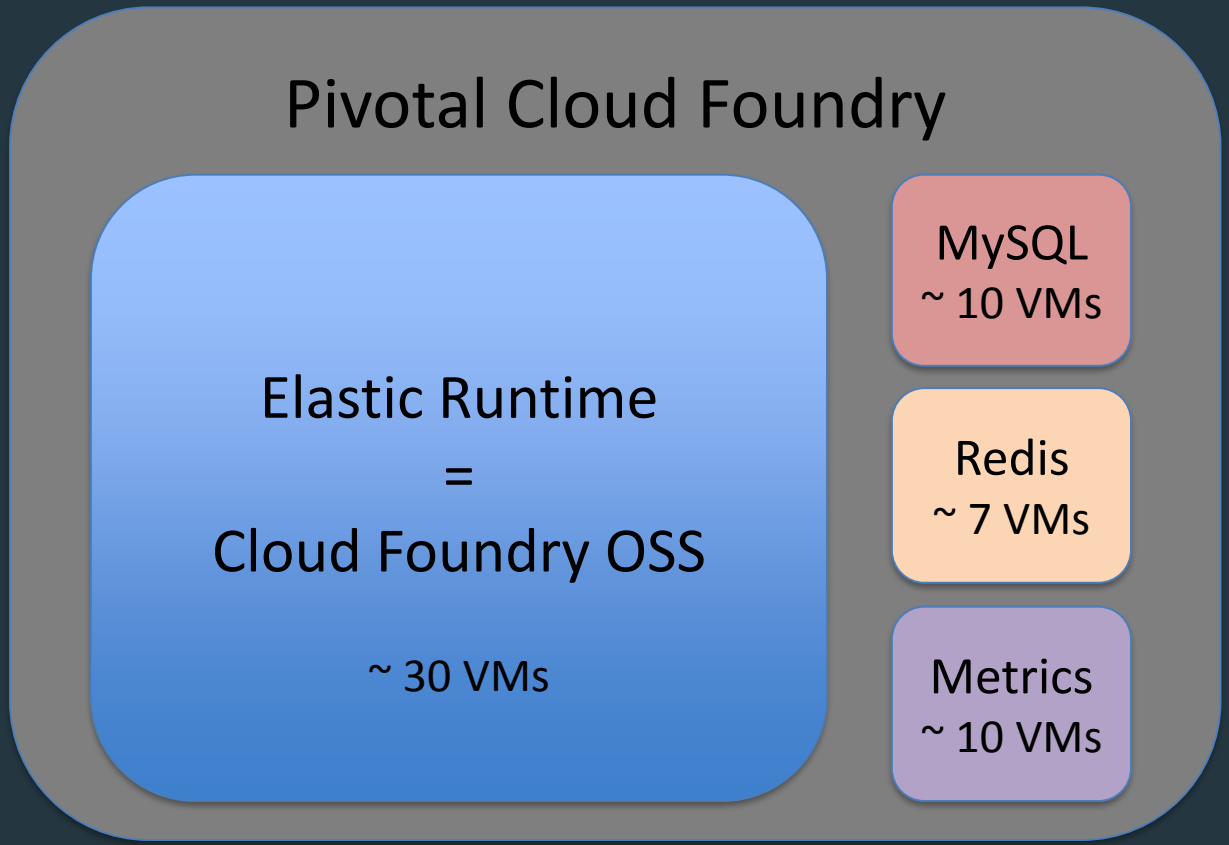
1. Cloud Native Apps
2. Elastic Runtime Architecture
3. High Availability

What is the Elastic Runtime?

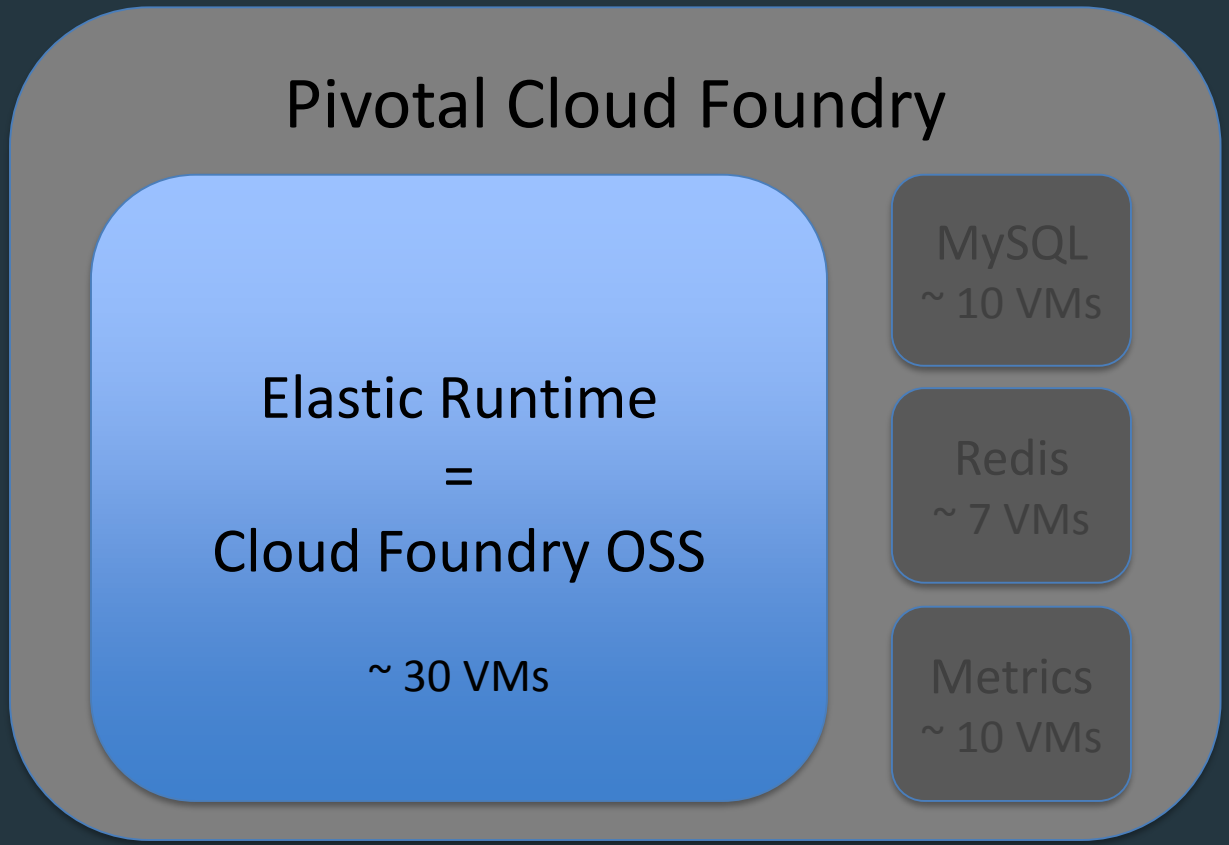
What is the Elastic Runtime?

The Elastic Runtime is Cloud Foundry.

Pivotal Cloud Foundry Simplified



Pivotal Cloud Foundry Simplified



Elastic Runtime Architecture

- Elastic Runtime Subsystems
  - Diego
  - Loggregator
  - Cloud Controller API
  - Routing
- Key Sequence Flows through the ER

Elastic Runtime
=
Cloud Foundry OSS

~ 30 VMs

DIEGO

https://docs.pivotal.io/pivotalcf/concepts/diego/diego-architecture.html

Diego

Schedules tasks and Long-Running Processes (LRPs).

Diego: Task

Is guaranteed to run at *most once*.

e.g. stage an application

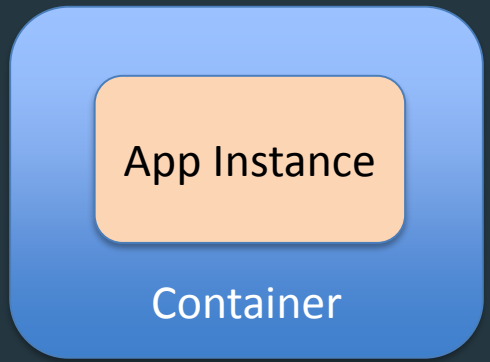
Diego: LRP

Is a Long-Running Process, typically represented as a web app.

LRPs can have multiple instances.

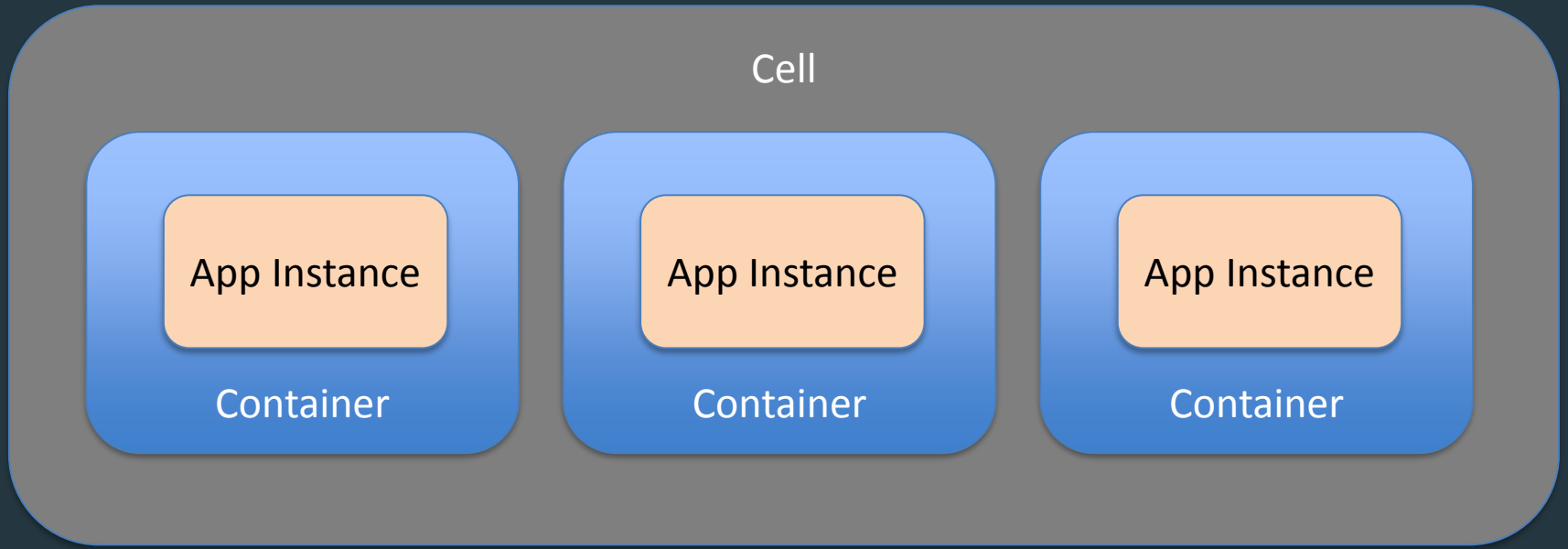
Diego: Container

An application instance is run within an immutable container.

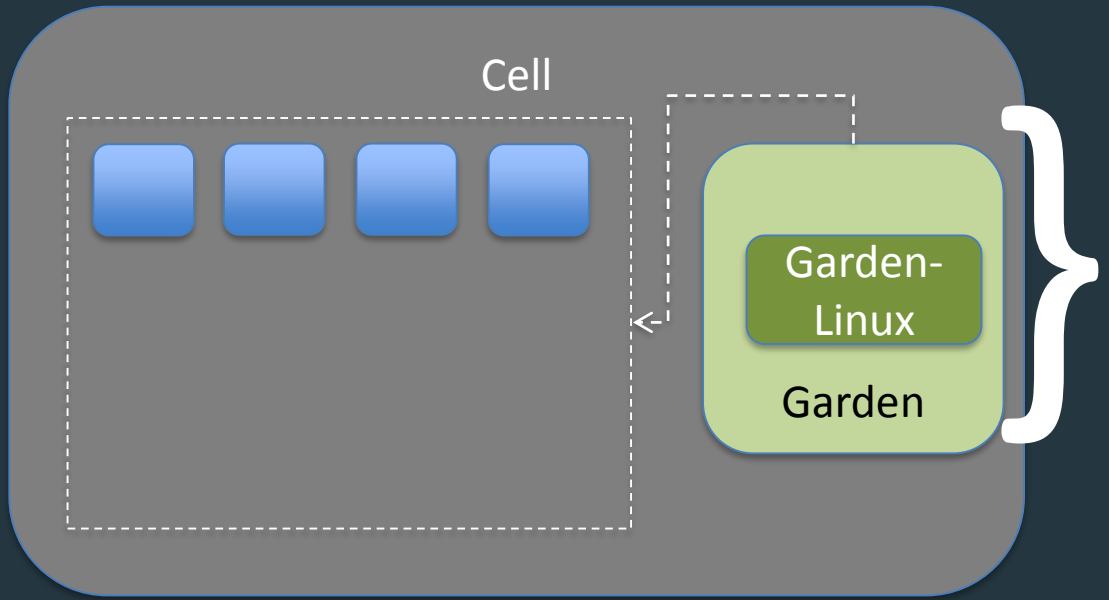


Diego: Cell

Containers are run within a cell.



Containers are managed by Garden.

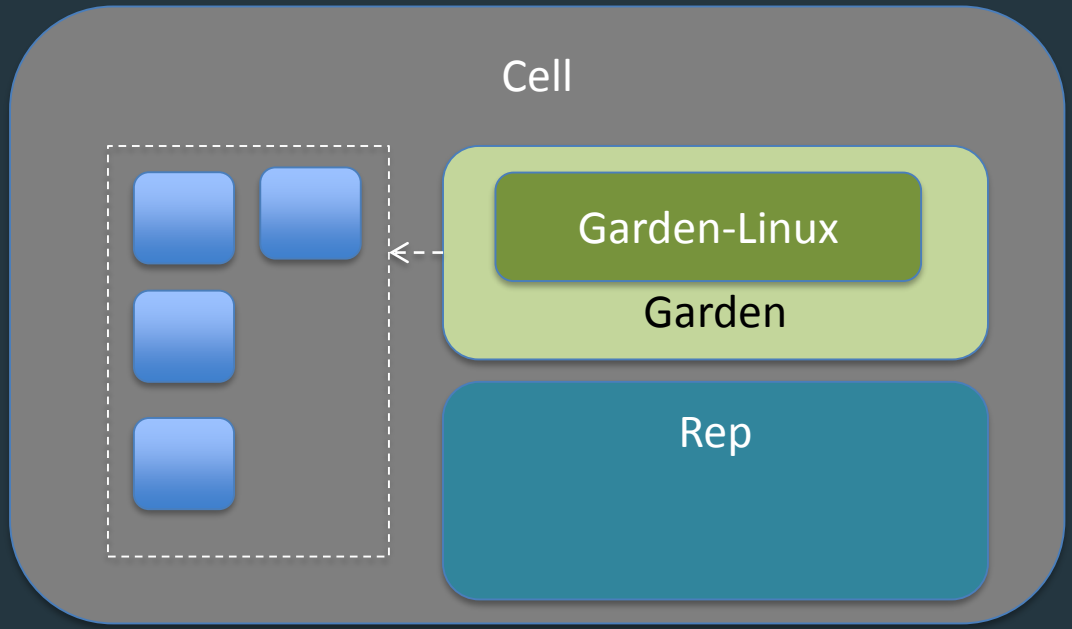


Garden is an interface.

Garden-Linux is a backend implementation.

Diego: Rep

Represents the Cell in the BBS/auctions.

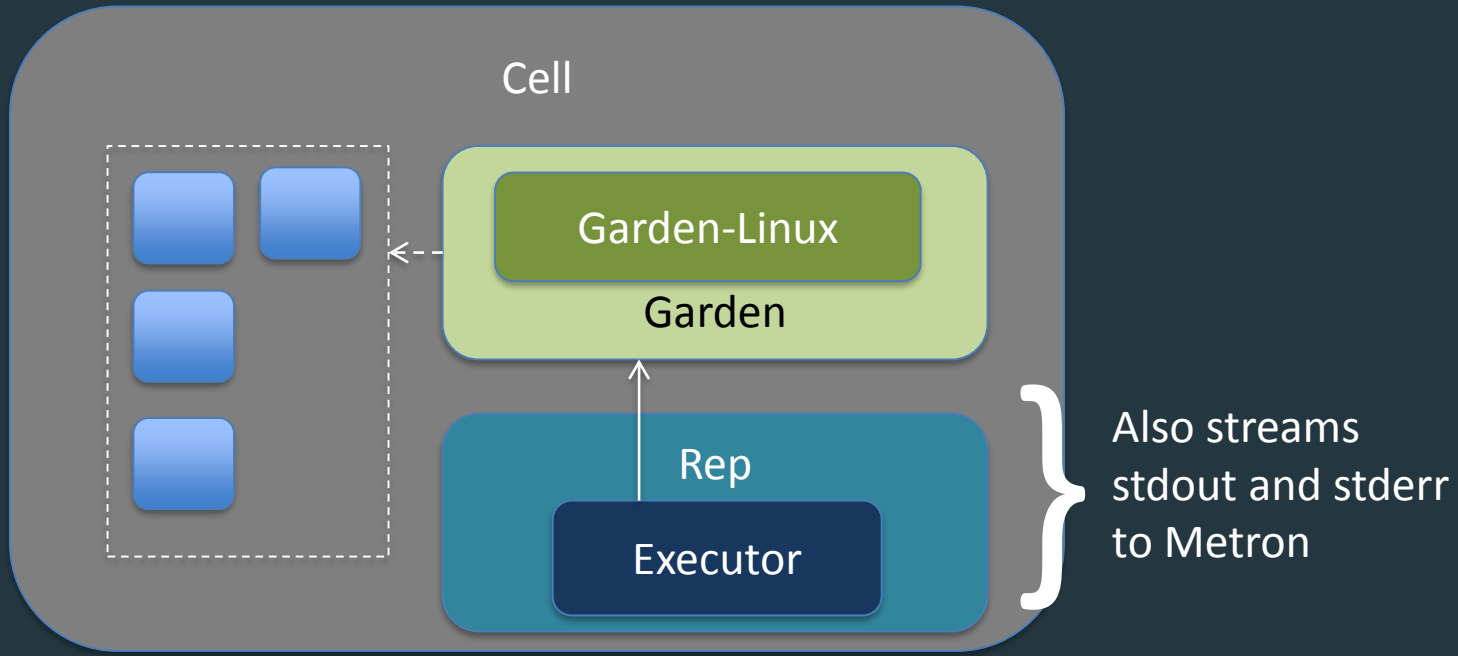


Diego: Auction

An auction is held to bid on executing a task or an LRP.

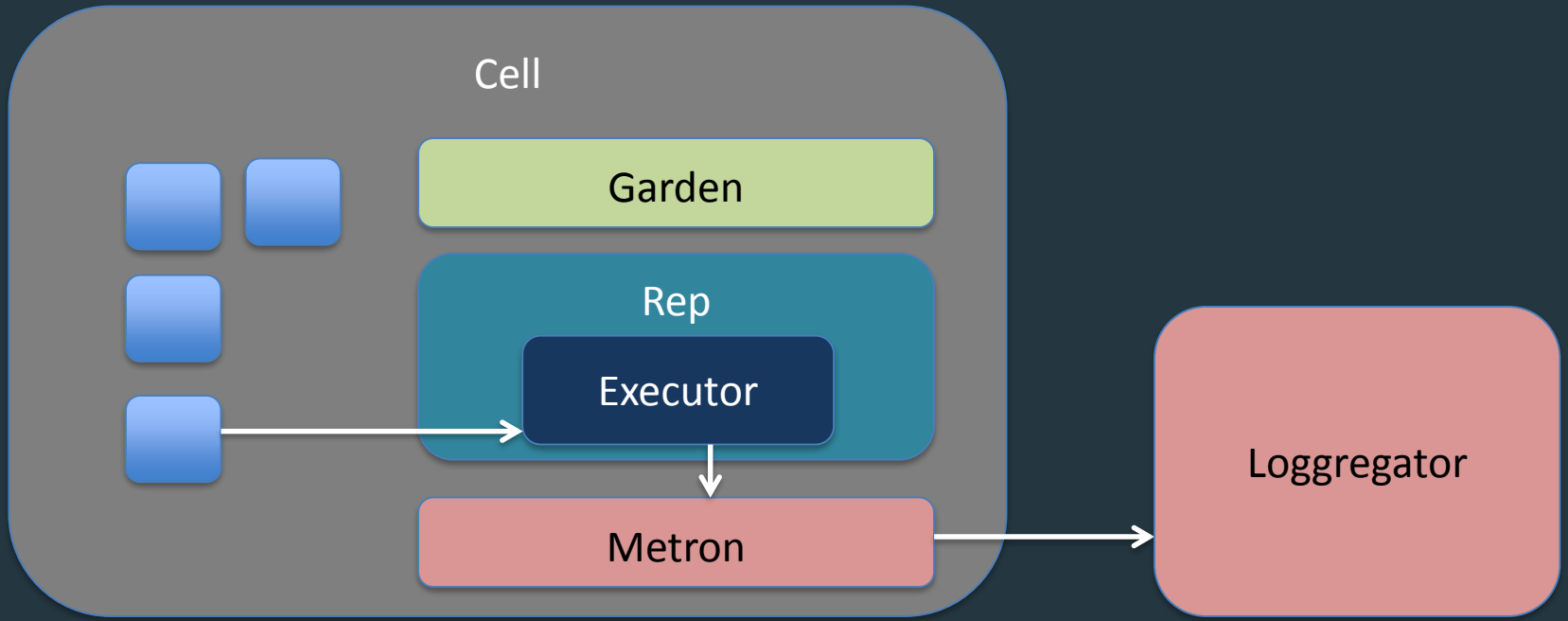
Diego: Executor

Manages container allocations on the cell.



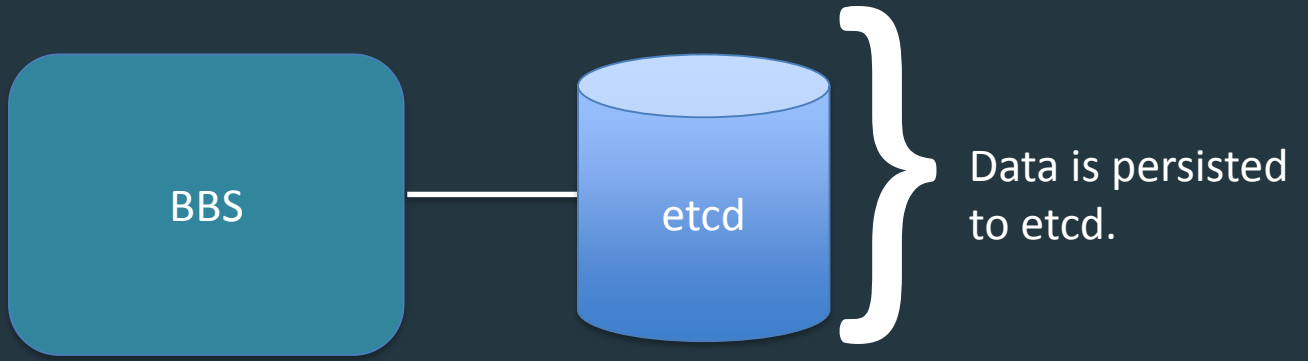
Diego: Metron

Forwards logs to the Loggregator subsystem.

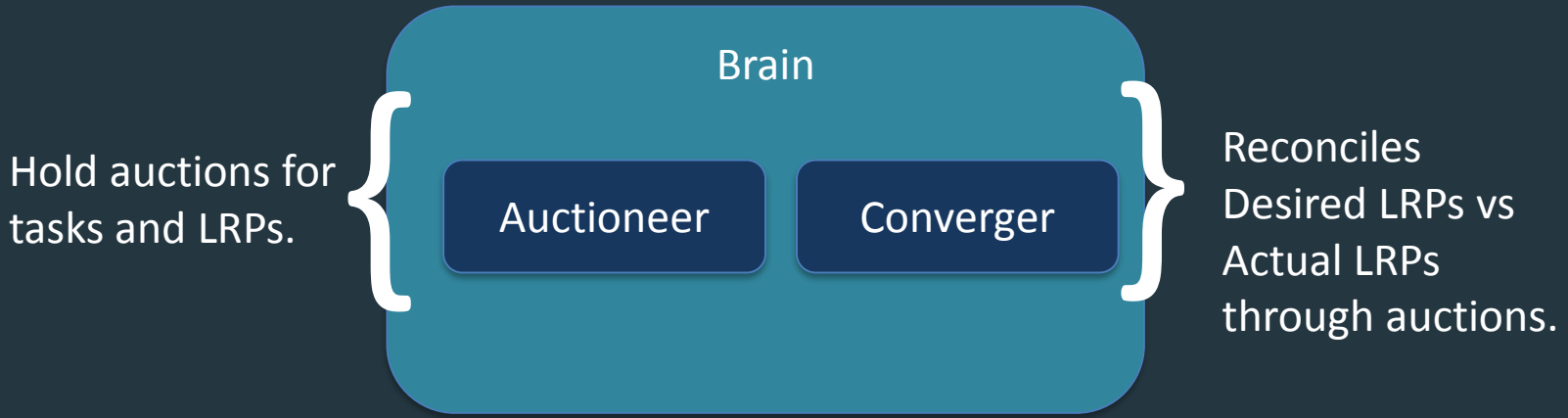


Diego: BBS

BBS (Bulletin Board System) is the API to access the Diego database for tasks and LRPs.



The Brain is composed of two components.

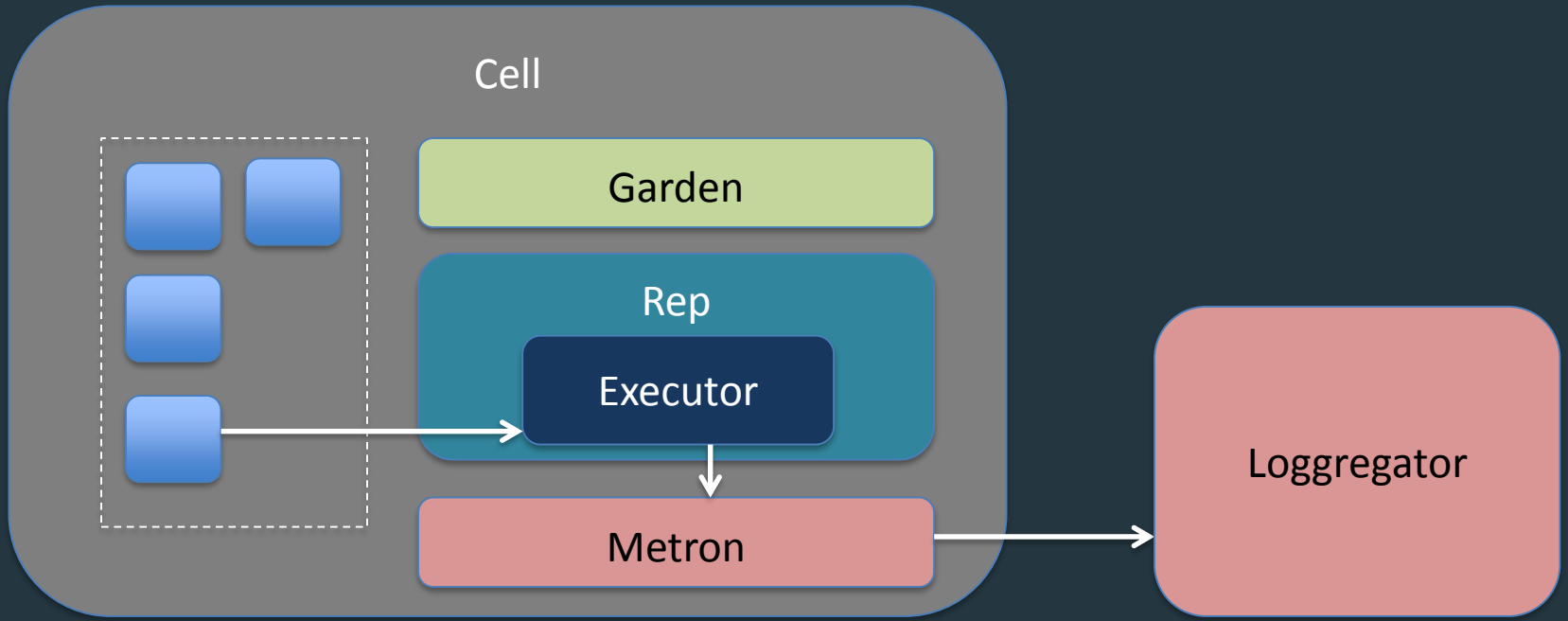


LOGGREGATOR

https://docs.pivotal.io/pivotalcf/loggregator/architecture.html

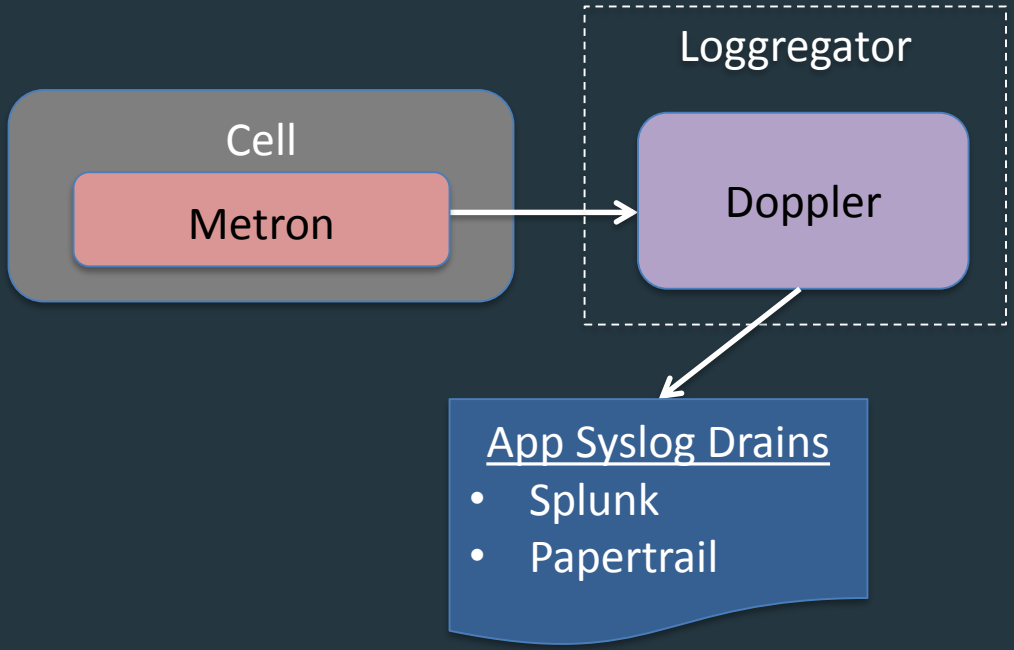
Loggregator: Metron

Forwards logs to the Loggregator subsystem.



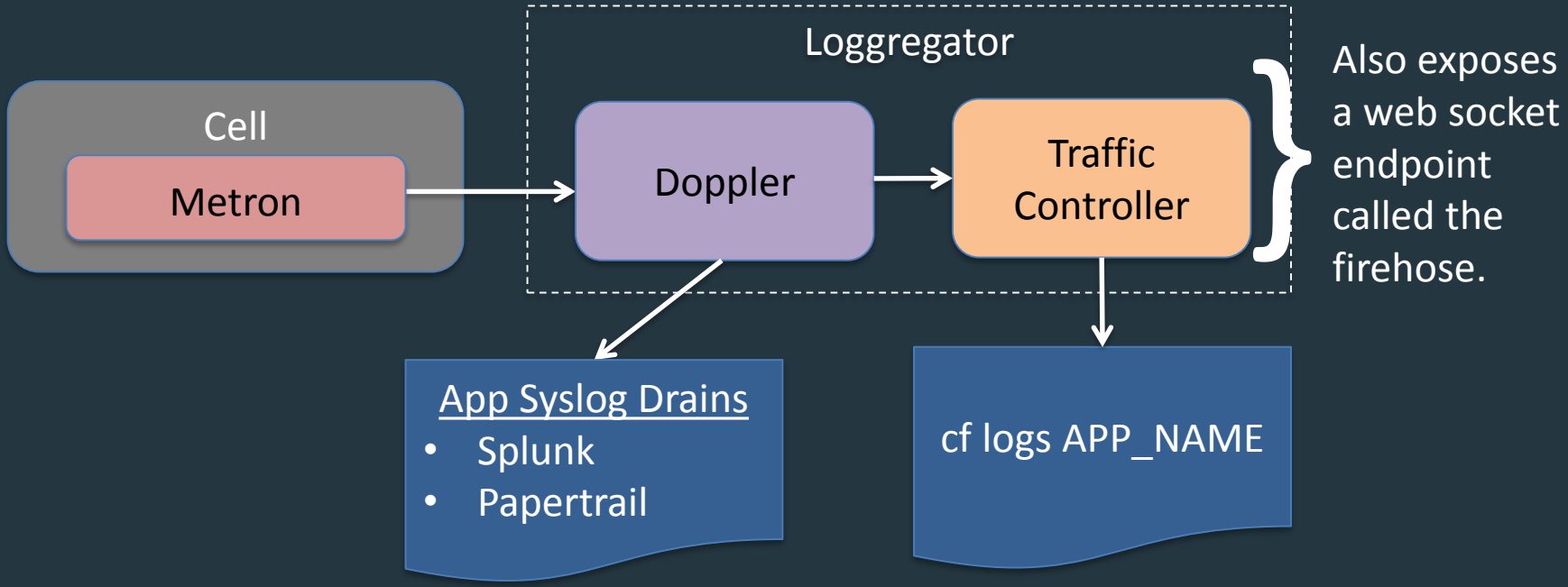
Loggregator: Doppler

Gathers logs from Metron.



Loggregator: Traffic Controller

Handles client requests for logs.



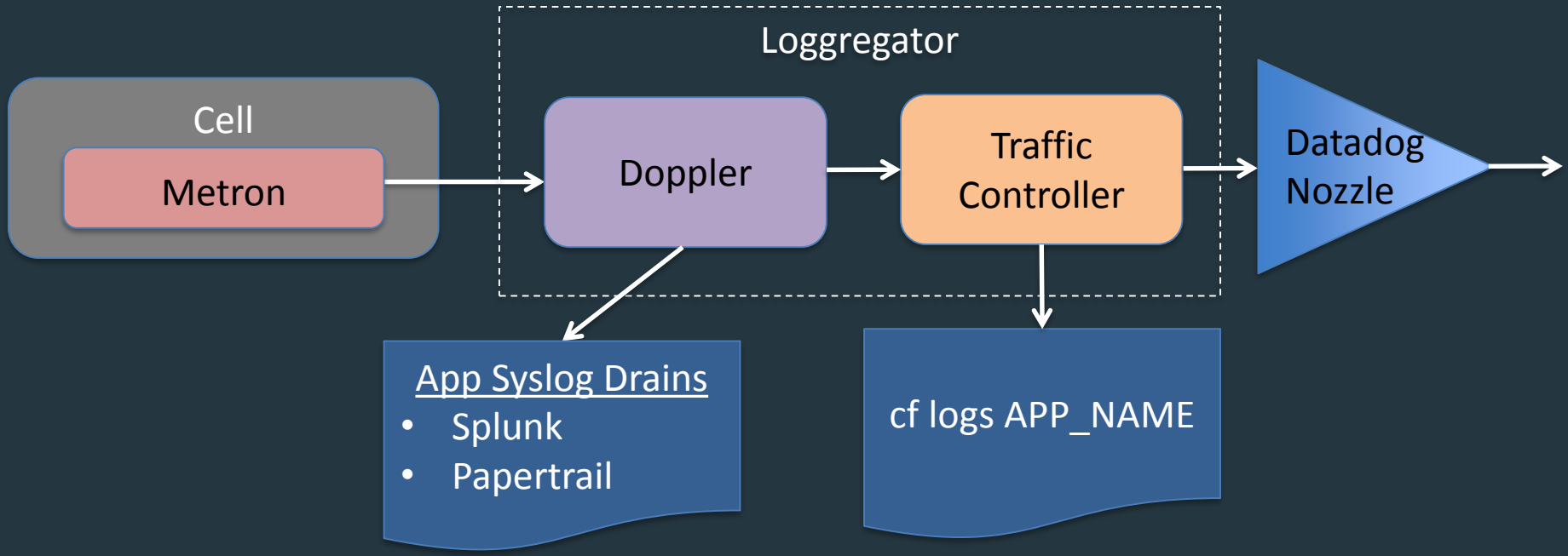
Loggregator: Firehose

A websocket endpoint that exposes app logs, container metrics and ER component metrics.

Does not include ER component logs.

Loggregator: Nozzles

Consume the firehose output.

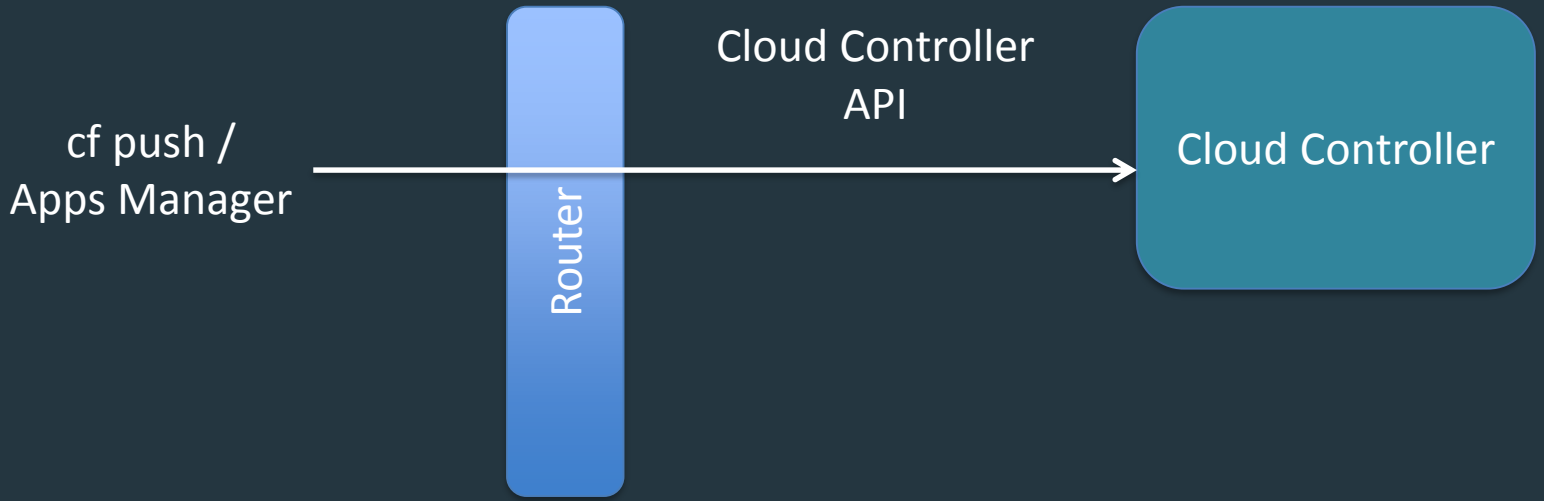


CLOUD CONTROLLER API

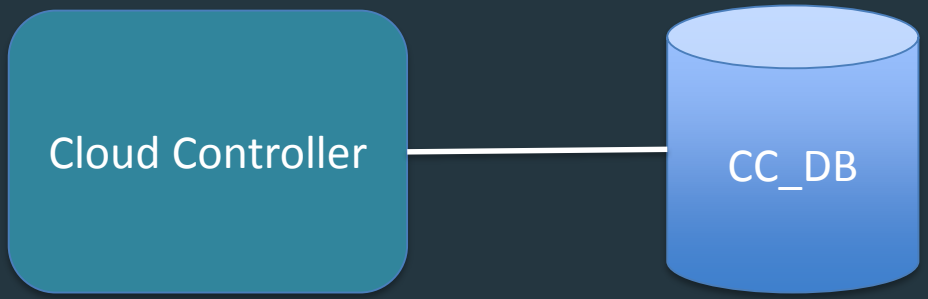
https://docs.pivotal.io/pivotalcf/concepts/architecture/cloud-controller.html

CCAPI: Cloud Controller

The Cloud Controller exposes an API for using and managing the Elastic Runtime.

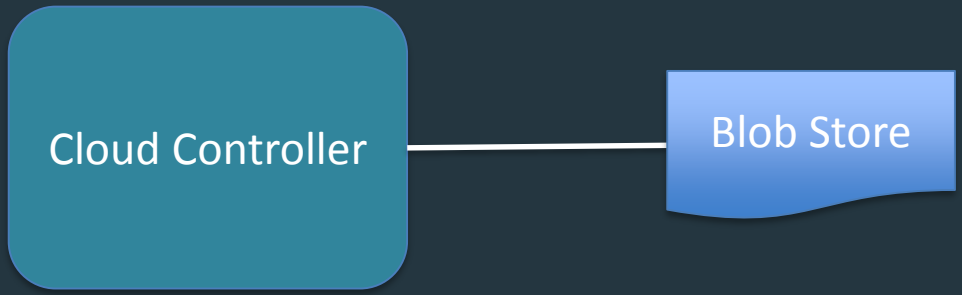


The Cloud Controller persists Org/Space/App data in the Cloud Controller Database.



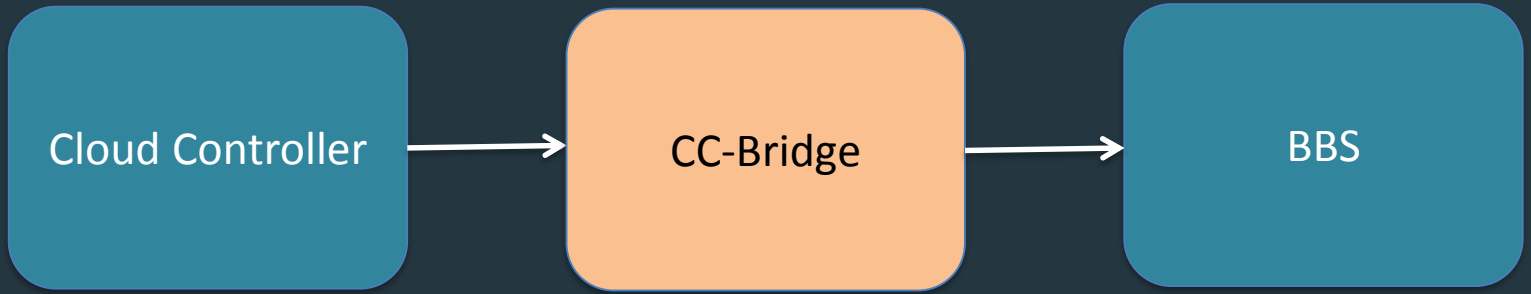
CCAPI: Blob Store

The Cloud Controller persists app packages and droplets to the blob store.



CCAPI: CC-Bridge

The CC-Bridge translates app specific messages into the generic language of tasks and LRPs.

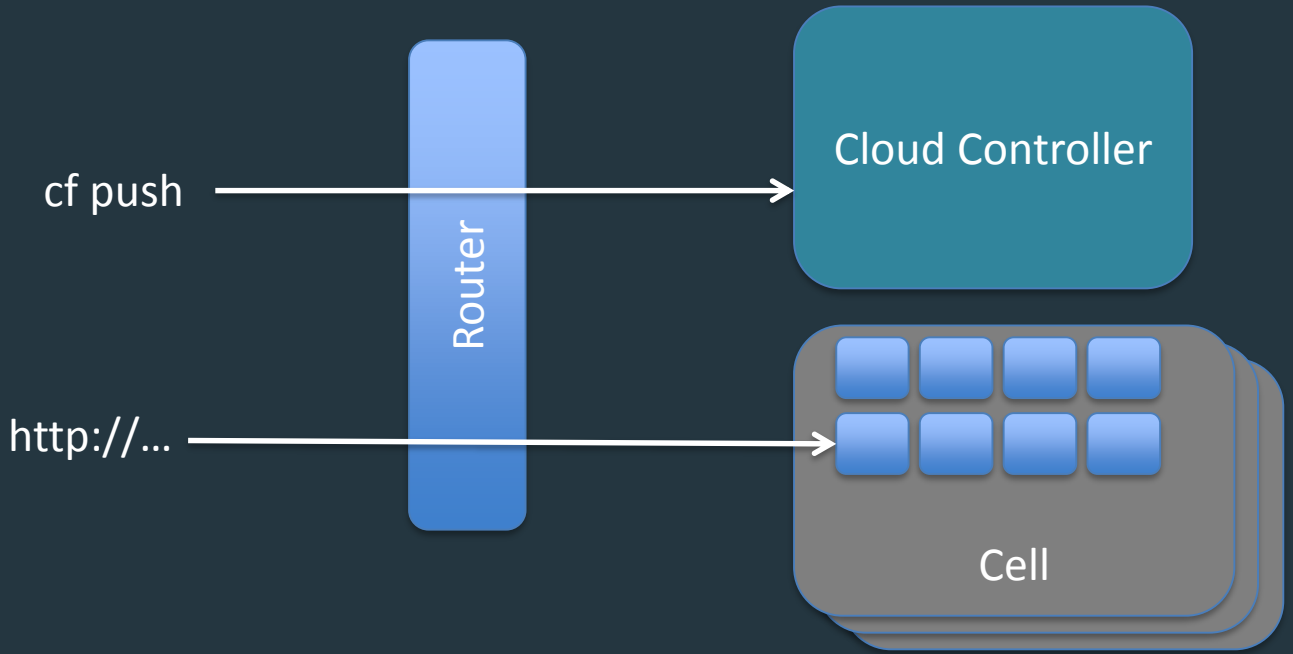


ROUTING

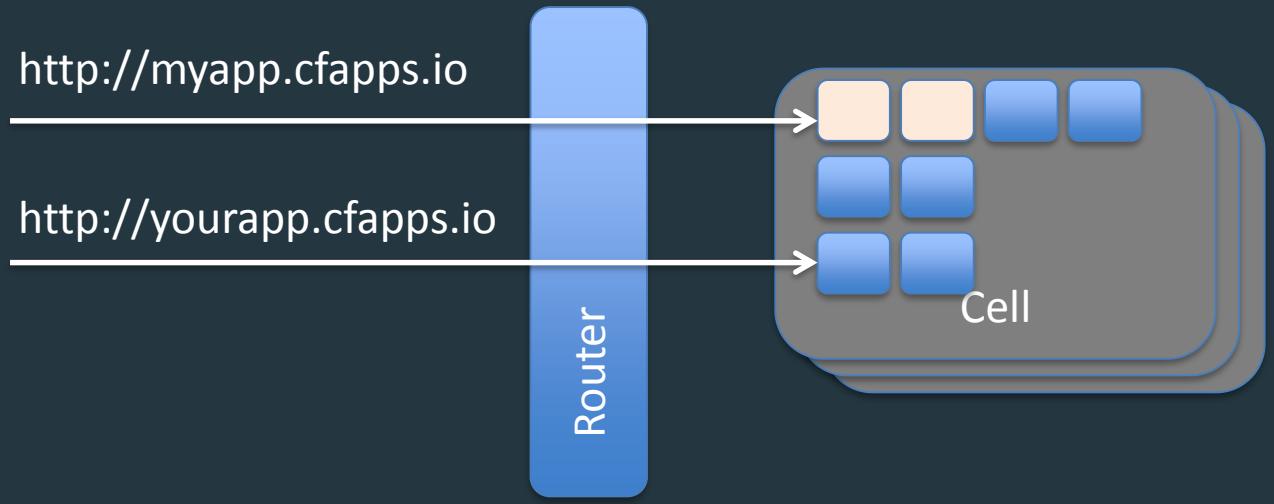
https://docs.pivotal.io/pivotalcf/concepts/architecture/router.html

Router

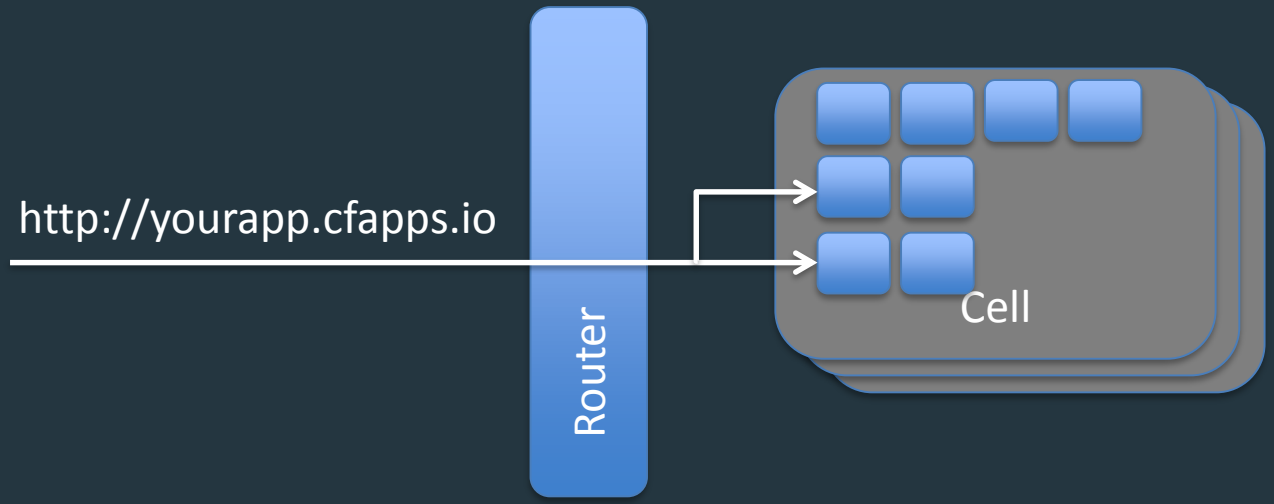
The router routes traffic to appropriate component.



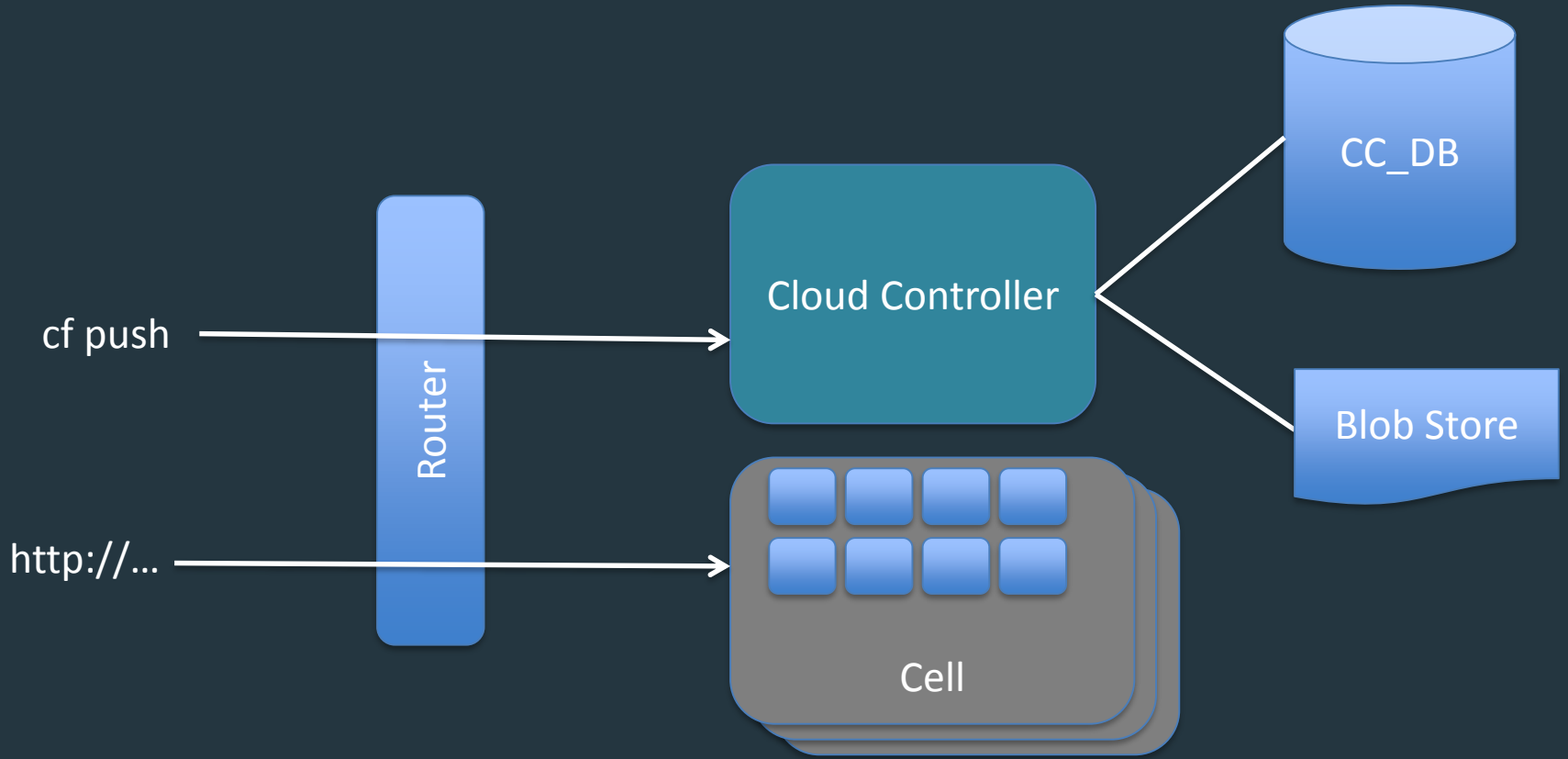
The router routes requests to all app instances.



The router round robins between application instances.



Elastic Runtime Simplified Architecture



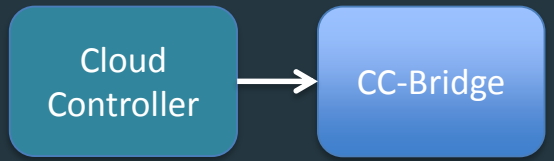
KEY FLOWS

https://docs.pivotal.io/pivotalcf/concepts/diego/diego-architecture.html

https://docs.pivotal.io/pivotalcf/concepts/how-applications-are-staged.html

Stage and Run Request Flow

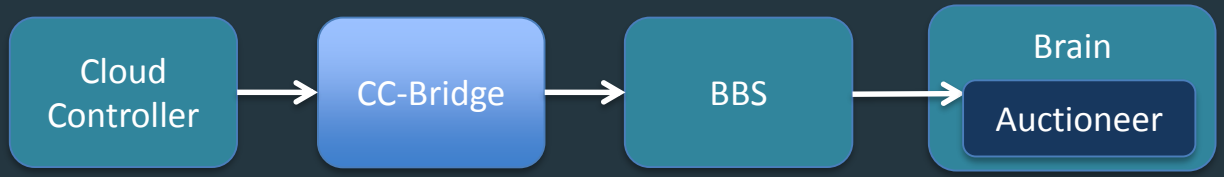
Cloud Controller passes requests to run and stage applications to the CC-Bridge.



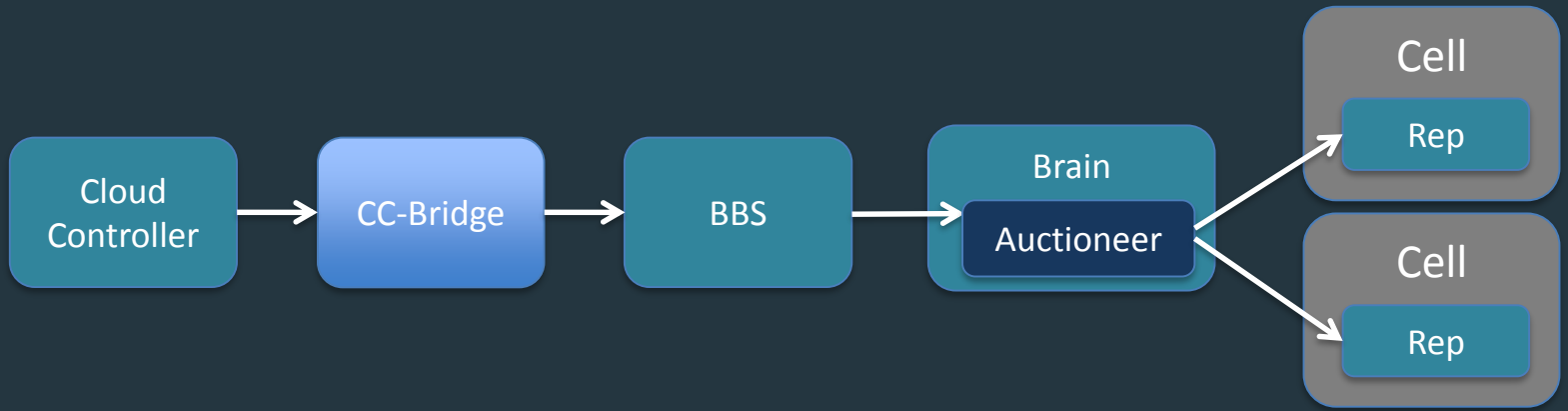
Stage and Run Request Flow

CC-Bridge translates stage and run requests into tasks and LRPs.

BBS submits tasks and LRPs to the Auctioneer (Brain)

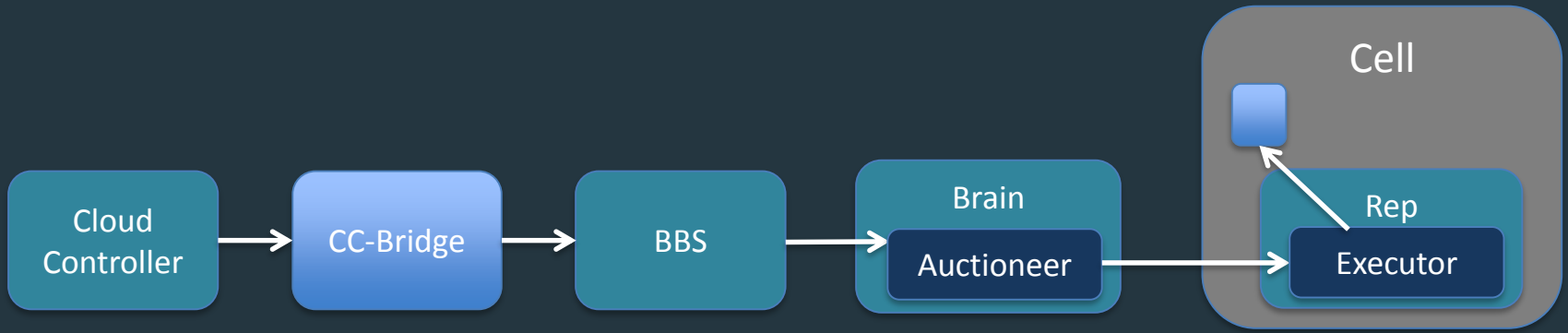


Auctioneer distributes tasks and LRPs through an auction.

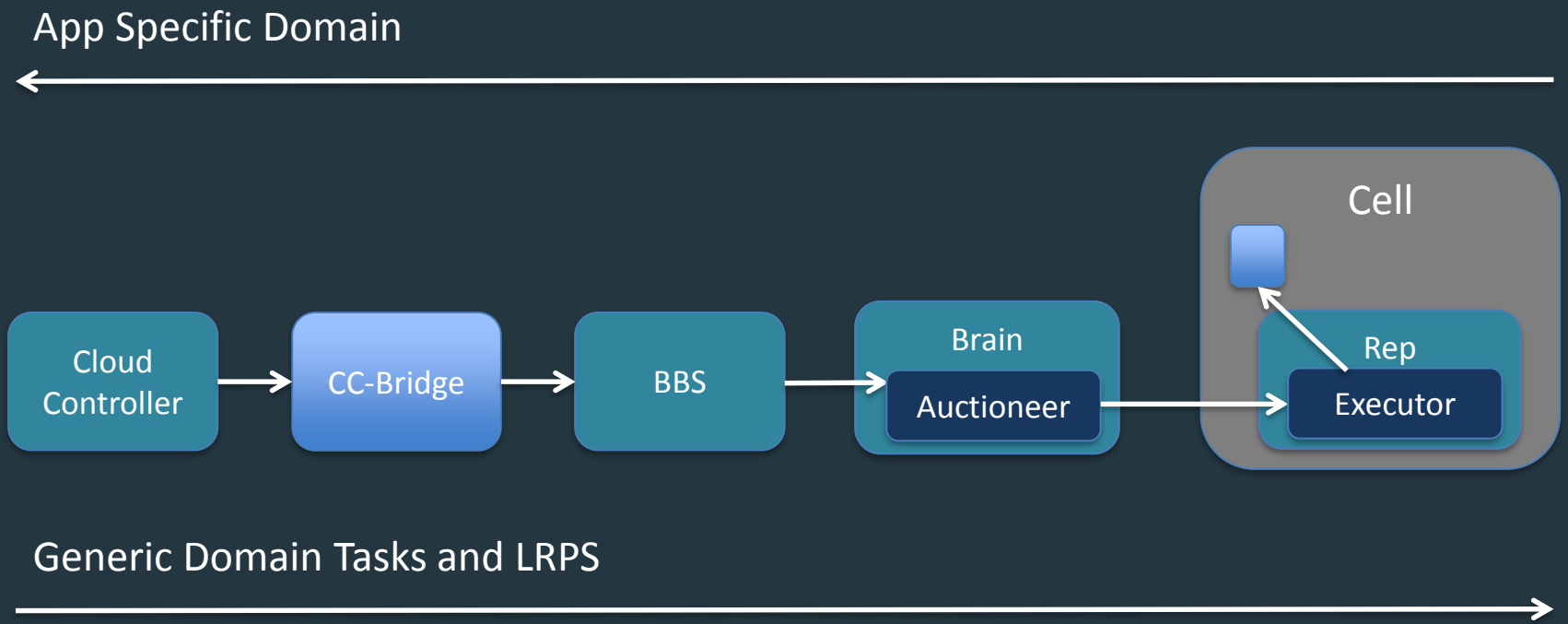


Stage and Run Request Flow

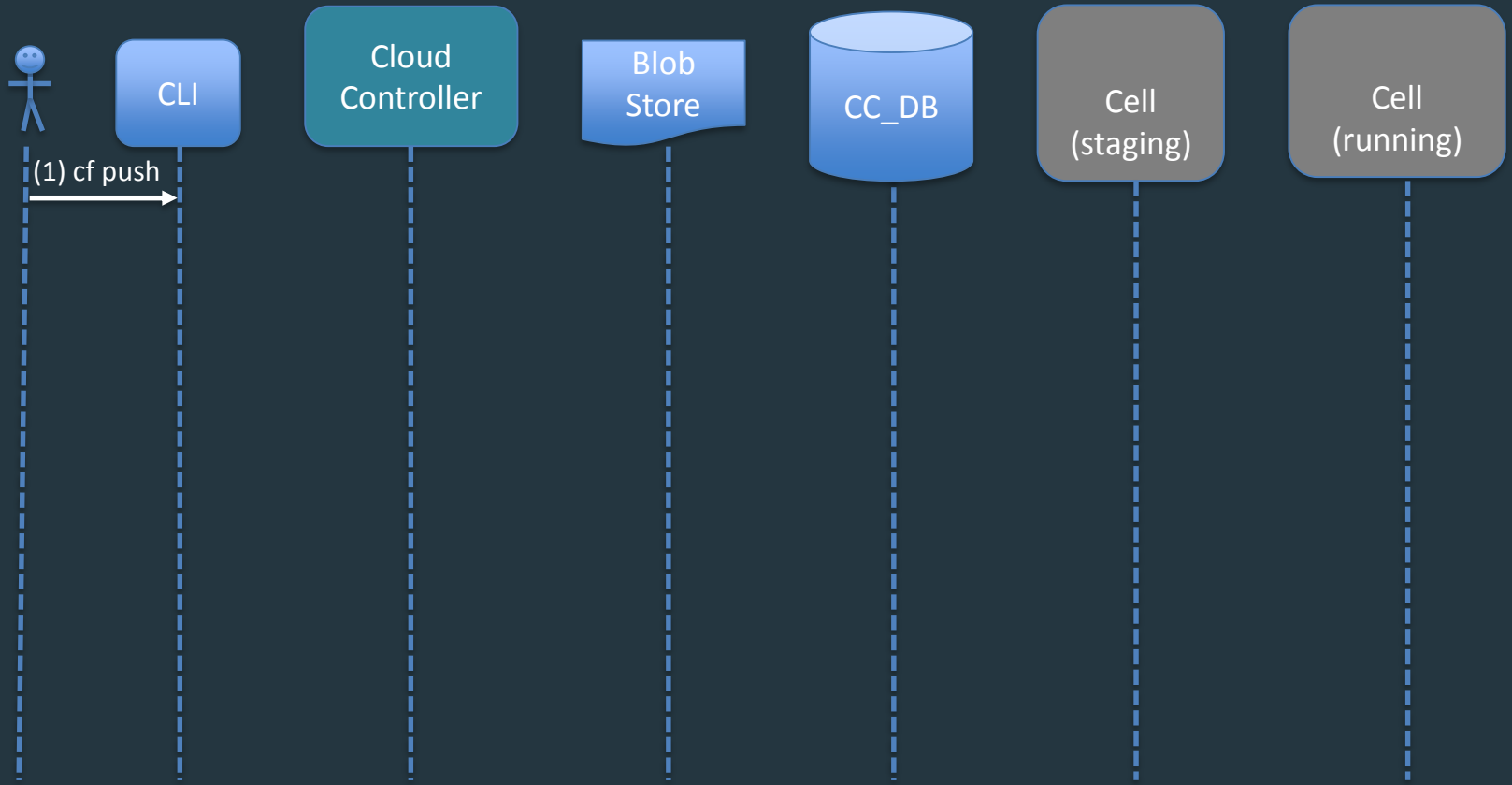
Auctioneer assigns the task or LRP to a cell. The Executor creates a Garden container in the cell. The task or LRP runs in the container.



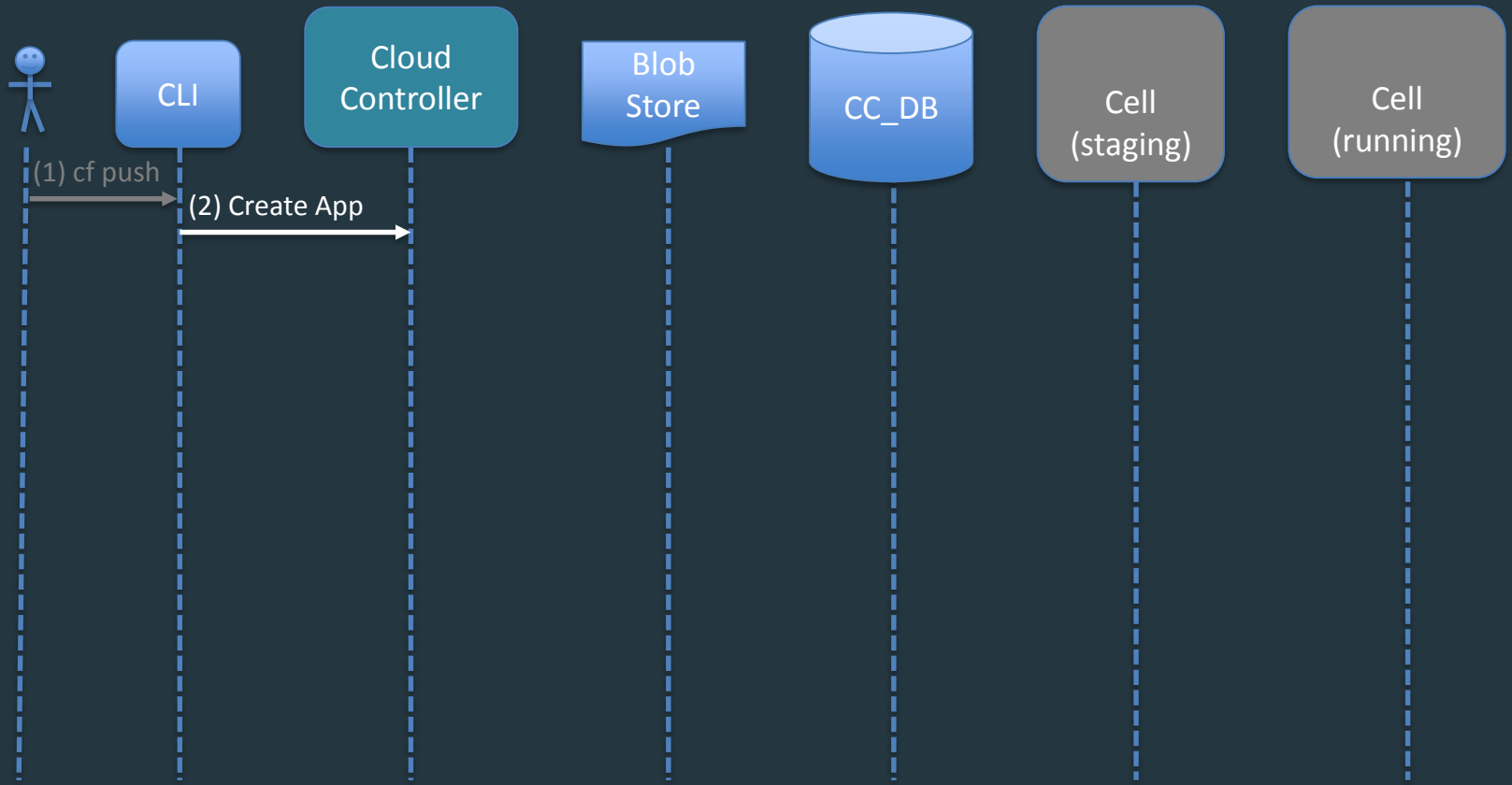
Stage and Run Request Flow



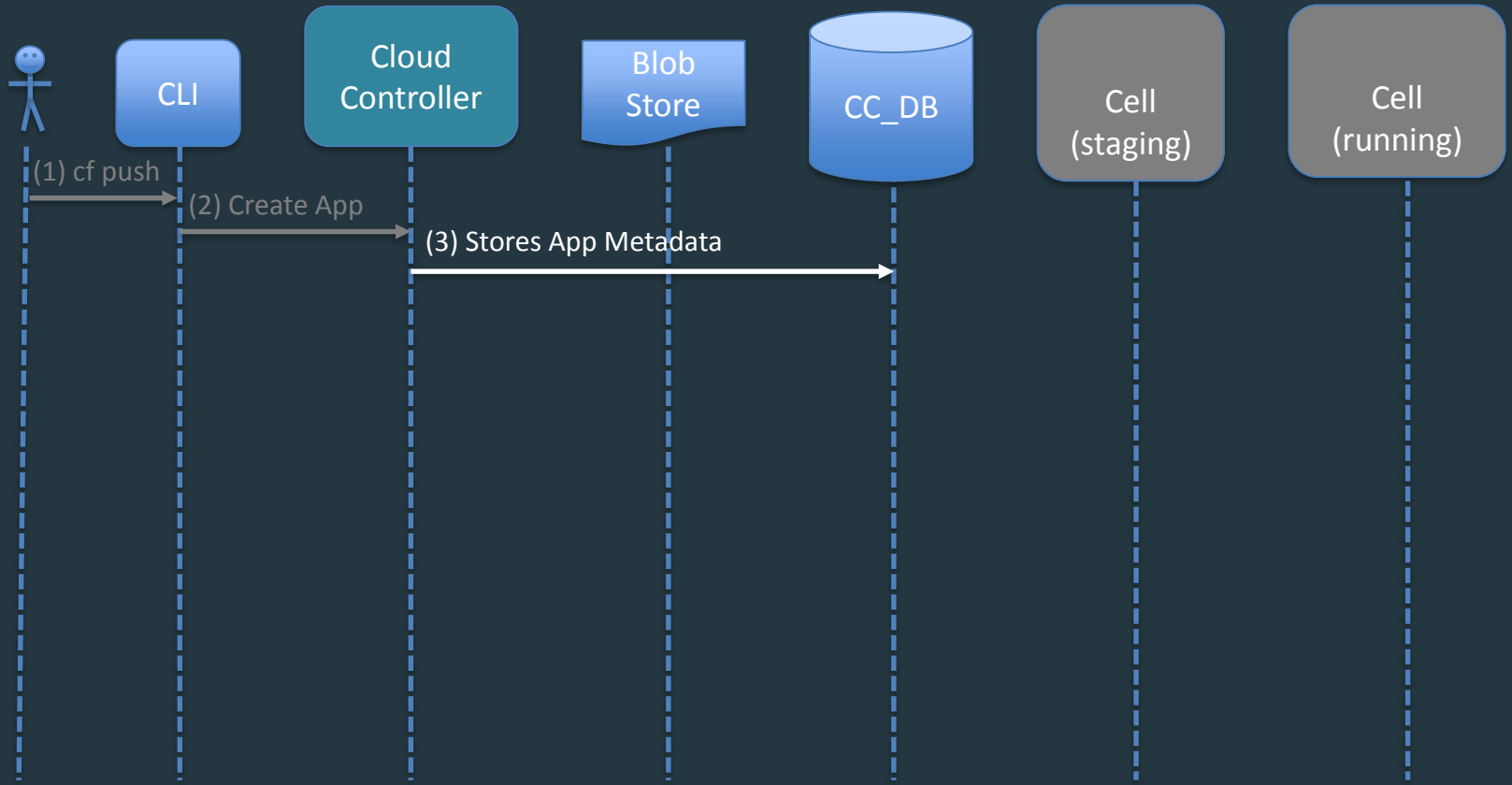
Push Sequence



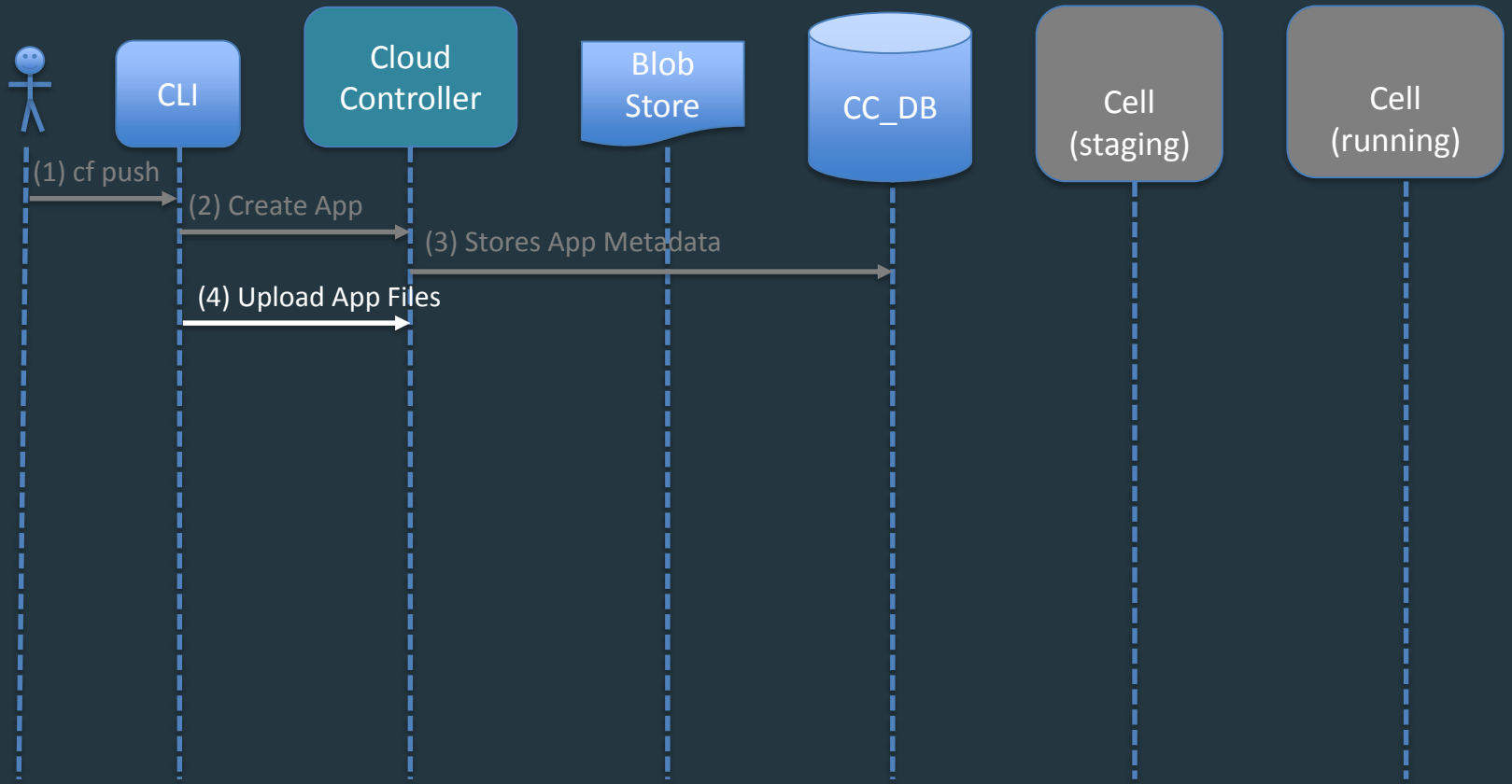
Push Sequence



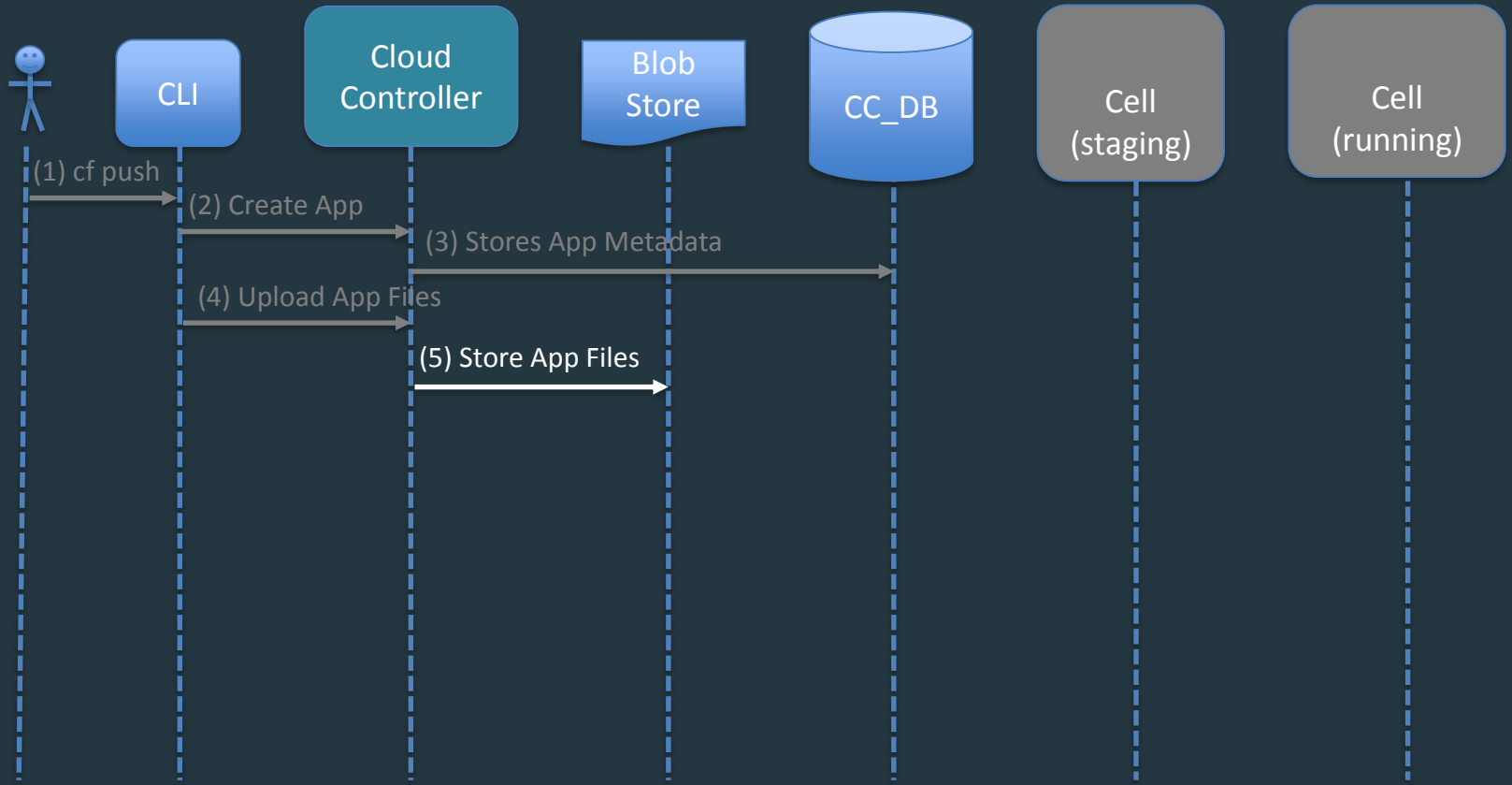
Push Sequence



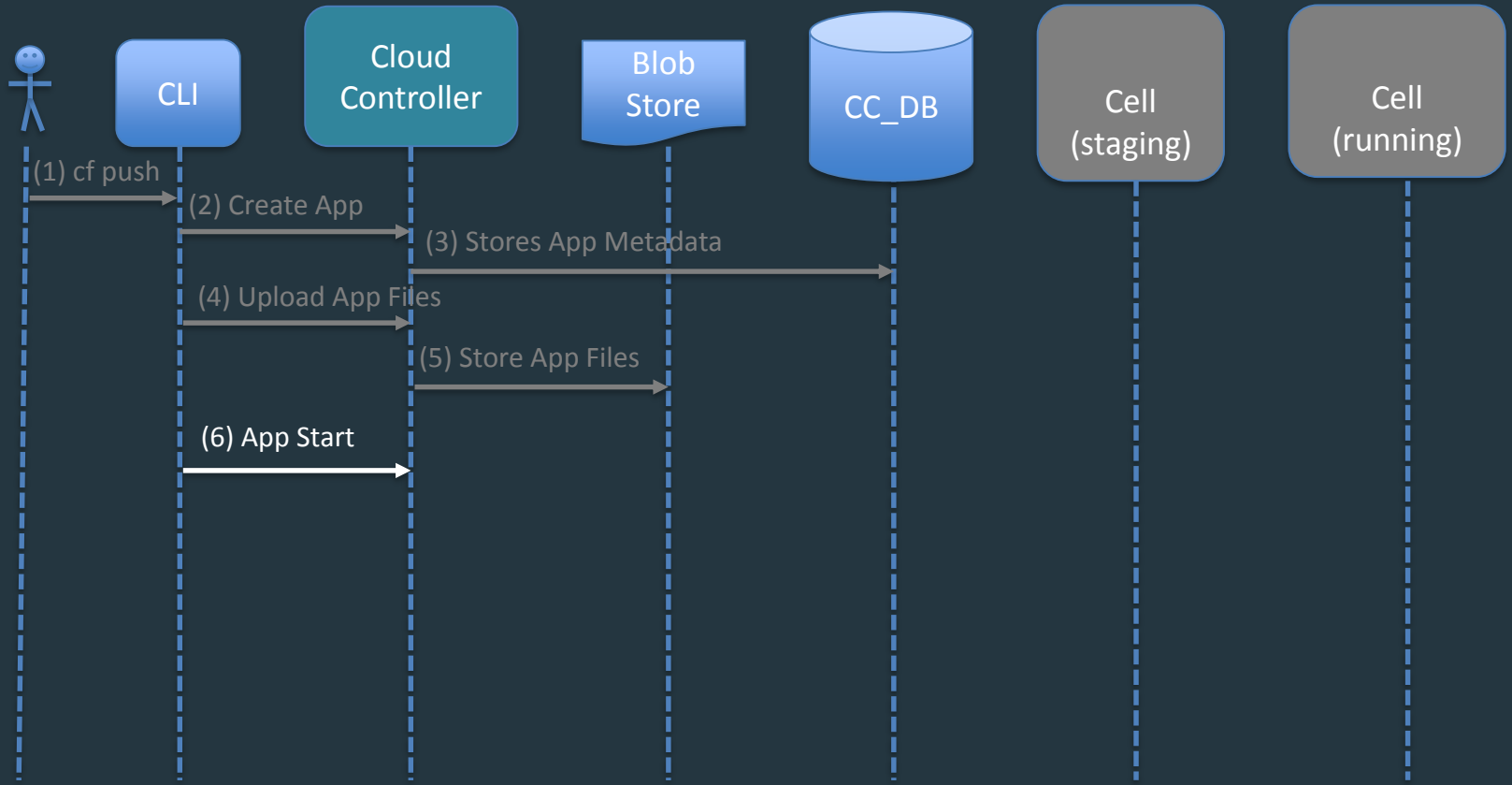
Push Sequence



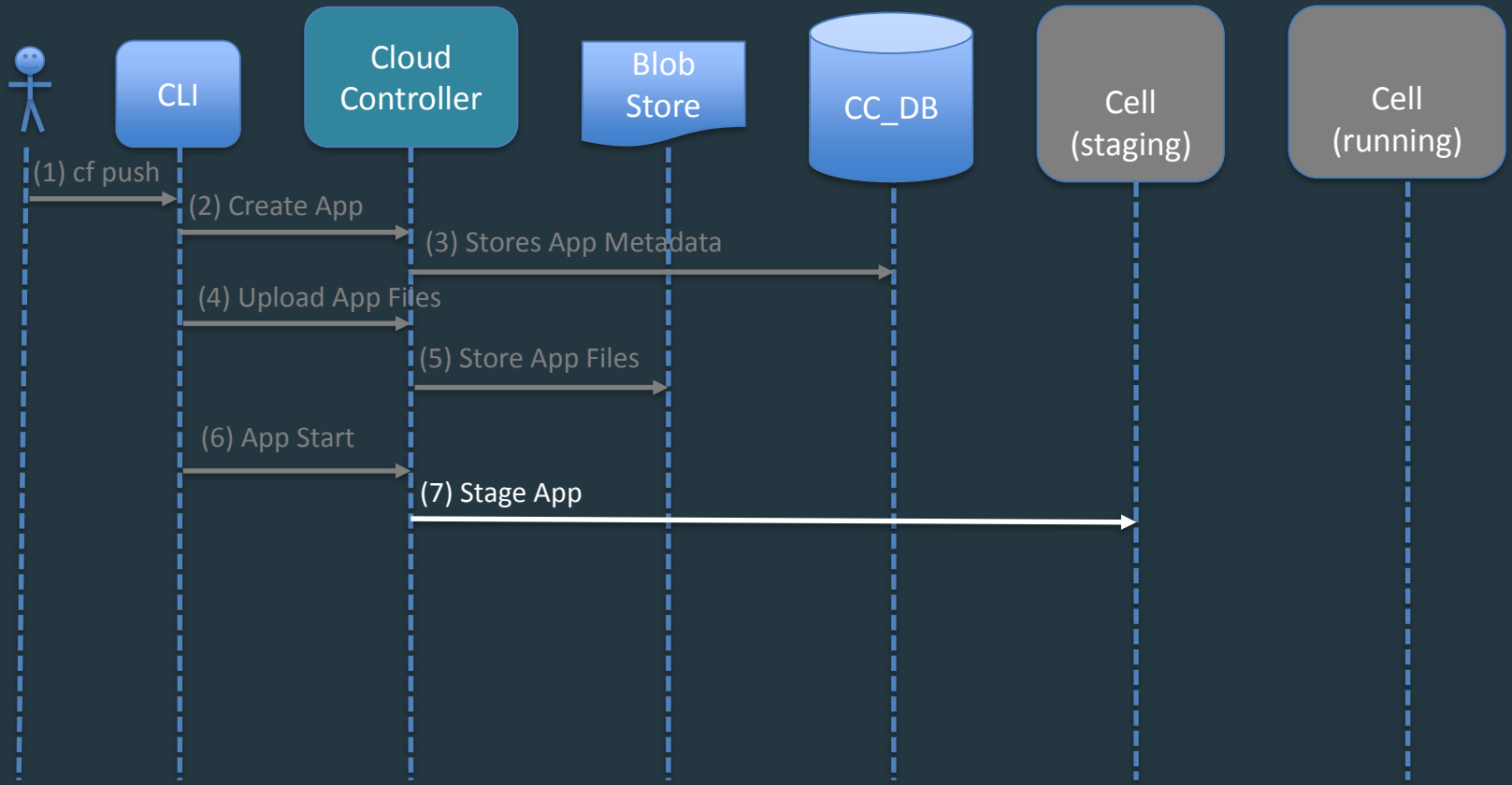
Push Sequence



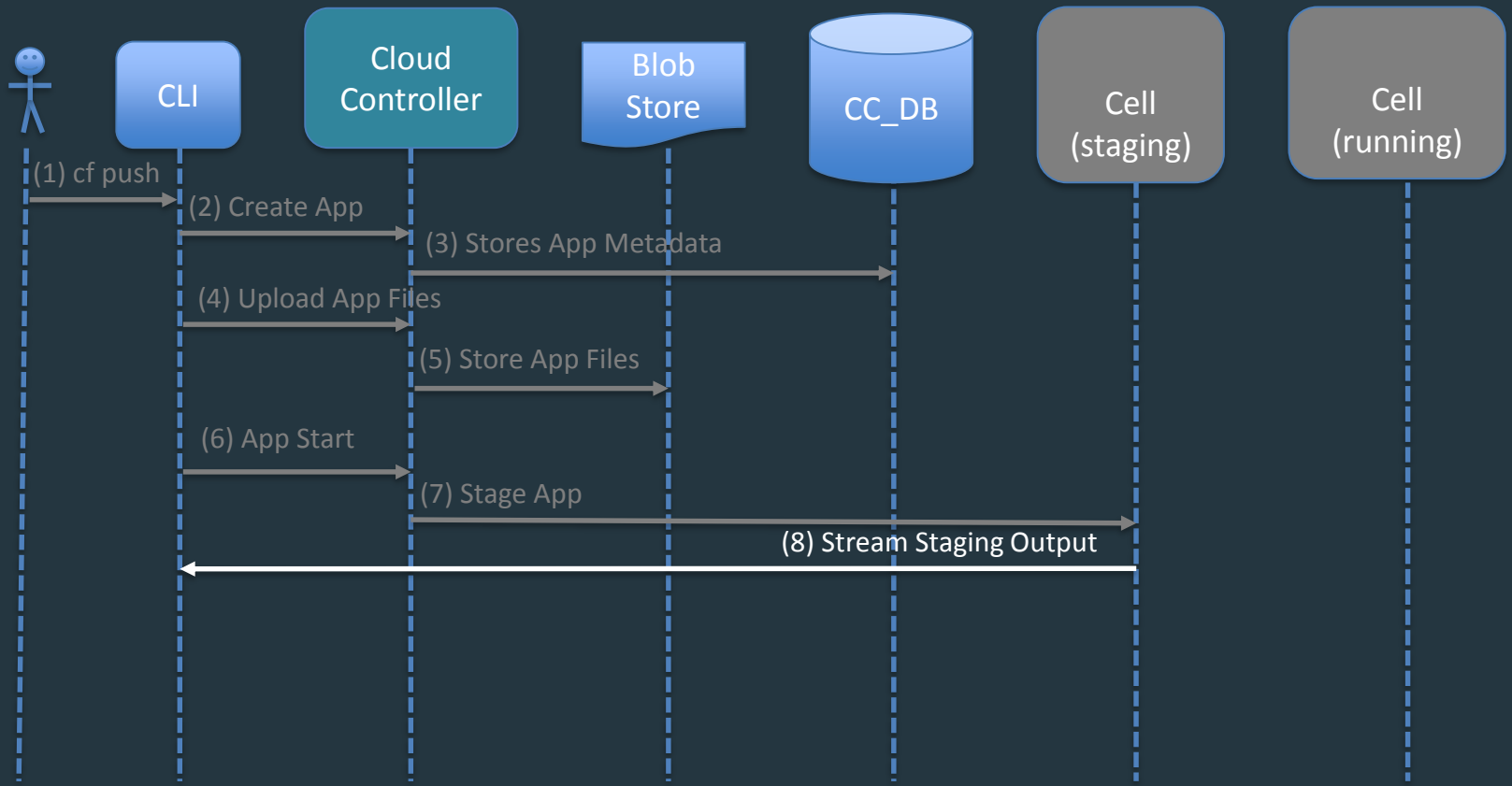
Push Sequence



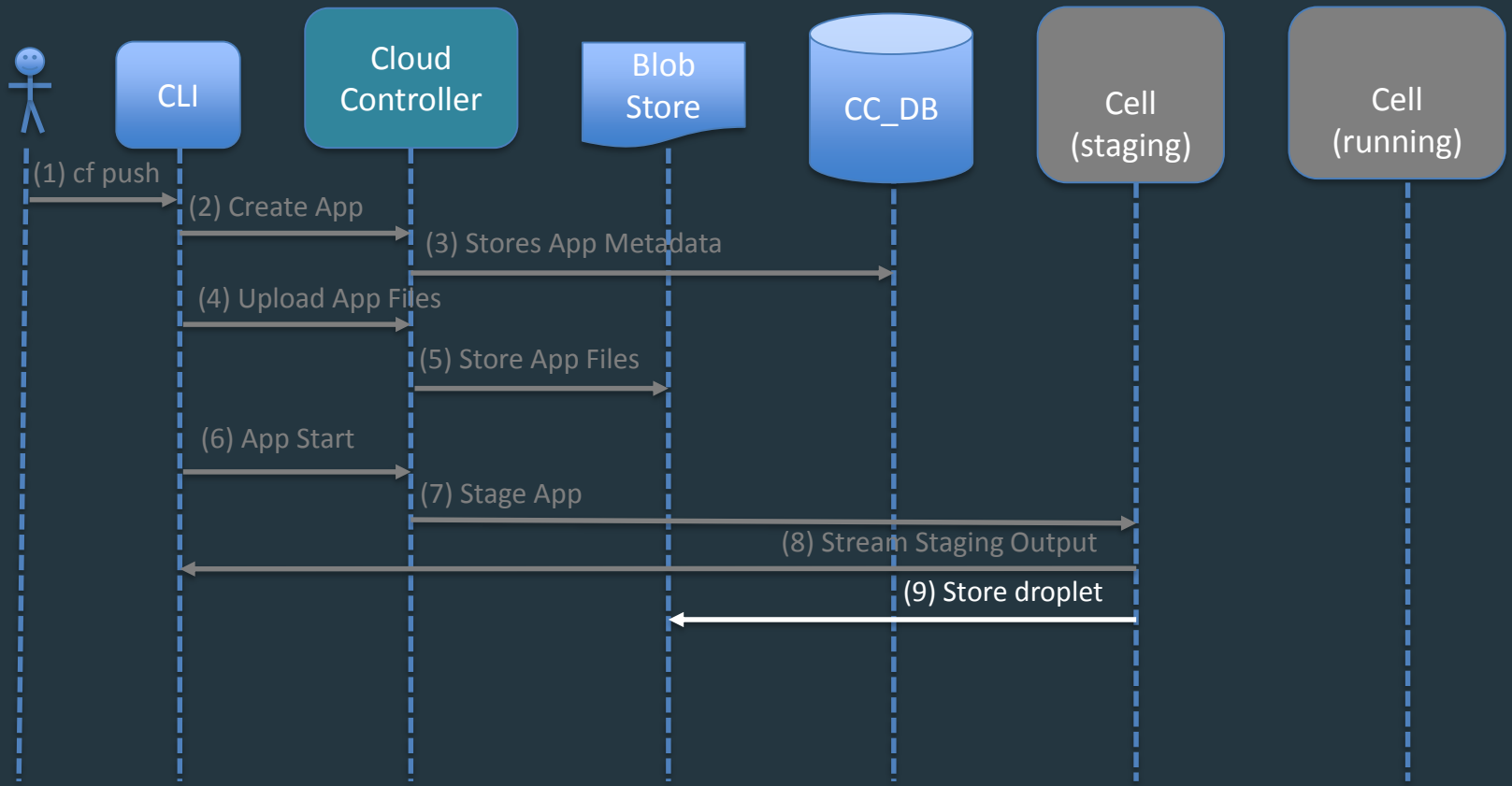
Push Sequence



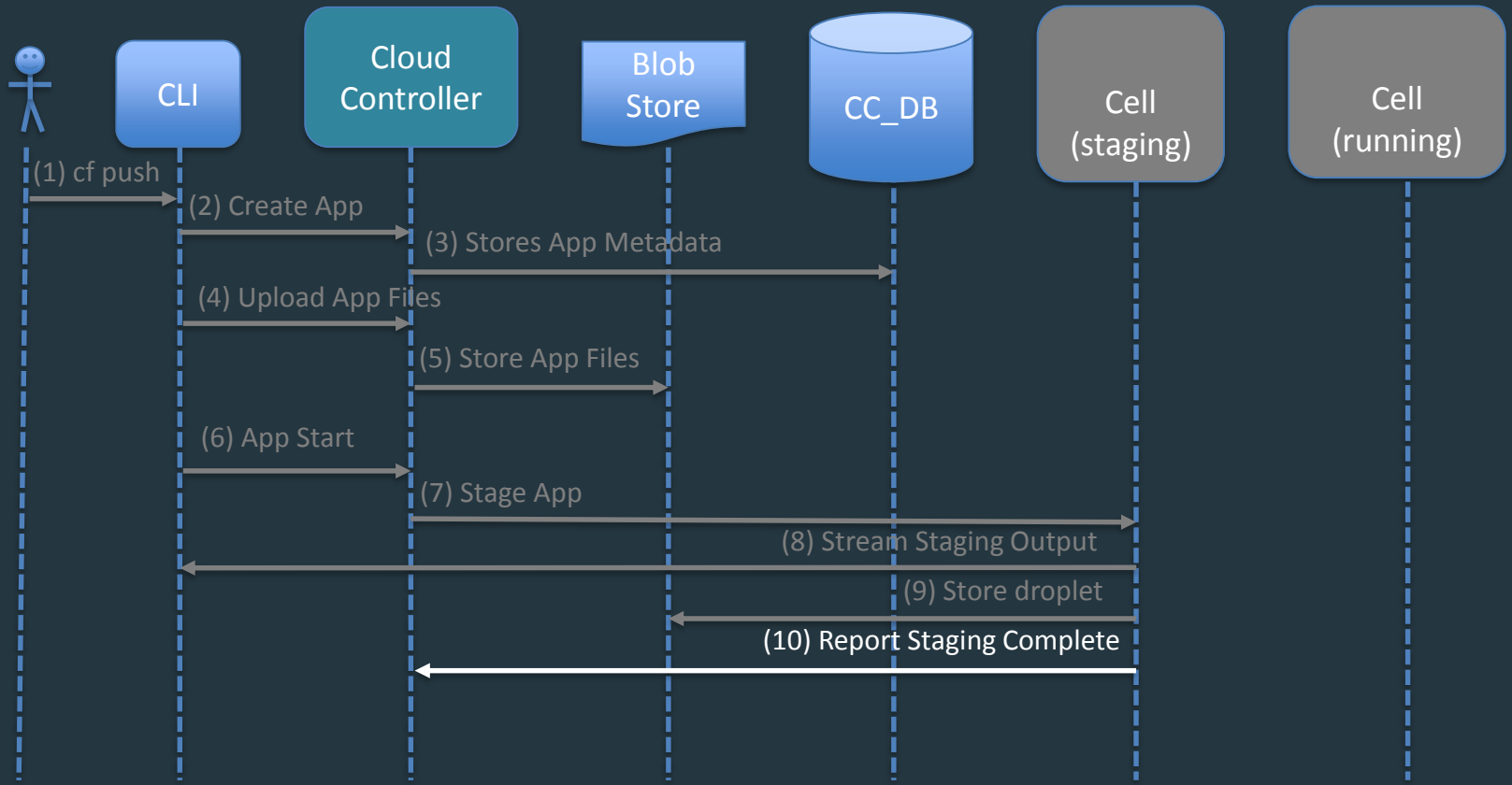
Push Sequence



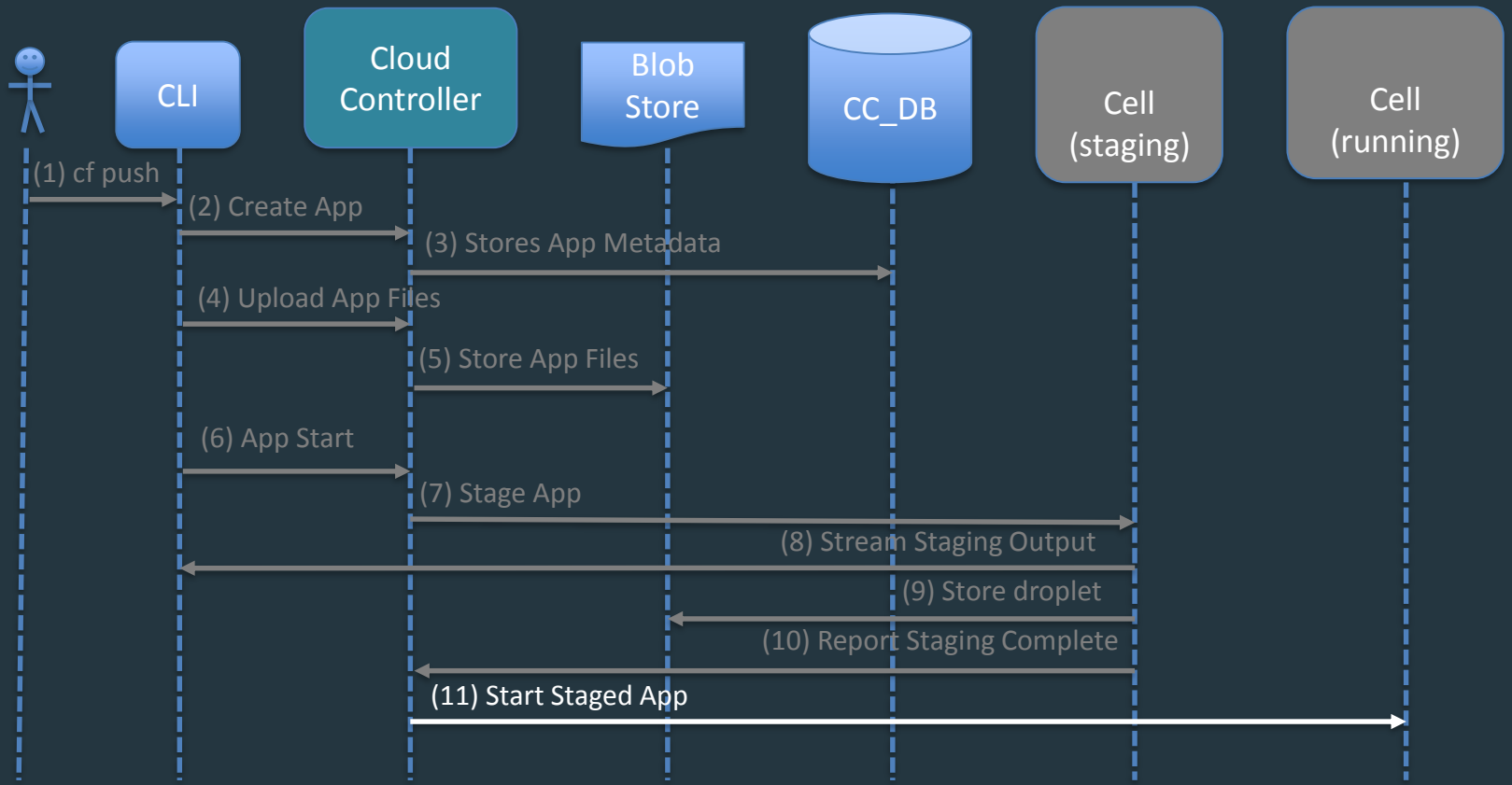
Push Sequence



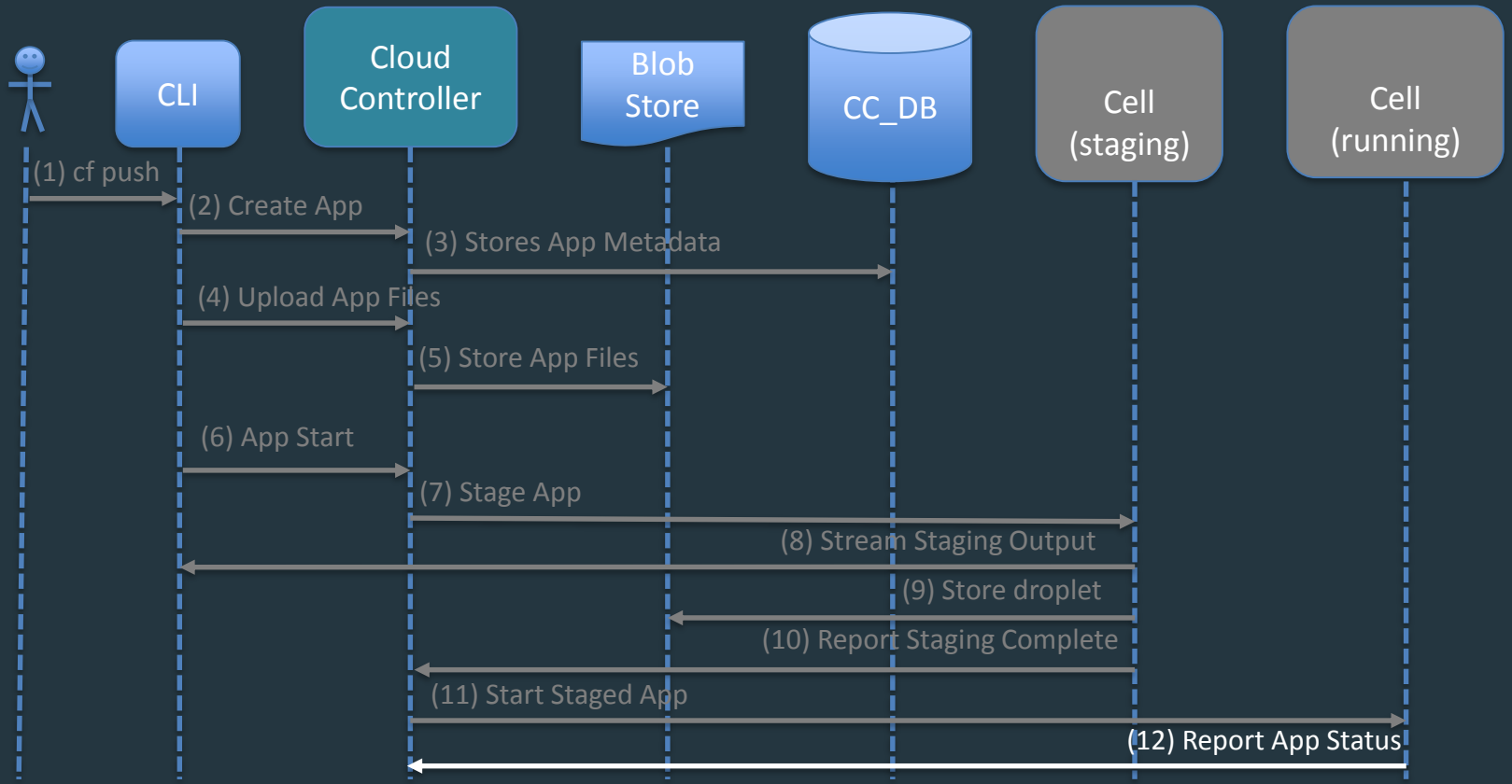
Push Sequence



Push Sequence



Push Sequence



Buildpack

Buildpacks provide framework and runtime support for your applications.

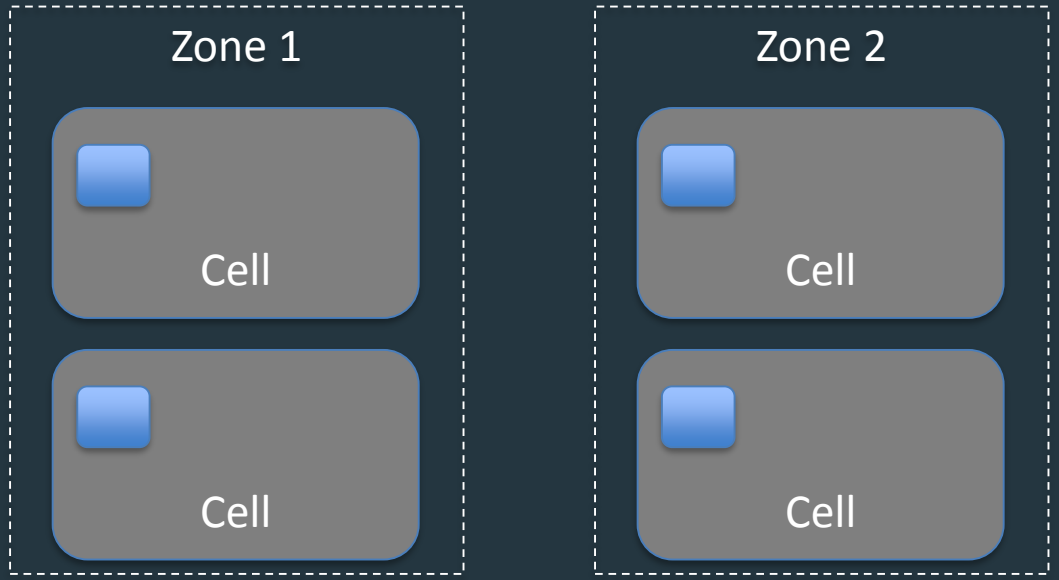
In other words, they build immutable droplets (stage your application).

Agenda

1. Cloud Native Apps
2. Elastic Runtime Architecture
3. High Availability

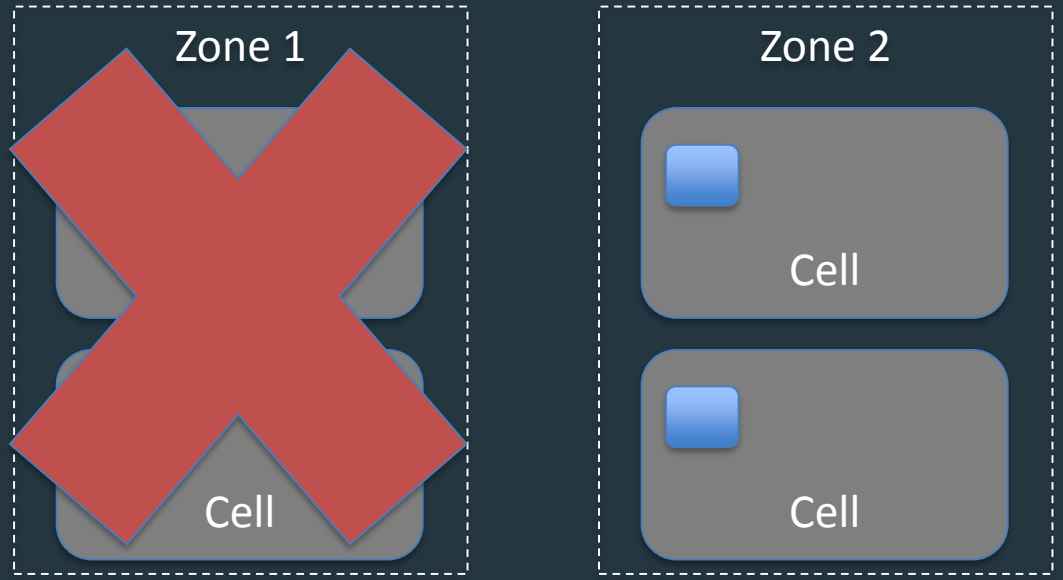
Availability Zones

Application instances are evenly distributed across availability zones.



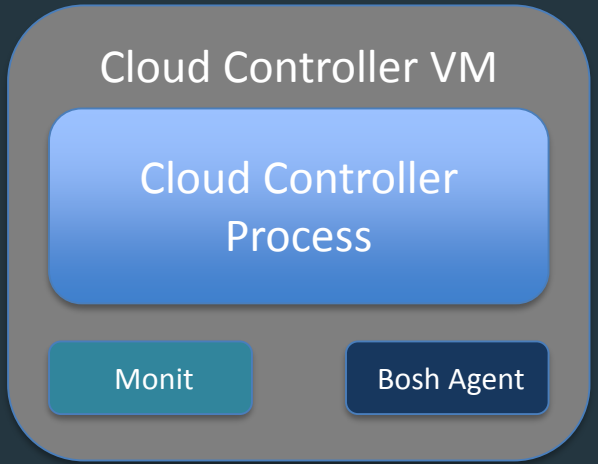
Availability Zones

Application stays up despite losing an AZ.



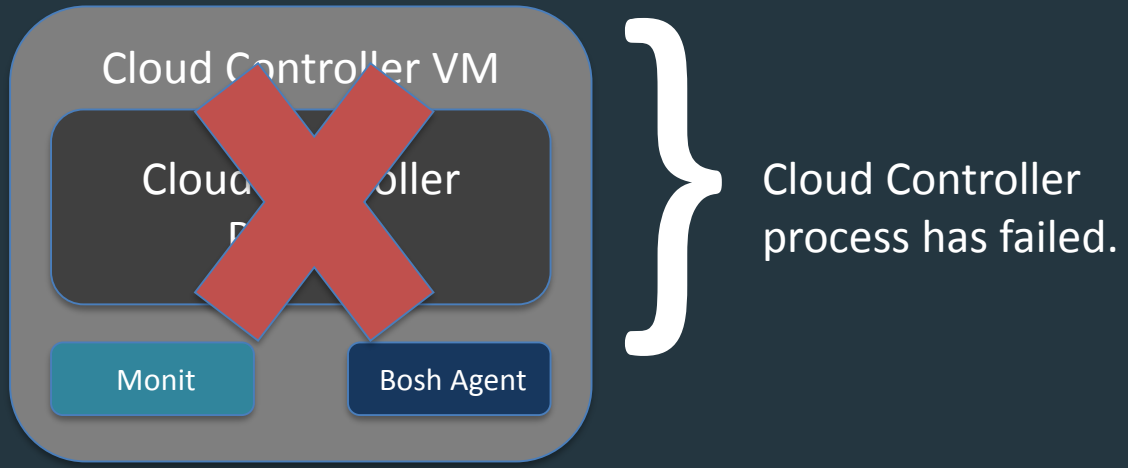
BOSH Managed Processes

Elastic runtime processes are monitored and automatically restarted.



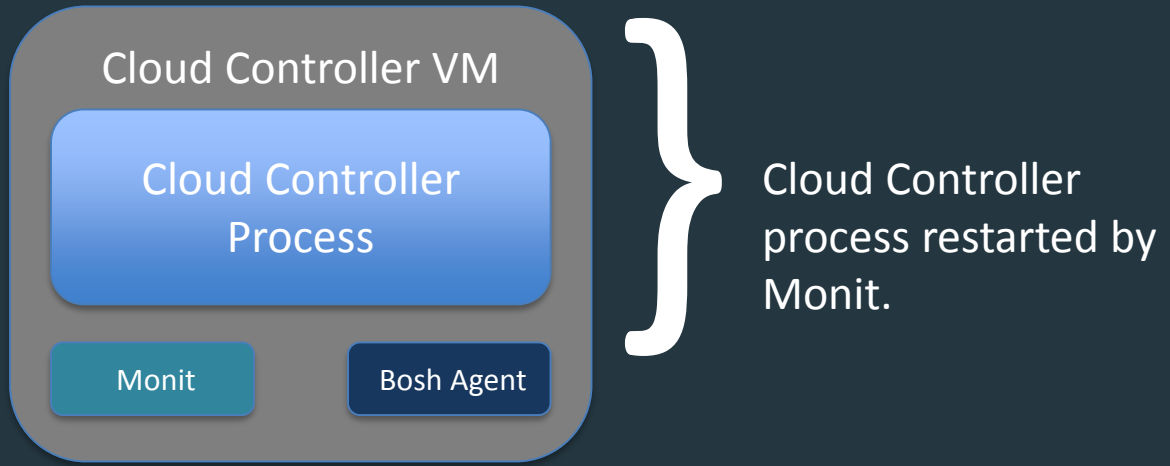
BOSH Managed Processes

Elastic runtime processes are monitored and automatically restarted.



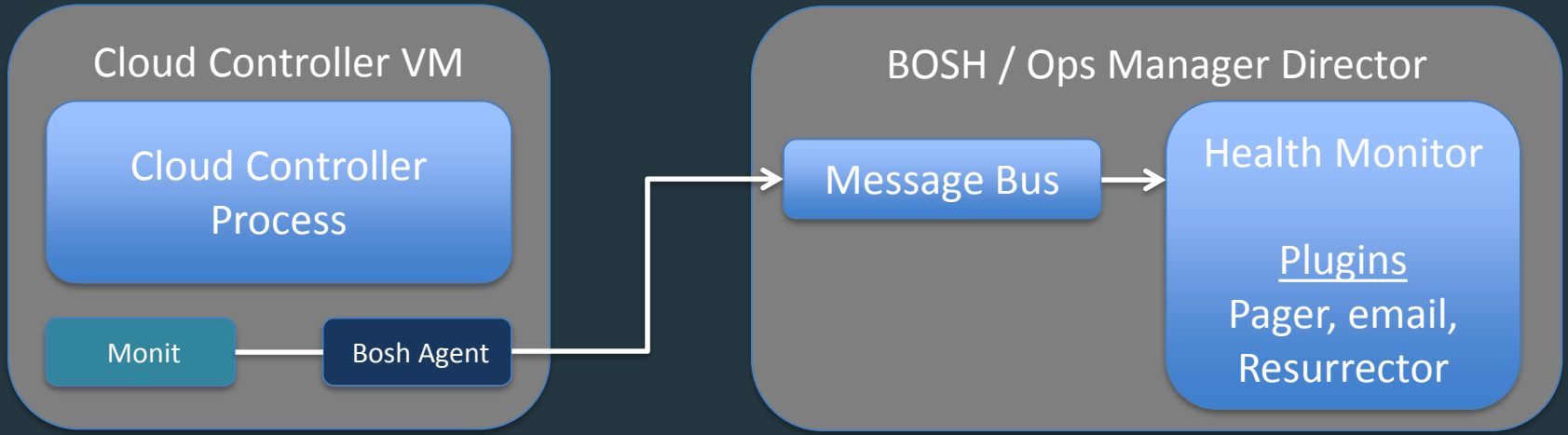
BOSH Managed Processes

Elastic runtime processes are monitored and automatically restarted.

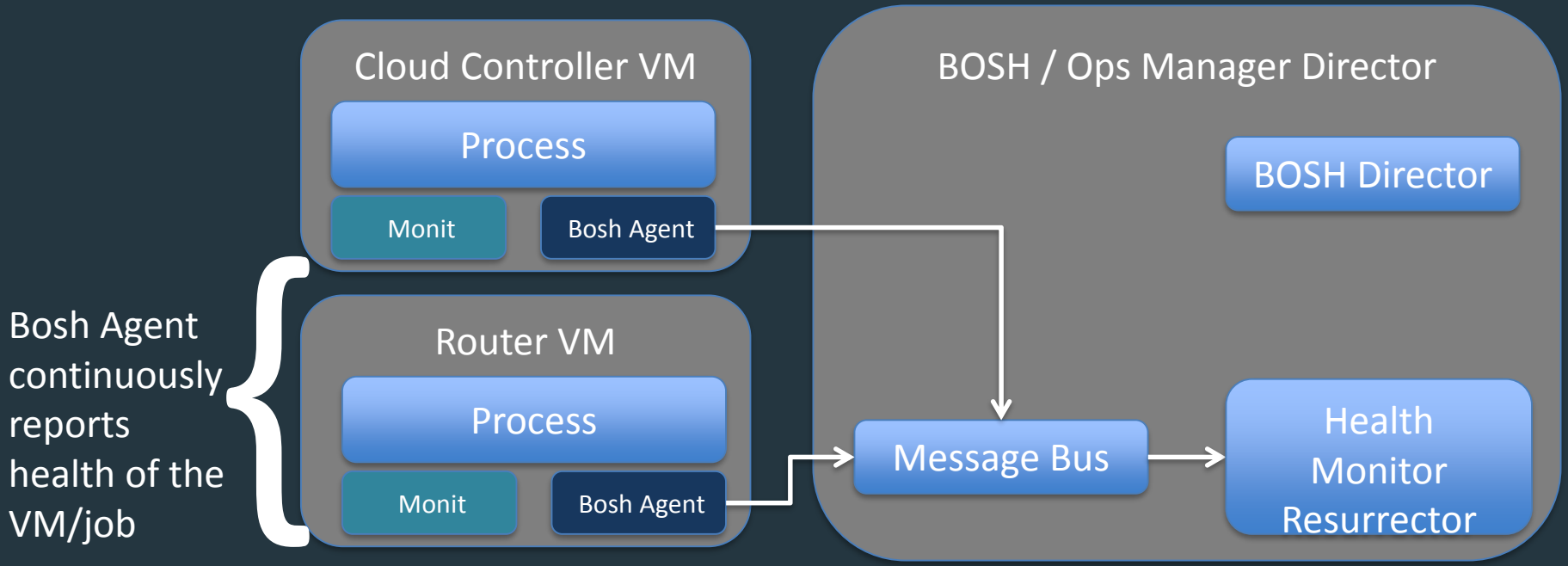


BOSH Managed Processes

Restart event is reported back to the Health Monitor for further investigation.

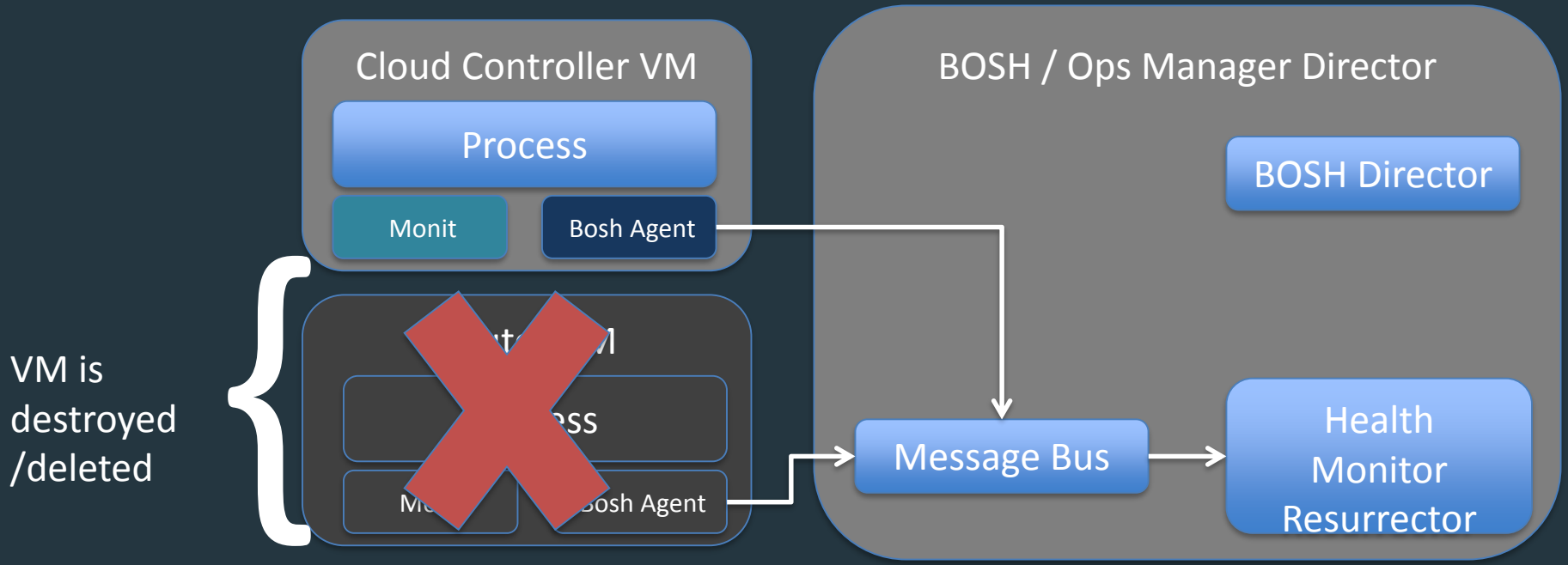


Failed VMs will be recreated automatically.

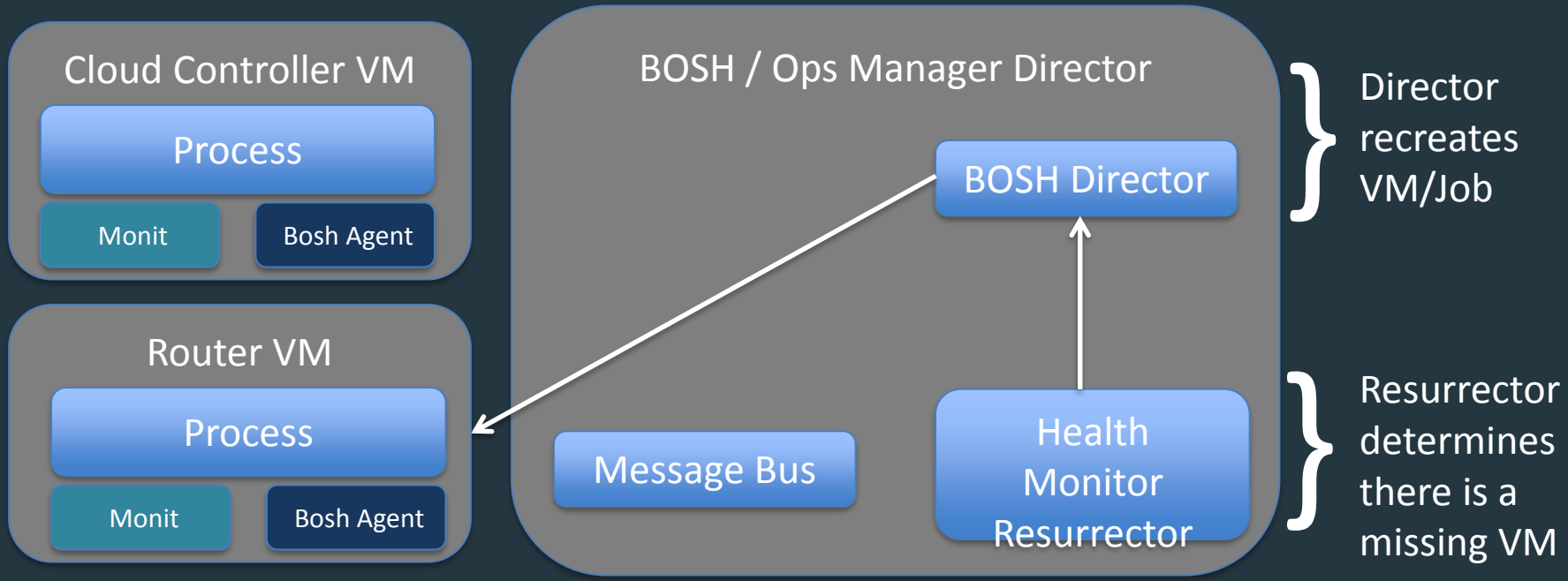


Failed VMs

Failed VMs will be recreated automatically.

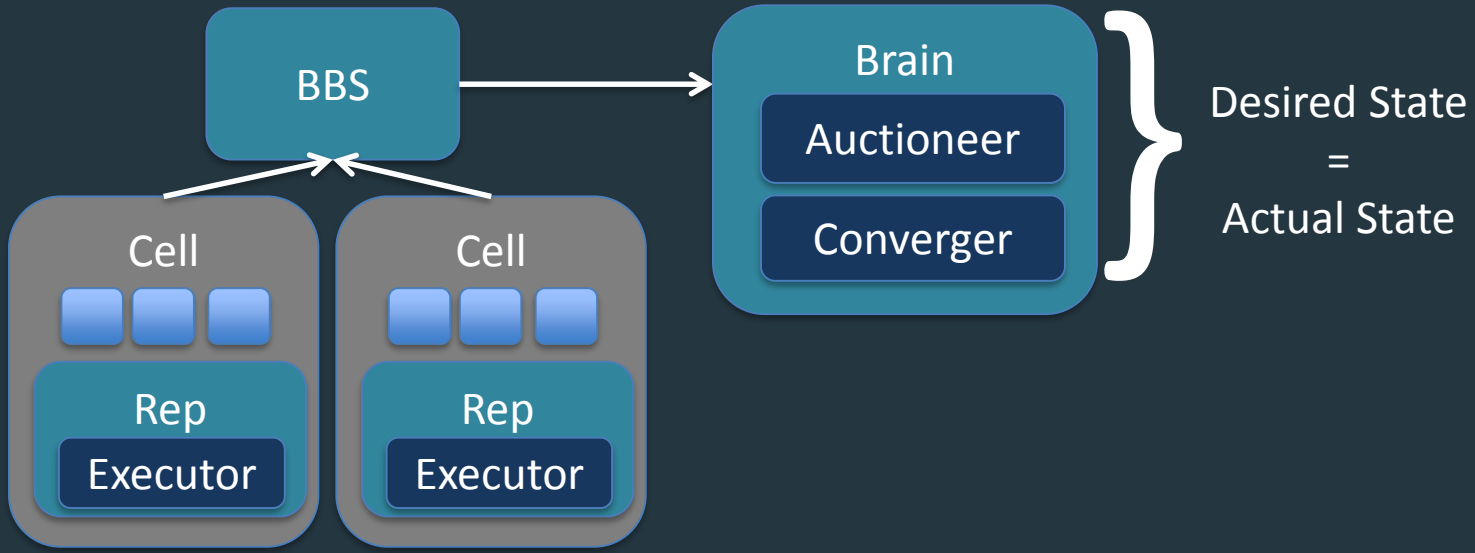


Failed VMs will be recreated automatically.



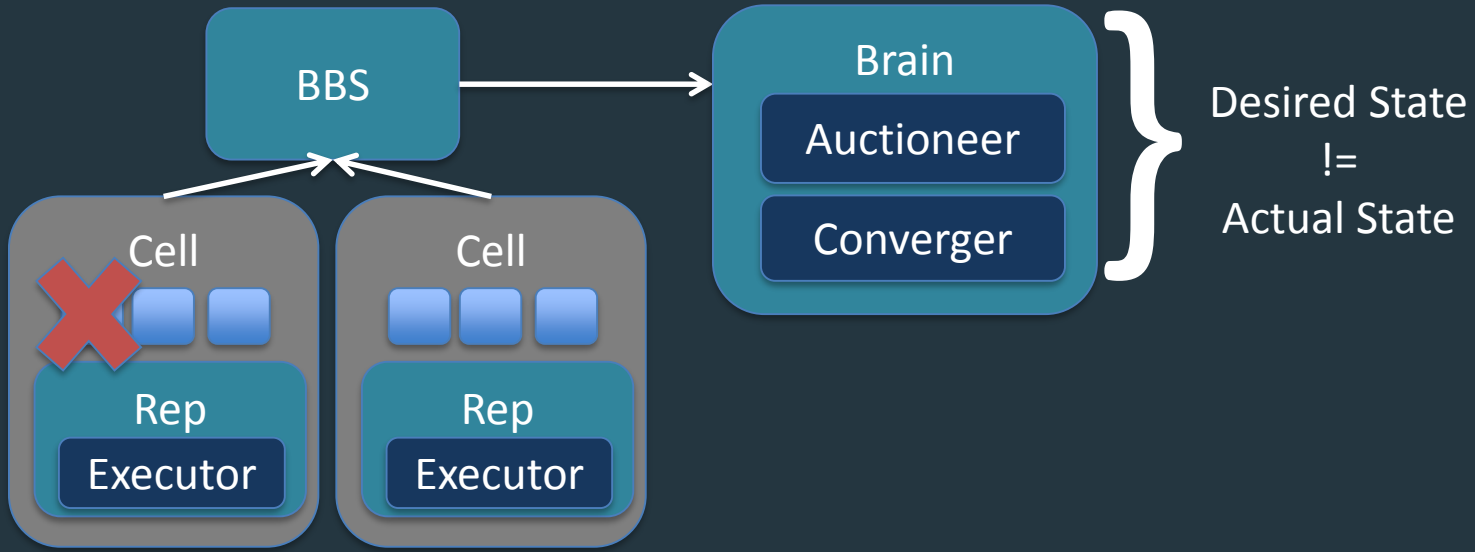
Self Healing Application Instances

Once running, failed application instances will be recreated.



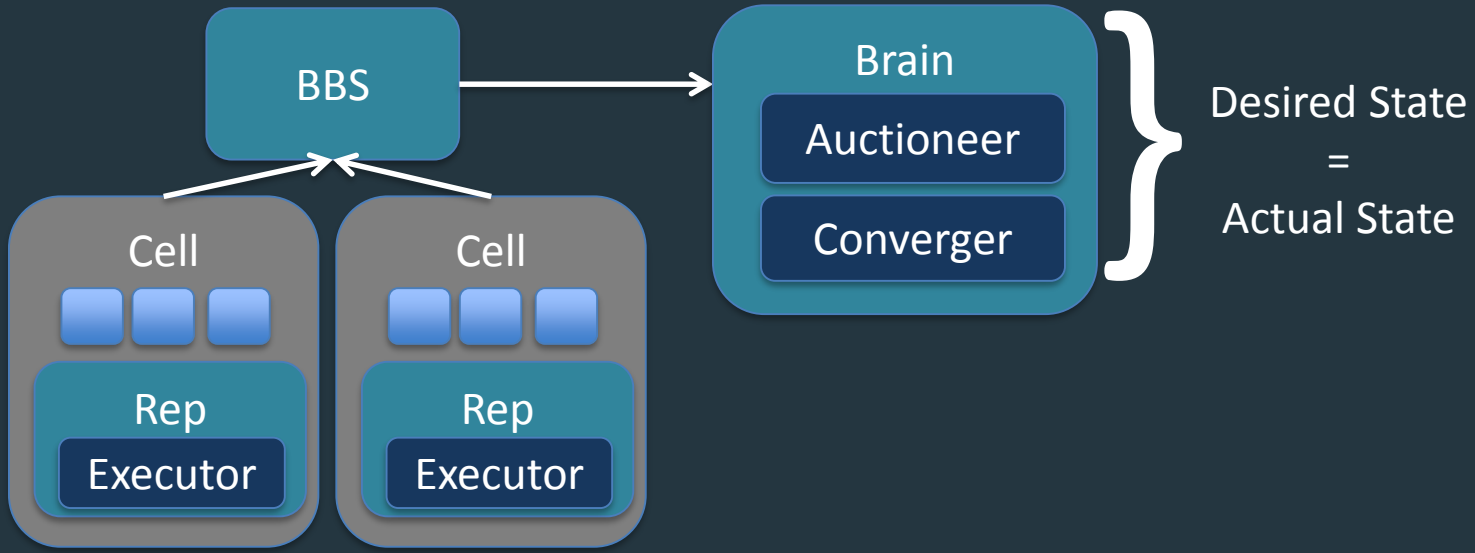
Self Healing Application Instances

Once running, failed application instances will be recreated.



Self Healing Application Instances

Once running, failed application instances will be recreated.




Pivotal
Transforming How The World Builds Software

Logging, Scale, and HA

Recap