



What is serverless computing?

Frank Pientka, Dortmund

Who is Frank Pientka?



Dipl.-Informatiker (TH **Karlsruhe**)

Principal Software Architect at **Dortmund**

iSAQB founding member



heise.de/developer/Federlesen column

over **20 years** IT experience
lots of articles and talks



Materna GmbH founded 1980

Agenda.

Why serverless?

Characteristics of serverless applications?

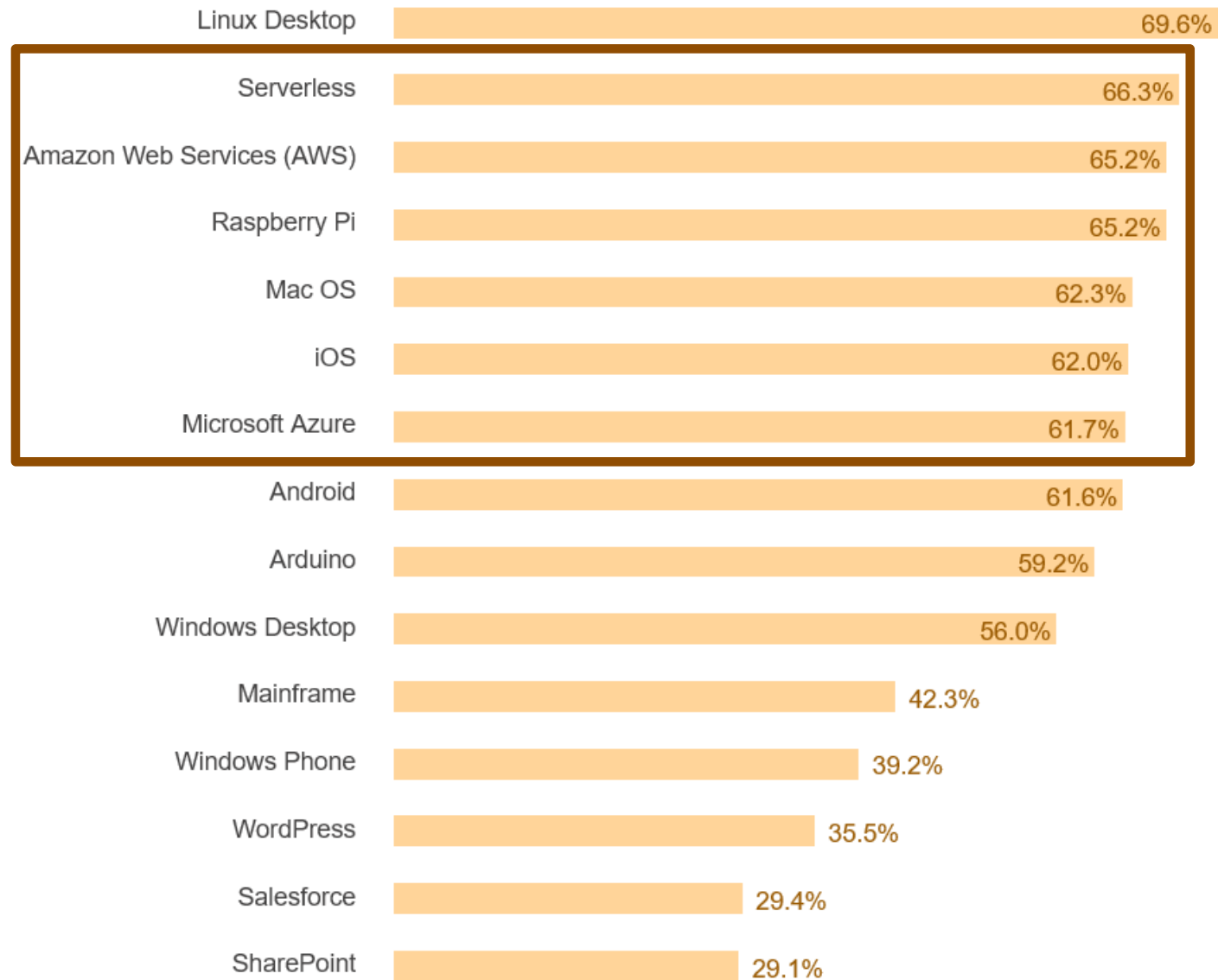
Overview serverless offers

AWS Lambda, IBM OpenWhisk, Azure Functions

The serverless framework

Resume

Stack Overflow 2017 survey: Most Loved, Dreaded, and Wanted Platforms

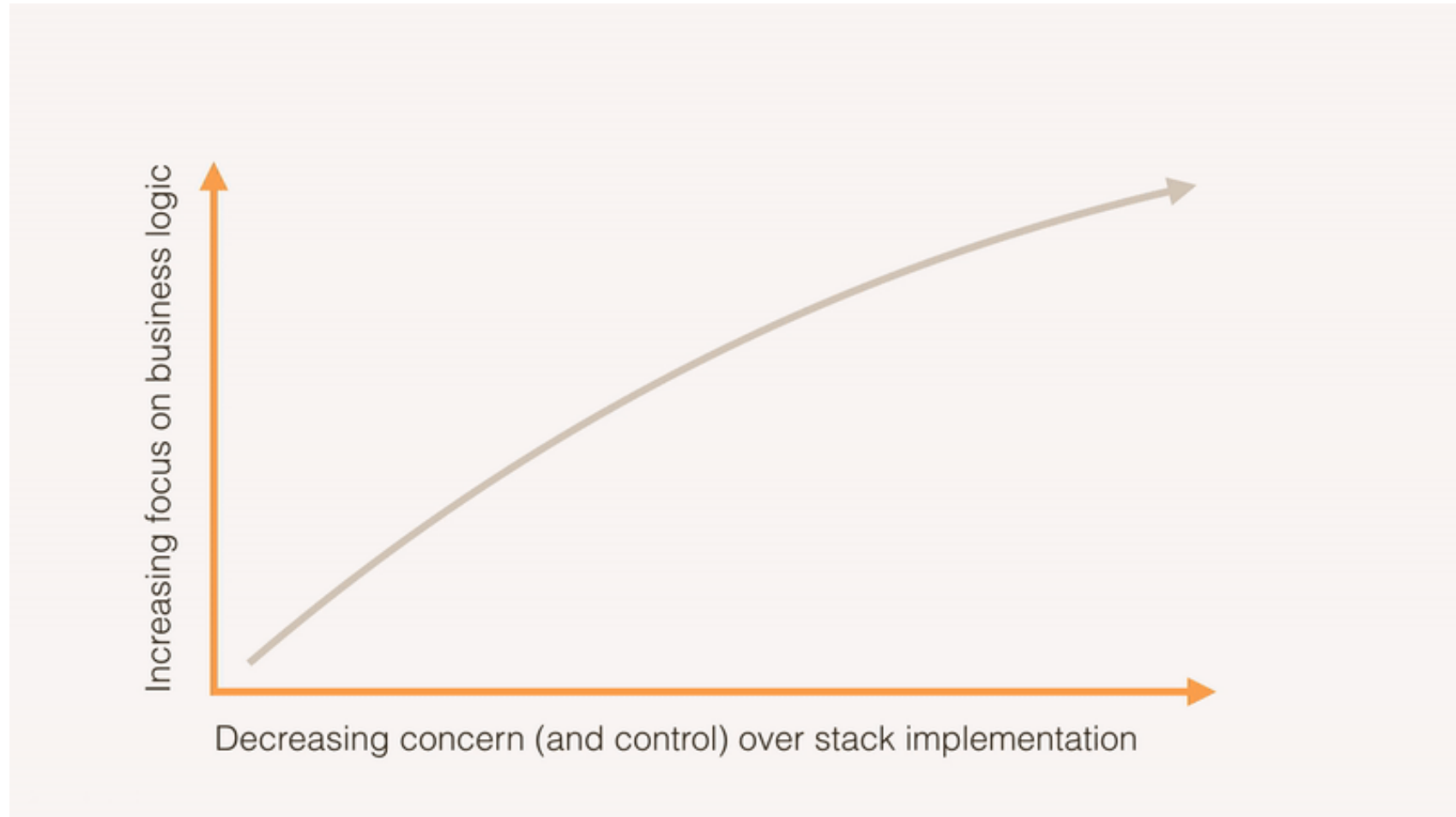


Your Cloud journey
without servers:
serverless the next
evolution step

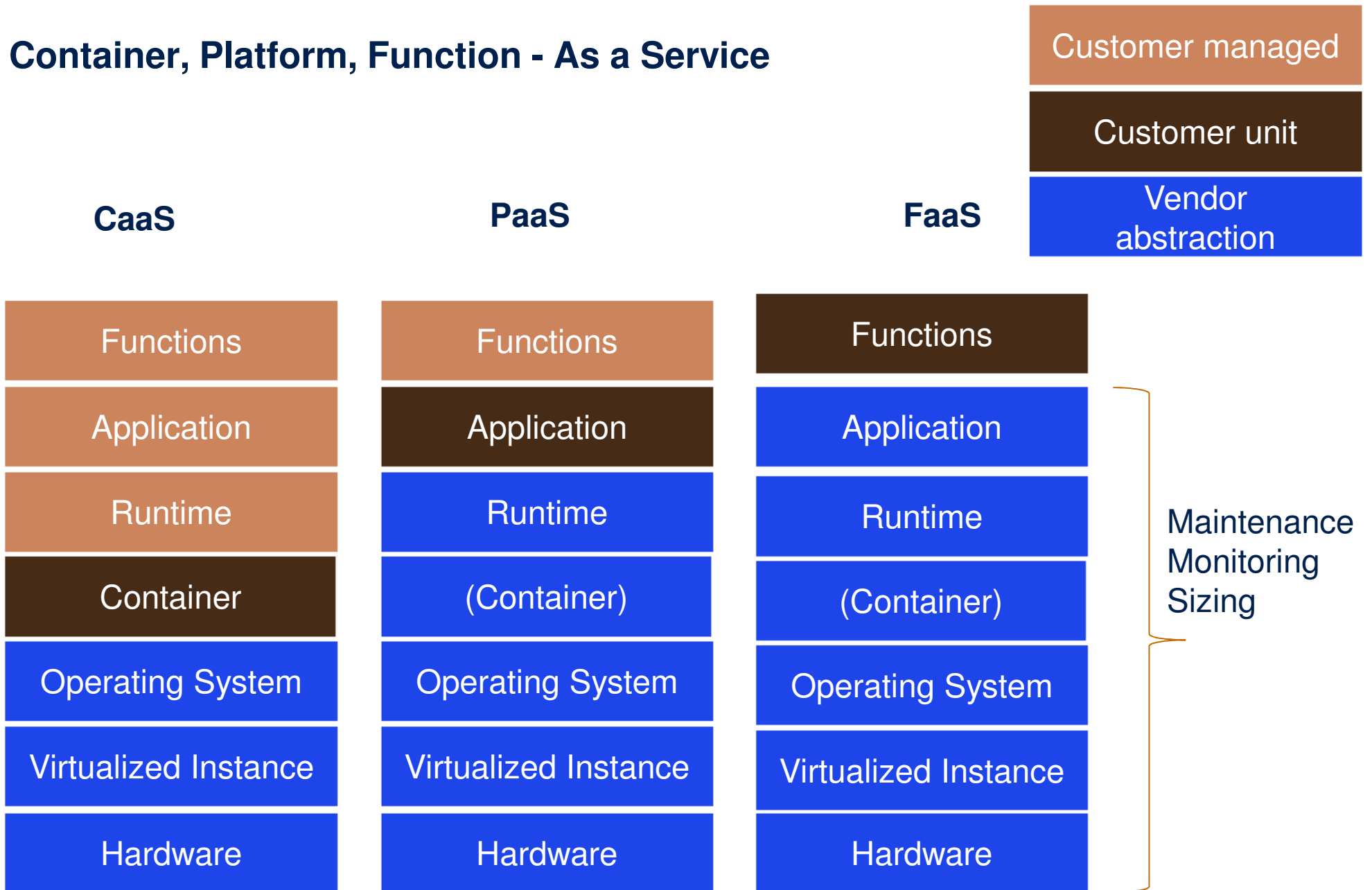
```
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
elif operation == "MIRROR_Z":
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

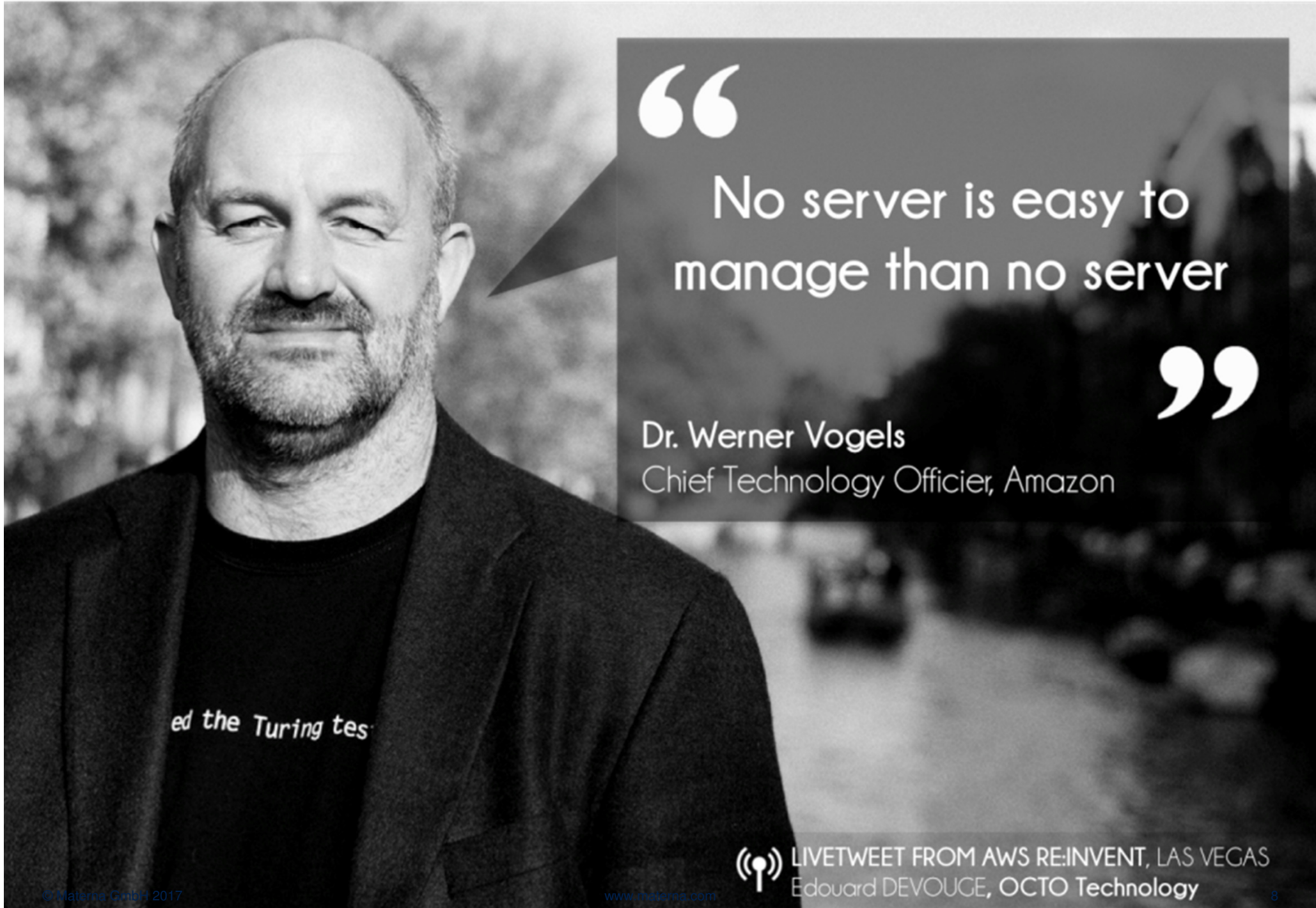
#selection at the end -add back the deselected mirror modifier object
mirror_ob.select= 1
modifier_ob.select=1
bpy.context.scene.objects.active = modifier_ob
print("Selected" + str(modifier_ob)) # modifier ob is the active ob
mirror_ob.select = 0
name = bpy.context.selected_objects[0]
modifier_ob.objects[name].select = 1
```

Serverless developers focus more on code, less on infrastructure



Container, Platform, Function - As a Service





“

No server is easy to manage than no server

”

Dr. Werner Vogels
Chief Technology Officer, Amazon

ed the Turing tes



LIVETWEET FROM AWS RE:INVENT, LAS VEGAS
Edouard DEVOUGE, OCTO Technology

TOOLS

Technology Radar April 2017

ADOPT

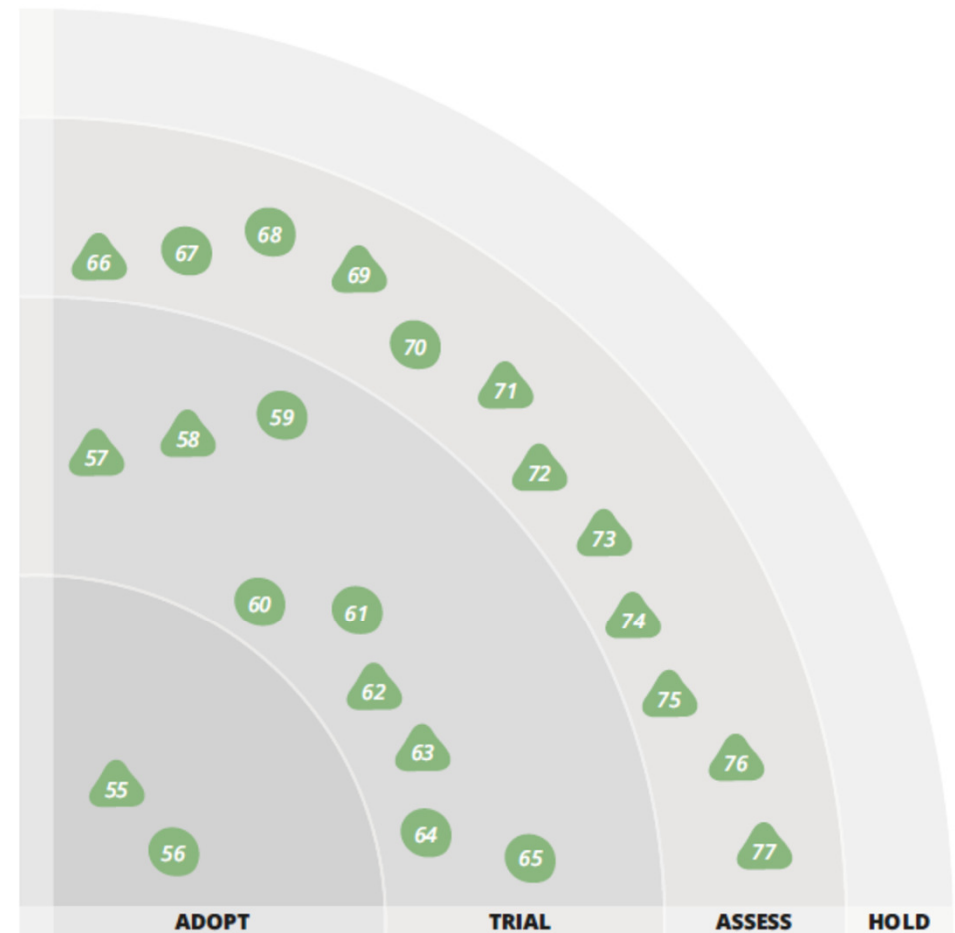
- 55. fastlane
- 56. Grafana

TRIAL

- 57. Airflow *NEW*
- 58. Cake and Fake *NEW*
- 59. Galen
- 60. HashiCorp Vault
- 61. Pa11y
- 62. Scikit-learn
- 63. Serverless Framework *NEW*
- 64. Talisman
- 65. Terraform

ASSESS

- 66. Amazon Rekognition *NEW*
- 67. Android-x86
- 68. Bottled Water
- 69. Claudia *NEW*
- 70. Clojure.spec
- 71. InSpec *NEW*
- 72. Molecule *NEW*
- 73. Spacemacs *NEW*
- 74. spaCy *NEW*
- 75. Spinnaker *NEW*
- 76. Testinfra *NEW*
- 77. Yarn *NEW*



Technology Radar 2017

ThoughtWorks®


A [serverless architecture](#) approach replaces long-running virtual machines with ephemeral compute power that comes into existence on request and disappears immediately after use. Our teams like the serverless approach; it's working well for us and we consider it a valid architectural choice. Note that serverless doesn't have to be an all-or-nothing approach: some of our teams have deployed a new chunk of their systems using serverless while sticking to a traditional architectural approach for other pieces. Although [AWS Lambda](#) is almost synonymous with serverless, the other major cloud providers all have similar offerings, and we also recommend assessing niche players such as [webtask](#).

History for Serverless architecture

MAR
2017

TRIAL ?

A [serverless architecture](#) approach replaces long-running virtual machines with ephemeral compute power that comes into existence on request and disappears immediately after use. Our teams like the serverless approach; it's working well for us and we consider it a valid architectural choice. Note that serverless doesn't have to be an all-or-nothing approach: some of our teams have deployed a new chunk of their systems using serverless while sticking to a traditional architectural approach for other pieces. Although [AWS Lambda](#) is almost synonymous with serverless, the other major cloud providers all have similar offerings, and we also recommend assessing niche players such as [webtask](#).



*Serverless architectures refer to applications that significantly depend on third-party services (known as Backend as a Service or "**BaaS**") or on custom code that's run in ephemeral containers (Function as a Service or "**FaaS**").*

*By using these ideas, and by moving much behavior to the front end, such architectures remove the need for the traditional **'always on' server** system sitting behind an application.*

*Depending on the circumstances, such systems can significantly **reduce operational cost and complexity** at a **cost of vendor dependencies** and (at the moment) immaturity of supporting services.*

<https://martinfowler.com/articles/serverless.html> August 2016 Mike Roberts

The Serverless Compute Manifesto re:Invent 2016

- **Functions** are the **unit of deployment** and scaling
- **No machines**, VMs, or containers visible in the programming model
- **Permanent storage** lives elsewhere
- **Scales per request**; Users cannot over- or under-provision capacity
- **Never pay for idle** (no cold servers/containers or their costs)
- **Implicitly fault-tolerant** because functions can run anywhere
- BYOC - **Bring Your Own Code**
- **Metrics** and **logging** are a universal right

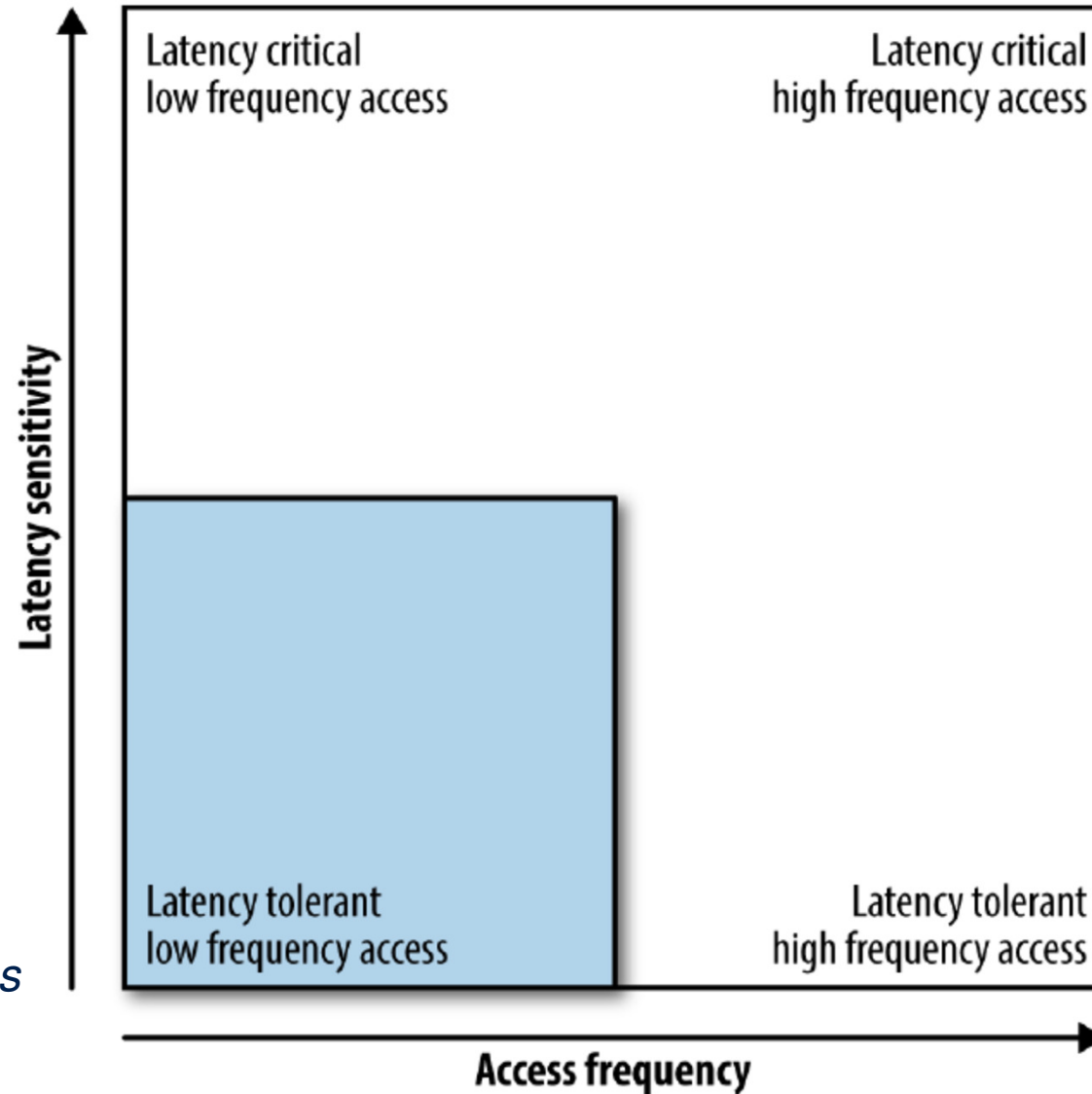
Serverless architectures abstract many of the operations specific cloud native 12 Factors

I	<u>Codebase</u>	Handled by developer (Manage versioning of functions on their own)
II	<u>Dependencies</u>	Handled by developer, facilitated by serverless platform (Runtimes and packages)
III	<u>Configuration</u>	Handled by platform (Environment variables or injected event parameters)
IV	<u>Backing services</u>	Handled by platform (Connection information injected as event parameters)
V	<u>Build, release, run</u>	Handled by platform (Deployed resources are immutable and internally versioned)
VI	<u>Processes</u>	Handled by platform (Single stateless containers often used)
VII	<u>Port binding</u>	Handled by platform (Actions or functions are automatically discovered)
VIII	<u>Concurrency</u>	Handled by platform (Process model is hidden and scales in response to demand)
IX	<u>Disposability</u>	Handled by platform (Lifecycle is hidden from the user, fast startup and elastic scale is prioritized)
X	<u>Dev/prod parity</u>	Handled by developer (The developer is the deployer. Scope of what differs is narrower)
XI	<u>Logs</u>	Handled by platform (Developer writes to console.log, platform handles log streaming)
XII	<u>Admin processes</u>	Handled by developer (No distinction between one off processes and long running)



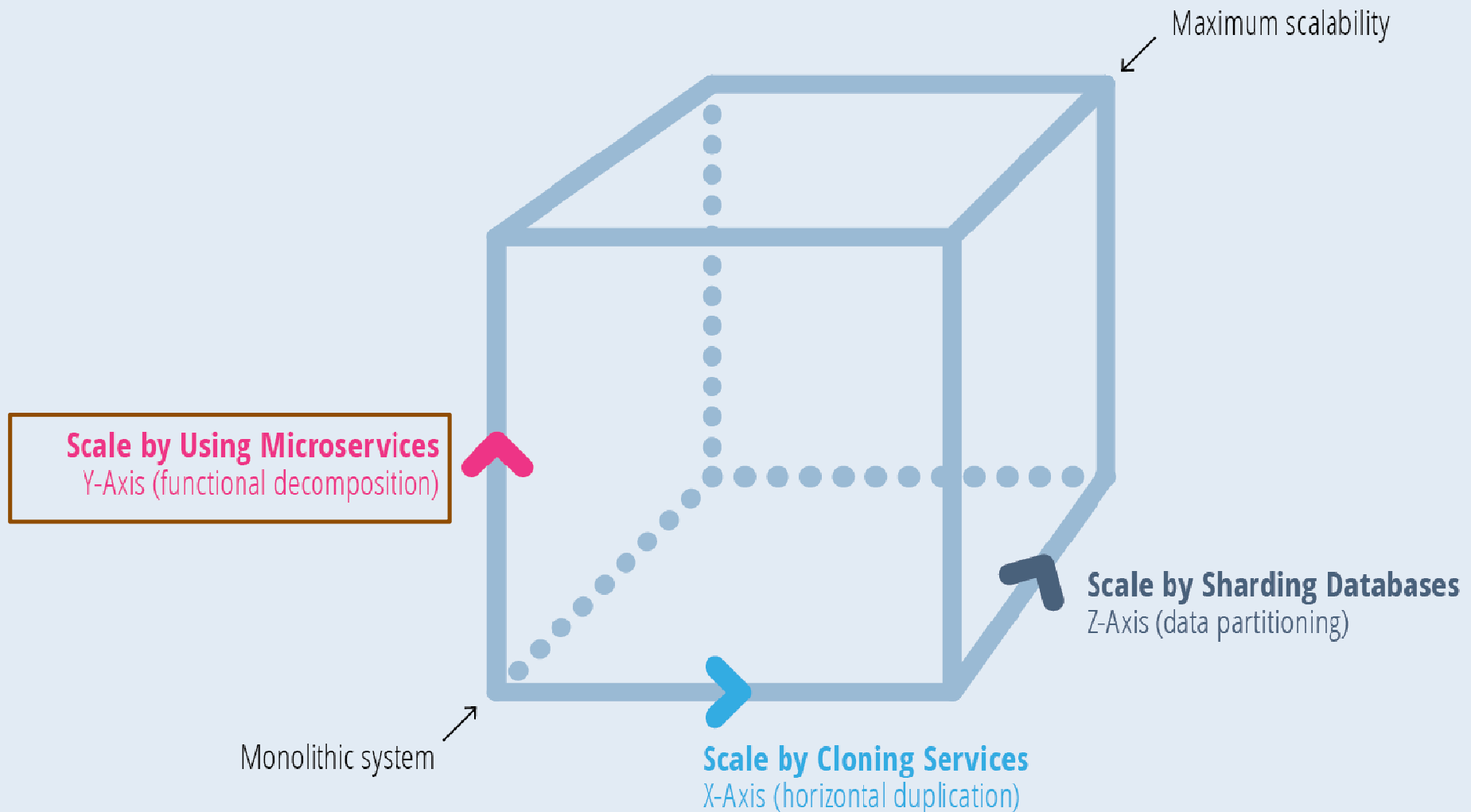
No Operations?
More Automation!

Where serverless fits in? Latency sensitivity vs access frequency



Serverless Ops
Michael Hausenblas
2017 O'Reilly

The Scale Cube and Microservices: 3 Dimensions to Scaling



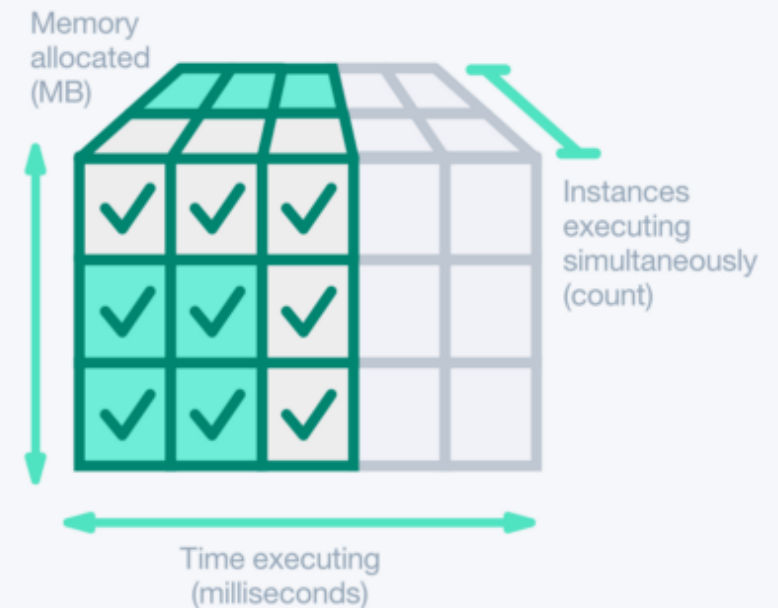
Source: The New Stack. Based on the "The Art of Scalability," by Martin Abbott & Michael Fisher.

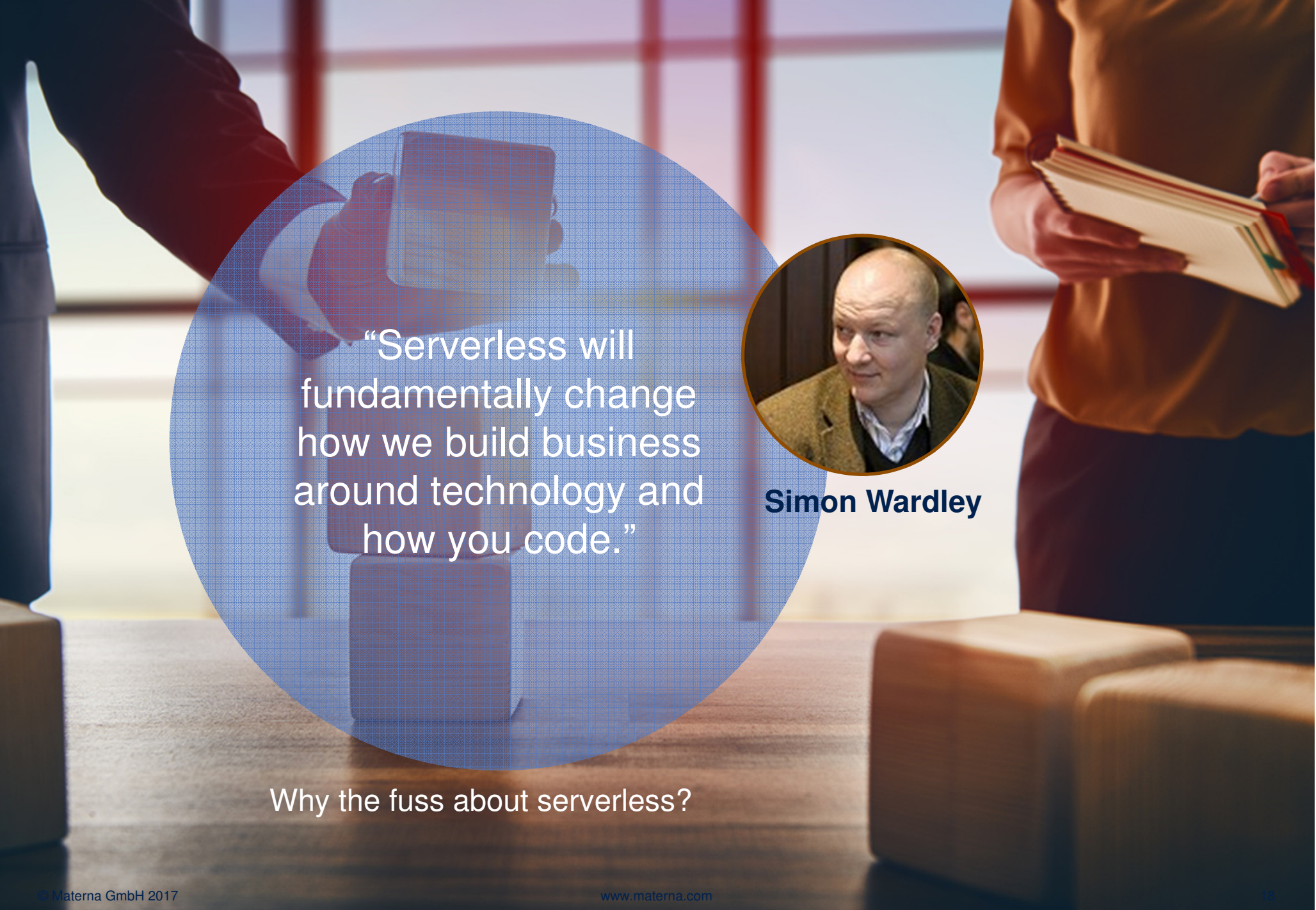
THE NEW STACK

Serverless cost models charge more accurately for usage

Cloud resource cost
better matches
business value gained

Not a silver bullet, but this can result in substantial savings for many workloads





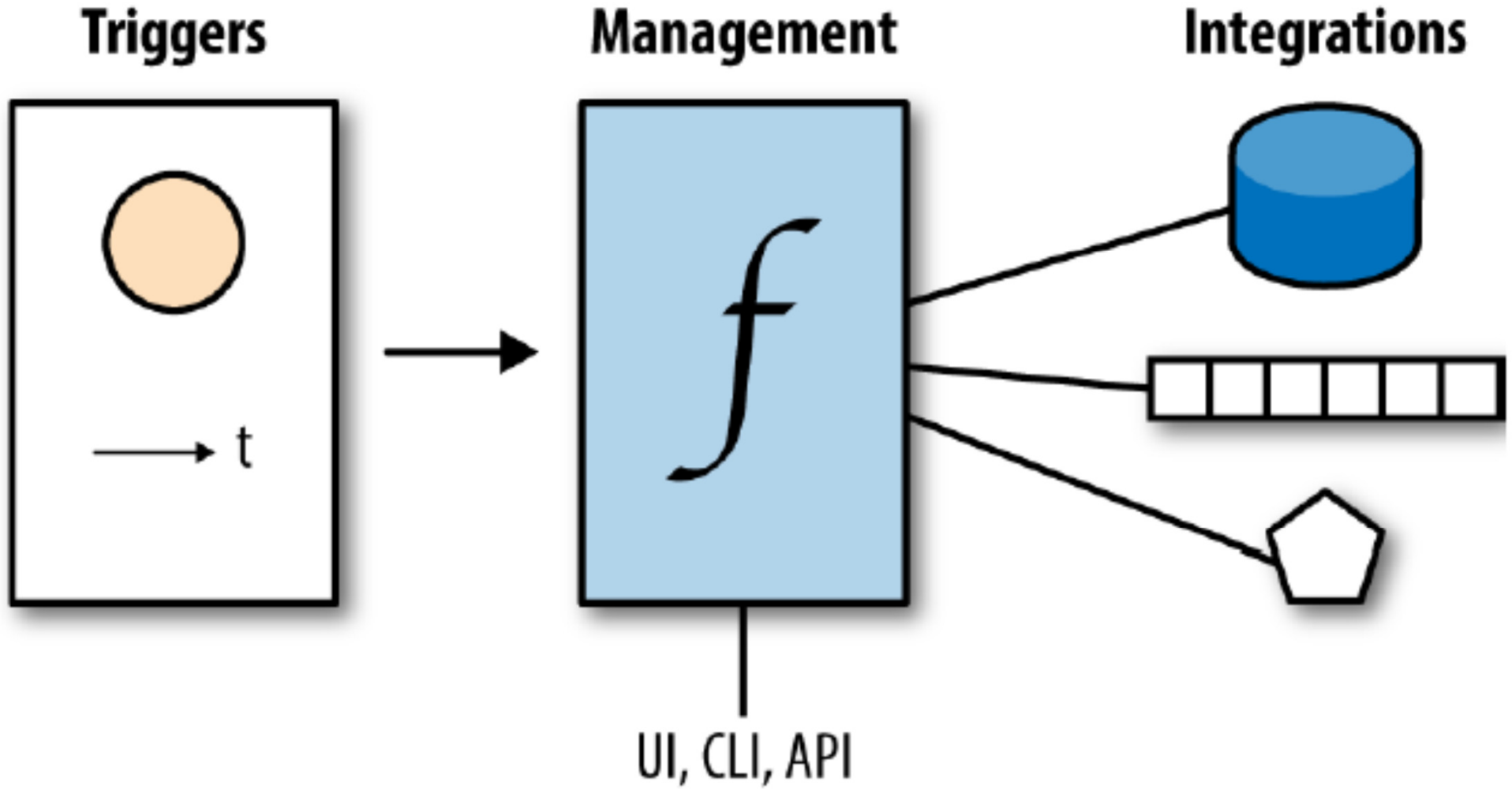
“Serverless will fundamentally change how we build business around technology and how you code.”



Simon Wardley

Why the fuss about serverless?

concept of serverless computing

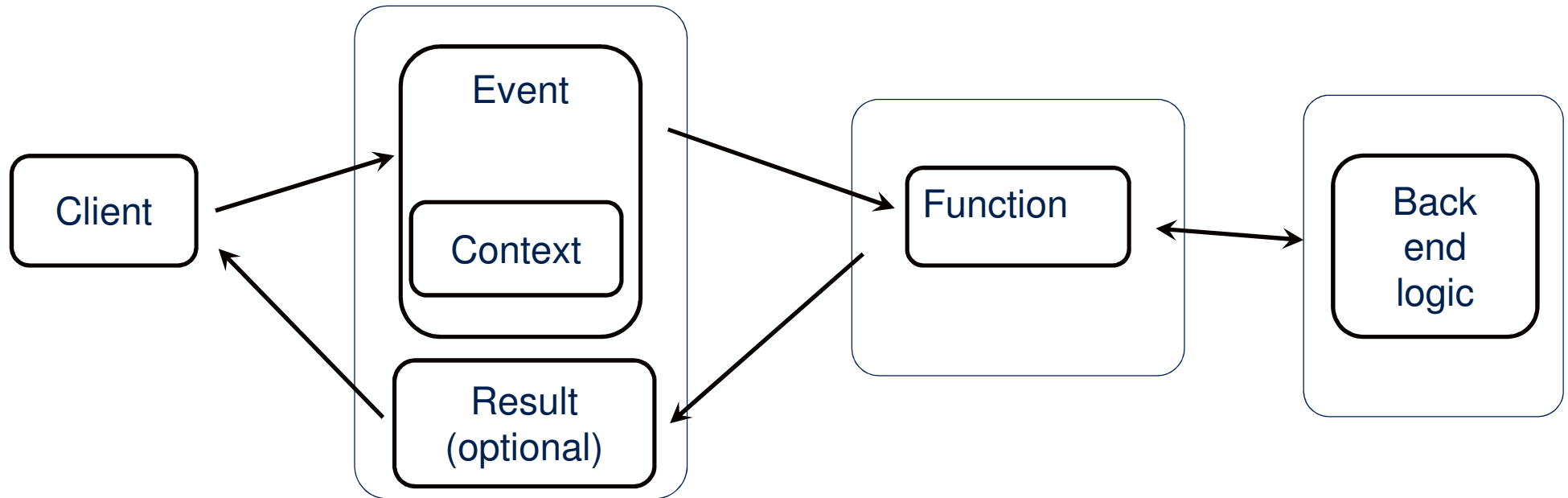


What are Lambdas?

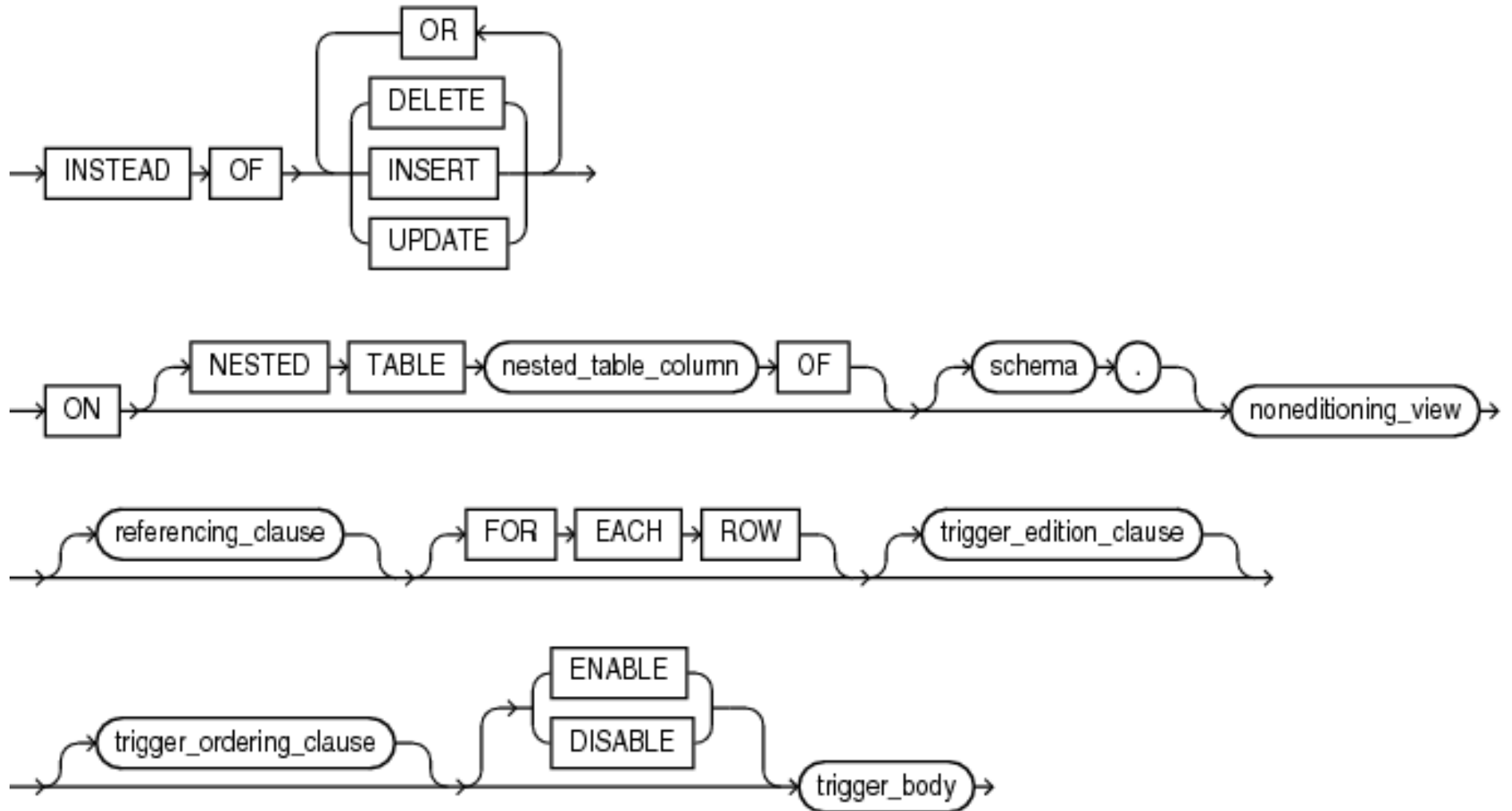


- **λ - calculus** (Alonso Church 1936) Undecidability
- **Lambda-Expression** (Java 8 Functional Interface)
 $(i, j) \rightarrow i + j$
- **Lambda-Architecture** (Nathan Marz 2014)
query = $\lambda(\text{all data}) = \lambda(\text{live streaming data})^*$
 $\lambda(\text{historical data})$
- **AWS Lambda** (2014): runtime for small functions

PaaS = FaaS + BaaS



SQL Trigger



Agenda.

Why serverless?

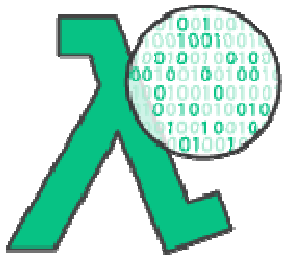
Characteristics of serverless applications?

Overview serverless offers
AWS Lambda, IBM OpenWhisk, Azure Functions

The serverless framework

Resume

Serverless provider



AWS Lambda



Azure Functions



IBM Open Whisk



Cloud Functions

Azure functions elements





Azure Functions architecture

Search	Search
Functions	Functions
+ New Function	+ New Function
⚡ GenericWebhookJS1	⚡ GenericWebhookJS1
Proxies (preview)	⚡ GenericWebhookJS1
+ New proxy	⚡ TimerTriggerJS1
	</> Develop >
	⚡ Integrate
	⚙ Manage
	🔍 Monitor
	Proxies (preview)
	+ New proxy
	🔧 Function app settings
	📄 API Definition (preview)


index.js

Save

▶ Run

```
1 module.exports = function (context, myTimer) {
2   var timeStamp = new Date().toISOString();
3
4   if(myTimer.isPastDue)
5   {
6     context.log('JavaScript is running late!');
7   }
8   context.log('JavaScript timer trigger function ran!', timeStamp);
9
10  context.done();
11  };
```

Apache OpenWhisk Serverless platform <http://openwhisk.org/>



Apache OpenWhisk

This project contains the source code for the Apache OpenWhisk Serverless platform

✉ dev@openwhisk.incubator.apache.org

Repositories 👤 People **13**

Included repositories

<h4>openwhisk</h4> <p>Apache OpenWhisk is a serverless event-based programming service and an Apache Incubator project.</p> <p>● Scala ★ 1.4k 🍴 289</p>	<h4>openwhisk-catalog</h4> <p>Curated catalog of Apache OpenWhisk packages to interface with event producers and consumers</p> <p>● JavaScript ★ 6 🍴 16</p>	<h4>openwhisk-wskdeploy</h4> <p>Utility for deploying and managing Apache OpenWhisk packages and projects</p> <p>● Go ★ 6 🍴 9</p>
<h4>openwhisk.github.io</h4> <p>Apache OpenWhisk website (openwhisk.org) code built using Jekyll</p> <p>● HTML ★ 4 🍴 10</p>	<h4>openwhisk-client-go</h4> <p></p> <p>● Go ★ 4 🍴 4</p>	

<http://openwhisk.org/>



APACHE
OpenWhisk

Fine-grain
consumpti



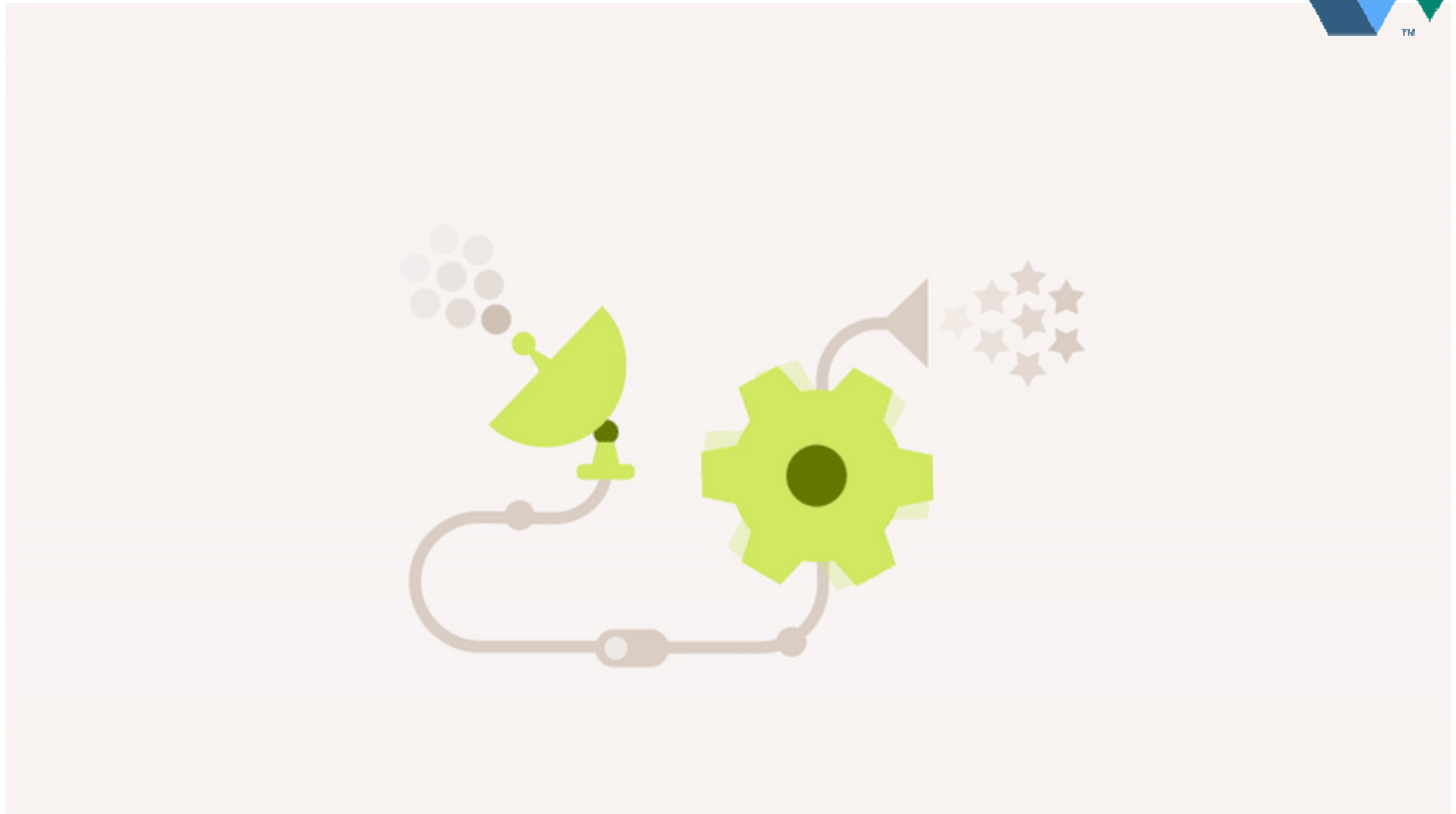
Plug your service into Apache OpenWhisk.

Apache OpenWhisk makes it simple for thousands of developers to integrate with your service. Make it easy for developers to perform actions when something happens in your service, or to send data into your service when things happen somewhere else.

Become part of a thriving, open source ecosystem. Get your service in the hands of more developers, and make it easier than ever for other providers to integrate with your services.

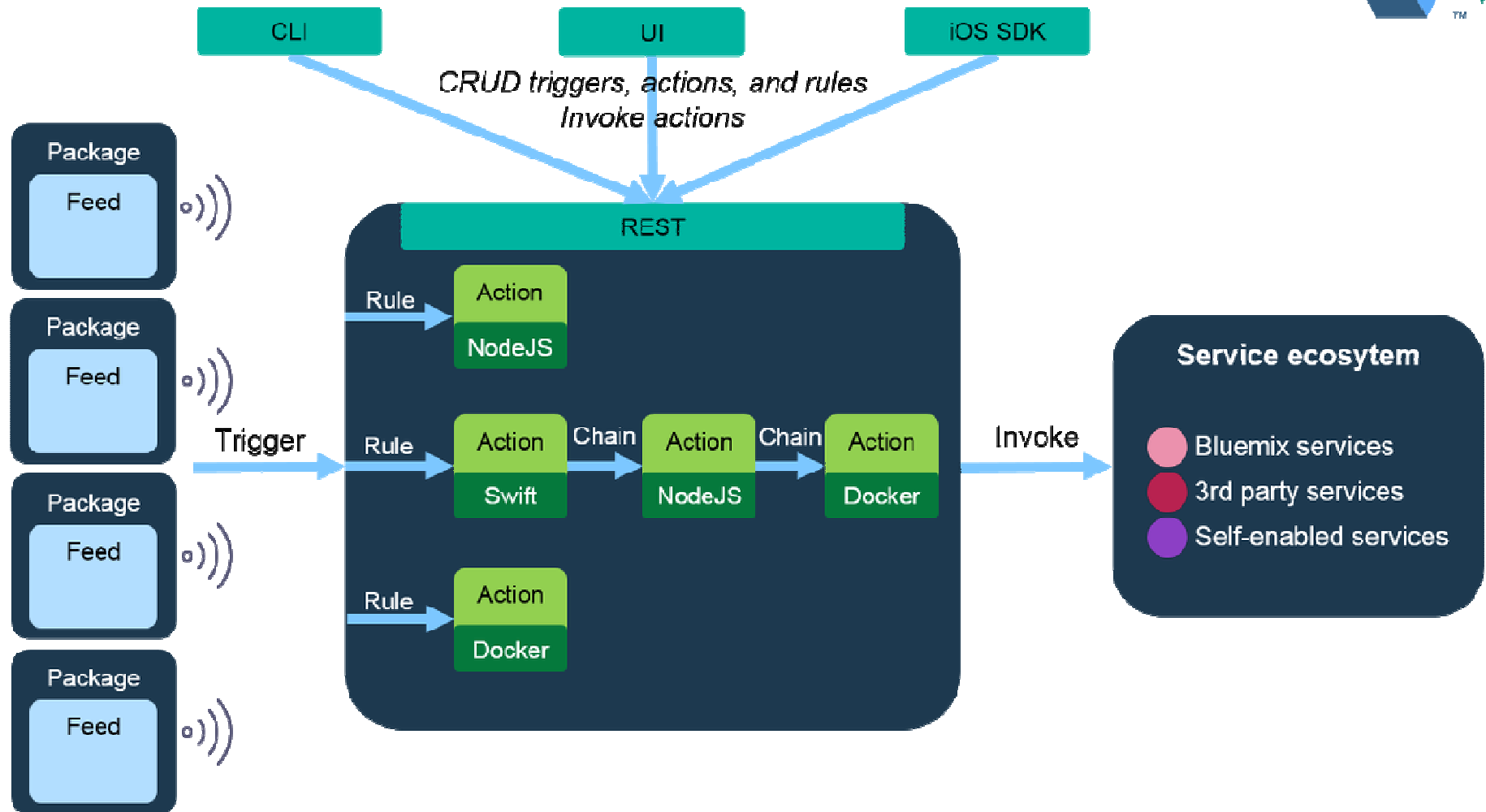
Pay only for the actions use, do
an action is not
so you don't pay

Developers work with packages, triggers, actions, and rules

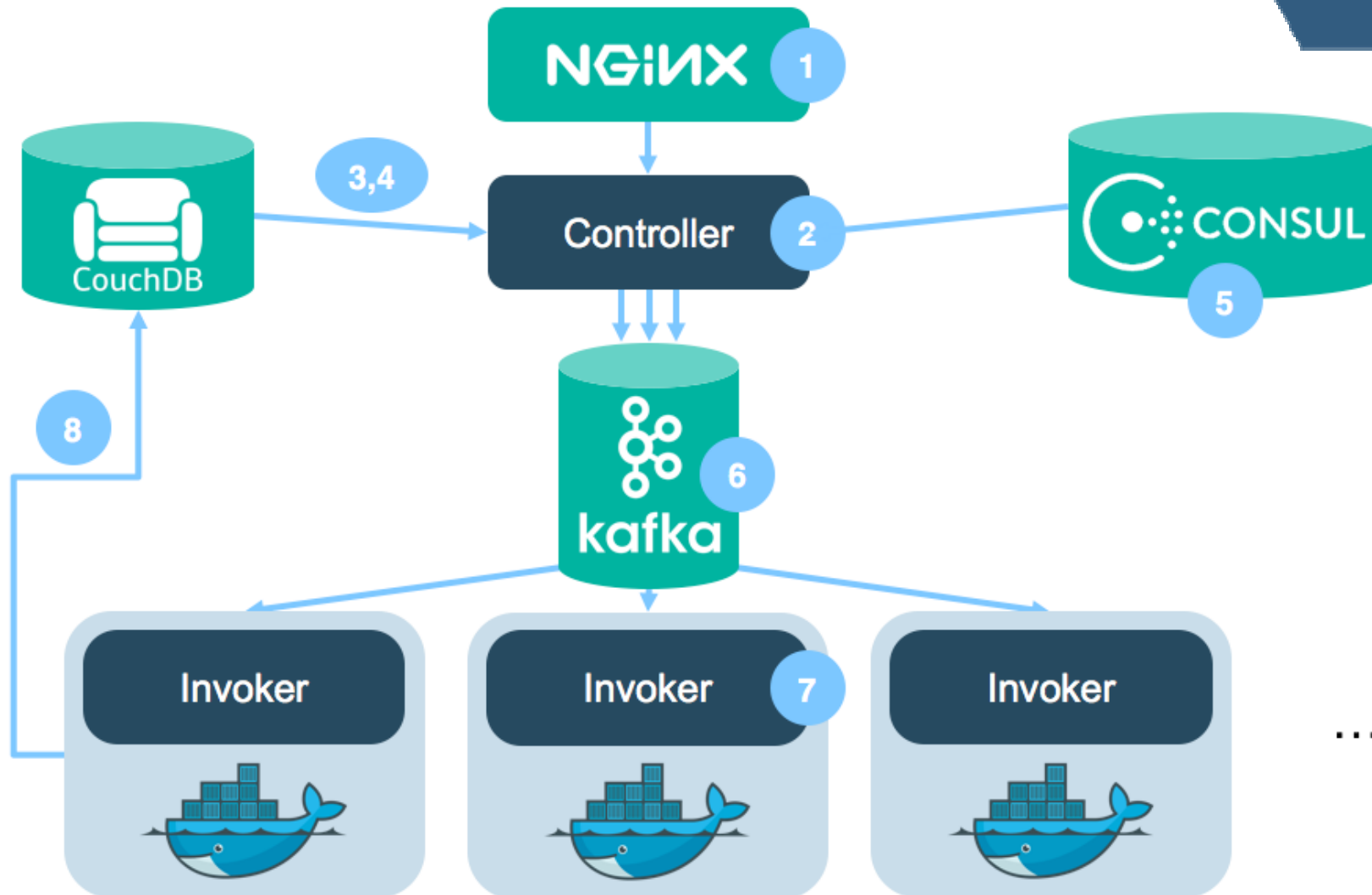




OpenWhisk architecture



OpenWhisk is built on solid open source foundations



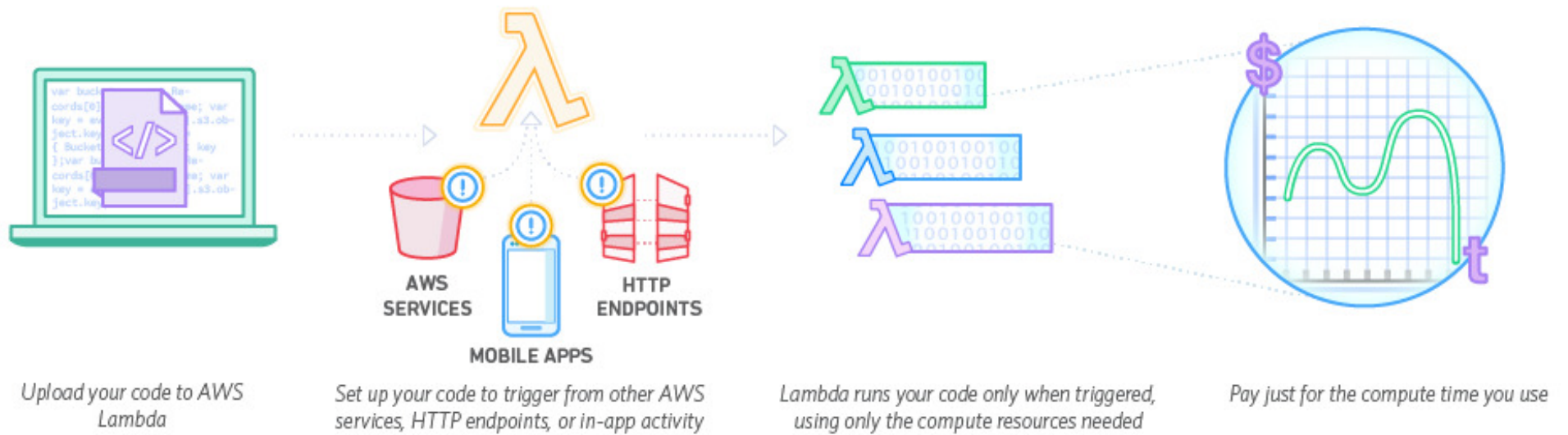
Bluemix offers management, tooling, and monitoring Apache OpenWhisk

The screenshot displays the IBM Bluemix OpenWhisk interface. At the top, there's a navigation bar with 'Docs', '101 Trial Days Remaining', 'Daniel Krook's Account', 'US South', 'krook@us.ibm.com', and 'dev'. Below this is a secondary navigation bar with 'Catalog', 'Support', and 'Manage'. The main interface has a top navigation bar with 'Getting Started', 'Manage', 'Develop', 'Monitor', and 'APIs'. The 'Develop' tab is active, showing a sidebar with 'MY ACTIONS' (including 'Hello World', 'Hello World With Params', 'alert-customer-event', 'analyze-service-event', 'check-warranty-renewal', and 'create-order-event'), 'MY SEQUENCES' (including 'order-sequence' and 'service-sequence'), 'MY RULES' (including 'check-warranty-rule', 'openridge-feed-rule', 'order-rule', and 'service-rule'), and 'MY TRIGGERS'. The main area shows the 'create-order-event' action, which is a 'NODE.JS 6 ACTION'. It contains the following code:

```
16
17 var Cloudant = require('cloudant');
18
19 /**
20  * 1. Fetches the customer object on service database changes trigger (which has the appliance_serial)
21  * 2a. If in warranty, create request in order database with status of 'ordered', which will invoke alert-customer-event to send the appropriate email.
22  * 2b. If not in warranty, create request in order database with status of 'pending', which will invoke alert-customer-event to send the appropriate email.
23  *
24  * @param params._id The id of the record in the Cloudant 'service' database
25  * @param params.appliance_serial The appliance serial inserted in the Cloudant 'service' database
26  * @param params.warranty_expiration The warranty expiration inserted in the Cloudant 'service' database
27  * @param params.part_number The part number inserted in the Cloudant 'service' database
28  * @param params.CLOUDANT_USERNAME Cloudant username (set once at action update time)
29  * @param params.CLOUDANT_PASSWORD Cloudant password (set once at action update time)
30  * @return Standard OpenWhisk success/error response
31  */
32 function main(params) {
33
34   // console.log(params);
35
36   return new Promise(function(resolve, reject) {
37
38     // Configure database connection
39     var cloudant = new Cloudant({
40       account: params.CLOUDANT_USERNAME,
41       password: params.CLOUDANT_PASSWORD
42     });
43     var applianceDb = cloudant.db.use('appliance');
44     var orderDb = cloudant.db.use('order');
45
46     // Find the appliance object by the serial (which is its _id)
47     applianceDb.get(params.appliance_serial, function(err, body, head) {
48       if (err) {
49         console.log('[create-order-event.main] error: applianceDb');
50         console.log(err);
51         reject({
```

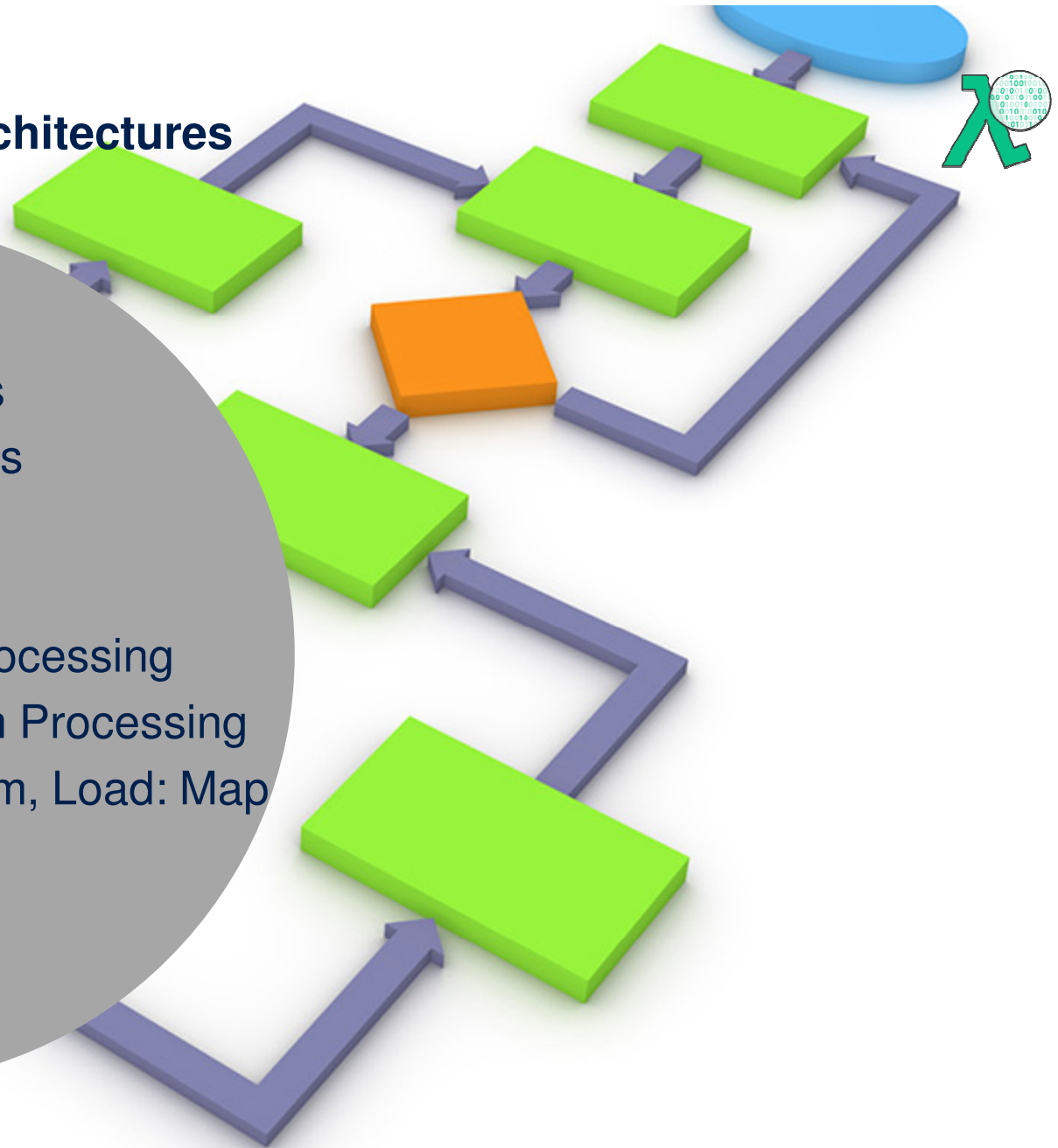
At the bottom right of the code editor, there is a 'TM' logo and the text 'This Code Is Live! [View REST Endpoint](#)'. Below the code editor, there are two buttons: 'Link into a Sequence' and 'Automate this Action'.

AWS Lambda functions

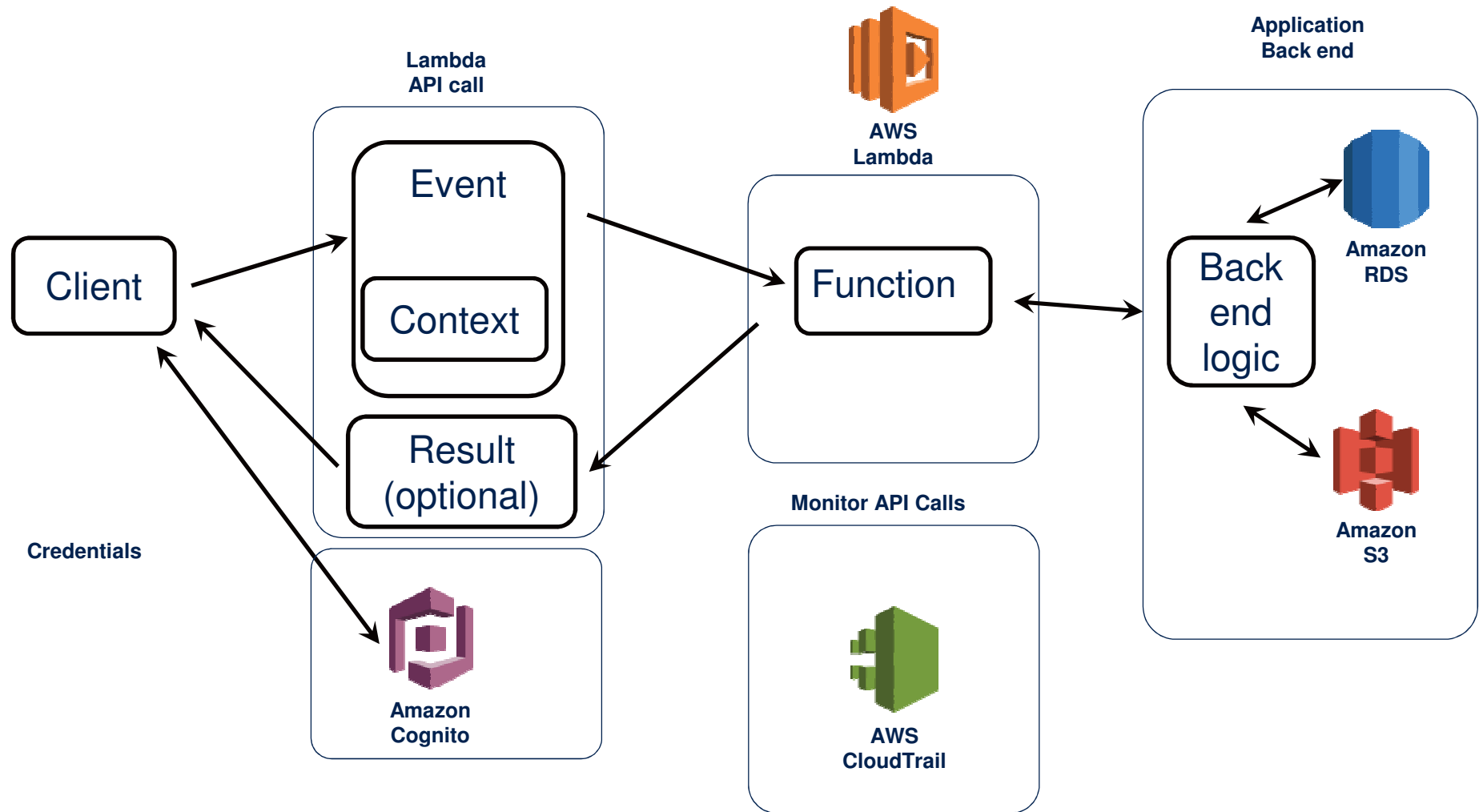


AWS Lambda Reference Architectures

- Web Applications
- Mobile-Back-Ends
- IoT-Back-Ends
- Data Processing
- Real-time File Processing
- Real-time Stream Processing
- Extract, Transform, Load: Map Reduce
- Chatbots



Lambda in Action: PaaS = FaaS + BaaS





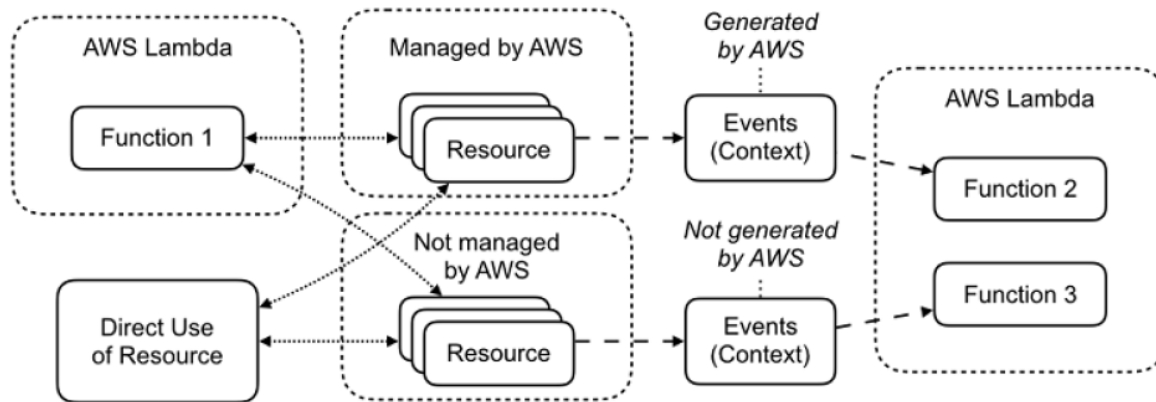
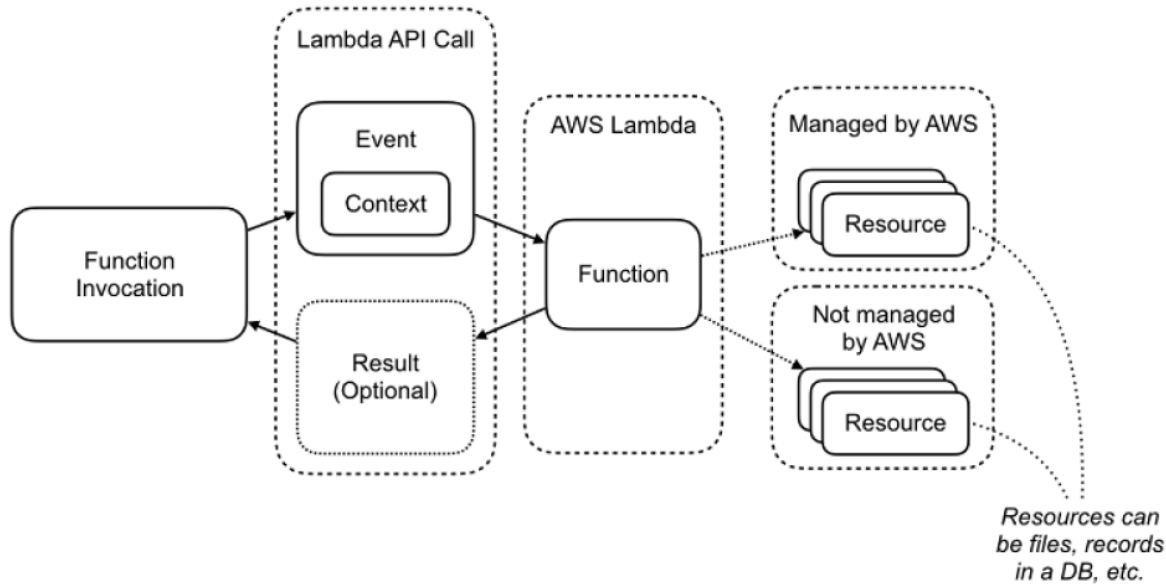


Steps to a lambda blank function:

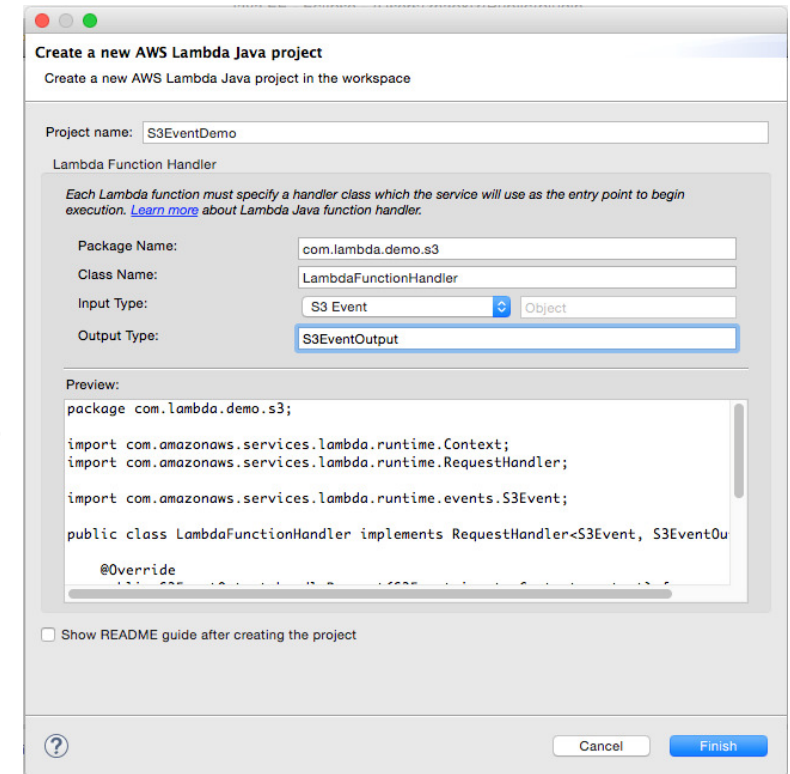
- Step 1: Create a lambda blank function
- Step 2: Configure triggers
- Step 3: Configure the function, choose Java 8 as the language
- Step 4: Upload your executable JAR and set the environmental variables
- Step 5: Set the Lambda function handler and role
- Step 6: Test invoke the function

AWS Lambda with and from AWS resources

-  AWS Java Web Project
-  AWS Lambda Java Project
-  AWS Java Project



For example, a file is uploaded or something is written in a database



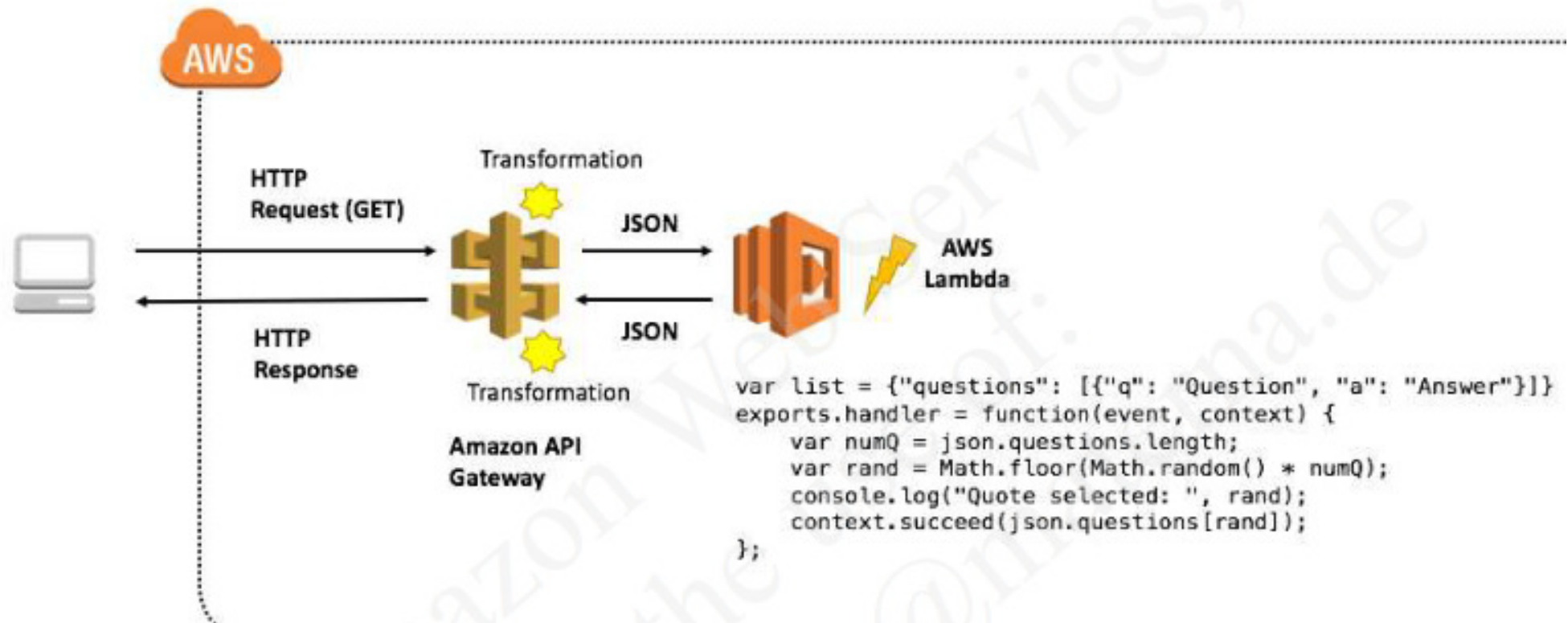
 Amazon Web Services

Team

▶ Run function on AWS Lambda...

▶ Upload function to AWS Lambda...


AWS Lambda with JavaScript function




File: 'serverless.template'

Resource types


- ApplicationAutoScaling
- ApiGateway
- AutoScaling
- CertificateManager
- CloudFormation
- CloudFront
- CloudTrail
- CloudWatch
- CodeBuild
- CodeCommit
- CodeDeploy
- CodePipeline
- Config
- DataPipeline
- DirectoryService




ArticleBu...
Bucket



Article Ta...
Table




PutArticle
Function



GetArticle
Function

Properties | Metadata | DependsOn | Condition

PutArticle 

```

1 {
2   "Resources": {
3     "PutArticle": {
4       "Type": "AWS::Serverless::Function",
5       "Properties": {
6         "Handler": "PutArticle",
7         "Policies": [
8           "AmazonDynamoDBFullAccess",
9           "AmazonS3FullAccess"
10        ],
11        "Environment": {
12          "Variables": {
13            "ARTICLE_TABLE_NAME": {

```

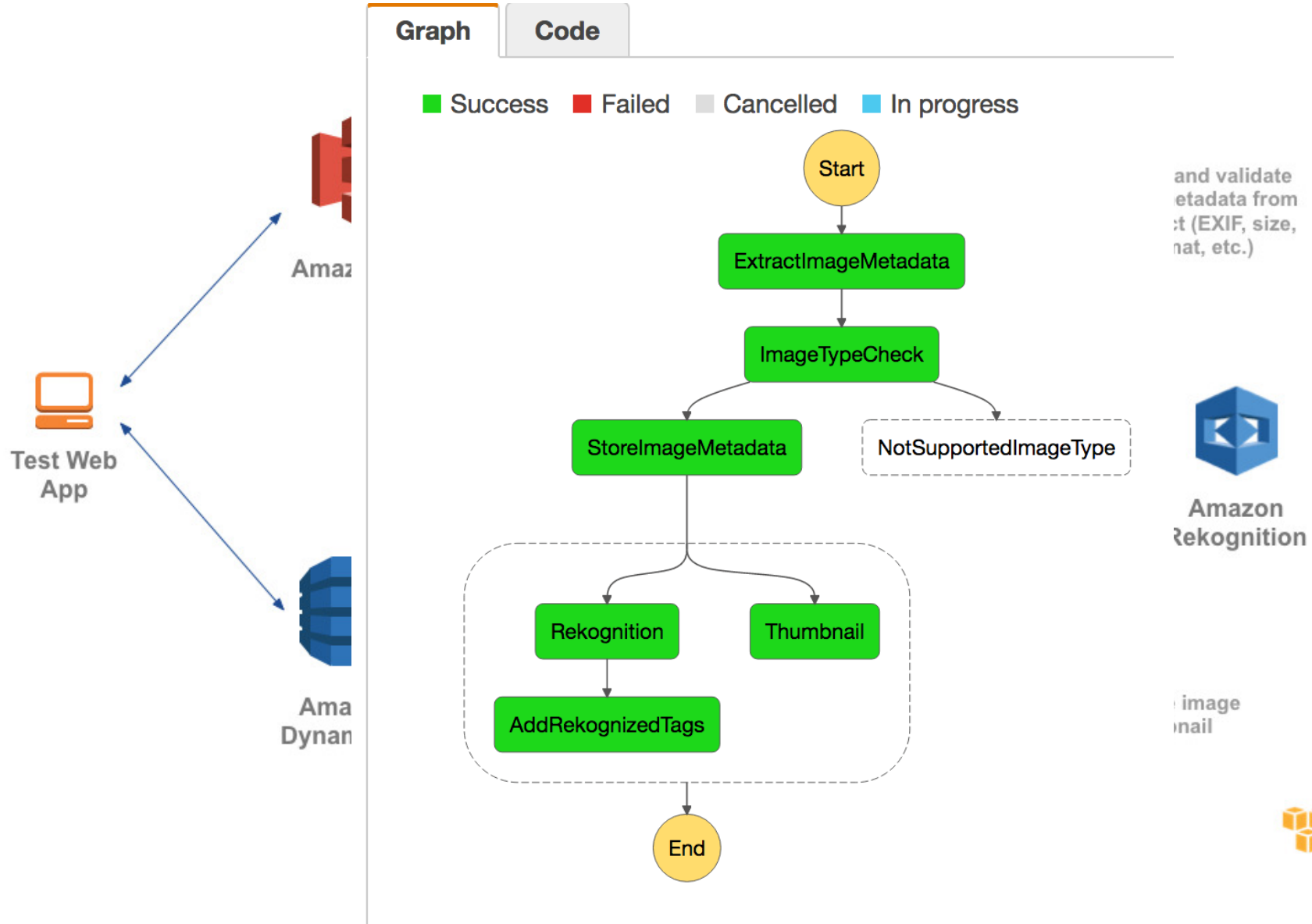


Eclipse Lambda HelloWorldService Test Case with test data

```
1 package de.materna;
2
3 import com.amazonaws.services.lambda.runtime.Context;
4
5
6
7
8 public class LambdaFunctionHandler implements RequestHandler<S3Event, Object> {
9
10     @Override
11     public Object handleRequest(S3Event input, Context context) {
12         context.getLogger().log("Input: " + input);
13
14         // TODO: implement your handler
15         return null;
16     }
17 }
18
19
```

Step 1: Implement your function
Step 2: Test your function locally as JUnit test
Step 3: Upload your function to AWS Lambda
Step 4: Invoke your function on AWS

AWS Step Functions orchestrate a serverless processing workflow



Agenda.

Why serverless?

Characteristics of serverless applications?

Overview serverless offers
AWS Lambda, IBM OpenWhisk, Azure Functions

The serverless framework

Resume

<http://www.serverless.com>

SERVERLESS FRAMEWORK VERSION 1.0

Build auto-scaling, pay-per-execution,
event-driven apps on AWS Lambda

▶ WATCH THE VIDEO

📖 READ THE DOCS

```
# Install serverless globally
$ npm install serverless -g

# Create an AWS Lambda function in Node.js
$ serverless create --template aws-nodejs

# Deploy to live AWS account
$ serverless deploy

# Function deployed!
$ http://api.amazon.com/users/update

-> Read the docs or connect with the community
```

Powered by



Serverless Framework – Build web, mobile and IoT applications with serverless architectures via AWS Lambda and API Gateway

Serverless is composed of Plugins. A group of default Plugins ship with the Framework, and here are some others you can add to improve/help your workflow:

- **Meta Sync** - Securely sync your the variables in your project's `_meta/variables` across your team.
- **Offline** - Emulate AWS Lambda and Api Gateway locally to speed up your development cycles.
- **Hook Scripts** - Easily create shell script hooks that are run whenever Serverless actions are executed.
- **CORS** - Adds support for CORS (Cross-origin resource sharing).
- **Serve** - Simulate API Gateway locally, so all function calls can be run via localhost.
- **Webpack** - Use Webpack to optimize your Serverless Node.js Functions.
- **Serverless Client** - Deploy and config a web client for your Serverless project to S3.
- **Alerting** - This Plugin adds Cloudwatch Alarms with SNS notifications for your Lambda functions.
- **Optimizer** - Optimizes your code for performance in Lambda. Supports coffeefify, babelify and other transforms
- **CloudFormation Validator** - Adds support for validating your CloudFormation template.
- **Prune** - Delete old versions of AWS lambdas from your account so that you don't exceed the code storage limit.
- **Base-Path** - Sets a base path for all API Gateway endpoints in a Component.
- **Test** - A Simple Integration Test Framework for Serverless.
- **SNS Subscribe** - This plugin easily subscribes your lambda functions to SNS notifications.
- **JSHint** - Detect errors and potential problems in your Lambda functions.
- **ESLint** - Detect errors and potential problems in your Lambda functions using eslint.
- **Mocha** - Enable test driven development by creating test cases when creating new functions

„No server is easier to manage than no server“
Dr. Werner Vogels AWS re:Invent 2015

<https://github.com/serverless/serverless>



<https://serverless.com/framework/docs/>



- [Intro](#)
- [Installation](#)
- [Credentials](#)
- [Services](#)
- [Functions](#)
- [Events](#)
- [Resources](#)
- [Deploying](#)
- [Testing](#)
- [Variables](#)
- [Packaging](#)
- [IAM](#)
- [Plugins](#)
- [Workflow](#)
- [Serverless.yml](#)



- [Config Credentials](#)
- [Create](#)
- [Install](#)
- [Deploy](#)
- [Deploy Function](#)
- [Deploy List](#)
- [Invoke](#)
- [Invoke Local](#)
- [Logs](#)
- [Metrics](#)
- [Info](#)
- [Rollback](#)
- [Remove](#)
- [Serverless Stats](#)



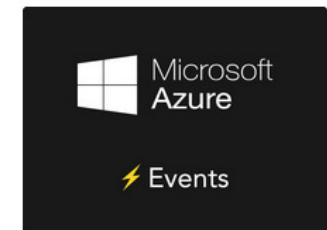
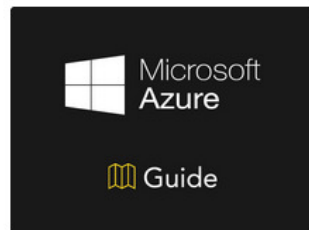
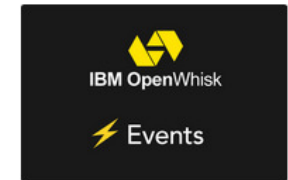
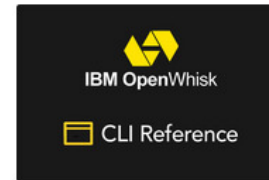
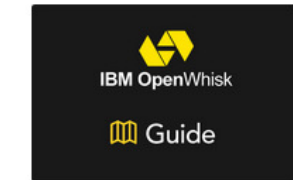
- [API Gateway](#)
- [Streams](#)
- [S3](#)
- [Schedule](#)
- [SNS](#)
- [Alexa Skill](#)
- [IoT](#)
- [CloudWatch Event](#)



- [Hello World](#)

<https://serverless.com/framework/docs/>

support
AWS Lambda
Apache OpenWhisk
Microsoft Azure
Google Cloud Functions



- [Intro](#)
- [Installation](#)
- [Credentials](#)
- [Services](#)
- [Functions](#)
- [Events](#)
- [Resources](#)
- [Deploying](#)
- [Testing](#)
- [Variables](#)
- [Packaging](#)
- [API](#)
- [Plugins](#)
- [Workflow](#)
- [Serverless.yml](#)

- [Config Credentials](#)
- [Create](#)
- [Install](#)
- [Package](#)
- [Deploy](#)
- [Deploy Function](#)
- [Deploy List](#)
- [Invoke](#)
- [Invoke Local](#)
- [Logs](#)
- [Metrics](#)
- [Info](#)
- [Rollback](#)
- [Revoke](#)
- [Serverless State](#)

- [API Gateway](#)
- [S3](#)
- [SQS](#)
- [SNS](#)
- [Step Functions](#)
- [Lambda@Edge](#)
- [CloudWatch Events](#)
- [CloudWatch Log](#)

- [Intro](#)
- [Installation](#)
- [Credentials](#)
- [Services](#)
- [Functions](#)
- [Events](#)
- [Deploying](#)
- [Testing](#)
- [Variables](#)
- [Packaging](#)
- [Plugins](#)
- [Workflow](#)
- [Serverless.yml](#)

- [Config Credentials](#)
- [Create](#)
- [Install](#)
- [Deploy](#)
- [Deploy Function](#)
- [Invoke](#)
- [Invoke Local](#)
- [Logs](#)
- [Info](#)
- [Revoke](#)
- [Serverless State](#)

- [API Gateway](#)
- [Cloudant DB](#)
- [Message Hub](#)
- [Schedule](#)
- [Trigger](#)

- [Intro](#)
- [Cloudant](#)
- [Installation](#)
- [Credentials](#)
- [Services](#)
- [Functions](#)
- [Events](#)
- [Deploying](#)
- [Testing](#)
- [Variables](#)
- [Plugins](#)
- [Workflow](#)

- [Install](#)
- [Deploy](#)
- [Deploy Function](#)
- [Invoke](#)
- [Logs](#)
- [Revoke](#)

- [HTTP](#)
- [Timer](#)
- [Queue Storage](#)
- [Service Bus](#)
- [Event Hub](#)
- [Blob Storage](#)
- [Other Bindings](#)

Agenda.

Why serverless?

Characteristics of
serverless
applications?

Overview
serverless offers
AWS Lambda, IBM
OpenWhisk, Azure
Functions

The serverless
framework

Resume

A man in a dark suit is rappelling down a thick rope on the side of a skyscraper. He is looking upwards and holding the rope with both hands. The background shows a dense cityscape under a cloudy, golden sky. A large, semi-transparent circular graphic is overlaid on the left side of the image, containing the text.

Serverless ! = headless

Characteristics of serverless computing

- Limited resources (RAM, File, Requests)
- Short or long running jobs
- Event-, stream based
- stateless

Challenges

- sequence calls
- Local testing
- Restart
- Tuning
- Tracing, Debugging
- Monitoring



five serverless patterns for use cases:

1. Event-driven data processing
2. Web applications
3. Mobile and Internet-of-Things applications
4. Application ecosystems
5. Event workflows



Resume

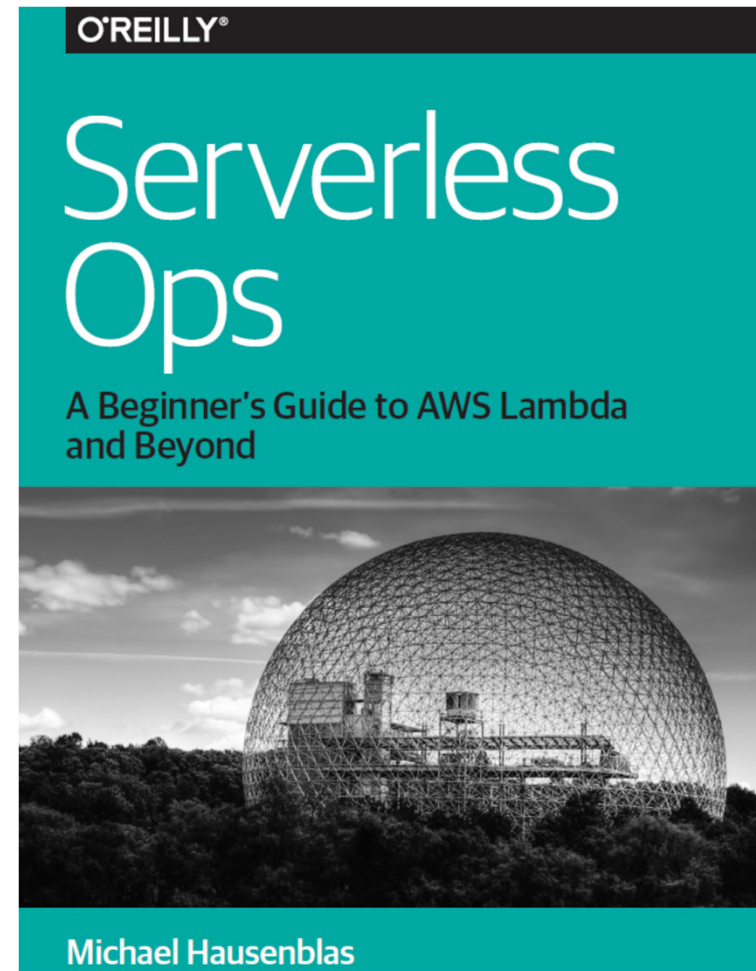
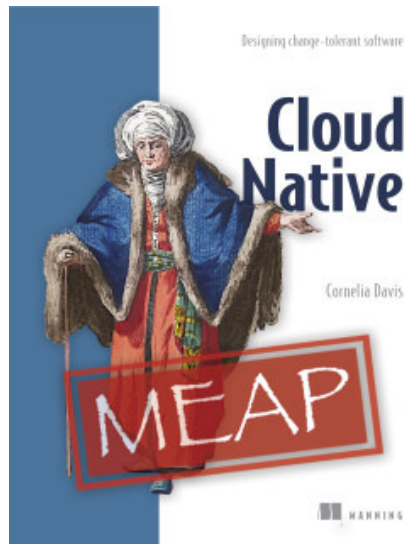
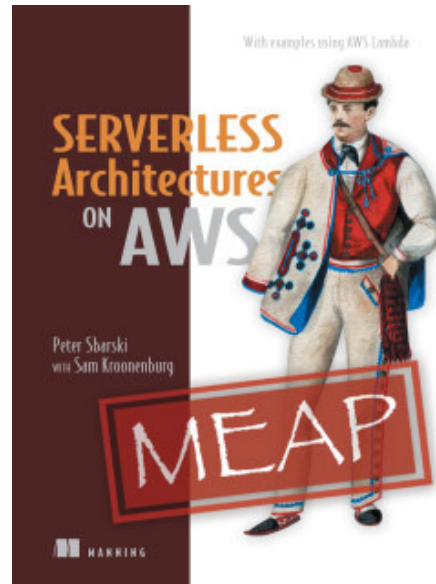
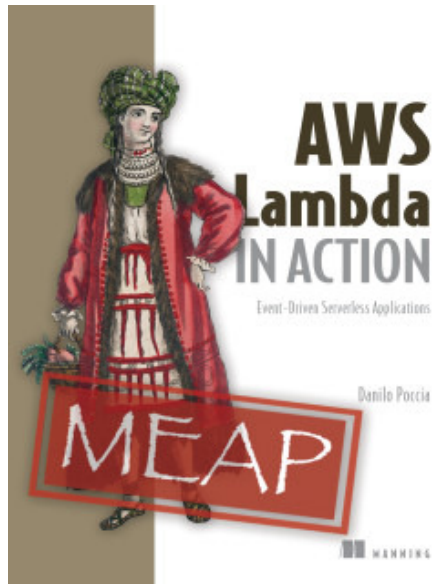
- **Service without server means less Ops**
- **Like convenience food**
- **Reduced costs (development, operation (BaaS), scaling (FaaS))**
- **Vendor lock-in**
- **Stateless (CaaS)**
- **Less complexity more flexibility**
- **Less maintenance more elasticity**
- **Nondeterministic, latency, start-up, a synchronous**
- **Monitoring, Debugging, Logging**
- **Optimization**



Conclusion

- **Newer workloads** moving to cloud better fit for event driven programming
- **Serverless cost** models promise a better match between resources used and value delivered
- platform for **cloud native, event-driven applications**
- **Scaling** Per-Request & Optimal Utilization
- **Flexible** Programming in different languages
- **Crowing Ecosystem**
- Another **emerging alternative architecture style**

Some more books...



further information

- <https://www.informatik-aktuell.de/betrieb/server/function-as-a-service-was-ist-serverless-computing.html>
- <https://cloud.google.com/functions/>
- Michael Hausenblas, Serverless Ops, O'Reilly Media, 2017
- Badri Janakiraman, Serverless, June 2016
<https://martinfowler.com/bliki/Serverless.html>
- AWS Lambda <https://aws.amazon.com/lambda>
- Azure Functions <https://azure.microsoft.com/de-de/services/functions/>
- Danilo Poccia, AWS Lambda in Action, Event-driven serverless applications, Manning, 2016
- Mike Roberts, Serverless Architecture, August 2016
<https://martinfowler.com/articles/serverless.html>
- Peter Sbarski, Serverless Architectures on AWS, Manning, 2017
- Serverless Cost Calculator <http://serverlesscalc.com/>
- Serverless Framework <http://serverless.com/>
- IBM OpenWhisk <http://openwhisk.org/>

Vernetzt.



Kontakt.

Materna GmbH
Frank Pientka
Voßkuhle 37
44141 Dortmund
+49 231 5599-8854
Frank.Pientka@materna.de