Improving Performance with Server-side Rendering and Serverless Computing

Peter Kellner

DEVELOPER, CONSULTANT AND AUTHOR

@pkellner   linkedin.com/in/peterkellner99   ReactAtScale.com

Serverless Computing

All about GraphQL server and no interaction with client

Absolutely a server involved

Every request handled potentially by a different server

No server idle time

Server-side Rendering

All about GraphQL client and React

Web application running inside Node server

HTML rendered in Node server and downloaded as response
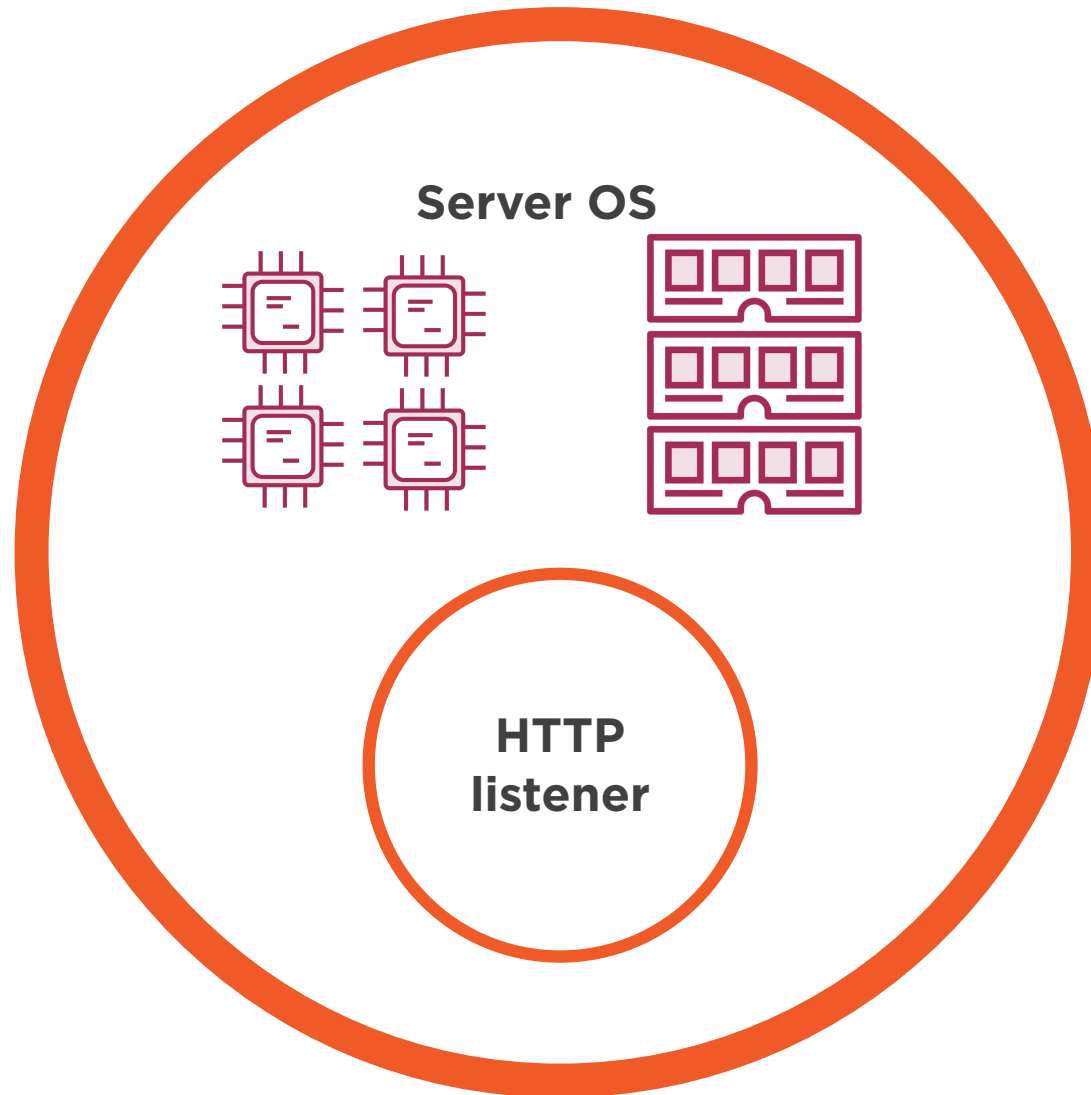
Coming up in This Module

Serverless Computing

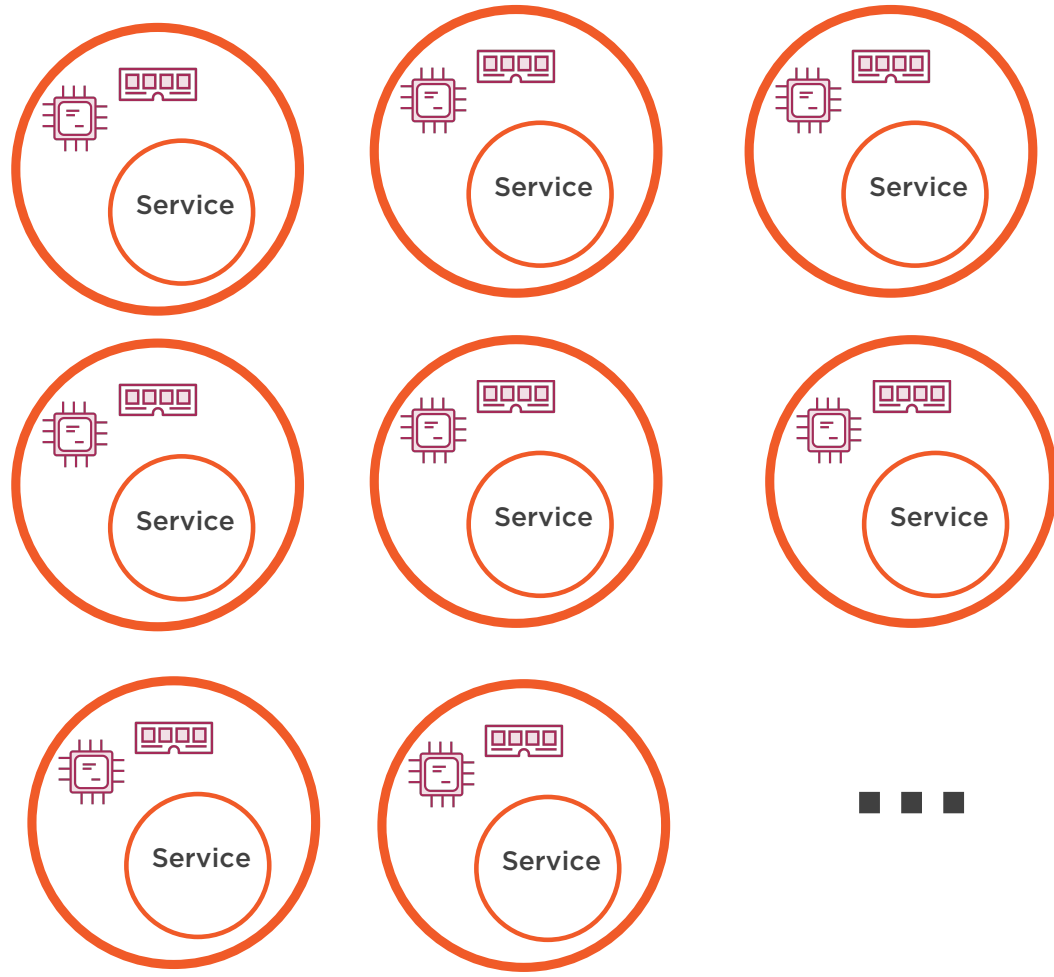**Learn pluses and minuses of Serverless computing**

Server-side Rendering

Build out a full implementation of Server-side rendering

Typical Server Based Solution With Apollo
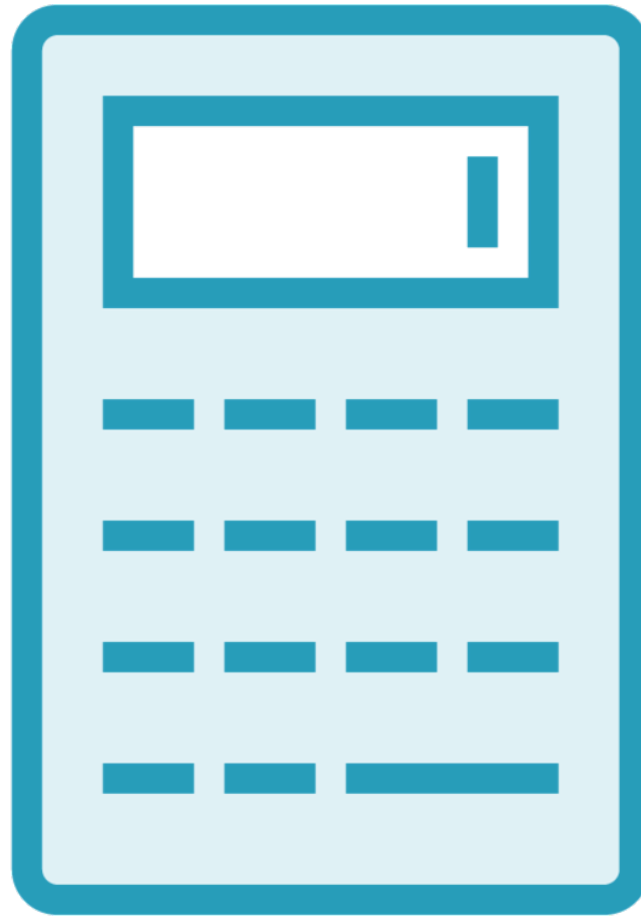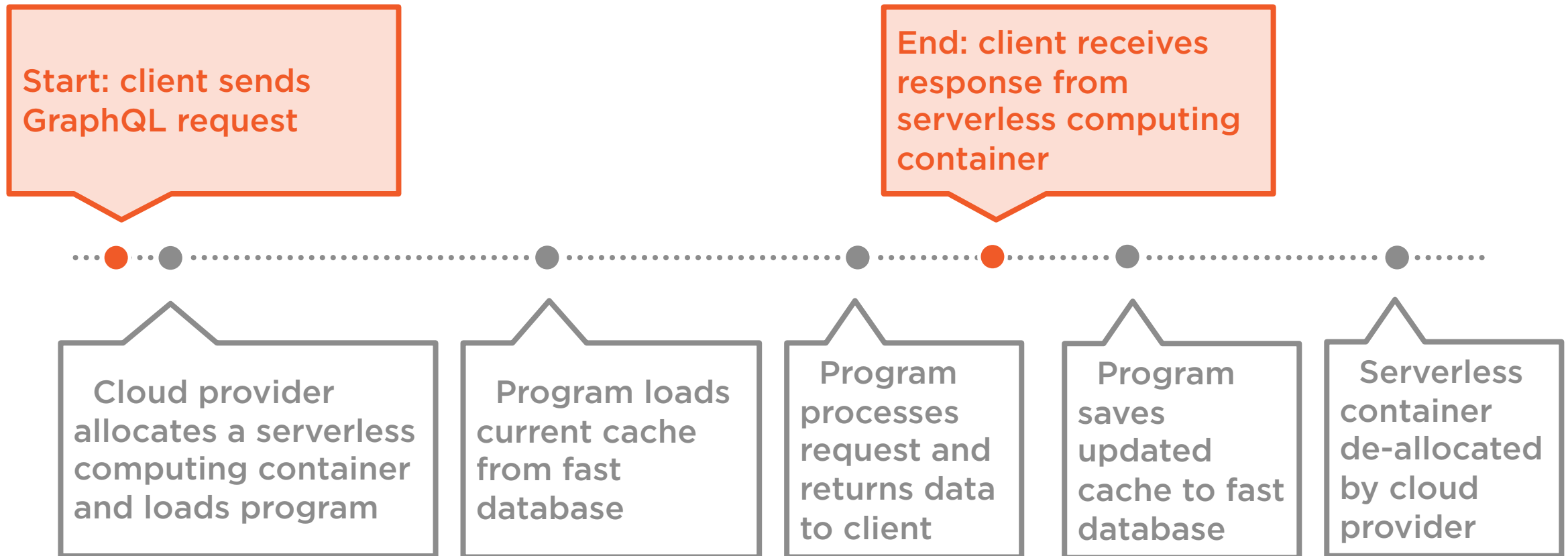
Typical Serverless Solution With Apollo

Service

Service

Service

Service

Service

Service

Service

Service

. . .

Common fast database

Common file system

Serverless App Easy To Implement

Apollo GraphQL Serverless Instance Timeline

Start: client sends GraphQL request

End: client receives response from serverless computing container

Cloud provider allocates a serverless computing container and loads program

Program loads current cache from fast database

Program processes request and returns data to client

Program saves updated cache to fast database

Serverless container de-allocated by cloud provider

When running React apps in Node, all async calls must complete before returning.

Next.js designed to make server-side rendered apps easy.

Takeaways

Serverless and Server-side bring big benefits

Serverless computing focused on devops

Server-side benefits

> **Faster response times**

> **Better SEO**

Feedback on forums and GitHub

Stay safe!