The Serverless Survivalist

A Development Blueprint

COME TO CODE
DEVELOPERS AND MAKERS KINGDOM

About Me

Tech Lead @ Docebo

LinkedIn

Substack

GitHub

Sessionize

Glossary

- Managed - PaaS, no runtime code
- Serverless Compute - Compute w/o server management
- Serverful - Compute w/ server management
- Containerized - Runs on a container

Some Context

- AWS
- Serverless Compute / Managed mostly
- CDK/Cloudformation
- DevOps deam - Fully Autonomous
- Many deployments a day

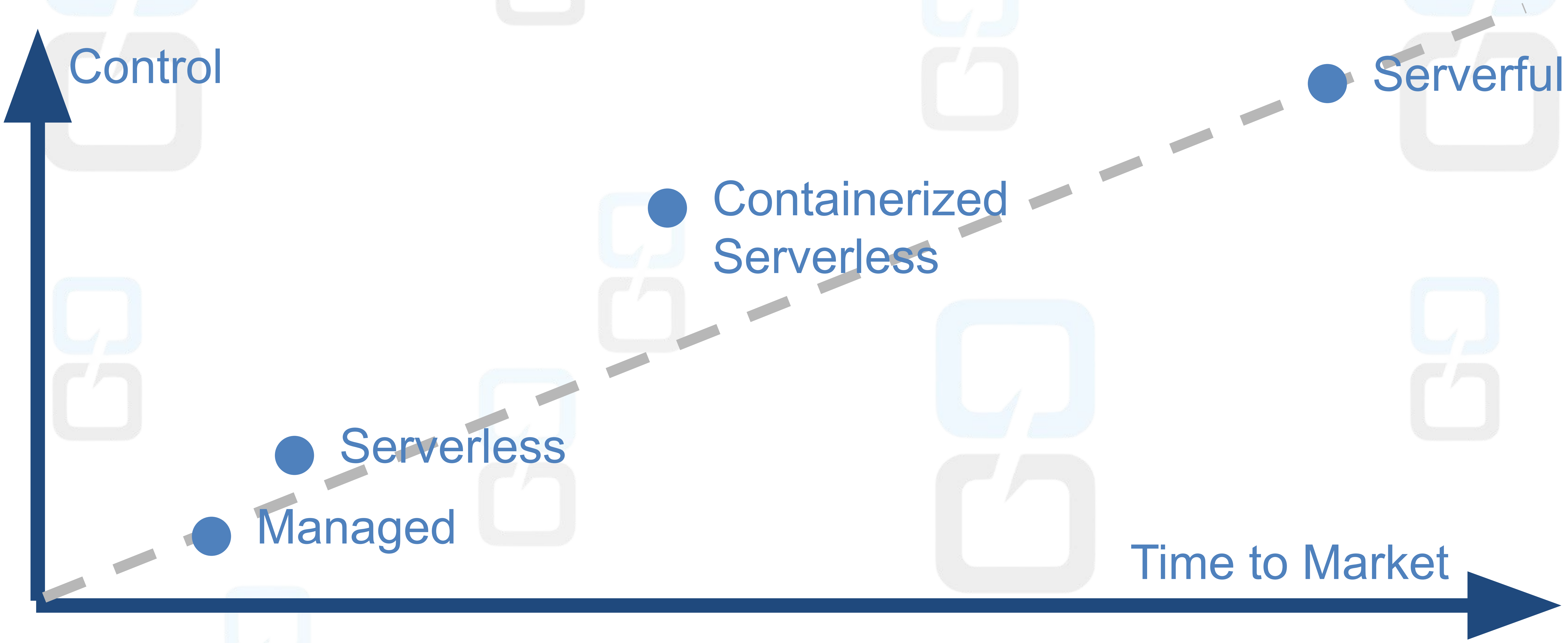
Reasons for Choosing Serverless

- Time to market
- Low operational cost / complexity
- Pay per use model
- Can scale higher than you think
- Autonomy

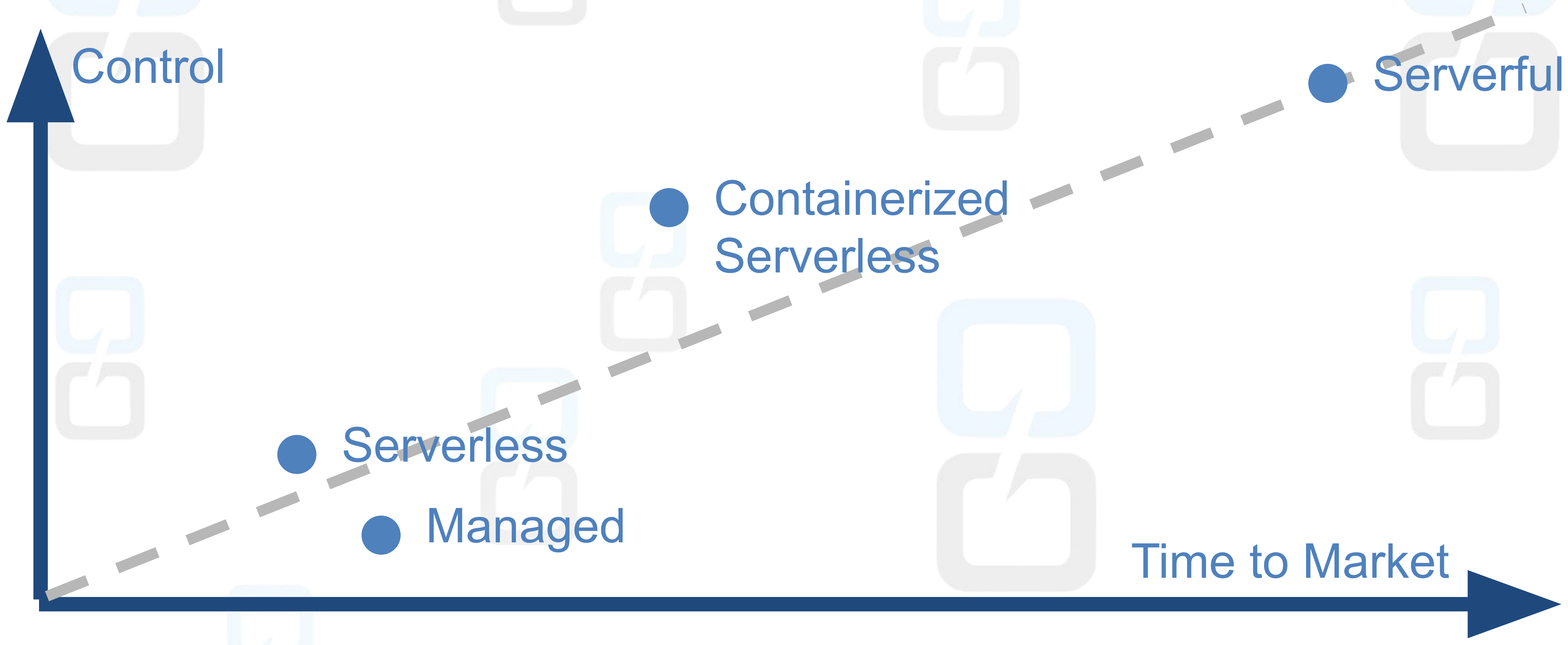
Wrong Reasons for Choosing Serveless

- I don't maintain/own any infra
- Less testing needed - are you sure?
- Just deploy it, it scales automatically

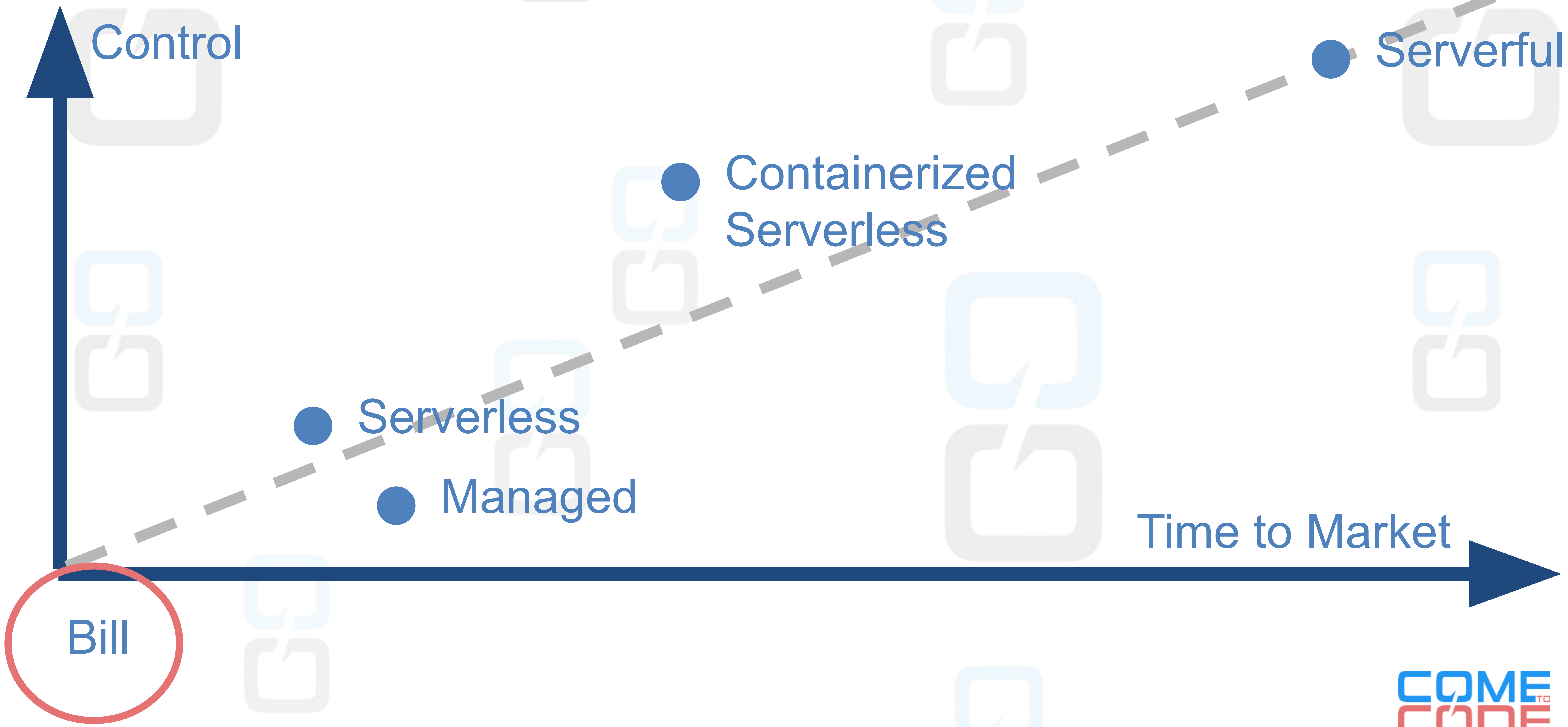
Managed to Serverless to Serverful



But Sometimes



Yes, but...


Control
Serverful
Containerized Serverless
Serverless
Managed
Time to Market
Bill

Average
Serverless
Developer

Survivalist Serverless Developer

Survivalist Development Blueprint

- Continuous Delivery - the whole set of practices
- Testing strategy
- No downtime policy
- Cost Management
- Get good at Architecting

Continuous Delivery

COME TO CODE
DEVELOPERS AND MAKERS KINGDOM

Continuous Delivery

Deploy small changes often

Continuous Delivery

- Deploy small changes often

Use ephemeral environments to shift left feedback

Continuous Delivery

- Deploy small changes often
- Use ephemeral environments to shift left feedback

Always have a rollback strategy

Continuous Delivery

- Deploy small changes often
- Use ephemeral environments to shift left feedback
- Always have a rollback strategy

But favour fixing forward

Continuous Delivery

- Deploy small changes often
- Use ephemeral environments to shift left feedback
- Always have a rollback strategy
- Favour fixing forward

Testing Strategy

Testing Strategy

You cannot have Cloud Services on your machine

Testing Strategy

But Cloud Services are part of your business logic

Testing Strategy

What now?

COME TO CODE
DEVELOPERS AND MAKERS KINGDOM

Testing Strategy

Test only runtime code with mocks

Mock Cloud Services

Test Manually in Dedicated envs

LocalStack

Testing Strategy

A casual friday deployment later…

The customer
I will look for you, I will find you and I will kill you.

COME TO CODE

Testing Strategy

Where do Cloud Services behave like Cloud Services?

Testing Strategy

The Cloud

Effective Testing Strategy

Pyramid to Prisma

Effective Testing Strategy

- Pyramid to Prisma

Deps Check/Licenses/Code Quality/Secrets Detection on whole codebase

Effective Testing Strategy

- Pyramid to Prisma
- Deps Check/Licenses/Code Quality/Secrets and more on whole codebase

Snapshots/unit + SAST for IaC code

Effective Testing Strategy

- Pyramid to Prisma
- Deps Check/Licenses/Code Quality/Secrets and more on whole codebase
- Snapshots/unit + SAST for IaC code

Unit tests + SAST for runtime code

Effective Testing Strategy

- Pyramid to Prisma
- Deps Check/Licenses/Code Quality/Secrets and more on whole codebase
- Snapshots/unit + SAST for IaC code
- Unit tests + SAST for runtime code

Effective Testing Strategy

Integration/DAST/Fuzz tests against deployed services

COME TO CODE
DEVELOPERS AND MAKERS KINGDOM

Effective Testing Strategy

- Integration/DAST/Fuzz tests against deployed services

Acceptance tests against integrated team’s system

Effective Testing Strategy

- Integration/DAST/Fuzz tests against deployed services
- Acceptance tests against integrated team's system

Production Observability

Effective Testing Strategy

- Integration/DAST/Fuzz tests against deployed services
- Acceptance tests against integrated team’s system
- Production Observability/Monitoring/Tracing

No Downtime Policy

No Downtime Policy

We can use a maintenance window…

Still the customer

No Downtime Policy

Deployment plan

No Downtime Policy

- Deployment plan

Use N-steps deployments

No Downtime Policy

- Deployment plan
- Use N-steps deployments

Validate each deployment

No Downtime Policy

- Deployment plan
- Use N-steps deployments
- Validate each deployments

Create new, switch, destroy old

No Downtime Policy

- Deployment plan
- Use N-steps deployments
- Validate each deployments
- Create new, switch, destroy old

Avoid maintenance windows at all costs

No Downtime Policy

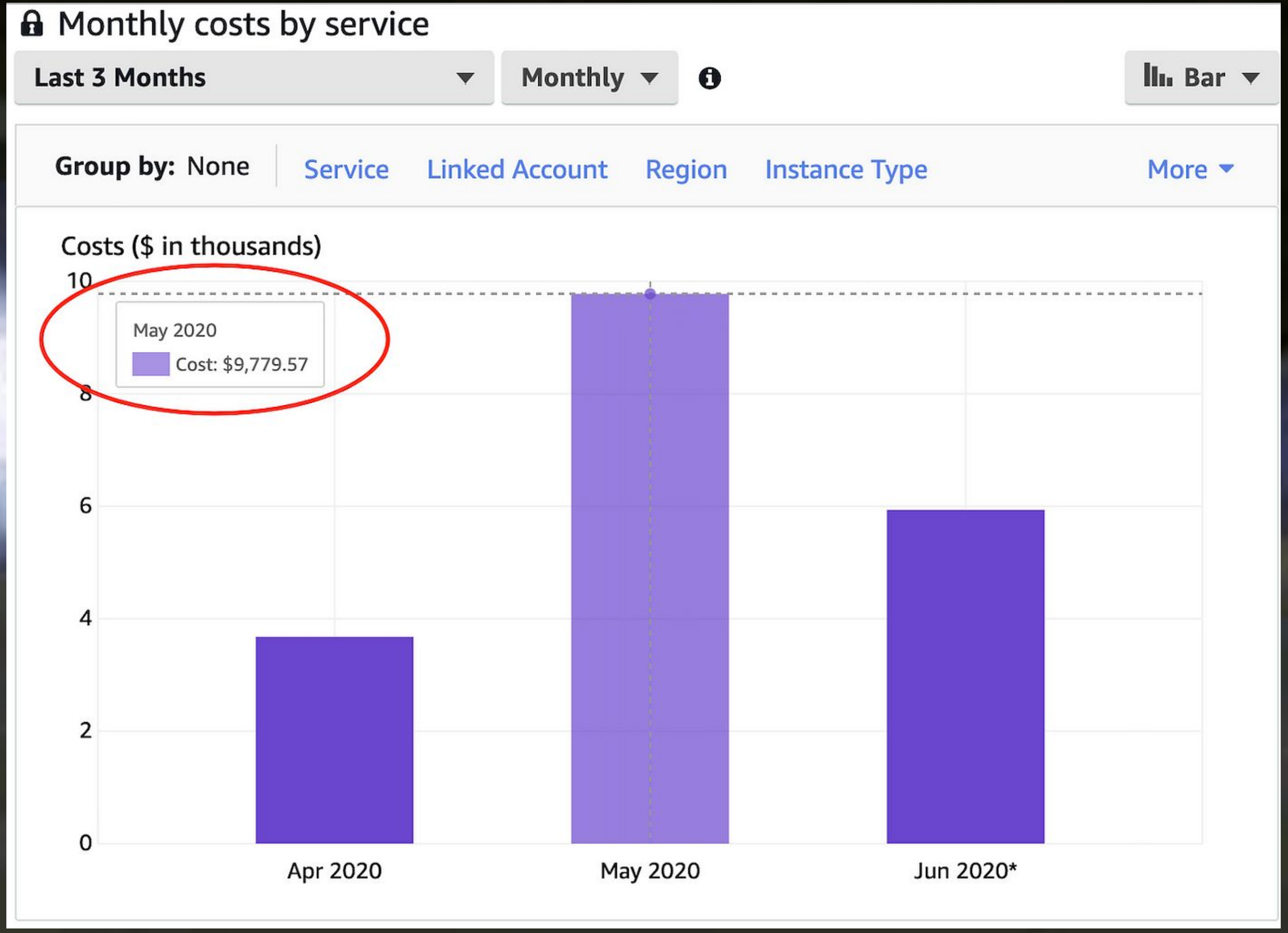
- Deployment plan
- Use N-steps deployments
- Validate each deployments
- Create new, switch, destroy old
- Avoid maintenance windows at all costs

Costs Management

Costs Management

My app went viral, it's awesome!

AWS Bill



Monthly costs by service

Last 3 Months | Monthly | Bar

Group by: None | Service | Linked Account | Region | Instance Type | More

Costs Management

Monitor Costs

Costs Management

- Monitor Costs

Set budgets as needed

Costs Management

- Monitor Costs
- Set budgets as needed

Set service limits

Costs Management

- Monitor Costs
- Set budgets as needed
- Set service limits

Move load to containerized/serverful when convenient

Costs Management

- Monitor Costs
- Set budgets as needed
- Set service limits
- Move load to containerized/serverful when convenient

Take infra into account in the business model (and the other way around)

Costs Management

- Monitor Costs
- Set budgets as needed
- Set service limits
- Move load to containerized/serverful when convenient
- Take infra into account in the business model (and the other way around)

Get Good at Architecting

Get Good at Architecting

Architectural Components > Cloud Services

Get Good at Architecting

- Architectural Components > Cloud Services

Data flows, ADRs, Illities

Get Good at Architecting

- Architectural Components > Cloud Services
- Data flows, ADRs, Illities

Design for failure

Get Good at Architecting

- Architectural Components > Cloud Services
- Data flows, ADRs, Illities
- Design for failure

Wrap Up

Wrap Up

- Frequent baby steps deployments
- Rethink testing strategy
- Share ownership across the whole team
- Avoid any downtime - Including maintenance windows
- Manage Costs
- Everything fails all the time

Additional Resources

Streamline Cloud Development with Ephemeral Environments

Continuous Delivery in Practice: the Pipeline

How to test serverless functions and applications - AWS Lambda

Testing serverless applications on AWS - AWS Prescriptive Guidance

COME TO CODE
DEVELOPERS AND MAKERS KINGDOM

Additional Resources

Managing your costs with AWS Budgets

Practicing Continuous Integration and Continuous Delivery on AWS

Architect Elevator Blog

COME TO CODE
DEVELOPERS AND MAKERS KINGDOM

Questions?

COME TO CODE
DEVELOPERS AND MAKERS KINGDOM

Leave feedback!

COME TO CODE
DEVELOPERS AND MAKERS KINGDOM

Thank you!

COME TO CODE
DEVELOPERS AND MAKERS KINGDOM