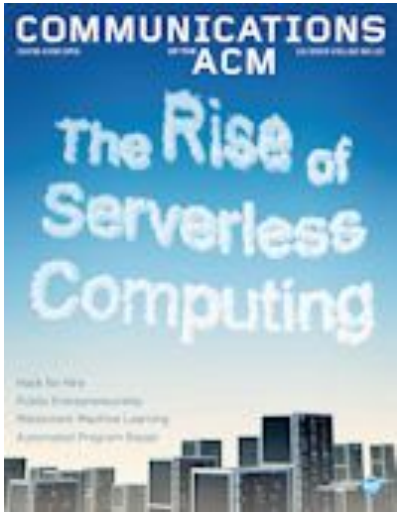
Rise of Containers and Serverless: Past, Present, Future



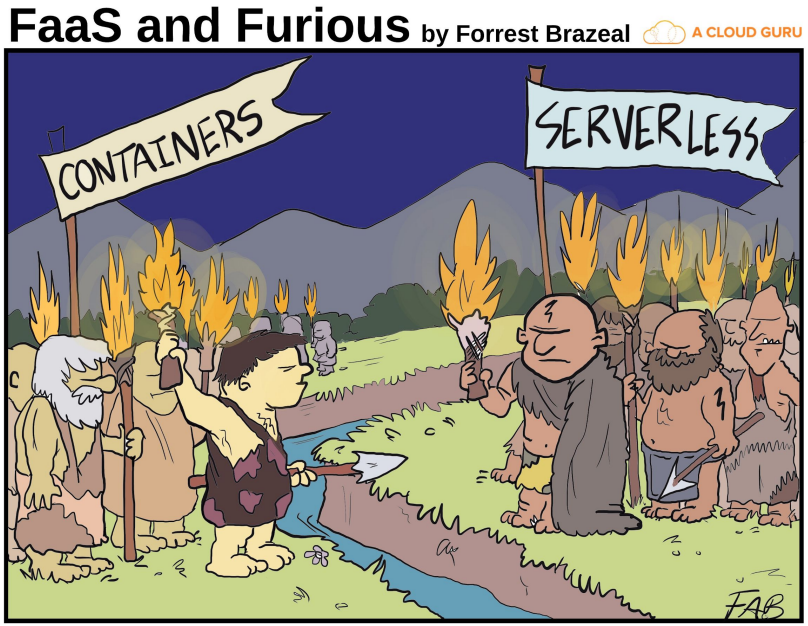
Aleksander Slominski

WoC *2019* : The Fifth International *Workshop on Container* Technologies and *Container* Clouds

Why containers **and** serverless matter for future?

- All marketing, there are servers in serverless after all!!???

“The serverless market is expected to reach $7.7B by 2021, up from $1.9B in 2016.



The two tribes regarded each other suspiciously in the glow of their brightly blazing production environments.

Public Cloud

The worldwide public cloud services market is projected to **grow 17.3 percent in 2019 to total $206.2 billion**, up from $175.8 billion in 2018, according to Gartner, Inc.

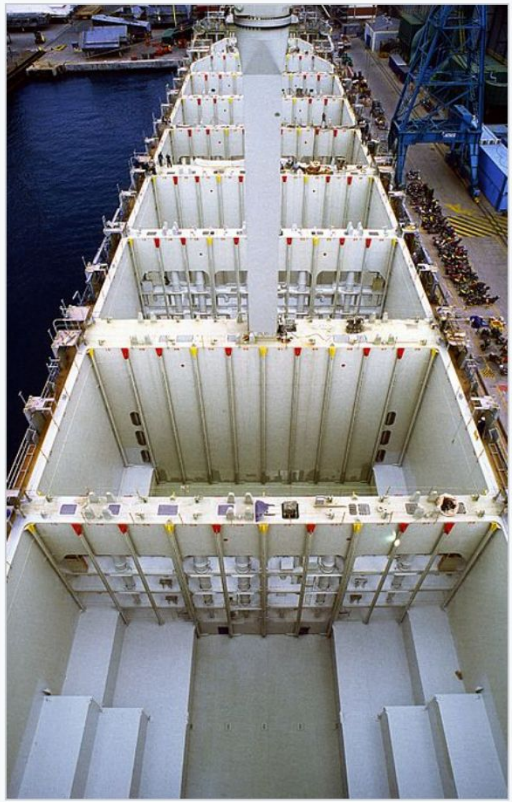
“At this point, **cloud adoption is mainstream**,” said Sid Nag, research vice president at Gartner. “The expectations of the outcomes associated with cloud investments therefore are also higher. Adoption of next-generation solutions are almost always ‘cloud-enhanced’ solutions, meaning they build on the strengths of a cloud platform to deliver digital business capabilities.”

Past

Containers Timeline

Docker Released in 2013

Why containers?



A view into the holds of a container ship. The vertical cell guides organize containers athwartships.

Container ships avoid the complex stevedoring of break-bulk shipping.

A 40-foot long (12.19 m) shipping container. Each of the eight corners has an essential twistlock fitting for hoisting, stacking, and securing

Making containers *stackable* made loading and transport on large ships feasible and efficient

https://en.wikipedia.org/wiki/Container_ship
https://en.wikipedia.org/wiki/Intermodal_container

(Short) History of shipping containers - success story

“**In the early 1950’s** McLean began his own “container concept”

Matson, on the West coast of the US also attempted a type of container concept, but failed sorely.

In the late 1960’s Sea-Land got some large boosts from the US Government and US Military.

McLean and Sea-Land were able to finally standardize the container concept in the early 70's.

“The History Of ISO Shipping Containers From Wooden Crates To Steel Boxes”
http://www.isbu-association.org/history-of-shipping-containers.htm

(Short) History of shipping containers - success story

For the first few years, from 1956 to 1965 virtually all use of the shipping containers were only by McLean on his own ships.

We must remember, the ISO shipping container as we know it today was developed in 1956, but was not re-designed and standardized Internationally until in 1972.

The standardization process which began by the ISO and IMO in 1967, was only pushed-forward at the urging of the US Government for their use by the US military for transport and housing overseas.

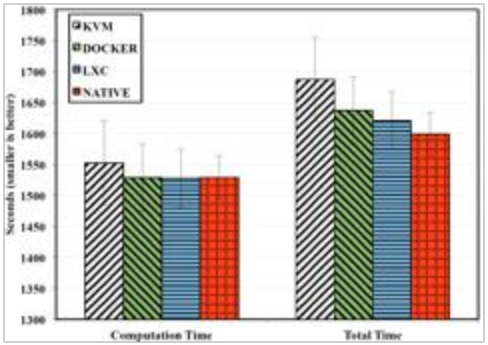
“The History Of ISO Shipping Containers From Wooden Crates To Steel Boxes”
http://www.isbu-association.org/history-of-shipping-containers.htm

Containers performance

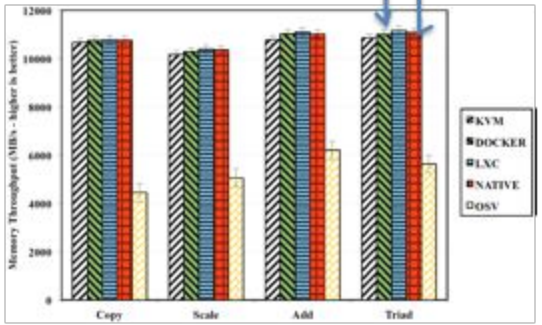
Docker

Native

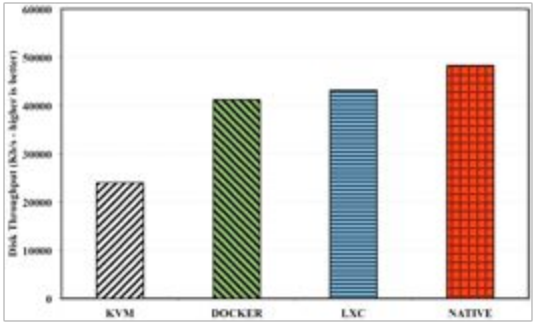
CPU (Pi Calculation)



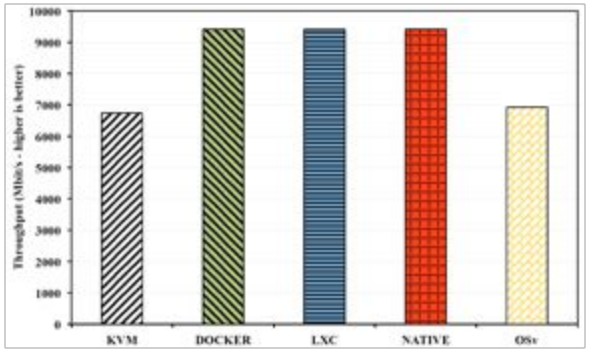
Memory



Disk I/O



TCP_STREAM Test

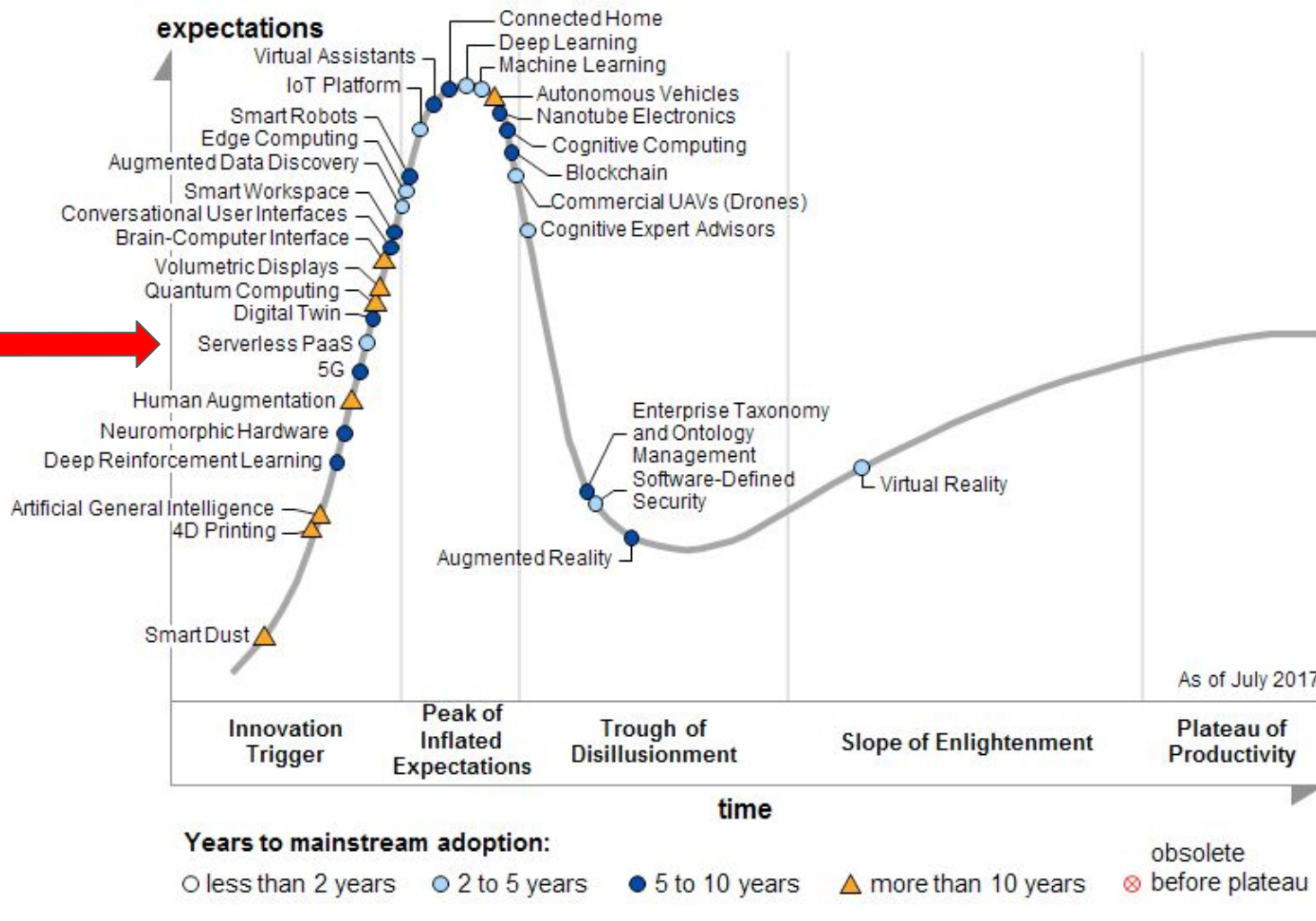


Source: Roberto Morabito, Jimmy Kjällman and Miika Komu. Hypervisors vs. Lightweight Virtualization: a Performance Comparison

Containers solved everything?

Gartner Hype Curve 2017

Serverless PaaS
2-5 years



Gartner Top 10 Trends Impacting Infrastructure

#1 “Serverless Computing

Serverless computing is an emerging software architecture pattern that promises to eliminate the need for infrastructure provisioning and management. I&O leaders need to adopt an application-centric approach to serverless computing, managing APIs and SLAs, rather than physical infrastructures. **“The phrase ‘serverless’ is somewhat of a misnomer,” noted Mr. Winser. “The truth is that servers still exist, but the service provider is responsible for all the underlying resources involved in provisioning and scaling a runtime environment, resulting in appealing agility.”**

Serverless does not replace containers or virtual machines, so it’s critical to learn how best and where to use the technology. “Developing support and management capabilities within I&O teams must be a focus as **more than 20 percent of global enterprises will have deployed serverless computing technologies by 2020, which is an increase from less than 5 percent today**,” added Mr. Winser.

#2 Artificial Intelligence

...

Quick Demo: How do you use serverless FaaS?

FaaS (Function as a Service):

Create function

Invoke function

Profit

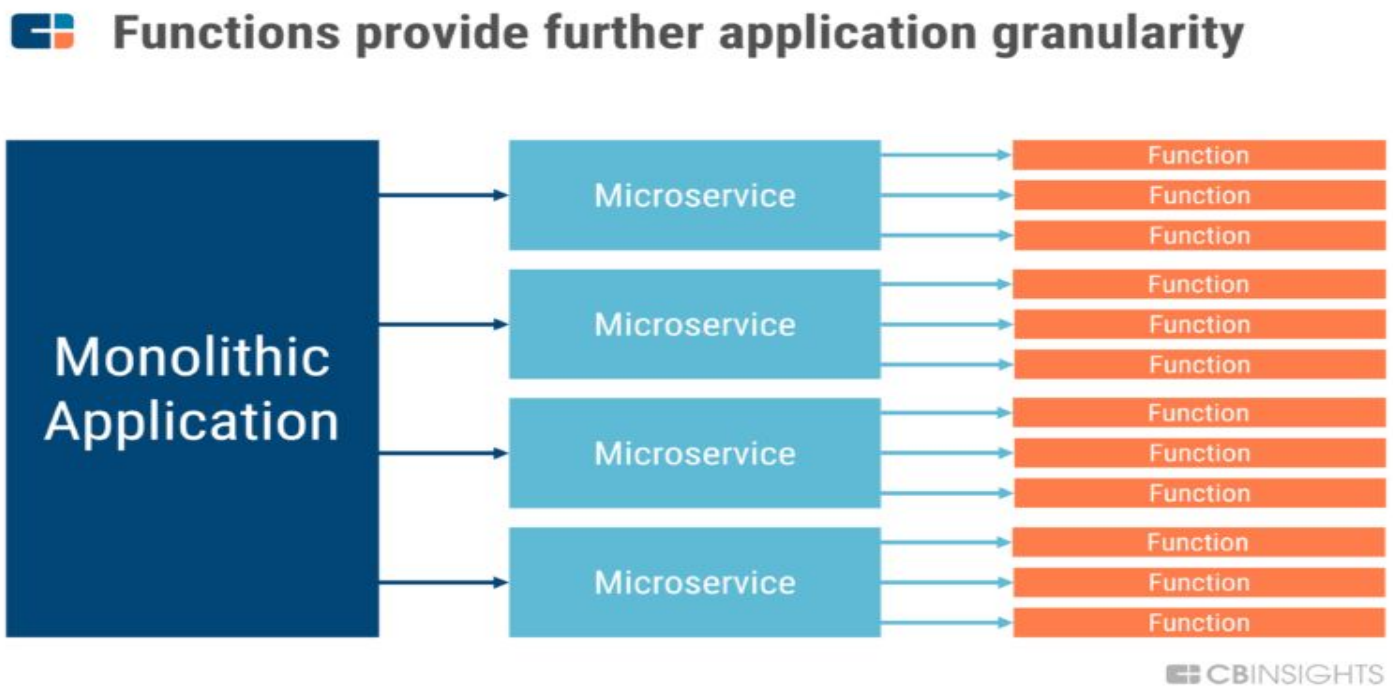
Quick Demo with IBM Cloud Functions

```
$ cat hellowosc5.js
function main() {
  return {payload: 'Hello WoC'};
}

$ ibmcloud fn action create hellowoc hellowoc.js --kind
nodejs:10
ok: created action hellowosc5

$ ibmcloud fn action invoke --result hellowoc
{
    "payload": "Hello WoC"
}
```

Why functions?



What happened during demo?

Where is server “hiding” in **server**less?

What happens when invoking cloud function?

$ ibmcloud fn action --verbose invoke --result hellowoc
REQUEST: **[POST]**
https://us-east.functions.cloud.ibm.com/api/v1/namespaces/_/actions/hellowoc?blocking=true&result=true
Req Headers {
"Authorization": ["Bearer eyJraWQiOiIyMD...zA"],
"Content-Type": ["application/json],
"User-Agent": ["CloudFunctions-Plugin/1.0 (2019-11-12T22:04:48+00:00) darwin amd64"],
"X-Namespace-Id": ["5a82404d-cd75-4cd9-8749-14c4d0d46435"]
}

Req Body
{}

RESPONSE:Got response with code 200
Resp Headers {
"Access-Control-Allow-Headers": ["Authorization, Origin, X-Requested-With, Content-Type, Accept, User-Agent"],
"Content-Length": ["23"],
"Content-Type": ["application/json"],
"Date": ["Tue, 10 Dec 2019 01:37:29 GMT"],
"X-Openwhisk-Activation-Id": ["52ea116e6b3c4805aa116e6b3cd80585"],
"X-Request-Id": ["deb5f8c60fbd0268ab07f7126f61c707"]}

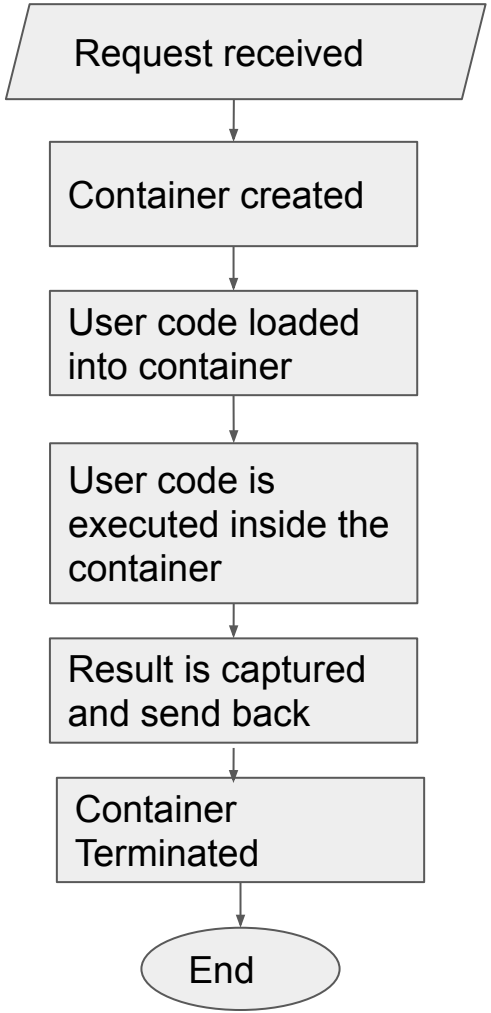
Response body size is 23 bytes
Response body received:
{"payload":"Hello WoC"}

What happened in demo?

Apache OpenWhisk is used by IBM Cloud Functions

OpenWhisk is Open Source!

When there is first invocation the user code is injected into a container and then executed.

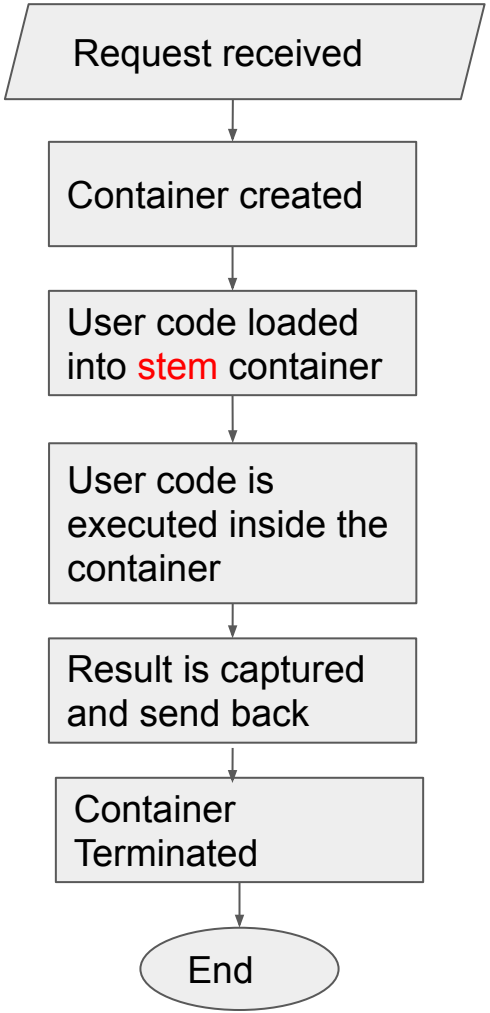


Optimization: stem containers

Keep a set of generic “stem” containers ready

They are ready to run code using popular runtimes with typical libraries for example nodejs

```
ibmcloud fn action create hellowoc
hellowoc.js --kind nodejs:10
```

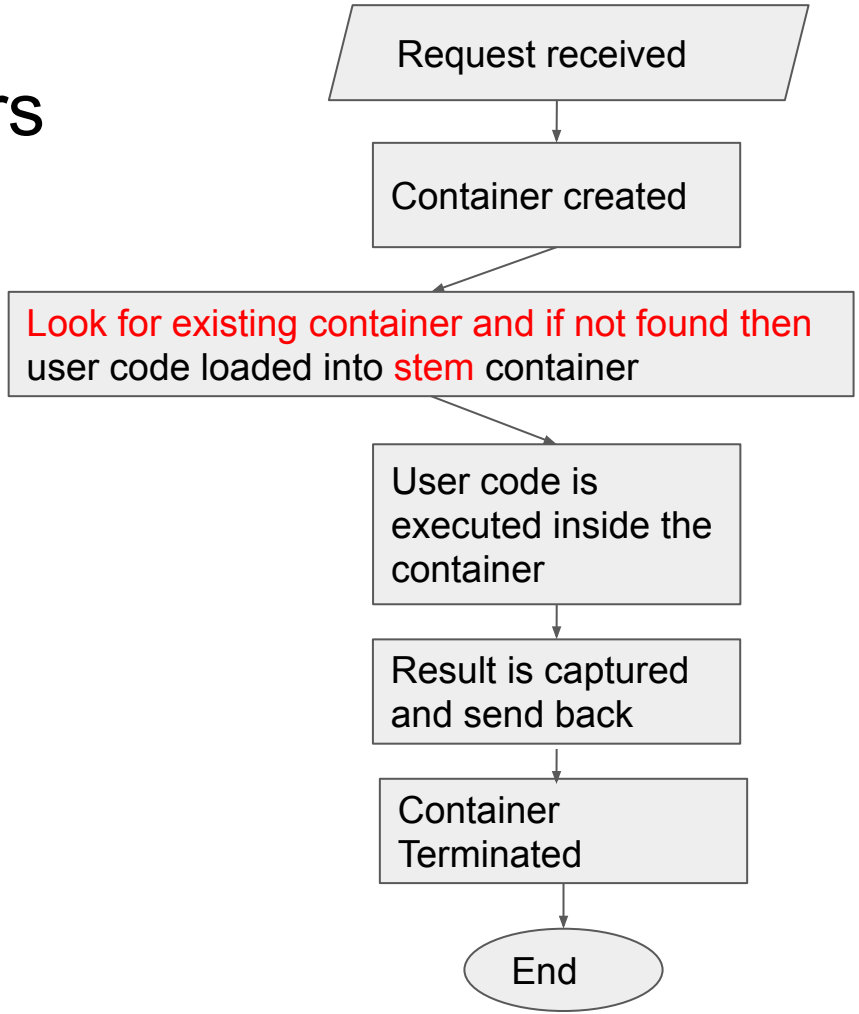


Optimization: warm containers

Do not terminate container

Wait for some time to see if the same code is to be executed again

Re-use (“warm”) container



Serverless Timeline

2014-11 AWS Lambda

2015-05 JAWS aka Serverless Framework

2016-02 IBM OpenWhisk, Google Cloud Functions

2016-03 Azure Functions

2017-02 Apache OpenWhisk

2017-05 Kubeless

2017-07 OpenFaas

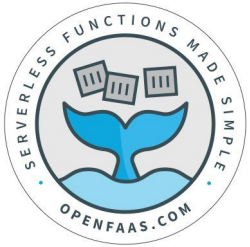
2018-07 Knative

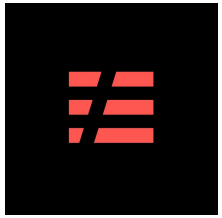
2019-04 Google Cloud Run

Google Cloud Functions

IBM Cloud Functions

Apache OpenWhisk





serverless.com

Why Serverless (FaaS)?

What developers want to do?

Write code that matters (direct business value)

Avoid undifferentiated heavy lifting

(leave it to experts to manage operations)

Serverless and Function-as-a-Service (FaaS)

Distinction between serverless computing and providing functions as a unit of computation (or container)?

CNCF (Cloud Native Computing Foundation) Serverless Definition

“Serverless computing refers to the concept of building and running applications that do not require server management. It describes a finer-grained deployment model where **applications**, **bundled as one or more functions**, are uploaded to a platform and then executed, scaled, and billed in response to the exact demand needed at the moment.

Serverless computing does not mean that we no longer use servers to host and run code; nor does it mean that operations engineers are no longer required. Rather, it refers to the idea that consumers of serverless computing no longer need to spend time and resources on server provisioning, maintenance, updates, scaling, and capacity planning. Instead, all of these tasks and capabilities are handled by a serverless platform and are completely abstracted away from the developers and IT/operations teams. As a result, developers focus on writing their applications’ business logic. Operations engineers are able to elevate their focus to more business critical tasks

Other definitions

“Serverless is eating the stack and people are freaking out — as they should be” by Forrest Brazeal

“The way I describe it is: **functions as a service are cloud glue.** So if I’m building a model airplane, well, the glue is a necessary part of that process, but it’s not the important part. Nobody looks at your model airplane and says: “Wow, that’s amazing glue you have there.” **It’s all about how you craft something that works with all these parts together**, and FaaS enables that.workload constraints.

Other definitions

What *is* Serverless? The “2020” edition by Paul Johnston

“I wanted to limit my exposure to managing servers as much as possible.

Notice that I’m not worried about managing servers. It’s my exposure to the risk that matters here.

I literally didn’t want to have to think about ssh-ing into any instances, and upgrading them regularly, monitoring them for hack attempts, maintaining any libraries on a long running system etc. unless I had no other choice due to workload constraints.”

Amazon AWS Definition of Serverless

“Serverless is the native architecture of the cloud that enables you to shift more of your operational responsibilities to AWS, ...

* No server management - There is no need to provision or maintain any servers. There is no software or runtime to install, maintain, or administer.

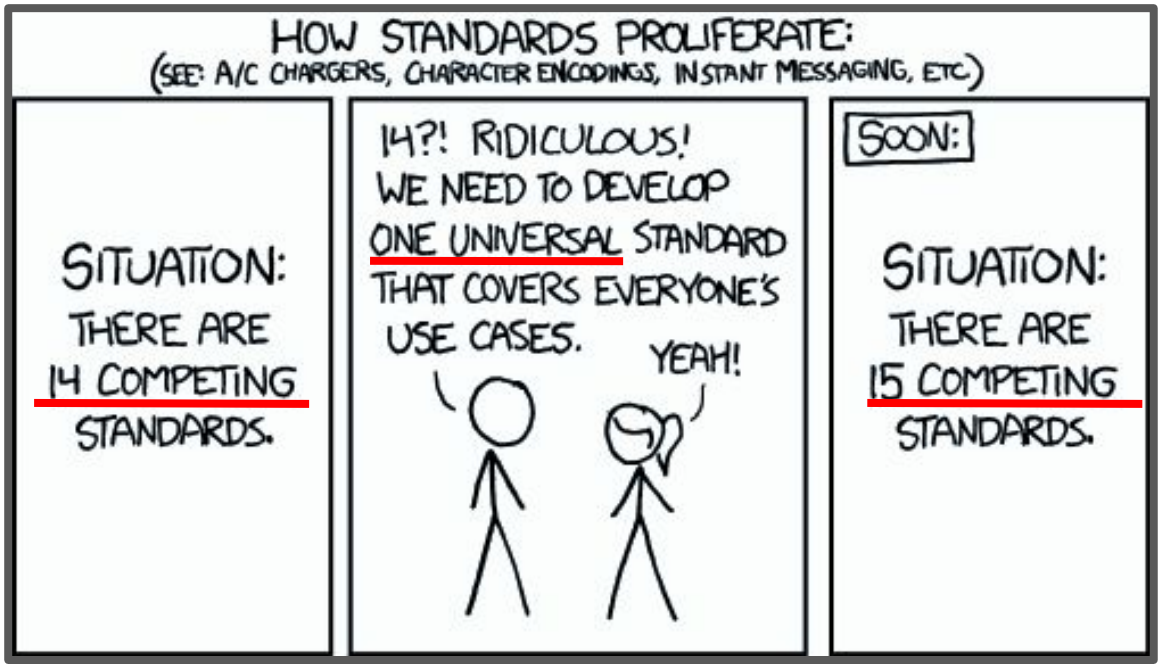
* Flexible scaling - Your application can be scaled automatically or by adjusting its capacity through toggling the units of consumption (e.g. throughput, memory) rather than units of individual servers.

* **Pay for value - Pay for consistent throughput or execution duration rather than by server unit.**

* Automated high availability - Serverless provides built-in availability and fault tolerance. You don't need to architect for these capabilities since the services running the application provide them by default.”

Definition of Serverless?

Let create the best definition for serverless?

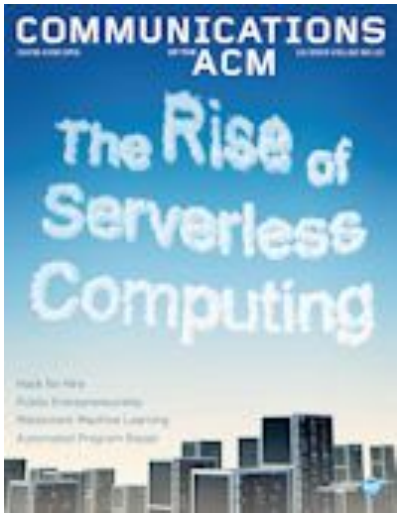


https://xkcd.com/927/

Our definition - YASD (Yet Another Serverless Definition)

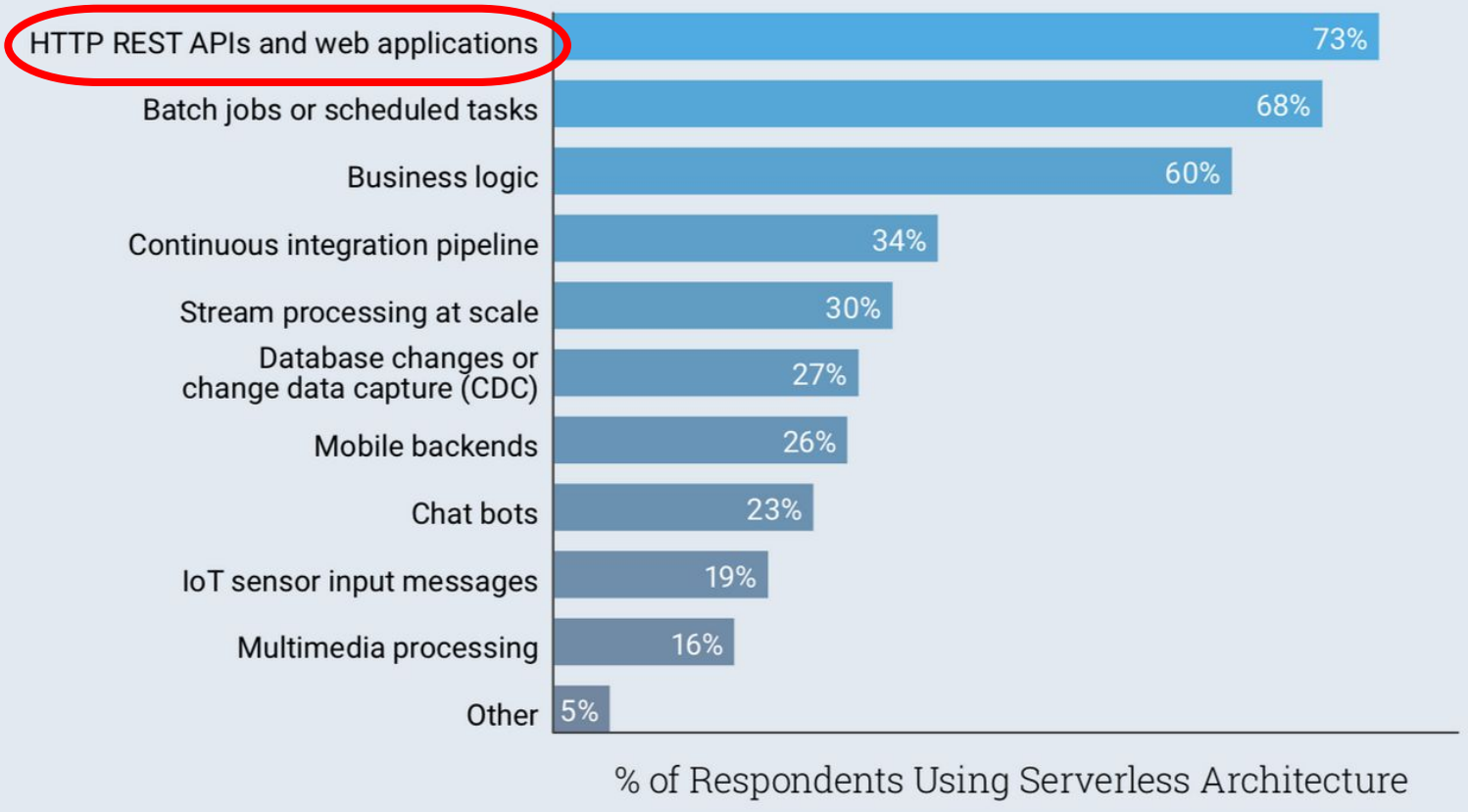
“Serverless computing is a platform that **hides server usage from developers** and runs code on-demand, automatically scaled, and billed only for the time the code is running.”

“Function-as-a-Service is a serverless computing platform where the **unit of computation is a function** that is executed in response to triggers such as events or HTTP requests.”



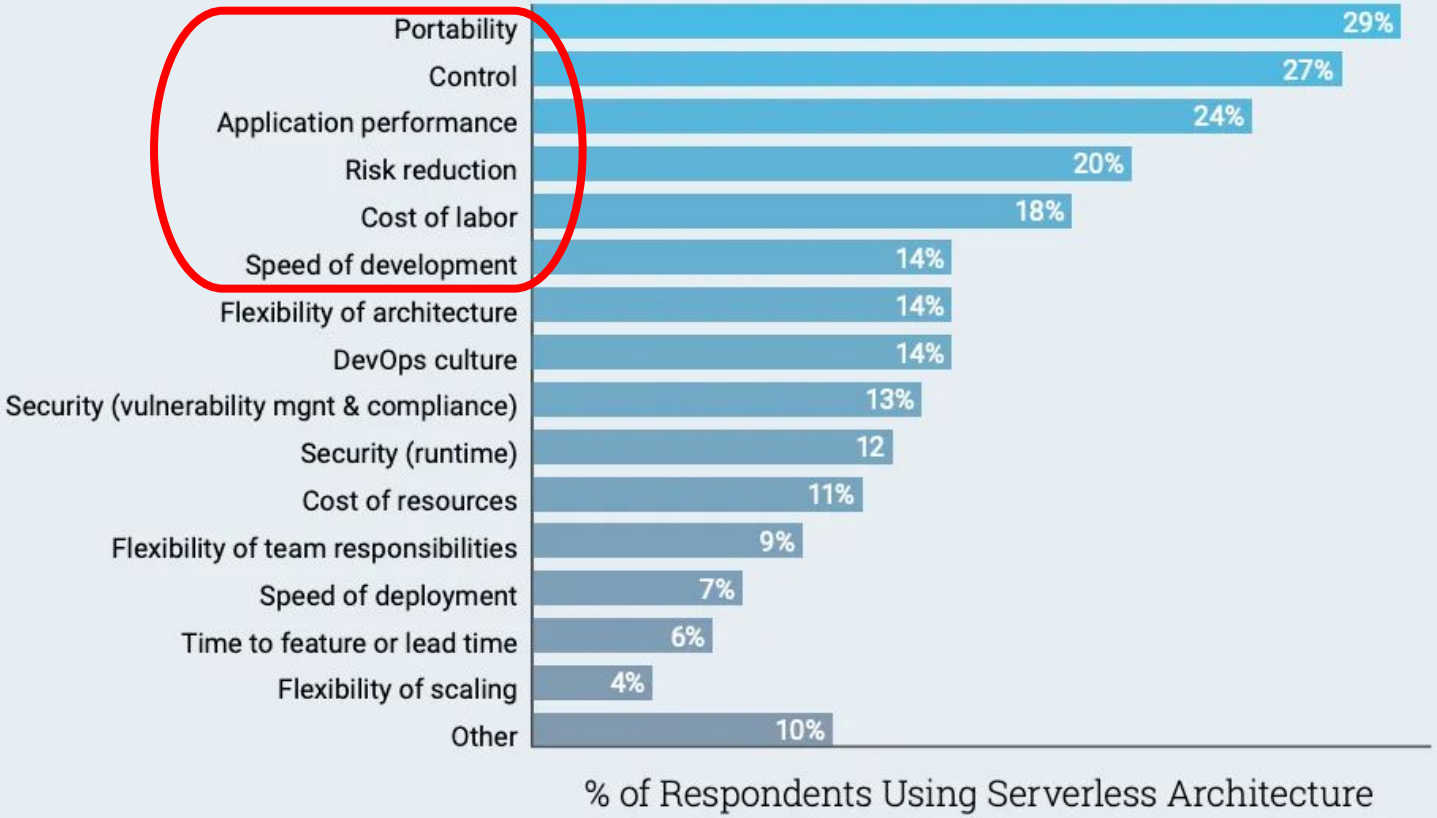
Communications of the ACM, December 2019

Technical Use Cases for Serverless



Source: The New Stack Serverless Survey 2018.
Q. What types of technical use cases are serverless technologies being used for? (check all that apply) n=278.

Areas Where Serverless Has Fallen Short



Source: The New Stack Serverless Survey 2018.
Q. What are the top three areas in which serverless architecture has fallen short of expectations? n=251.

Challenges and Limitations

Lack of standards and vendor lock-in

Programming models and tooling

Going beyond stateless short-lived FaaS: statefulness, composability, observability, ...

Future: containers + serverless?

Knative: serverless and containers and Kubernetes?

“Knative (pronounced kay-nay-tiv) extends Kubernetes to provide a set of middleware components that are essential to build modern, source-centric, and container-based applications that can run anywhere: on premises, in the cloud, or even in a third-party data center.”
https://knative.dev/docs/

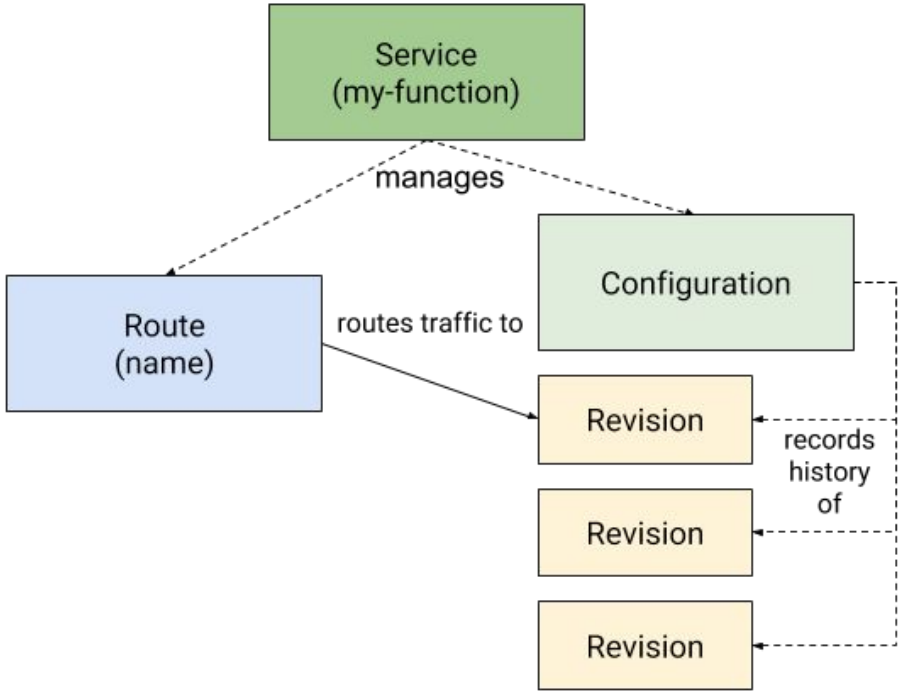
Serverless developer experience with containers?

“Knative Serving builds o**n Kubernetes and Istio to support deploying and serving of serverless applications and functions. Serving is easy to get started with and scales to support advanced scenarios.**

The Knative Serving project provides middleware primitives that enable:

- Rapid deployment of serverless containers
- Automatic scaling up and down to zero
- Routing and network programming for Istio components
- Point-in-time snapshots of deployed code and configurations”

https://knative.dev/docs/serving/



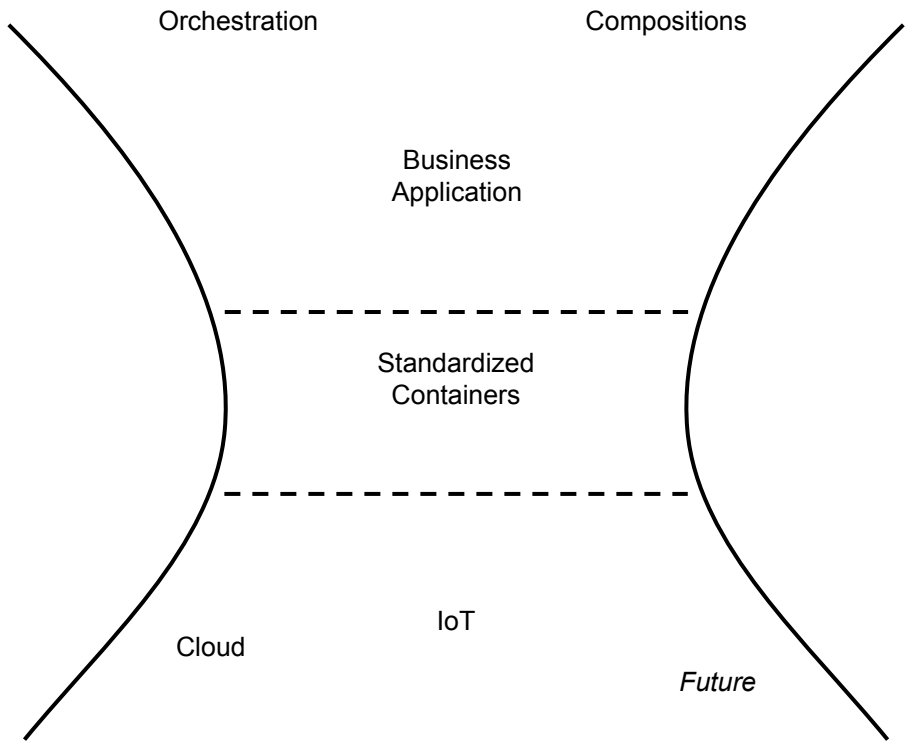
Future of computing

Evolution of computation

Year	Computing Paradigm	Expenses	Time to solution	Cost of compute with 1 GB of memory per hour
1960-2000	Mainframes, data centers	Upfront capital investment	Months to Years	$100K to $1000
2000s	HPC Cluster and Grid Computing, data centers, VMs	Capital investment, reusing idle machines	Days to Weeks	$10-50
2010s	Cloud and Serverless Computing, VMs and Containers	Pay-as-you-go, sometimes paying even if not using compute resources (VMs)	Minutes to Hours	$0.01 - 0.06
Future	**Standardized containers?**	Pay only for what is used	Seconds to Minutes	<$0.01

"Comparing Cloud Instance Pricing: AWS vs Azure vs Google vs IBM", November 18, 2017, https://www.rightscale.com/blog/cloud-cost-analysis/comparing-cloud-instance-pricing-aws-vs-azure-vs-google-vs-ibm

Universal Standardized Computing Container like Electricity?



Who builds and operates electric power plants?

Around the world, there are **about 62,500 power plants operating** today.
https://www.washingtonpost.com/news/wonk/wp/2012/12/08/all-of-the-worlds-power-plants-in-one-handy-map/?noredirect=on&utm_term=.af263d4f8a01

“In 2017, it was estimated that the number of data centers globally had fallen to 8.4 million.”
Predicted 7.2M in 2021 (**about 500 “hyperscale” datacenters**)
https://www.statista.com/statistics/500458/worldwide-datacenter-and-it-sites/

Inevitability of progress

“**People are mistaken when they think that technology just automatically improves**. It does not automatically improve. **It only improves if a lot of people work very hard to make it better and I actually, I think, by itself, degrade, actually.** You look at ancient civilizations like ancient Egypt and they were able to make the pyramids and they forgot how to do that. And the Romans, they built these incredible aqueducts, they forgot how to do it.

- Elon Musk

Today’s Computing becomes “Legacy” Computing in Future?

Legacy computing may be containerized, perhaps as virtual machines (VMs) inside containers, as from an economical point of view, **it may be cheaper to keep legacy code running than pay for developers to re-design and re-write code using new computing approaches**.

My predictions (may or may not become true …)

In the future computing works like an electric utility?

Pay-as-you-go only way to go for computation?

Like electricity today, it is just there, and we no longer think about it unless it is not working?

What will we call this future computing infrastructure?

Computing Fabric? Computing utility?

Perhaps simply computing?

Popular research topics for serverless

Use cases

Performance evaluations

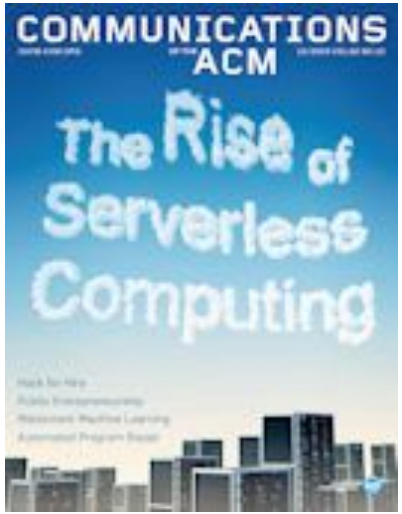
Evaluating architecture

For example https://www.serverlesscomputing.org/wosc5/#papers

Q&A

See previous serverless workshops and join mailing list for future CFP: https://www.serverlesscomputing.org/

Check our article:



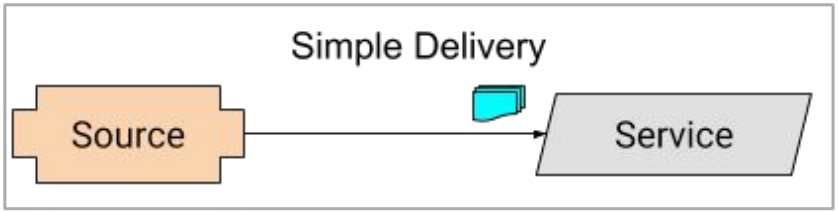
Communications of the ACM,
December 2019

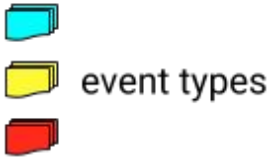
END

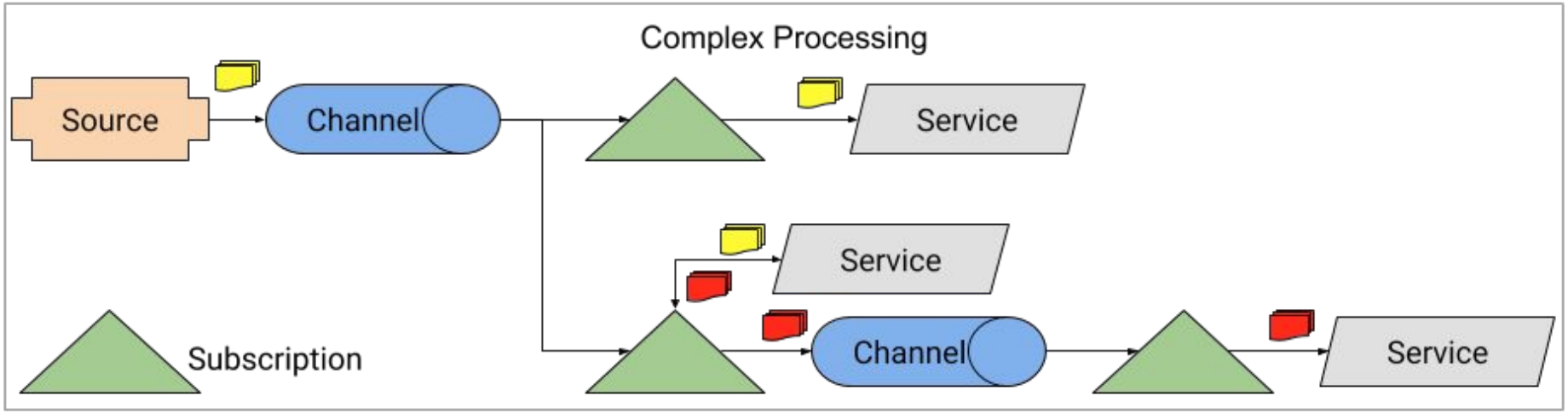
Can eventing be serverless?

Use event sources and compose them with serverless approach

Knative Eventing for eventing sources and compositions with serverless (containers) as functions



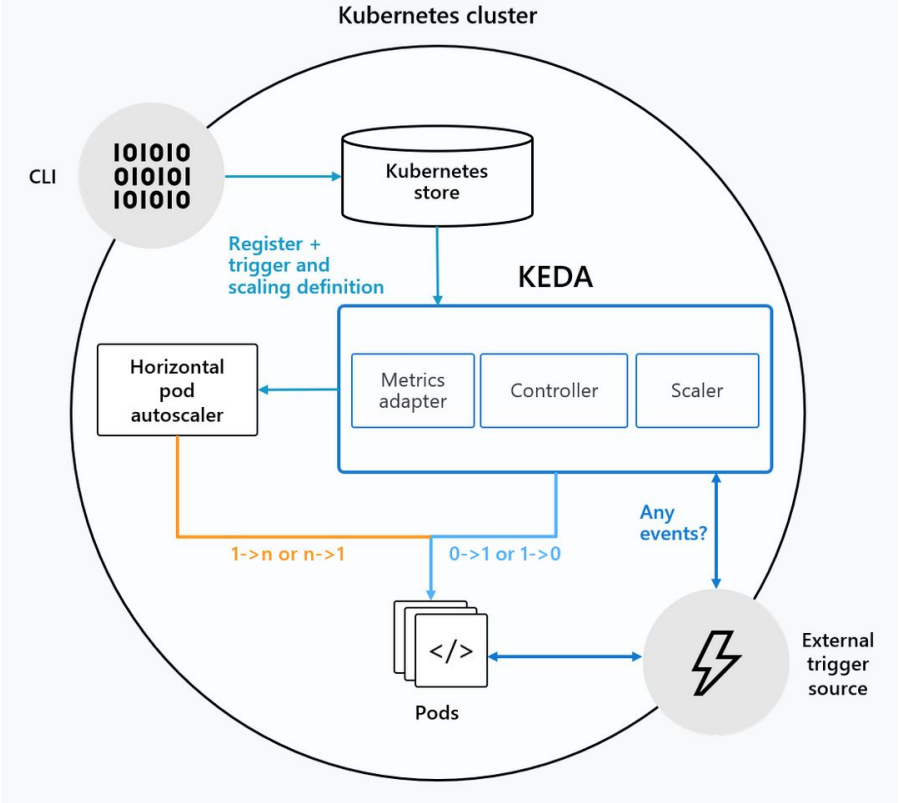




https://knative.dev/docs/eventing/

Scaling eventing

Kubernetes-based Event Driven Autoscaler can “drive the scaling of any container in Kubernetes based on the number of events needing to be processed”



https://keda.sh/concepts/overview/#how-keda-works

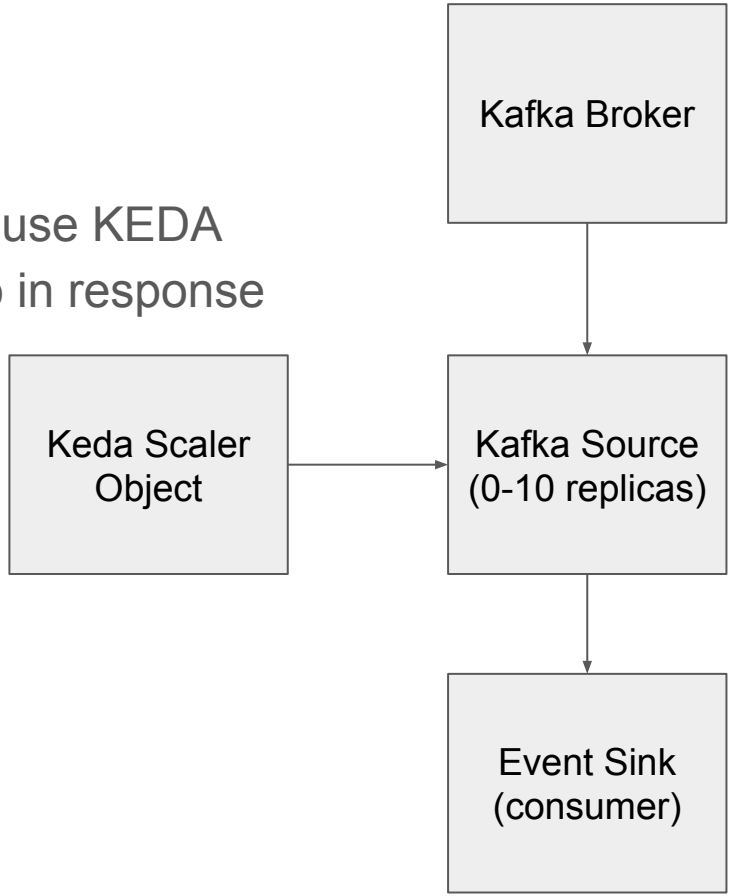
DEMO

Use Kafka source in Knative eventing and use KEDA to scale it from zero to 10 and back to zero in response to events sent

Live and if not working recording

https://youtu.be/7Fb3NThpMmk

https://vimeo.com/376958832



Back to (Near) Future

What is beyond FaaS in Serverless?

Can serverless “mindset” be applied to other areas?

“Serverless is a way to focus on business value.”

Can you have serverless computing paradigm in your own (private) cloud?

Rise of Containers

- Open Virtualization Format
  https://www.dmtf.org/standards/ovf
- The future of Linux Containers, **2013**
  https://www.youtube.com/watch?v=wW9CAH9nSLs
  - At PyCon Solomon Hykes shows docker to the public **for the first time**.
- Open Container Initiative (OCI) https://www.opencontainers.org/
- Firecracker – Lightweight Virtualization for Serverless Computing, **2018**
  https://github.com/firecracker-microvm/
- Future …?

(Short) History of shipping containers - success story

In 1956, loose cargo cost **$5.86 per ton to load**. Using an ISO shipping container, the cargo cost was **reduced to only 16 cents per ton**.

5.86 / 0.16 = 36x

There were many who had similar concepts previously but McLean was simply the guy, who with the push of the US military, really made the "standardized container" concept spread globally.

“The History Of ISO Shipping Containers From Wooden Crates To Steel Boxes”
http://www.isbu-association.org/history-of-shipping-containers.htm

AWS Definition of Serverless (2018-07-28)

Serverless computing allows you to build and run applications and services without thinking about servers. Serverless applications don't require you to provision, scale, and manage any servers. You can build them for nearly any type of application or backend service, and everything required to run and scale your application with high availability is handled for you.

* NO SERVER MANAGEMENT - There is no need to provision or maintain any servers. There is no software or runtime to install, maintain, or administer.

* FLEXIBLE SCALING - Your application can be scaled automatically or by adjusting its capacity through toggling the units of consumption (e.g. throughput, memory) rather than units of individual servers.

* AUTOMATED HIGH AVAILABILITY- Serverless applications have built-in availability and fault tolerance. You don't need to architect for these capabilities since the services running the application provide them by default.

https://web.archive.org/web/20180728003356/https://aws.amazon.com/serverless/