SERVERLESS COMPUTING: ONE STEP FORWARD, TWO STEPS BACK

JOSEPH M. HELLERSTEIN, JOSE FALEIRO, JOSEPH E. GONZALEZ, JOHANN SCHLEIER-SMITH, VIKRAM SREEKANTI, ALEXEY TUMANOV AND CHENGGANG WU

UC BERKELEY
WHAT IS SERVERLESS?

Has Serverless become a marketing term?
WHAT IS SERVERLESS?

Has Serverless become a marketing term?
- Yes and No

Ben Kehoe  
@ben1kehoe

🤔 “serverless” was always destined to become as meaningless as "cloud", and it’s always been a spectrum anyway. Don’t argue about what is or isn’t serverless, it’s to talk about what the desired “benefits” of serverless are, and how much a given technology provides them.

Linda Nichols @lynnaloo · May 26, 2021

Here's your annual reminder that #Serverless is a catchy, made-up marketing word.

I'm not sure it's worth all of the Twitter gatekeeping and gaslighting to protect the virtue of tech buzzwords.

Matt Coulter  
@NIDeveloper

Even though I have been a huge proponent of serverless 1st archs to enable rapid dev of business value over the past 5ish years I no longer say serverless because it is a hijacked term with no meaning. We need a new approach to realise what serverless wanted to achieve.

Matt Coulter @NIDeveloper · Jan 5
Replies to @LeeJamesGilmore
I have flipped on this but when I think deeply, the agile manifesto is more toxic overall than helpful. I honestly think we should stop basing our strategies on marketing terms that shift and define an architecture style but the name serverless is gone for that purpose

5:31 AM · Jan 5, 2023 · 23.5K Views
WHAT IS SERVERLESS?

Is Serverless a marketing term?

- Yes and No

Simon Wardley
swardley

X: How do you define serverless?
Me: Roughly speaking it's an event driven, utility based, stateless, code execution environment in which you write code and consume services. A boundary condition is "write code" i.e. any lower than this and it's not serverless.

Kelsey Hightower
kelseyhightower

I now understand what all the Serverless fuss is about. When you have a great idea the last thing you want to do is setup infrastructure.
COMPUTING TRENDS WITH SERVERLESS

1st Gen Serverless platforms place autoscaling at odds with dominant trends in modern computing:

- Data-centric
- Distributed Computing
- Open source
- Custom hardware

These gaps limit current serverless offerings!!

DevOps

Serverless is awesome (if you overlook inflated costs, dislike distributed computing, love vendor lock-in), say boffins

If 2019 is the year you try AWS Lambda et al, then here are pitfalls to look out for

By Thomas Claburn in San Francisco 19 Dec 2018 at 21:06 25 ▼ SHARE ▼
New computing platforms have fostered innovation in PLs and there’s now new PLs for the Cloud.

Cloud today is used as an outsourcing platform for standard enterprise data services.

Multitenancy & Administrative simplicity is desirable given the millions of cores and exabytes of storage.
DEVELOPERS NEED NOT WORRY ABOUT PROVISIONING

There has been an increasing interest from the research community:

Figure 1: Google Trends for “Serverless” and “Map Reduce” from 2004 to time of publication.
"SERVERLESS" GOES FAAS

"A FaaS offering by itself is of little value, since each function execution is isolated and ephemeral. Building applications on FaaS requires data management in both persistent and temporary storage, in addition to mechanisms to trigger and scale function execution. As a result, cloud providers are quick to emphasize that serverless is not only FaaS. It is FaaS supported by a “standard library”: the various multitenanted, autoscaling services provided by the vendor."
FORWARD, BUT ALSO BACKWARD

- Faas offerings are autoscaling
  - Faas offerings ignore efficient data processing
  - Faas offerings stymie the development of distributed systems
IS SERVERLESS MORE? (USE CASES)

Embarrassingly parallel functions: Independent tasks that don't need communication with other functions

Orchestration functions: Orchestrate calls to proprietary autoscaling services i.e. Analytics at scale

Function Composition: Collection of functions that pass along outputs as inputs. (Event-driven)
WHY SERVERLESS TODAY IS TOO LESS

**Limited Lifetimes:** 15 min timeouts & No recoverable state across invocations

**I/o Bottlenecks:** Connect to shared storage across a network interface

**Communication through slow storage:** Lambdas are not directly network-accessible

**No specialized hardware:** No mechanism/API to access specialized hardware
LIMITATIONS IN SCOPE OF SERVERLESS APPS
FAAS IS A DATA-SHIPPING ARCHITECTURE

- Functions run on VMs separate from data
- Functions are short lived
- Functions are non-addressable
- Internal state caching is limited
Functions pass data through slow expensive storage

DS protocols depend on:

- Fine-grained communication
- Leader election
- Membership
- Data consistency
- Transaction commit
MORE LIMITATIONS...

Hardware-accelerated software innovation: Big Data setups lack GPU specs for DL.

Open Source innovation: OSS deployment need human operation

IT IS NOT OVER YET?
MORE CASE STUDIES!!

- Experiments were set up to prove the problems of serverless computing
- **Experiment Settings:** Big Data & Distributed Computing
- **Experiment:** Machine Learning Model

<table>
<thead>
<tr>
<th></th>
<th>Func. Invoc. (1KB)</th>
<th>Lambda I/O (S3)</th>
<th>Lambda I/O (DynamoDB)</th>
<th>EC2 I/O (S3)</th>
<th>EC2 I/O (DynamoDB)</th>
<th>EC2 NW (0MQ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latency</td>
<td>303ms</td>
<td>108ms</td>
<td>11ms</td>
<td>106ms</td>
<td>11ms</td>
<td>290μs</td>
</tr>
<tr>
<td>Compared to best</td>
<td>1.045×</td>
<td>372×</td>
<td>37.9×</td>
<td>365×</td>
<td>37.9×</td>
<td>1×</td>
</tr>
</tbody>
</table>

**Table 1: Latencies.** We compare the latency of “communicating” 1KB in various ways. To model pure functional event-driven communication, we show the cost of invoking a no-op Lambda function on a 1KB argument, averaged over 1,000 calls. We then show the cost of two explicit 1KB I/Os (write-read) from Python Lambda function and an EC2 instance to S3 and DynamoDB, averaged across 5k trials. Finally we show the cost of direct messaging by measuring a 1KB network message roundtrip, measured using python and the ZeroMQ message library running across two EC2 instances, averaged across 10k trials.
SILVER LINING

- Operational flexibility over developer control
- Enable easy to write and debug code
- Think deeply about why & when to use coordination protocols
EARLY OBJECTIONS

Paper addresses limitations of Public FaaS as solution to general-purpose data rich programing

O: “You keep using that word. I do not think it means what you think it means.”

R: The delivery of a particular special purpose autoscaling backend service does not solve the problem of enabling general-purpose cloud programming.

O: “Just wait for the next network announcement!”

R: … Data center networks will surely improve, yet continue to play a limiting role in a larger memory hierarchy

O: “The main point is simple economics: Serverless is inevitable.”

R: … this business motion will not accelerate the sea change in computing that the cloud offers. Specifically, it will not encourage—and may even deter—third-party and open-source development of new stateful services, which are the core of modern computing.
<table>
<thead>
<tr>
<th>Goal</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Push</td>
<td>Push core tech down to the playing fields</td>
</tr>
<tr>
<td>Rethink</td>
<td>Rethink Infra design &amp; programming models to spark innovation</td>
</tr>
<tr>
<td>Vision</td>
<td>Vision for the Future: Cloud programmers should leverage compute &amp; storage of the cloud in an auto-scaling cost effective manner</td>
</tr>
</tbody>
</table>
HOW?

- Fluid Code and Data Placement
- Long-Running, Addressable Virtual Agents
- Disorderly programming
- Flexible Programming, Common IR
HOW? (CONTINUED)

- **Heterogeneous Hardware Support: Follow user-defined SLOs**
  - Make specialized hardware cost effective
  - Allow devs target specific hardware features to foster/innovative hardware/software co-design

- **SLO Guarantees: No available APIs for SLOs. Pricing is based on RAM(#cores) & running time.**
  - Enable upfront SLO pricing with penalties for mis-estimation
  - Requires smooth cost-surface in optimization

- **Security Concerns**
  - Code between shared data storage
  - Security Mgt related to multitenancy, rogue code
Serverless platforms pose interesting & surmountable challenges.

FaasS platform are not open source yet but could be improved with features like container orchestration.

Program analysis & scheduling open up new formal research avenues.
CONCLUSION

The authors are optimistic, about research and its impact on the cloud's future

The subtitle of the paper is relatable to most of the great things that have happened (Ex: MAP REDUCE !!)
THANK YOU