What Serverless Computing Is and Should Become: The Next Phase of Cloud Computing

Autor: Acquila Santos Rocha
Disciplina: Sistemas Distribuídos (Mestrado)
Professor: Sérgio Teixeira de Carvalho

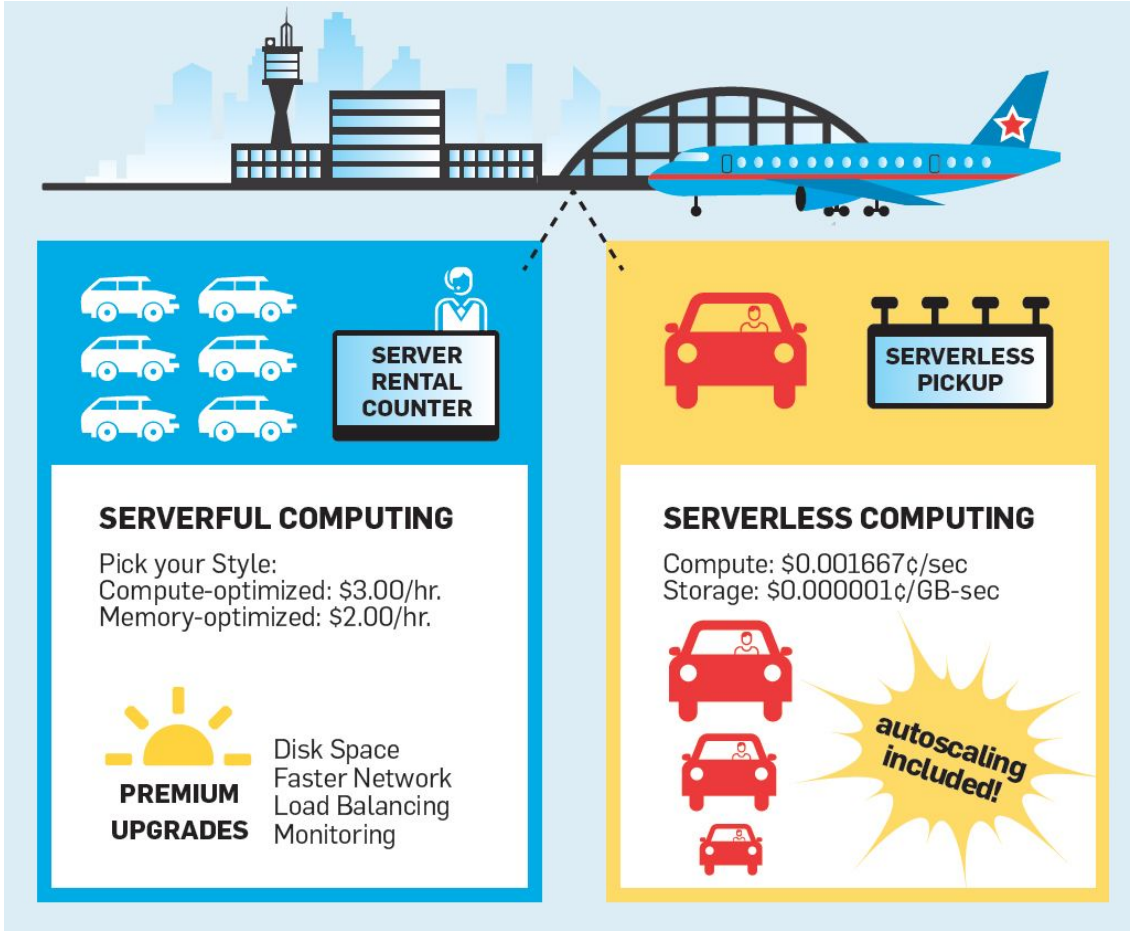
Introduction - Cloud Computing Phases

First Phase: mainly simplified system administration by making it easier to configure and manage computing infrastructure.

- **Target:** System Administrators
- **Called:** Serverful Computing

Second Phase: hides the servers by providing programming abstractions for application builders that simplify cloud development, making cloud software easier to write.

- **Target:** Programmers
- **Called:** Serverless Computing


SERVER RENTAL COUNTER
SERVERLESS PICKUP
SERVERFUL COMPUTING
Pick your Style:
Compute-optimized: $3.00/hr.
Memory-optimized: $2.00/hr.
PREMIUM UPGRADES
Disk Space
Faster Network
Load Balancing
Monitoring
SERVERLESS COMPUTING
Compute: $0.001667¢/sec
Storage: $0.000001¢/GB-sec
autoscaling included!

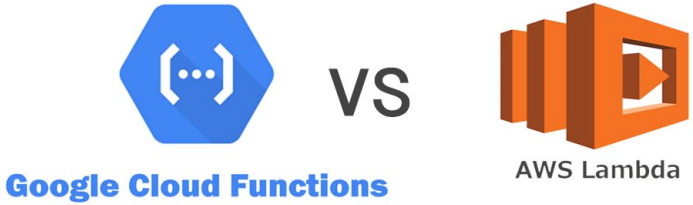
Serverless Computing - Cloud Functions

Function-as-a-Service (FaaS): Managed cloud function services

How it is done: programmers create applications using high-level (HL) abstractions offered by the cloud provider.

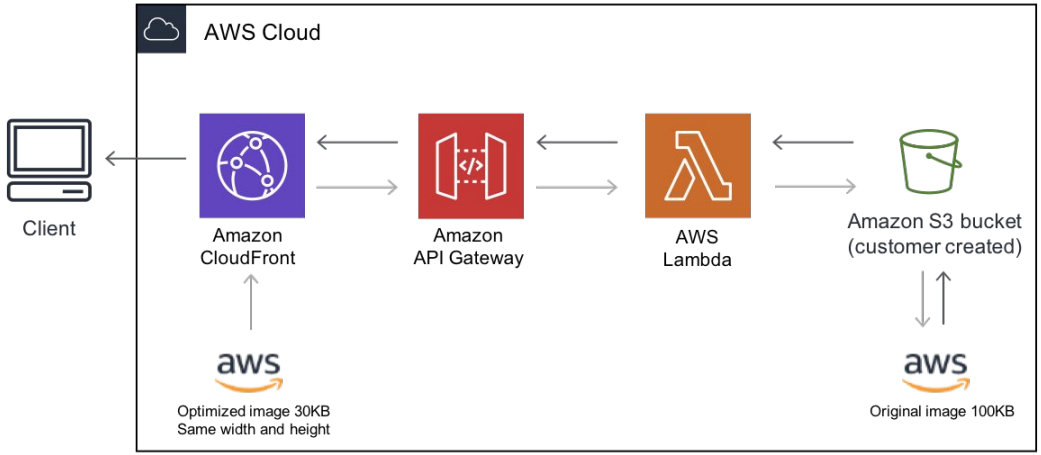
- **Example:** *cloud functions**

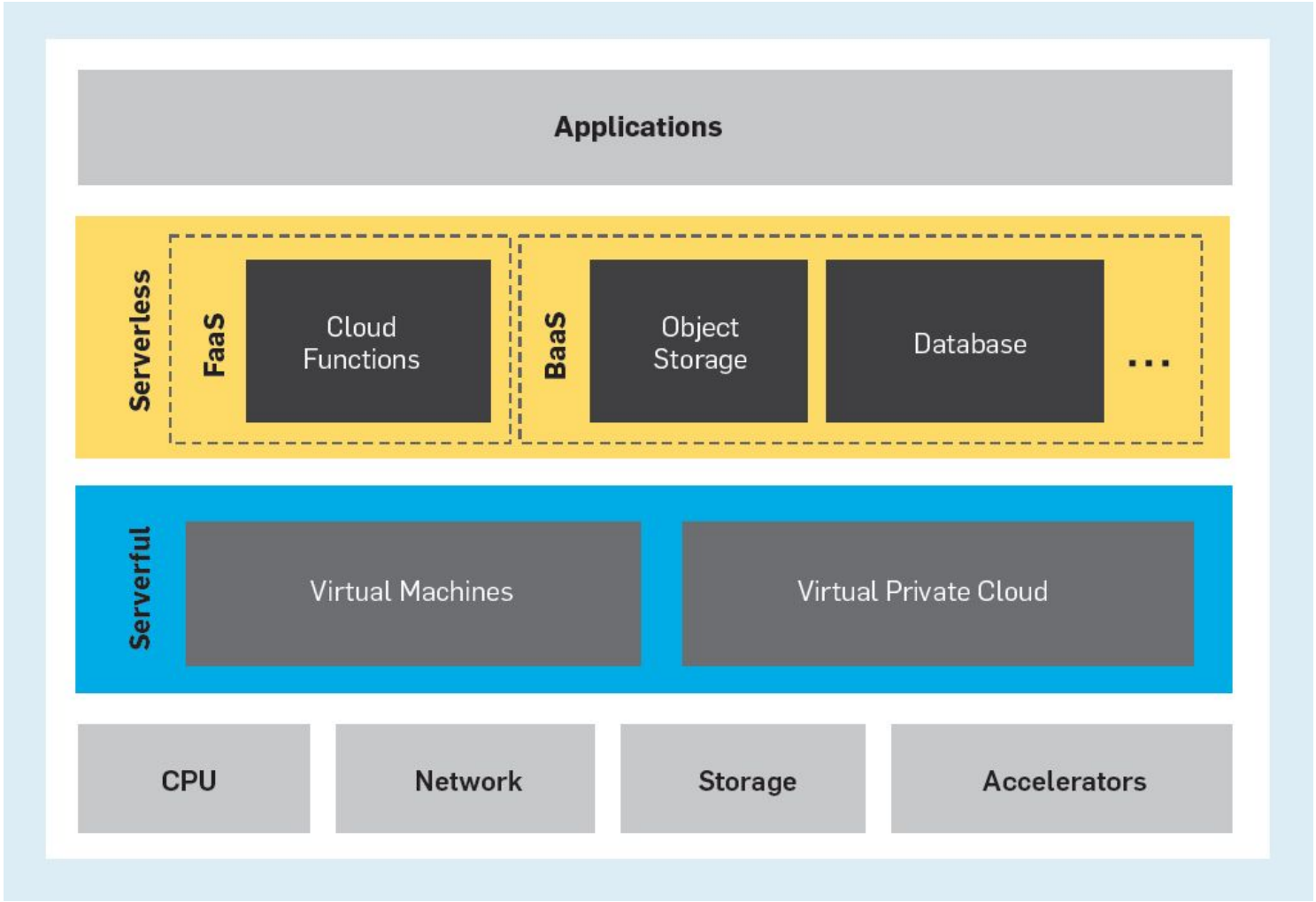
** using functional-style "stateless" programming in the language of their choice, then specify how the functions should run, whether in response to Web requests or to triggering events.*



Serverless Computing - Backend as a Service

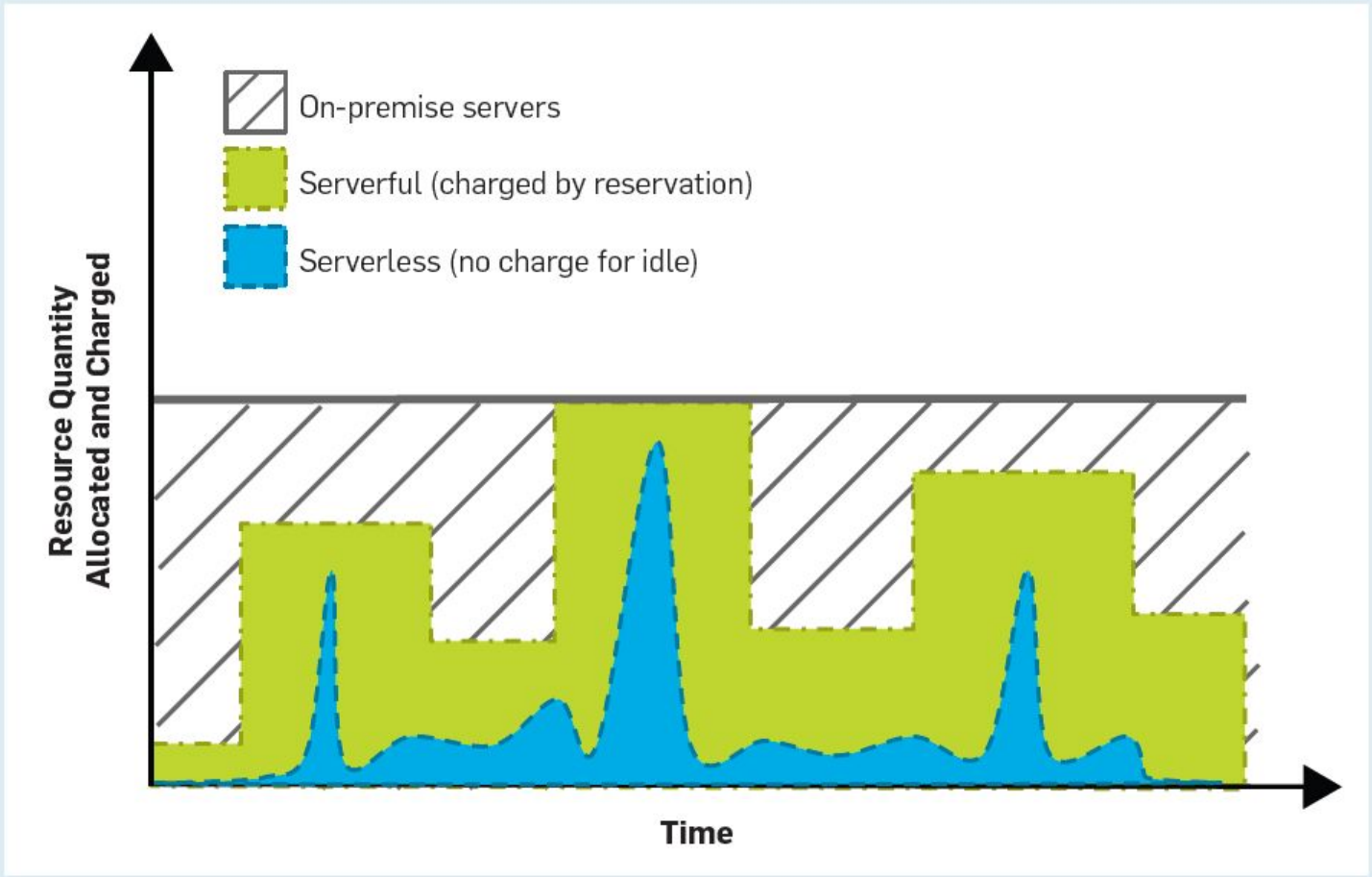
Cloud functions may also consume **serverless** object storage, message queues, key-value store databases, a group of services offerings known collectively as **Backend-as-a-Service** (BaaS).




Applications
Serverless
FaaS
Cloud Functions
BaaS
Object Storage
Database
...
Serverful
Virtual Machines
Virtual Private Cloud
CPU
Network
Storage
Accelerators

Serverless Computing - Innovation

- The main **innovation** of serverless is hiding servers, which have an inherently complex programming and operating model.
- Adjust capacity heuristically, a form of autoscaling, but these too require detailed configuration and ongoing monitoring.
- Establish autoscaling parameters.


On-premise servers
Serverful (charged by reservation)
Serverless (no charge for idle)
Resource Quantity
Allocated and Charged
Time

Summarizing - Qualities of Serverless Computing

1. Providing an abstraction that hides the servers and the complexity of programming and operating them.

2. Offering a pay-as-you-go cost model instead of a reservation-based model, so there is no charge for idle resources

3. Automatic, rapid, and unlimited scaling resources up and down to match demand closely, from zero to practically infinite.

“We project that the majority of data center computing will be dominated by serverless computing but we also believe that serverless computing will depart substantially from the serverless offerings of today”

A Brief History of Cloud

Data Centre

Traditional way to provide infrastructure

IAAS

Provides AWS CLI or AWS API to manage infrastructure (AWS EC2)

PAAS

Provides infrastructure based on developed application (AWS Elastic Beanstalk)

Serverless

Provides infrastructure automatically when the lambda function gets executed (AWS Lambda)

The Next Phase of Cloud Computing

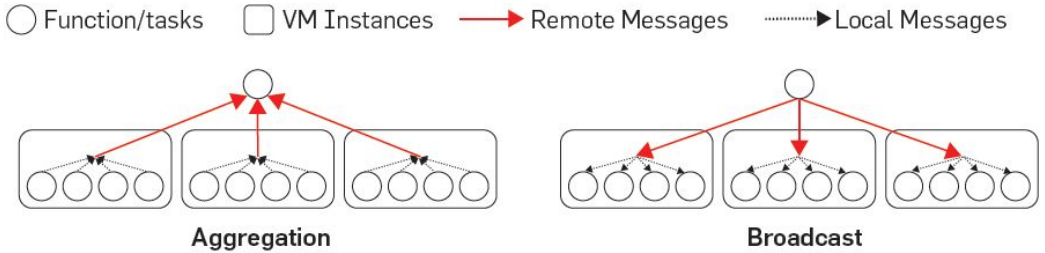
Application-specific abstractions solve a particular use case, and several of them exist in products today.

General-purpose abstractions must work well in a broad variety of uses and remain a research challenge.

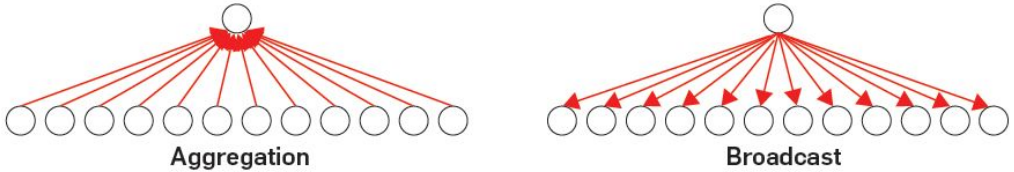
Cloud functions might appear to offer a general-purpose abstraction since they run arbitrary code, however due to their limitations they work only in some applications. More sophisticated derivatives might achieve the goal of general-purpose serverless computation.

Serverless Abstraction Approach		Big Data Example
Application-specific	Tool or component	AWS Athena
	Application framework	Cloud Dataflow
General-purpose	Hints to implementation	Affinity hints
	Automatic optimization	Communication-minimizing placement

Application-specific serverless frameworks (for example, Cloud Dataflow) can be implemented with serverful communication patterns. In this case (a) the fewer arrows indicate less network communication than in (b) the general-purpose serverless option. By packing K tasks per VM instance, an application-specific serverless solution, like a serverful solution, is able to achieve a communication complexity of $O(N/K)$ for a job with N tasks, as opposed to $O(N)$ for the cloud function based alternative which can not influence task placement. Typical values for K range from 10 to 100, leading to an overall difference of one to two orders of magnitude.



(a)
Serverful communication patterns in application-specific serverless



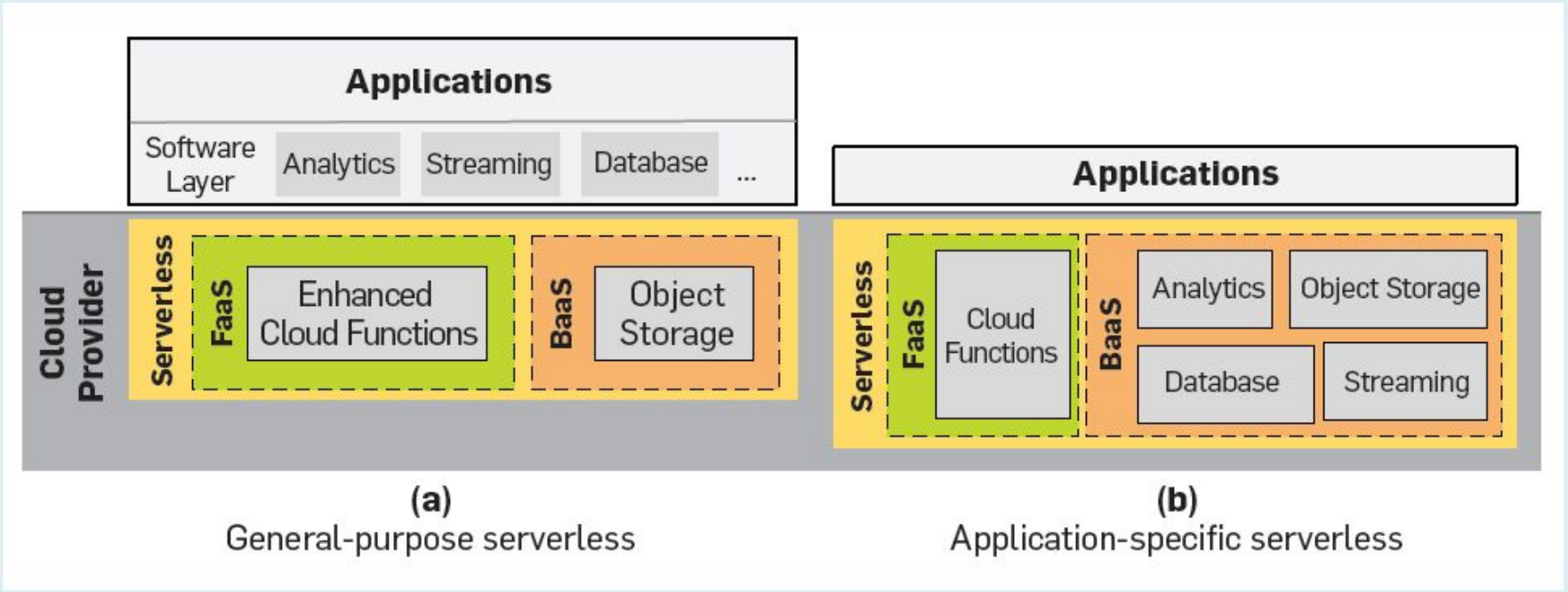
(b)
Communication when attempting general-purpose serverless with cloud functions

The Next Phase of Cloud Computing

Two suggested paths

1. **Hints** provided by the programmer might indicate how to achieve better performance.
2. Inefficiencies being removed by **automatic optimization**. Infer locality optimizations from observed communication patterns.

(a) General-purpose serverless abstractions support a wide range of needs, with application-specific functionality provided by software above the cloud provider interface, (b) application-specific serverless abstractions with many BaaS point-solutions.



The Next Phase of Cloud Computing

Today: serverless computing remains entirely of the application-specific variety. Even cloud functions, which can execute arbitrary code, are popular mainly for stateless API serving and event-driven data processing.

Expectation: potential emergence of general-purpose serverless abstractions, which could host software ecosystems catering to every need.

"In our view, only the general-purpose approach can ultimately displace servers to become the default form of cloud programming. However, general-purpose serverless technology does not exist today, and developing it presents research challenges."

Research Challenges

State management

Networking

Predictable performance

Security

Programming Languages

Machine learning

Hardware