CLOUD COMPUTING

Edge, Fog, and Serverless Computing

PAUL TOWNEND

ASSOCIATE PROFESSOR, UMEÅ

Background Trends

5G NETWORKS

4th Generation standard for cellular networks

Launched in 2008

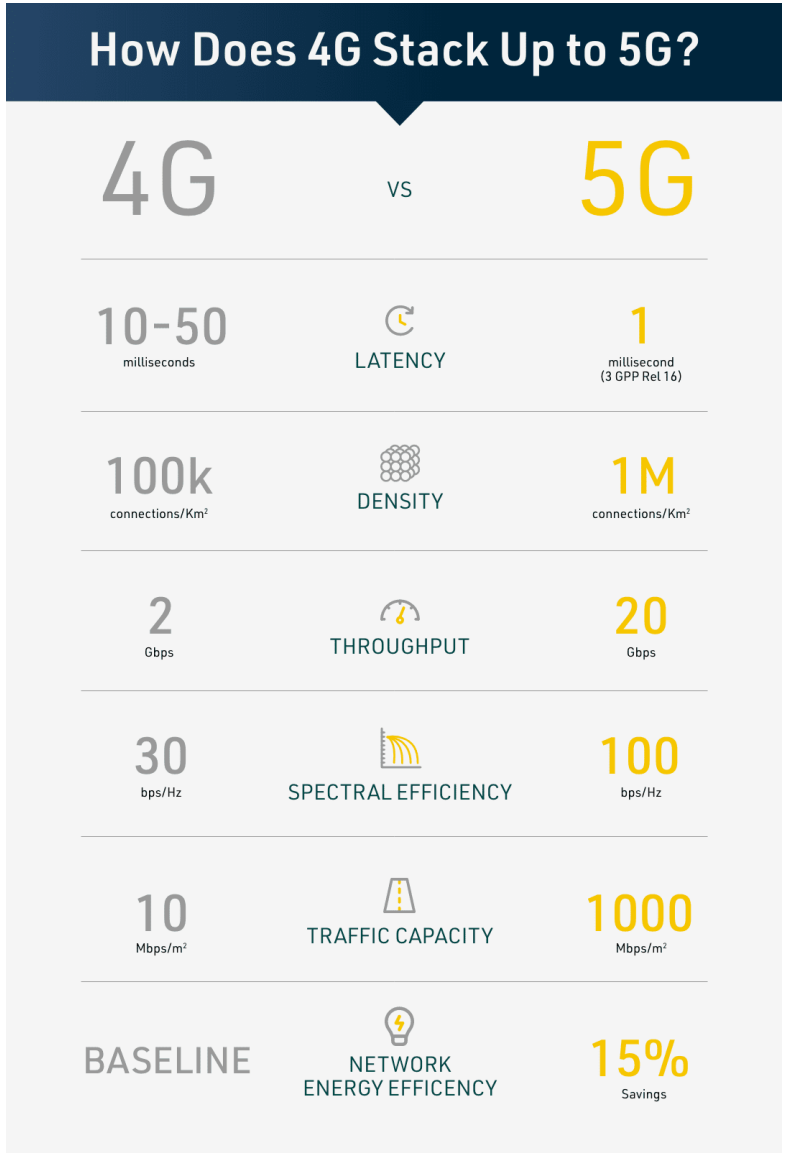
Designed for traditional mobile devices

5th Generation standard for cellular networks

Launched in 2020

Designed for a variety of devices

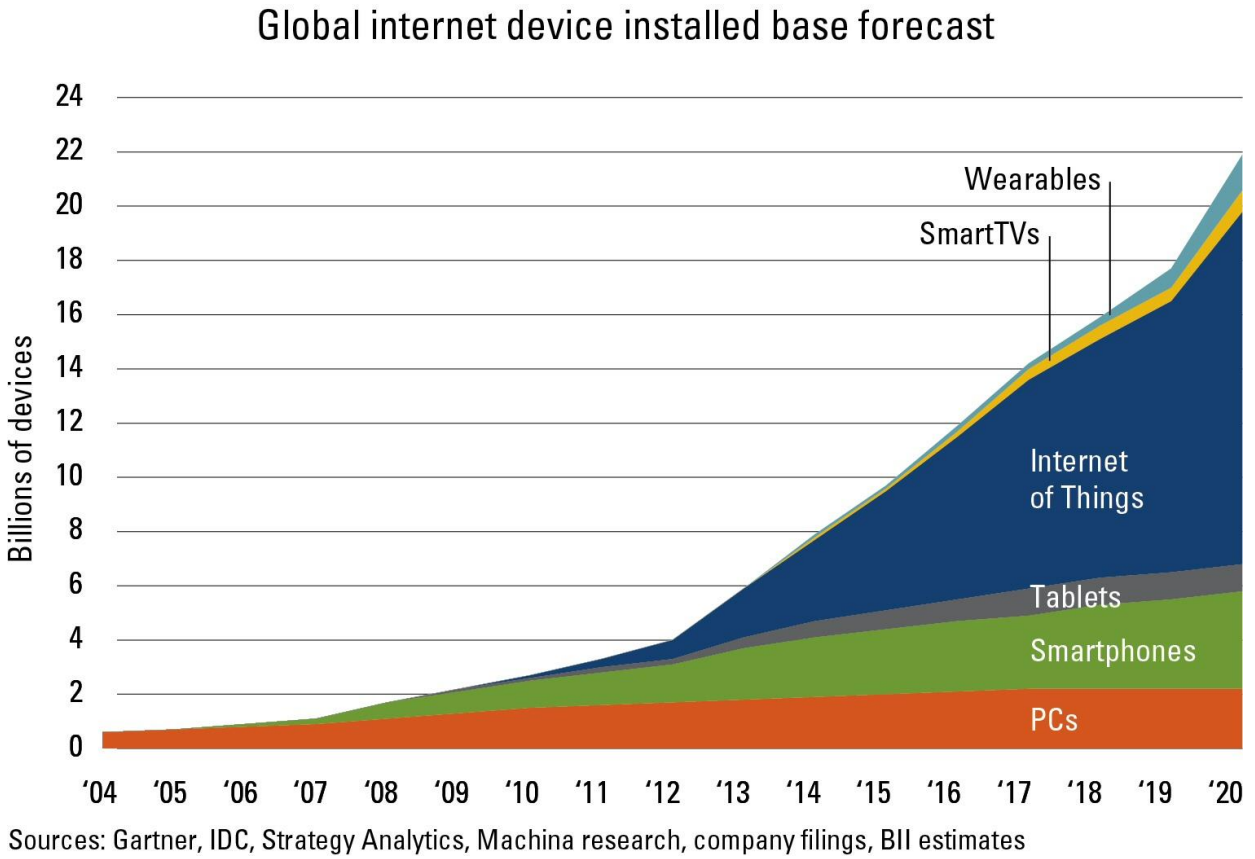
What does 5G enable for computing systems?



INTERNET OF THINGS

"The Internet of Things (IoT) is an environment in which objects, animals or people are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction... (it) may also be referred to as the **Internet of Everything**"

TechTarget IoT Agenda



INTERNET OF THINGS (2)

IoT represents a fundamental shift in how the internet looks and behaves.

Billions of mobile, interconnected devices, creating vast amounts of data and requiring low latency response, predictive analytics, and more.

GROWTH OF DATA IN MODERN SYSTEMS

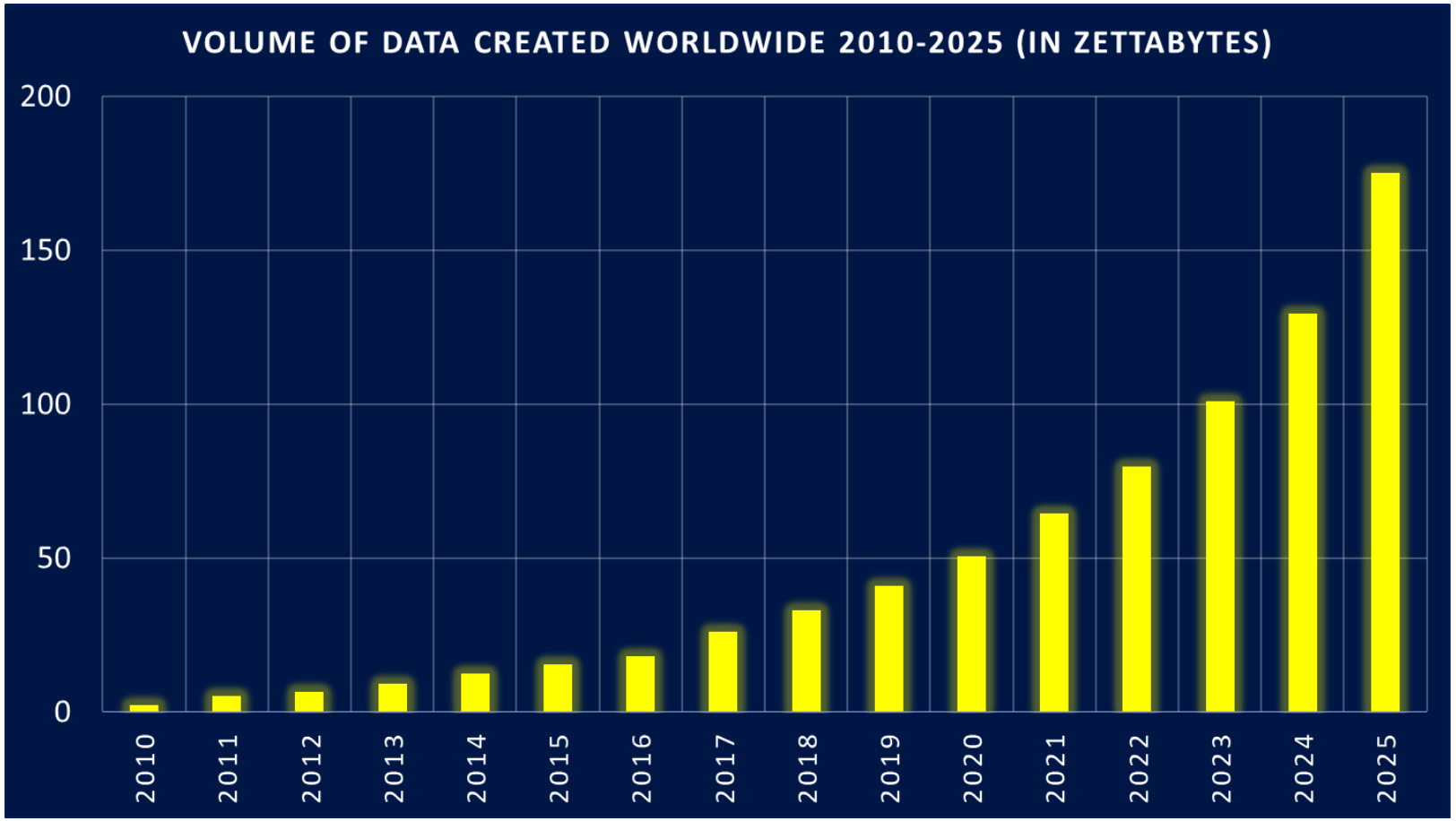
Computer systems are becoming pervasive and distributed

Smart cars generate 5-20TB per day

Medical data is doubling every 73 days

28 billion connected devices by 2021

DATA GROWTH



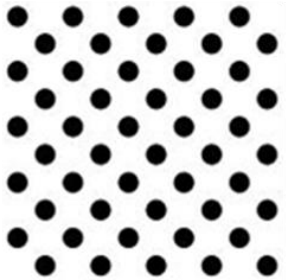
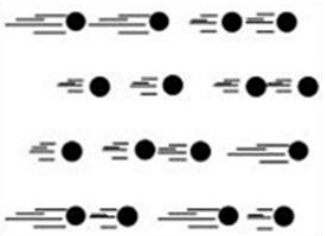
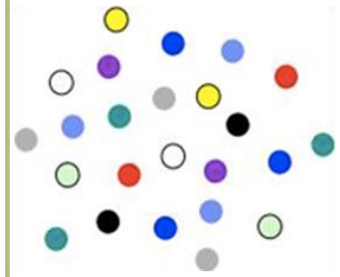
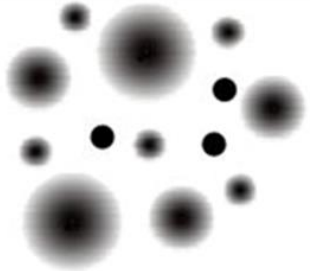
How do we store this data?

How do we process this data?

Source: IDC Global DataSphere, November 2018

3 (OR MORE) V'S

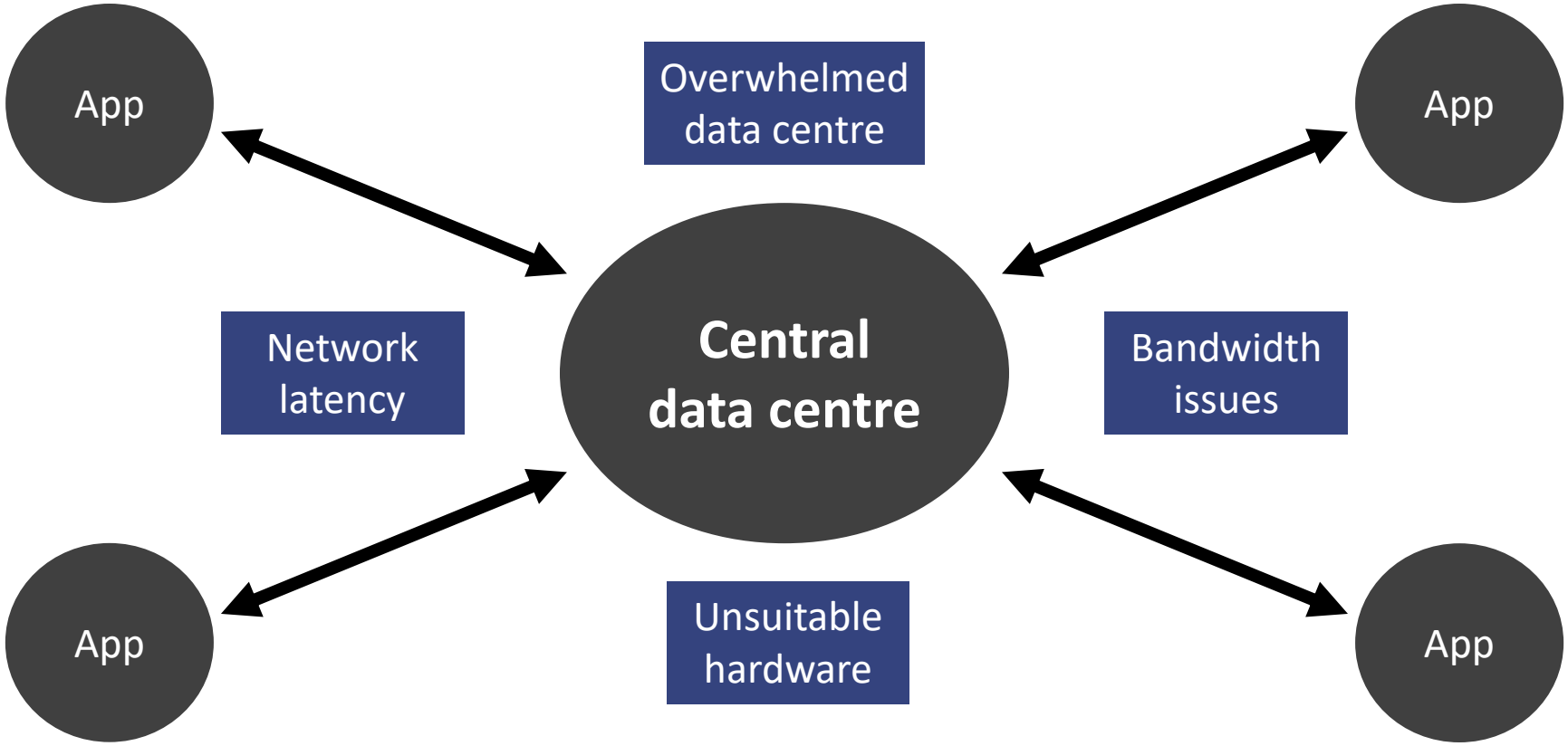
Big Data is not a single technology, technique, or initiative. Rather, it is a **characterisable trend.**

Volume	Velocity	Variety	Veracity	Value
				
Data at Rest	**Data in Motion**	**Data in Many Forms**	**Data in Doubt**	**Data into Money**
Terabytes to Exabytes of existing data to process	**Streaming data, requiring milliseconds to seconds to respond**	**Structured, unstructured, text, multimedia,…**	**Uncertainty due to data inconsistency & incompleteness, ambiguities, latency, deception, model approximations**	**Business models can be associated to the data**

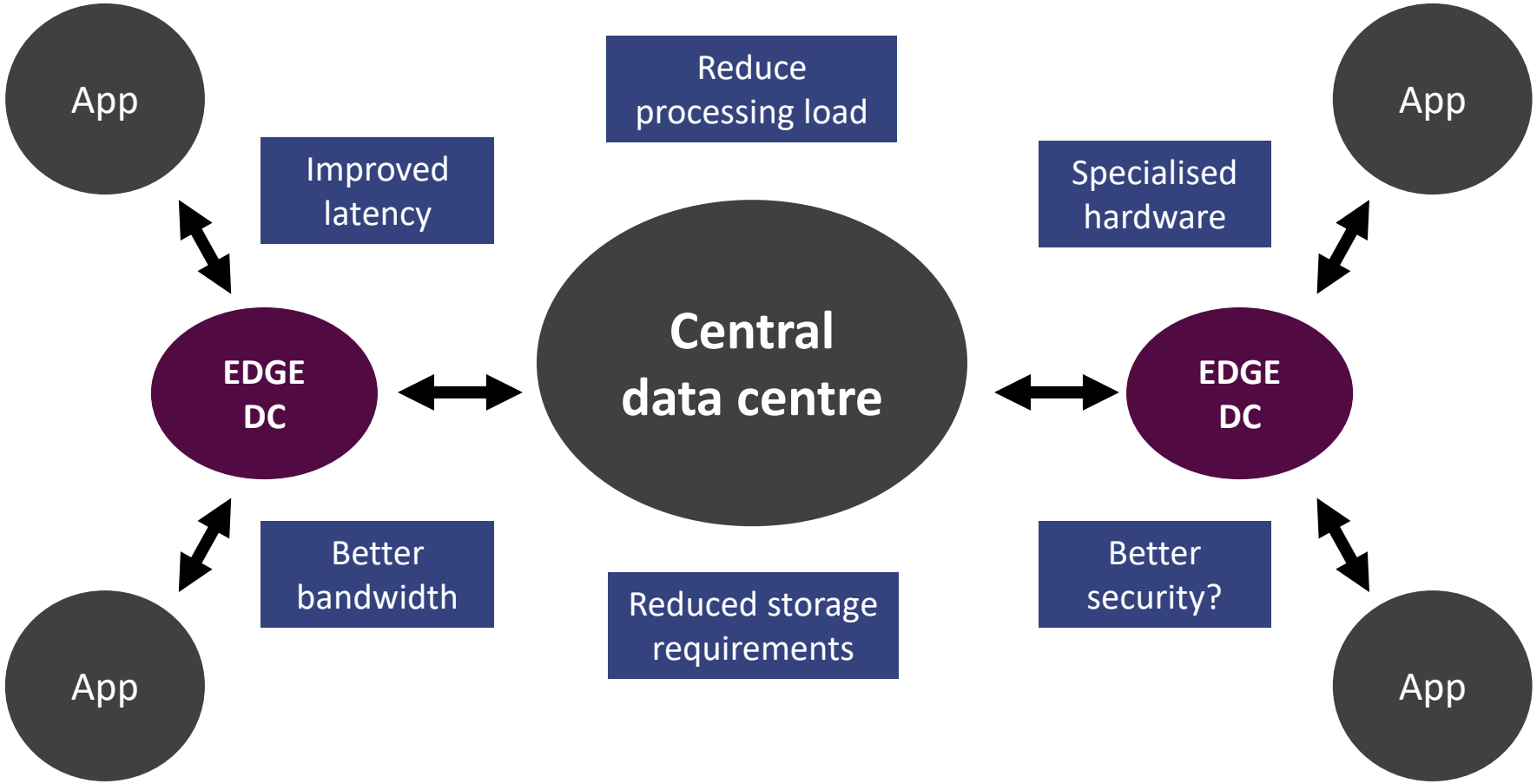
Adapted by a post of Michael Walker on 28 November 2012

Edge Computing

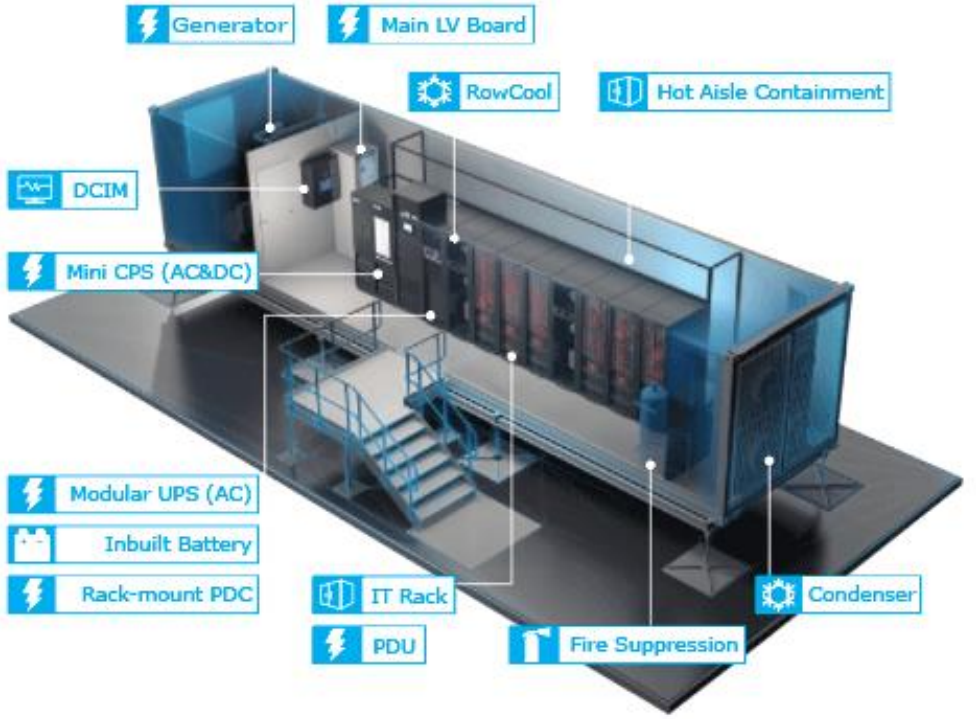
TRADITIONAL DATA CENTER APPROACH



EDGE COMPUTING



WHAT DO EDGE NODES LOOK LIKE?



Small and modular

Non-traditional networking (5G)

Mixture of hardware devices

ISSUES WITH A NON-FEDERATED EDGE MODEL?

Mobile (moving) devices

Discovery

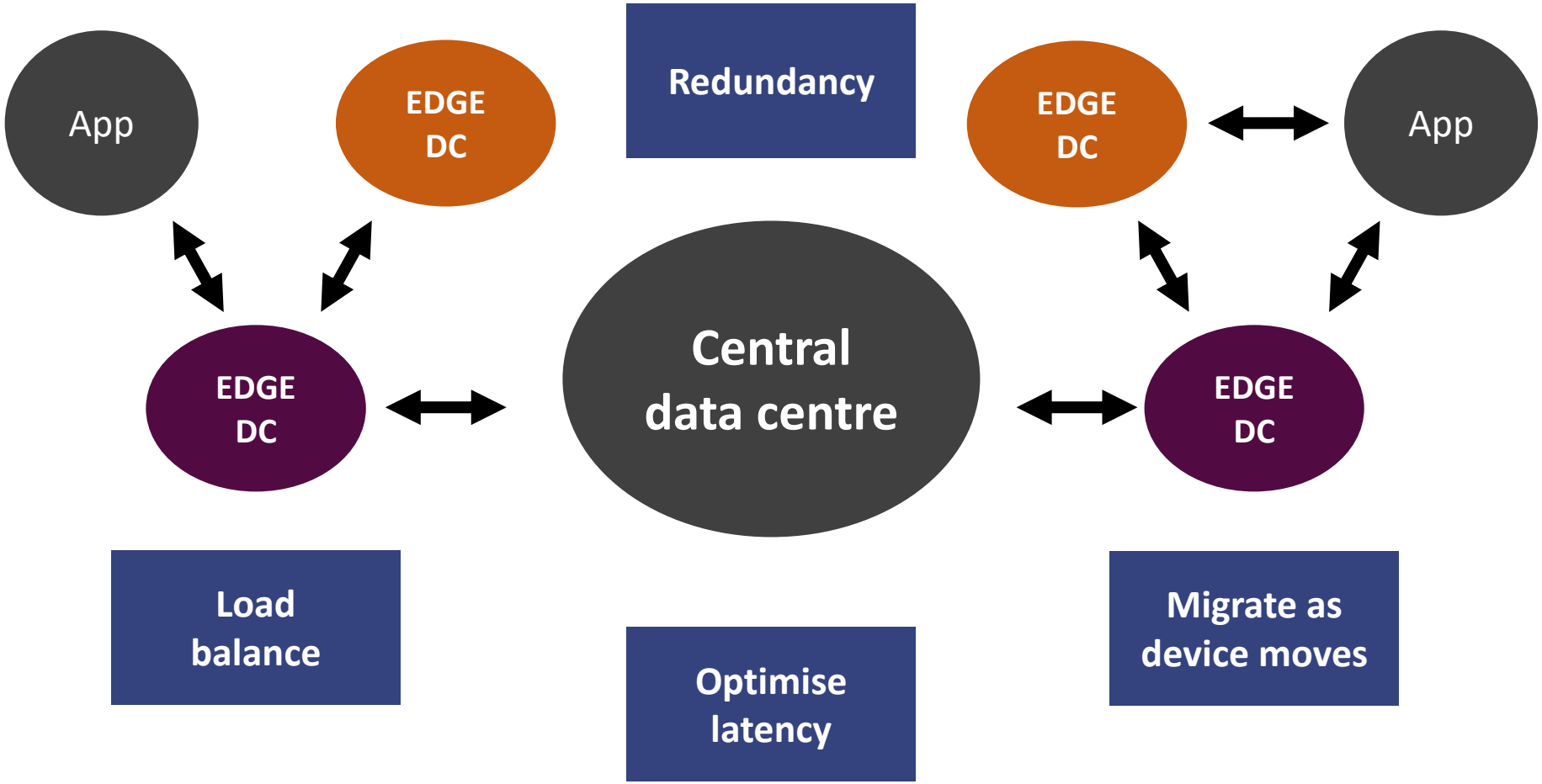
Load balancing

Multiple administrative domains

Migration

Predicting demand

FEDERATED EDGE IDEA



Fog Computing

WHAT IS FOG?

Ultimately, fog computing represents the integration between Edge and Cloud

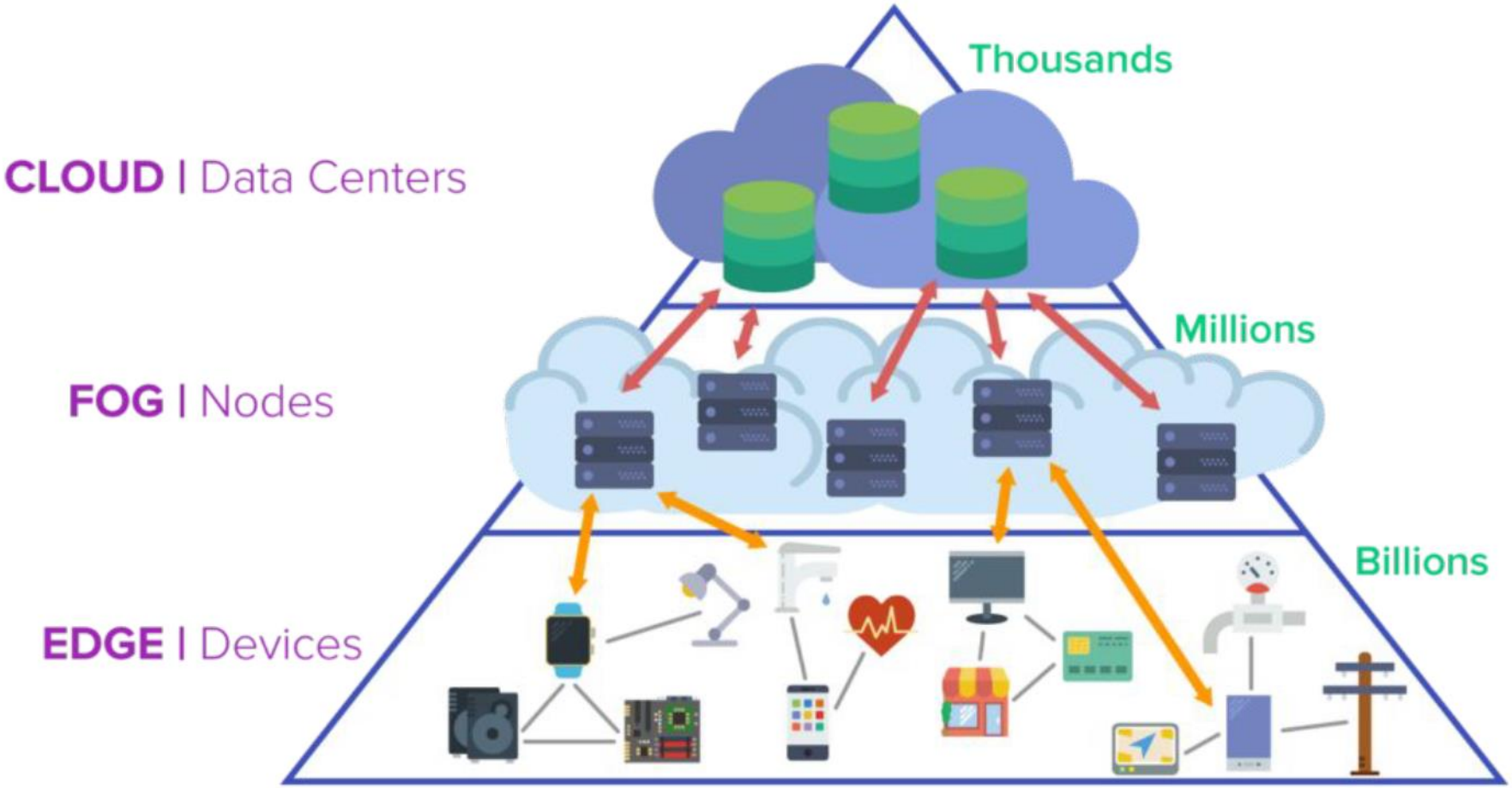
A layered, end-to-end architecture

Distributes resources and services along a continuum from Cloud to things

Solves problems that cannot be implemented using solely Cloud or intelligent Edge devices

A reference architecture has been published by the OpenFog consortium (55 organisations)

FOG ARCHITECTURE LAYERS



INDUSTRIAL IoT DATA PROCESSING LAYER STACK

CLOUD LAYER

Big Data Processing
Business Logic
Data Warehousing

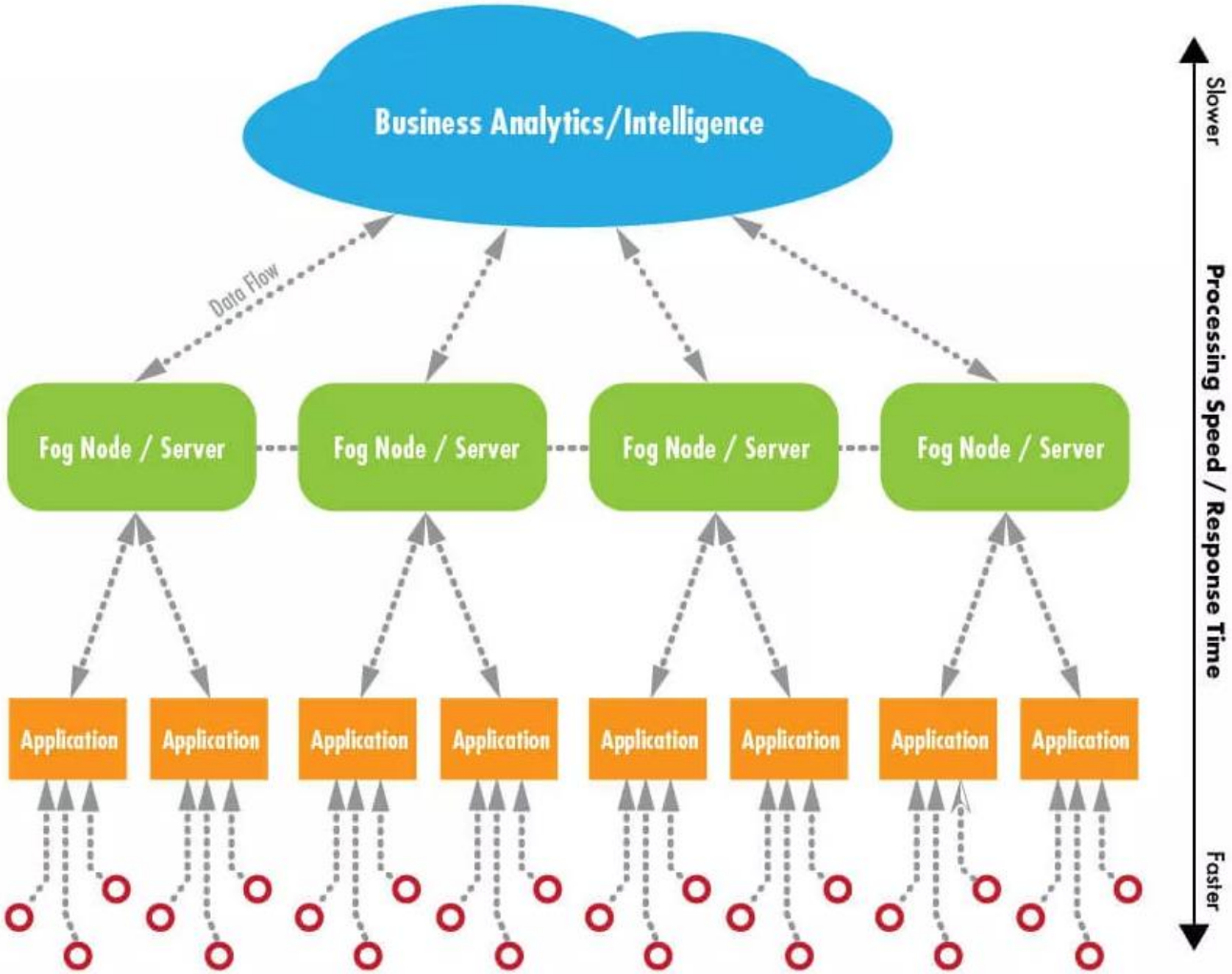
FOG LAYER

Local Network
Data Analysis & Reduction
Control Response
Virtualization/Standardization

EDGE LAYER

Large Volume Real-time Data Processing
At Source/On Premises Data Visualization
Industrial PCs
Embedded Systems
Gateways
Micro Data Storage

Sensors & Controllers (data origination)



OPENFOG CONSORTIUM

Established in November 2015 by ARM, Cisco, Dell, Intel, Microsoft and Princeton University

Published 162 page Fog Reference Architecture in November 2017

Aims to remove ”mandatory cloud connectivity” for IoT devices by emphasising information processing and intelligence at the logical edge

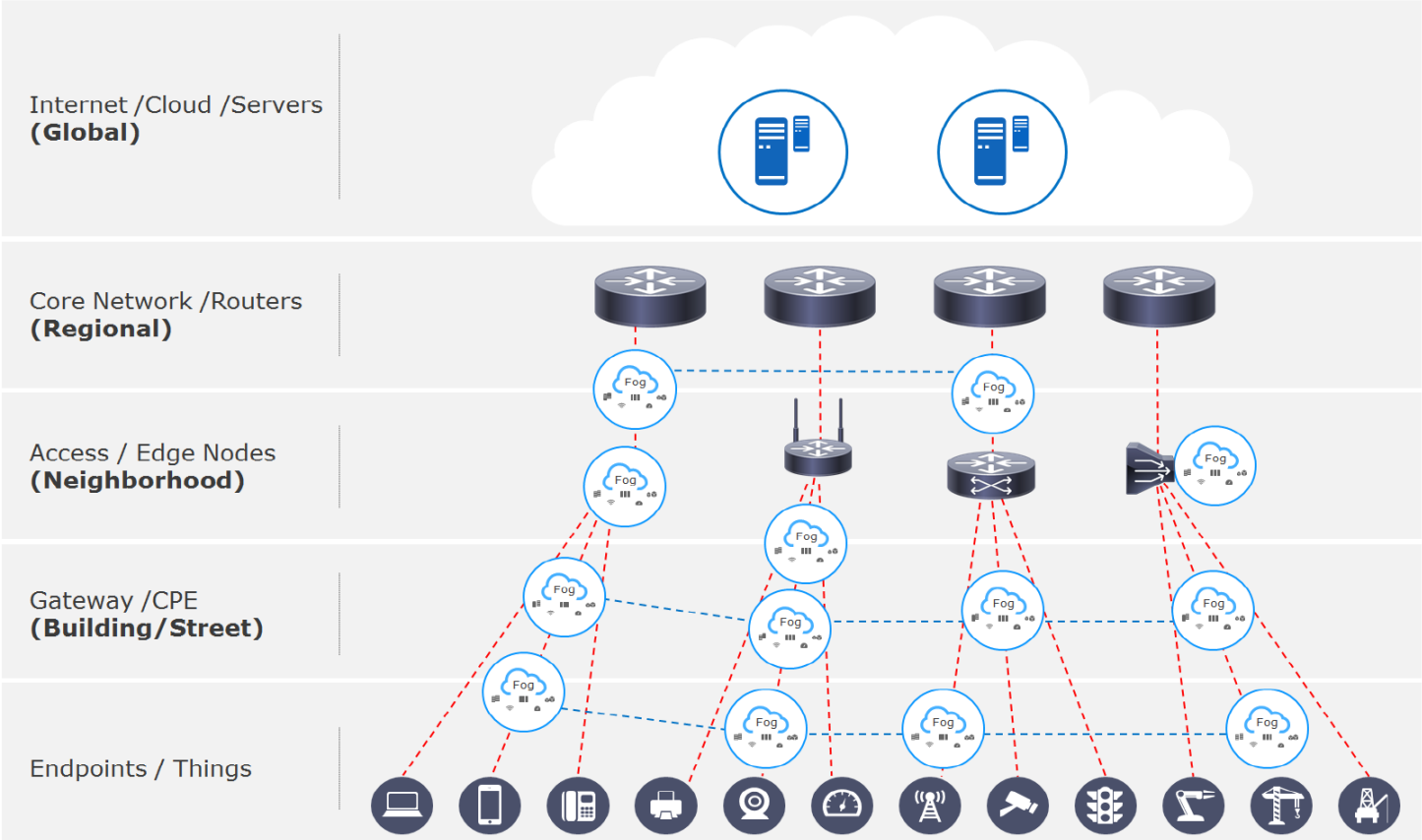
Made ”official” by IEEE Standards Association in 2018.

OPENFOG REFERENCE ARCHITECTURE

Form a mesh to provide load-balancing, resilience, fault-tolerance, and minimise communication.

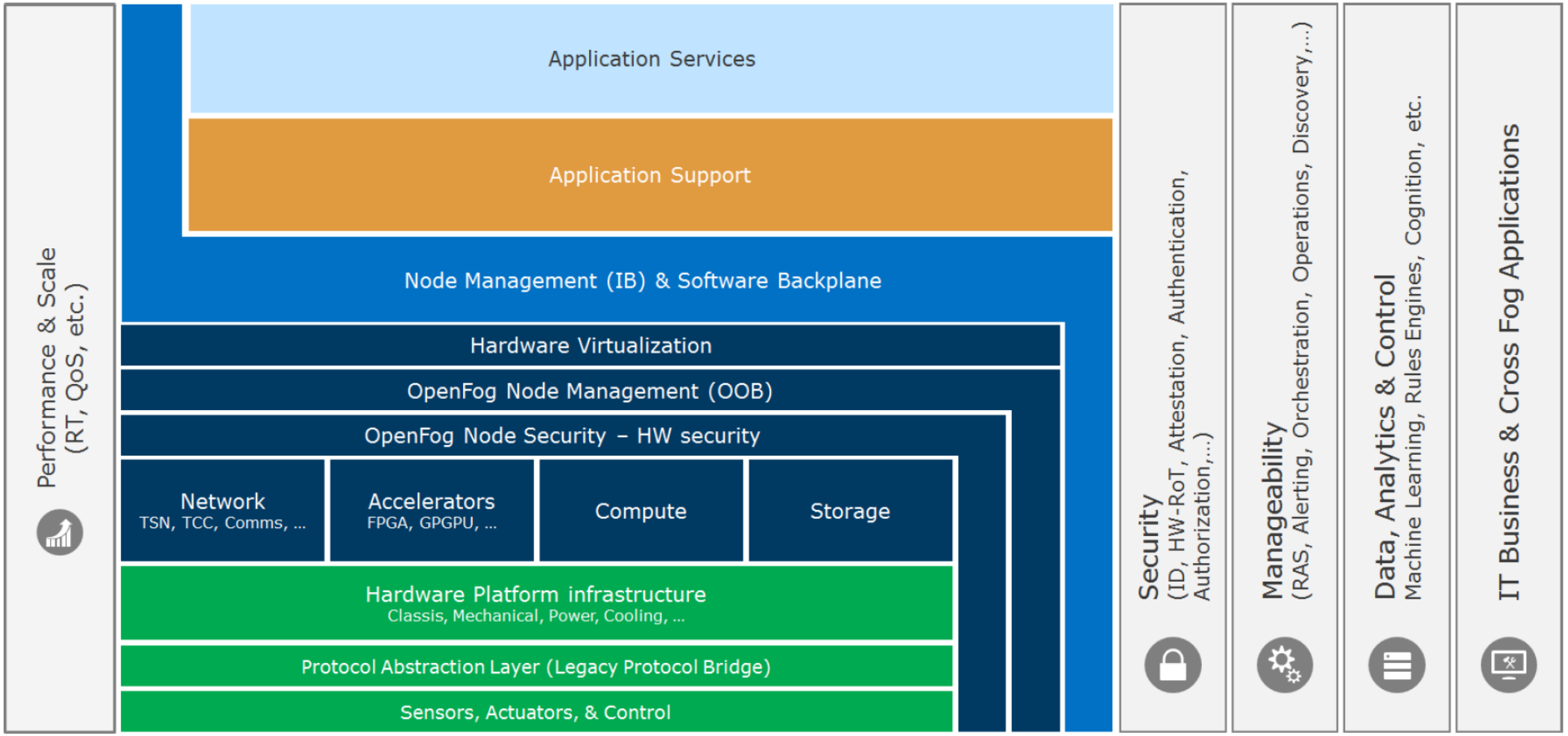
Communicate *laterally* and vertically

Able to discover, trust and utilise the services of other nodes to sustain relability, availability, serviceability.



Fog nodes in a Smart City: Buildings, neighborhoods & regions are connected to provide an infrastructure that may be optimized for service delivery.

OPENFOG RA “DESCRIPTION”



OPENFOG RA ARCHITECTURE “DESCRIPTION”

Performance

Low latency is a major driver. A cross-cutting concern, and involves time critical computing, time sensitive networking, network time protocols, etc.

Manageability

Managing all aspects of fog deployments is critical across all layers of the hierarchy

Security

End-to-end security is critical. Data integrity is a special aspect of security for devices that currently lack adequate security (including corruption).

Data analytics and control

For fog nodes to be autonomous, localised data analytics combined with control are essential. Control must happen at correct tier, which might not always be the physical edge but higher.

Edge / Fog / Cloud Comparison

EDGE VS FOG COMPARISON

Edge

- Distributed computing paradigm
- Computing and processing at Edge of network
- Closer to data sources
- Reduces volume of data
- Reduces distance data must be moved
- Ultimately, improves latency and reduceds pressure on central data centers

Fog

- Layered end-to-end architecture
- Is the integration between Cloud and Edge
- Distributed resources and services along a continuum from Cloud to things
- Solves problems that cannot be successfully implemented using soley Cloud or intelligent Edge devices

CLOUD VS FOG PERFORMANCE

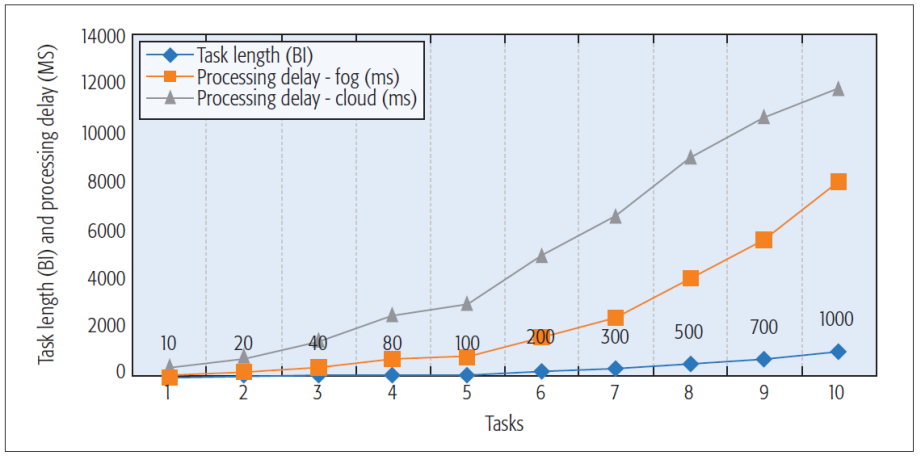


Figure 3. Processing delay with cloud and fog for different task lengths.

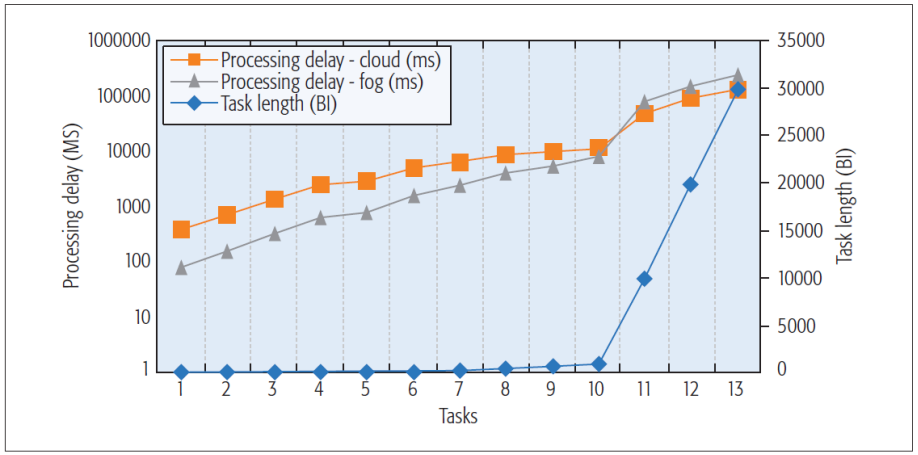


Figure 4. Delay comparison between cloud and fog for various task lengths.

M.Aazam, S.Zeadally, K.A. Harras. 2018. "Fog Computing Architecture, Evaluation, and Future Research Directions"

Parameter	Cloud	Fog
Proximity and geographic coverage	Global	Local (from a building to a city)
Distance between client and server node	Multiple hops	Typically, single-hop
Accessibility	Internet-only	Local area or Internet
Magnitude	Data center	From a single server to a micro-data center
Latency	High	Low
Target user	General Internet users	Mobile and resource constrained users
Resources	Practically unlimited	Limited storage, compute, and memory
Service scale	Global information	Customized, application- and user-oriented
Service type	Infrastructure as a service, platform as a service, software as a service	Software defined networking, network functions virtualization, network acceleration, content delivery, device management, data protection and security, complex event processing, task offloading
Geographical distribution and deployment	Centralized	Decentralized
Connectivity and communication	IP-based only	IP and non-IP
Communication overhead	Raw data (necessary and unnecessary communication through the core)	Refined and necessary communication through the core
Context awareness	Low	Medium to high
Location awareness	Low	High
Nodes	Servers	Routers, switches, gateways, servers, access points
Internode communication	Not supported	Supported
Access	Fixed and wireless	Mainly wireless (Bluetooth, ZigBee, WiFi, WiBro, 3G, 4G, LTE)
Main content generator	Human	Devices and sensors
Main content consumer	Devices such as smartphones, computers, servers	Any "thing"
Data storage	Days, years	Transient
Data and communication security	Undefined, depends on service	Adds additional layer of security before sending data to the core
Bandwidth required	High	Low
Mobility support	Limited	Full support
Price per server device	$1500–$3000	$50–$200 [15]
Operating expenses	High	Low

CLOUD VS FOG SUMMARY

More

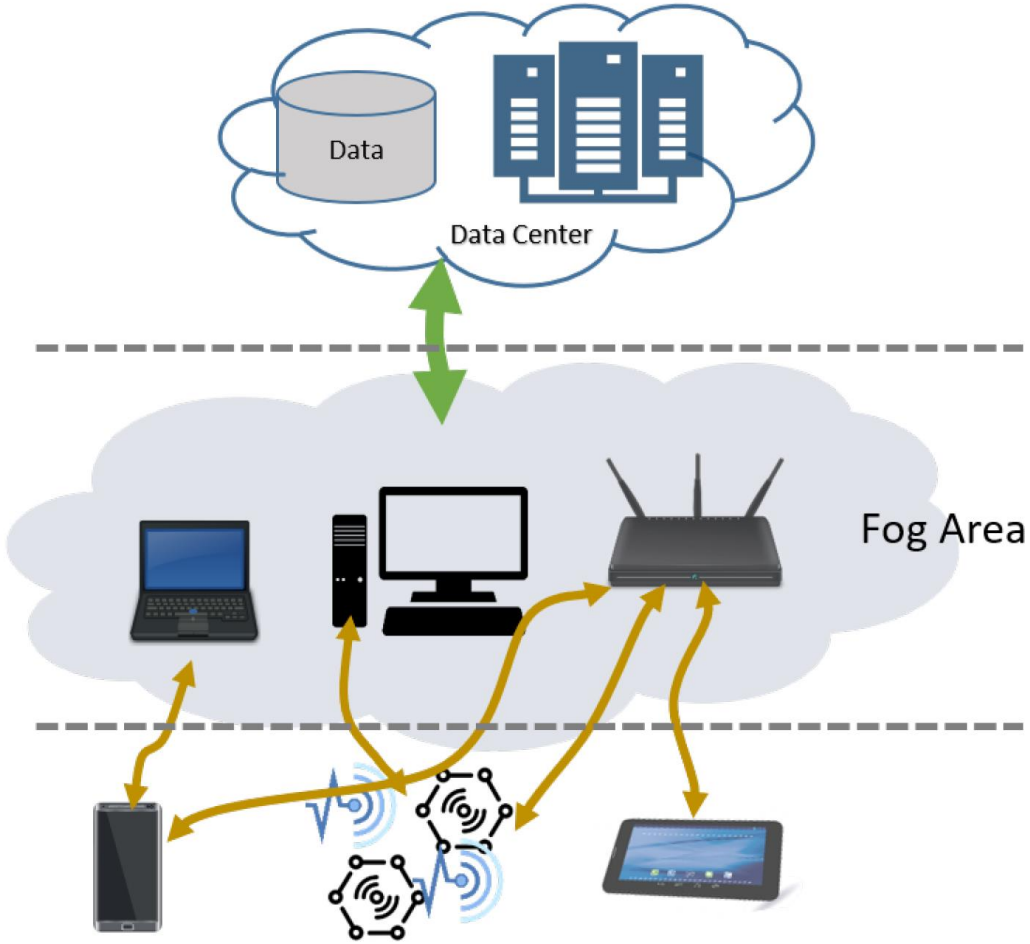
- ✓ Computing Power
- ✓ Reliability
- ✓ Data Storage
- ✓ Latency

Less

Cloud Layer
- Massive data processing,
- storage,
- mining,

Fog Layer
- Real- time data Storage,
- Processing,
- Analytics,
- Knowledge Discovery, …

End Devices Layer
- Data processing
- Human GUI

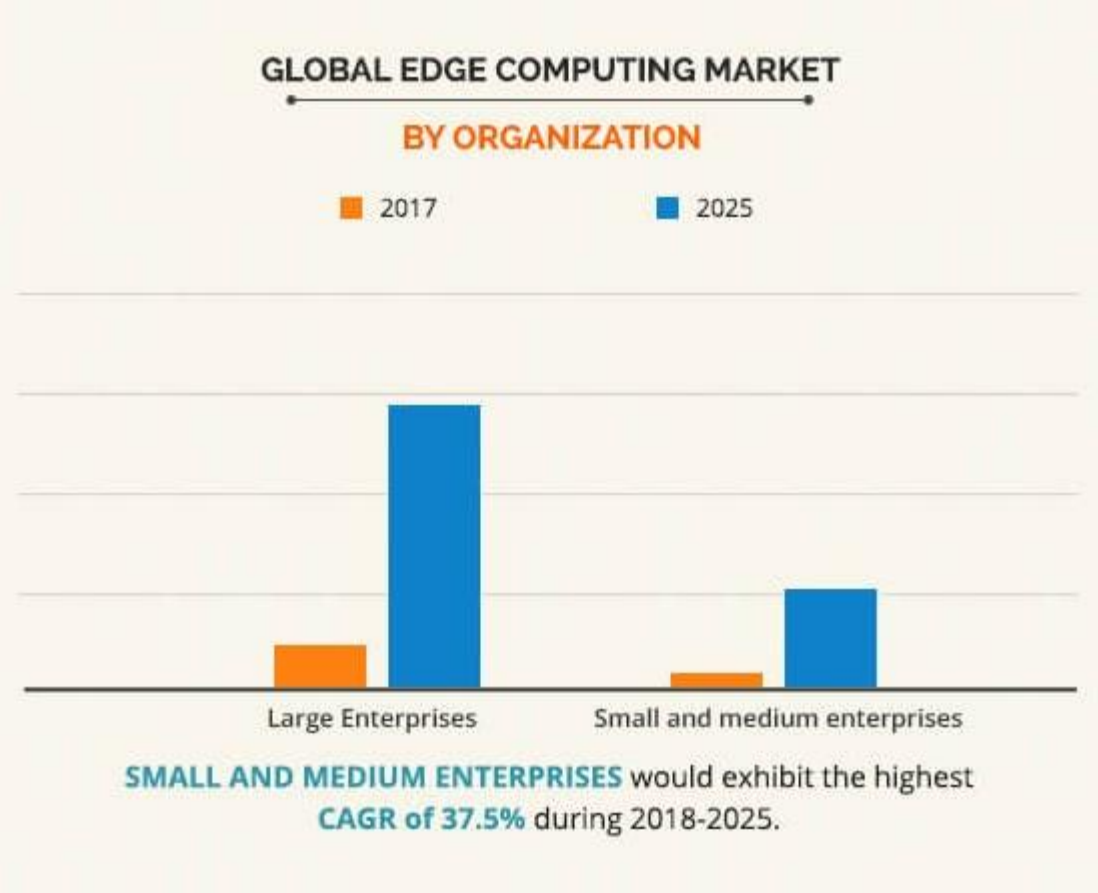


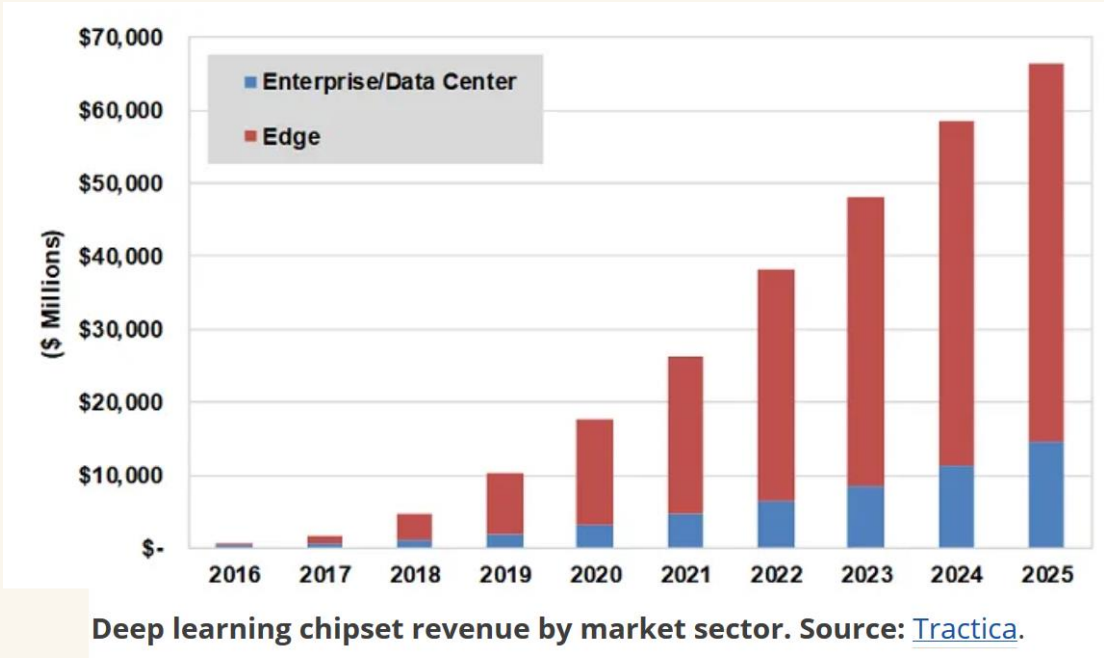
Less

- ✓ Location Awareness
- ✓ Interactive
- ✓ Mobility
- ✓ Devices

More

EDGE/FOG MARKET PREDICTIONS





Deep learning chipset revenue by market sector. Source: Tractica.

SOME RESEARCH CHALLENGES WITH EDGE/FOG

How many edge nodes do we need?	Where do we put them?	How do we discover Edge nodes?	How to hand-over workloads?
How do we manage multiple domains?	How often do we update topology?	How to detect Edge hardware resource?	How do we know workload types?
How to facilitate rapid migration?	How to load balance across Edge?	How to detect and mitigate failure?	What can be passed to the central DC?

Recap on Cloud Deployment

RECAP: TRADITIONAL CLOUD SERVICE MODELS

Software-as-a-Service (SaaS)

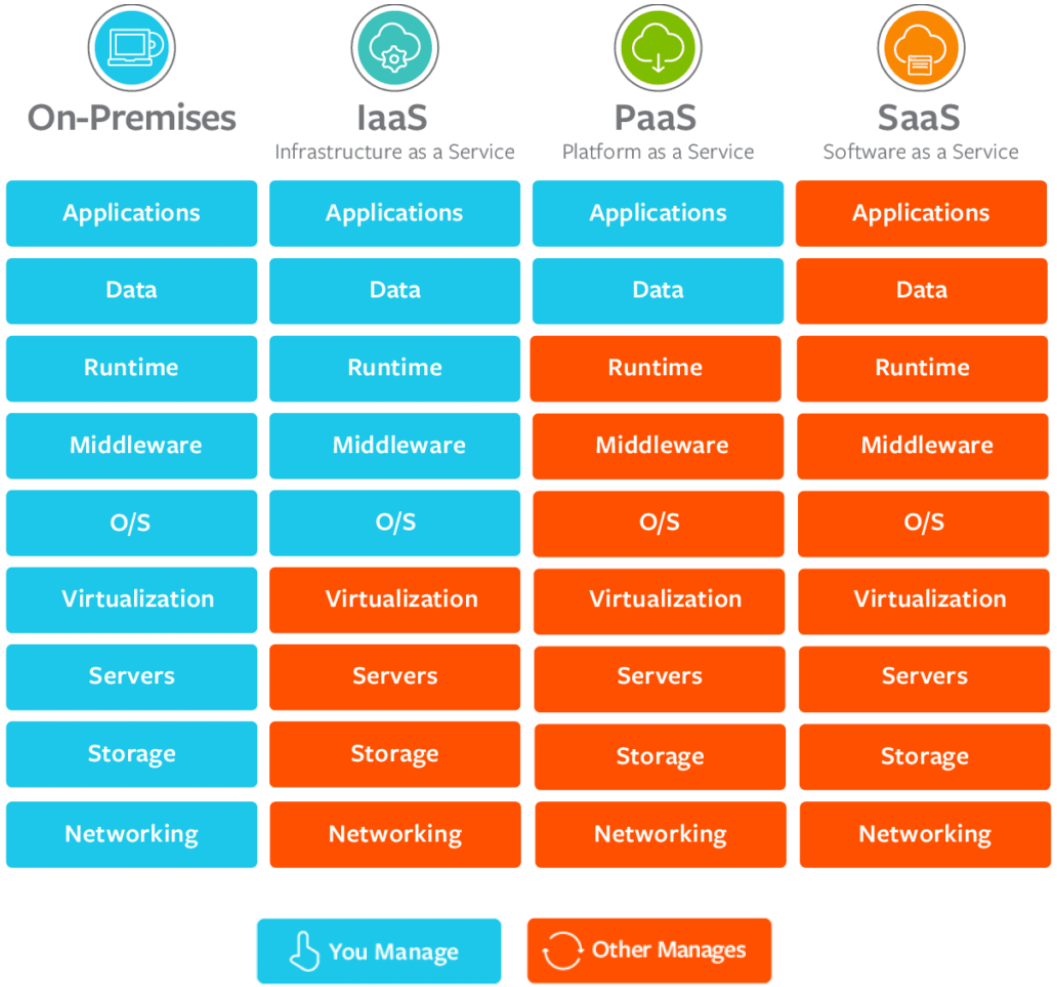
Ready-to-use applications
(O365, Spotify, Dropbox, etc)

Platform-as-a-Service (PaaS)

Ready-to-use platform
(Windows, Google Apps Engine, etc)

Infrastructure-as-a-Service (IaaS)

Full access to a hosted machine / VM
(Amazon EC2, Windows Azure, etc)



RECAP: FUNCTION-AS-A-SERVICE

Serverless – a new trend. Pay per request (no idle time)

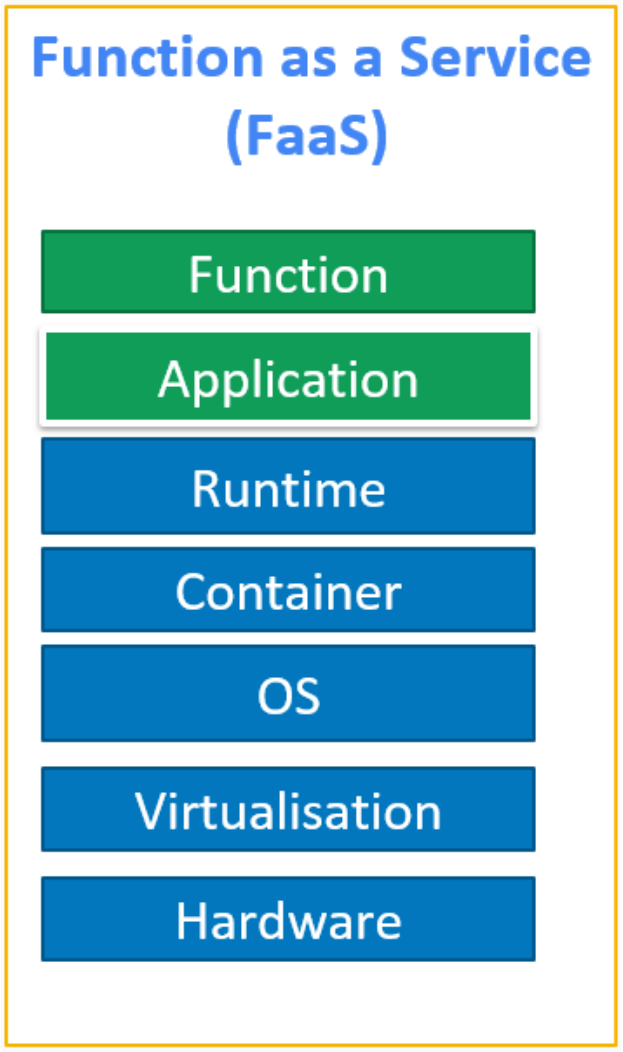
Auto-scaling and availability provided out-of-the-box

Computation is implemented as functions and execution is event driven

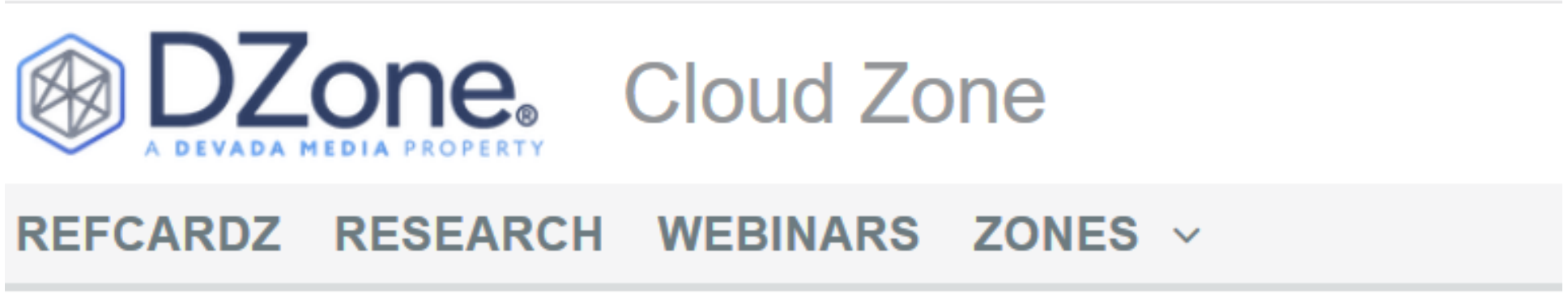
Customers define functions

Users select functions and specify the events triggering them

Examples include AWS Lambda, Google Cloud Functions, etc.



Serverless

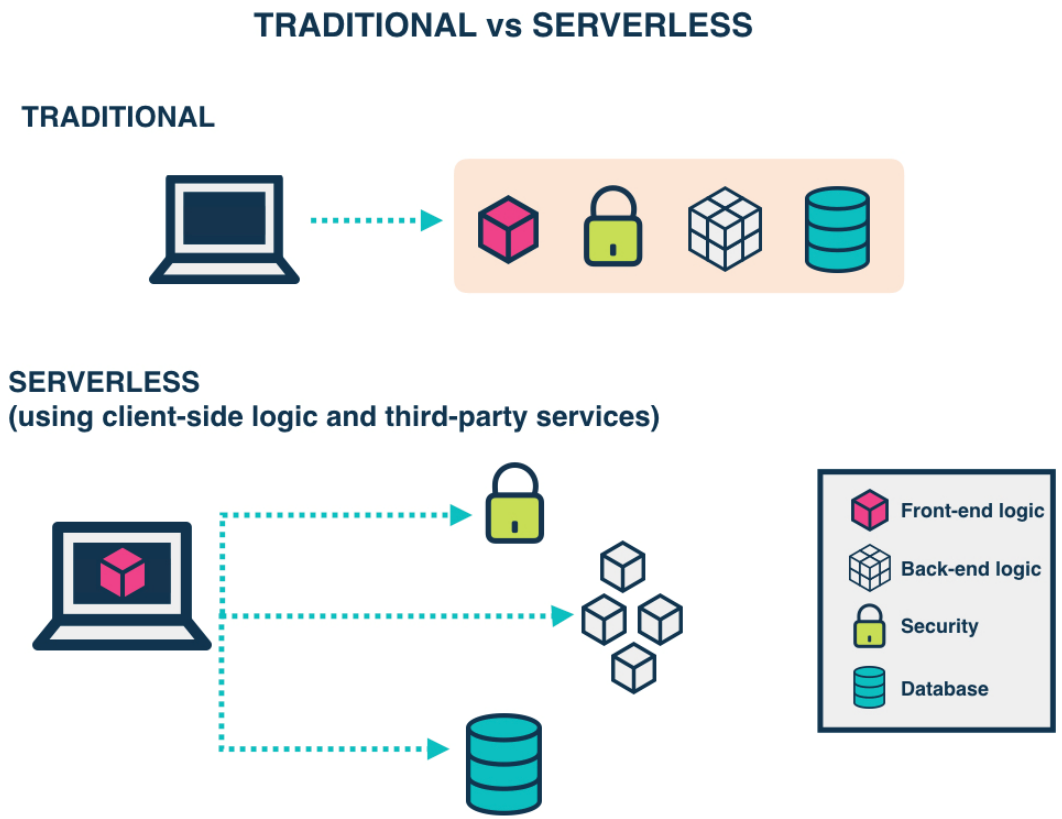


Serverless Is the Most Exciting Thing in Computing Right Now

https://dzone.com/articles/the-state-of-serverless-computing-2021

SERVERLESS ARCHITECTURES

- A “hot topic” in the data center industry
- Serverless DOES NOT mean no servers! (not P2P)
- Complex applications are built from simple functions
- No server management, including containers, VMs
- Stateless – results persisted to storage
- Flexible scaling
- Pay only for what resources you use
- Automated high availability



SERVERLESS VS "TRADITIONAL" CLOUD

	Characteristic	*AWS Serverless Cloud*	*AWS Serverful Cloud*
PROGRAMMER	When the program is run	On event selected by Cloud user	Continuously until explicitly stopped
	Programming Language	JavaScript, Python, Java, Go, C#, etc.[4]	Any
	Program State	Kept in storage (stateless)	Anywhere (stateful or stateless)
	Maximum Memory Size	0.125 - 3 GiB (Cloud user selects)	0.5 - 1952 GiB (Cloud user selects)
	Maximum Local Storage	0.5 GiB	0 - 3600 GiB (Cloud user selects)
	Maximum Run Time	900 seconds	None
	Minimum Accounting Unit	0.1 seconds	60 seconds
	Price per Accounting Unit	$0.0000002 (assuming 0.125 GiB)	$0.0000867 - $0.4080000
	Operating System & Libraries	Cloud provider selects[5]	Cloud user selects
SYSADMIN	Server Instance	Cloud provider selects	Cloud user selects
	Scaling[6]	Cloud provider responsible	Cloud user responsible
	Deployment	Cloud provider responsible	Cloud user responsible
	Fault Tolerance	Cloud provider responsible	Cloud user responsible
	Monitoring	Cloud provider responsible	Cloud user responsible
	Logging	Cloud provider responsible	Cloud user responsible

Eric Jonas et al.2019. Cloud Programming Simplified: A Berkeley View on Serverless Computing. 2020.

HOW DOES IT WORK?

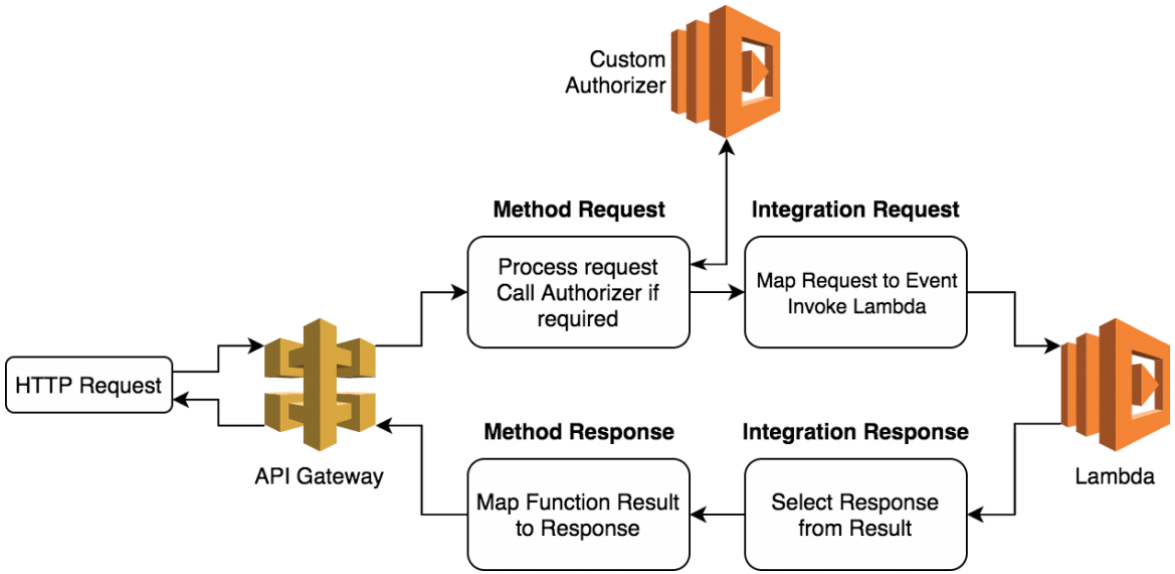
Developer writes/codes a function

Developer defines an ***event*** which will trigger the Cloud provider to execute the function

Event is triggered by a user (e.g. clicking a link, etc)

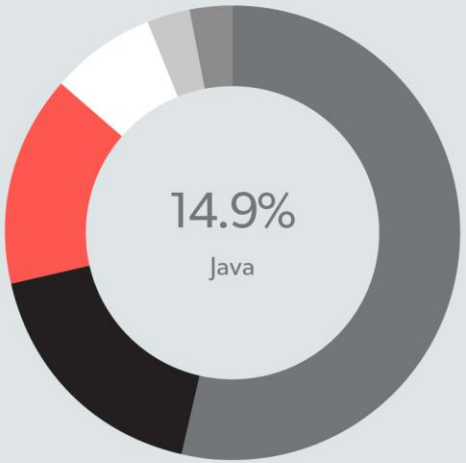
Function is executed by the Cloud. If the instance isn't running, it is started.

Result is sent to the Client



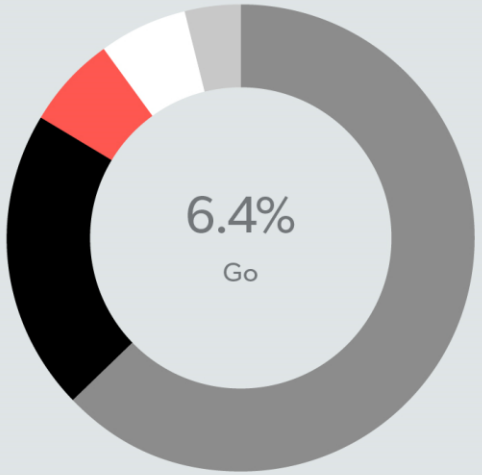
POPULAR LANGUAGES FOR CODING SERVERLESS

Languages used for serverless development in companies with >1000 employees



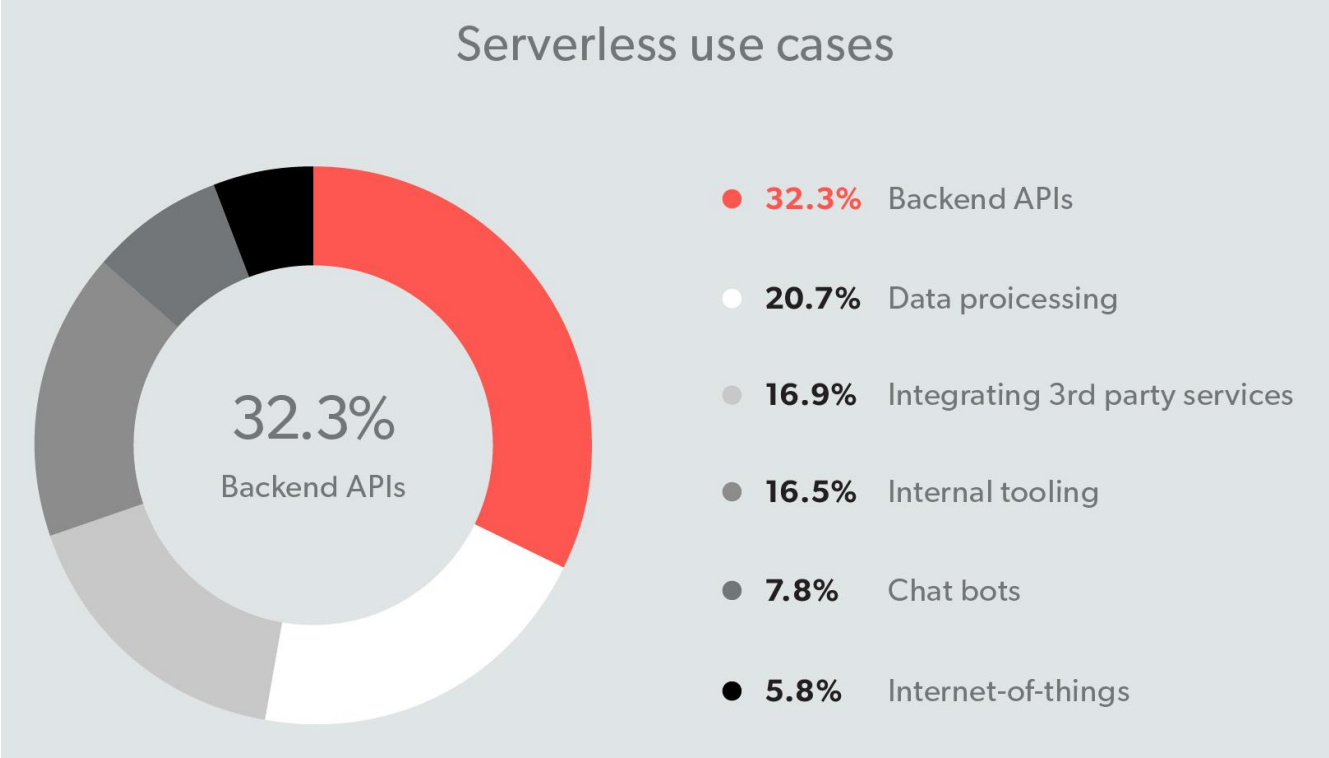
- **53.7%** Node.js
- **17.9%** Python
- **14.9%** Java
- **7.5%** Go
- **3.0%** C#
- **3.0%** Ruby

Languages used for serverless development



- **62.9%** Node.js
- **20.8%** Python
- **6.4%** Go
- **6.1%** Java
- **3.8%** C#

SERVERLESS USE CASES



Paul Castro, Vatche Ishakian, Vinod Muthusamy, Aleksander Slominski. 2019. The Rise of Serverless Computing. Communications of the ACM, December 2019, Vol. 62 No. 12, Pages 44-54, 2020

Table 2. Real-world applications that use serverless computing.

Where is serverless used?	What do they use serverless computing for?
Aegex	Xamarin application that customers can use to monitor real-time sensor data from IoT devices.[a]
Abilisense	Manages an IoT messaging platform for people with hearing difficulties. They estimated they could handle all the monthly load for less than $15 a month.[b]
A Cloud Guru	Uses functions to perform protected actions such as payment processing and triggering group emails. In 2017 they had around 200K users and estimated $0.14 to deliver video course to a user.[c]
Coca-Cola	Serverless Framework is a core component of The Coca-Cola Company's initiative to reduce IT operational costs and deploy services faster.[d] One particular use case is the use of serverless in their vending machine and loyalty program, which managed to have 65% cost savings at 30 million hits per month.[e]
Expedia	Expedia did "over 2.3 billion Lambda calls per month" back in December 2016. That number jumped 4.5 times year-over-year in 2017 (to 6.2 billion requests) and continues to rise in 2018.[f] Example applications include integration of events for their CI/CD platforms, infrastructure governance and autoscaling.[g]
Glucon	Serverless mobile backend to reduce client app code size and avoid disruptions.[h]
Heavywater Inc	Runs Website and training courses using serverless (majority of cost per user is not serverless but storage of video). Serverless reduced their costs by 70%.[i]
iRobot	Backend for iRobot products.[j]
Postlight	Mercury Web Parser is a new API from Postlight Labs that extracts meaningful content from Web pages. Serving 39 million requests for $370/month, or: How We Reduced Our Hosting Costs by Two Orders of Magnitude.[k]
PyWren	Map-reduce style framework for highly parallel analytics workloads.[l]
WeatherGods	A mobile weather app that uses serverless as backend.[m]
Santander Bank	Electronic check processing. Less than $2 to process all paper checks within a year.[n]
Financial Engines	Mathematical calculations for evaluation and optimization of investment portfolios. 94% savings on cost approximately 110K annually.[o]

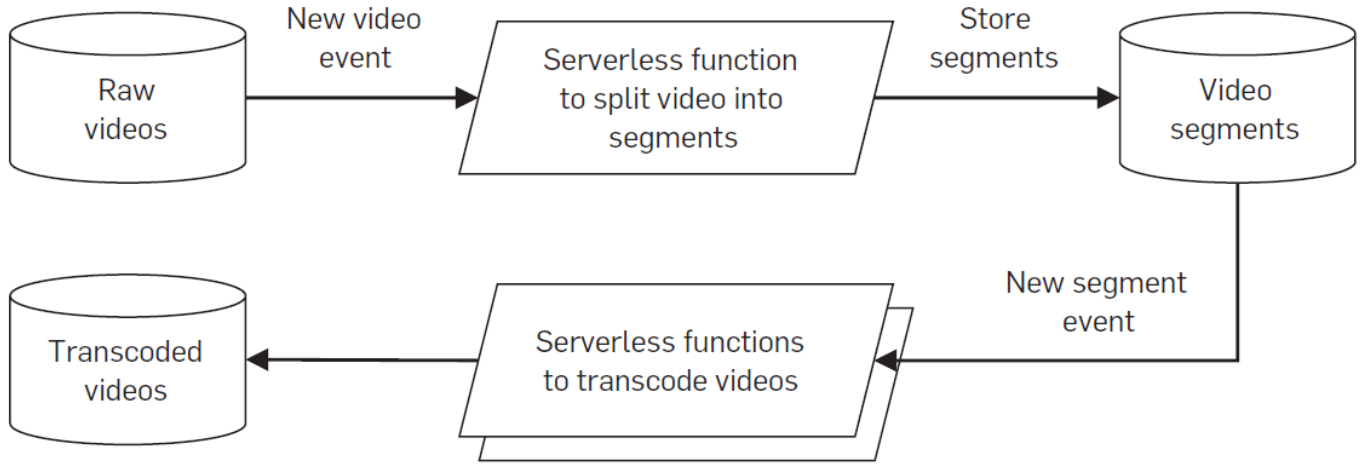
SERVERLESS USE CASE 1

Netflix uses serverless functions to process video files

Videos are uploaded to Amazon S3, which emits events that trigger the lambda functions

Functions are stateless and idempotent – good in case of failure, etc.

Video processing.



SERVERLESS USE CASE 2

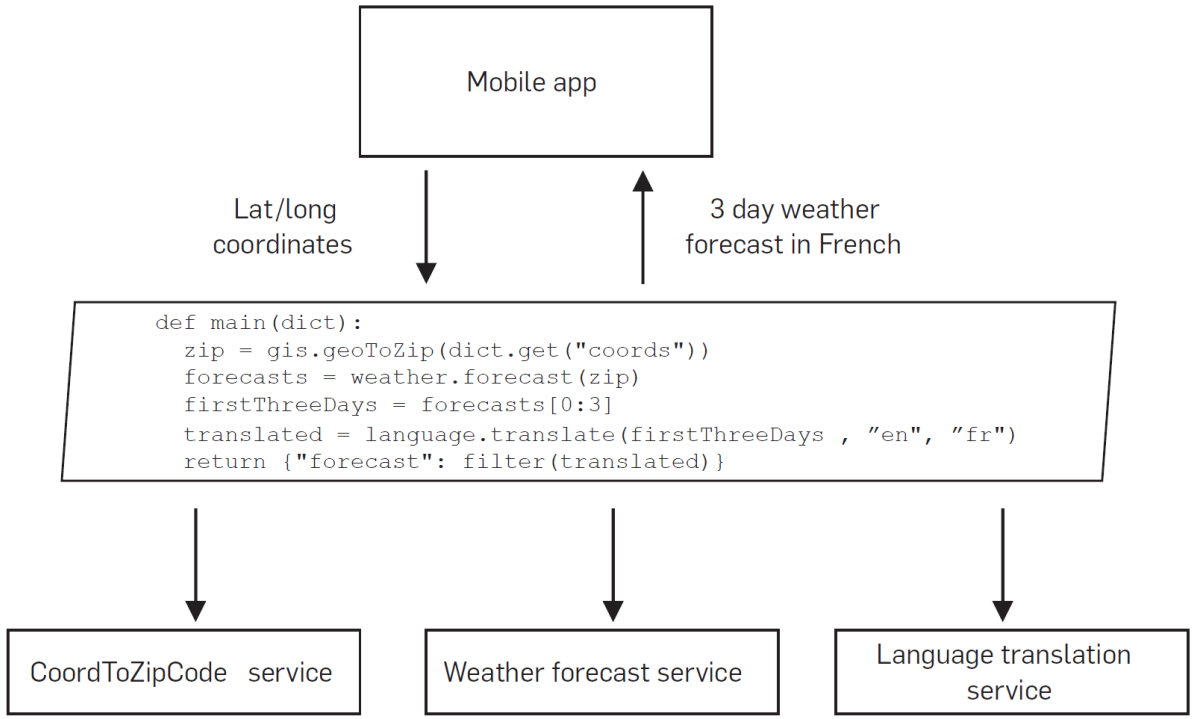
A mobile app providing a weather forecast based on current location

To avoid invoking multiple APIs over a resource constrained network, the app uses the main function as an orchestrator

This is an “anti-pattern” as the orchestrator costs money while waiting for results

Potential solution: chain functions using AWS Step Functions, IBM Composer, etc.

A serverless anti-pattern of offloading API calls from mobile app to backend.



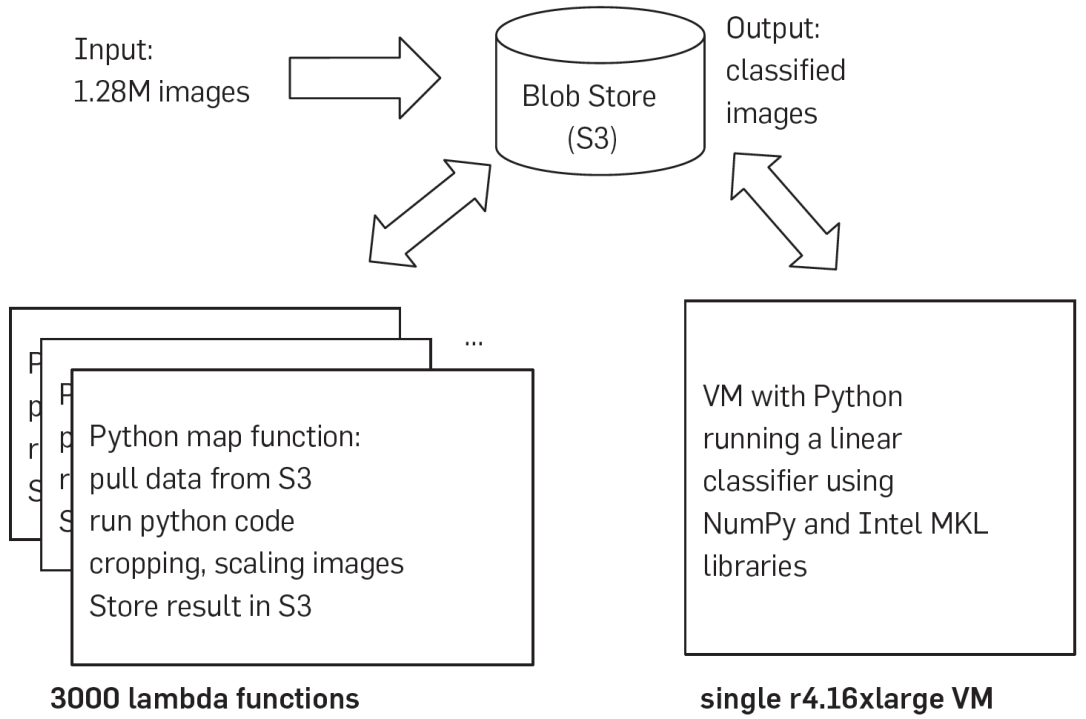
SERVERLESS USE CASE 3

PyWren uses serverless to reduce overhead on MapReduce jobs

Able to get up to 40 TFLOPS peak performance from AWS Lambda, using S3

Exemplifies a class of use cases that use serverless for highly parallel analytics.

Map + monolithic Reduce PyWren example implementing ImageNet Large Scale Visual Recognition Challenge.



3000 lambda functions

single r4.16xlarge VM

LIMITATIONS (1)

Inadequate storage for fine-grained operations

Cloud object stores (S3, Azure Blob, etc.) are highly scalable but have high access cost and latency.

Recent tests show all services take at least 10ms to read/write object. High throughput cost a lot (e.g. $30 per minute for 100K IOPS on S3)

Lack of fine-grained coordination

If Function B requires input from Function A, it needs to know when A has output available

Existing Cloud storage services do not come with notification capabilities (without significant latency and cost). Alternate methods need to be used.

LIMITATIONS (2)

Standard communication patterns have poor performance

Unpredictable performance

Common patterns include shuffle, aggregation, broadcast, etc. (especially in ML workloads)

Serverless functions have lower latency than VM-based instances, but have high start-up times.

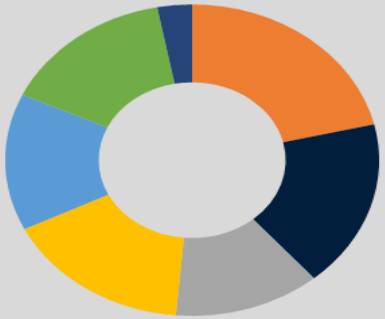
In a VM, local aggregation between tasks is relatively easy. In serverless, much more messaging required – 2 to 4 orders of magnitude more.

Must initialise the software environment of the function, as well as application-specific initialisation code.

LIMITATIONS (3)

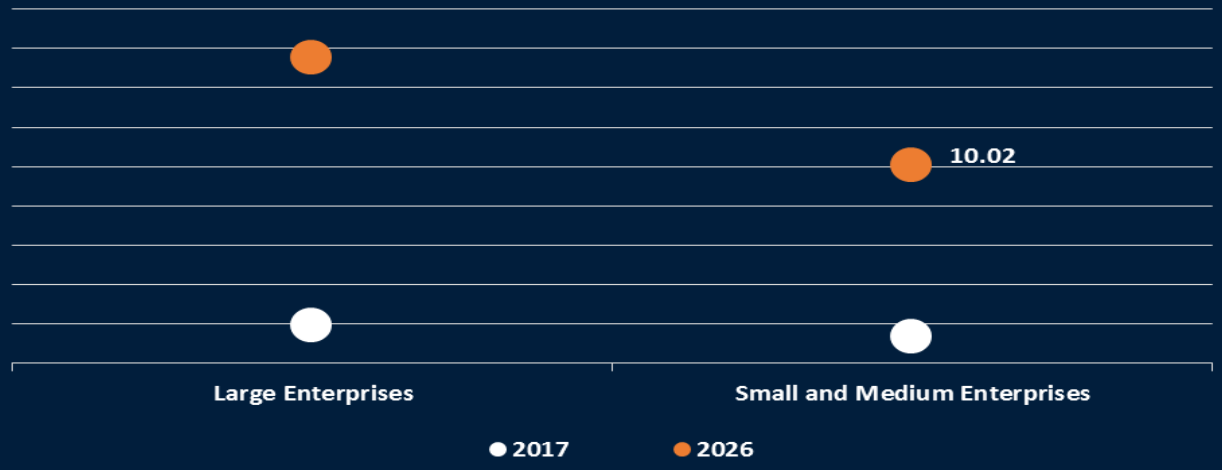
Execution duration	State
Not really appropriate for long-lived tasks	Stateless components must interact with stateful components to persist information. This introduces latency and complexity.
Google Cloud Functions: maximum 9 minutes AWS Lambda: maximum 15 minutes	Some serverless platforms **do** preserve some state between function calls (for optimisation) which can confuse the operational picture of a system.

2019-2026,
the market will
ACCELERATE
at a CAGR of
24.7%

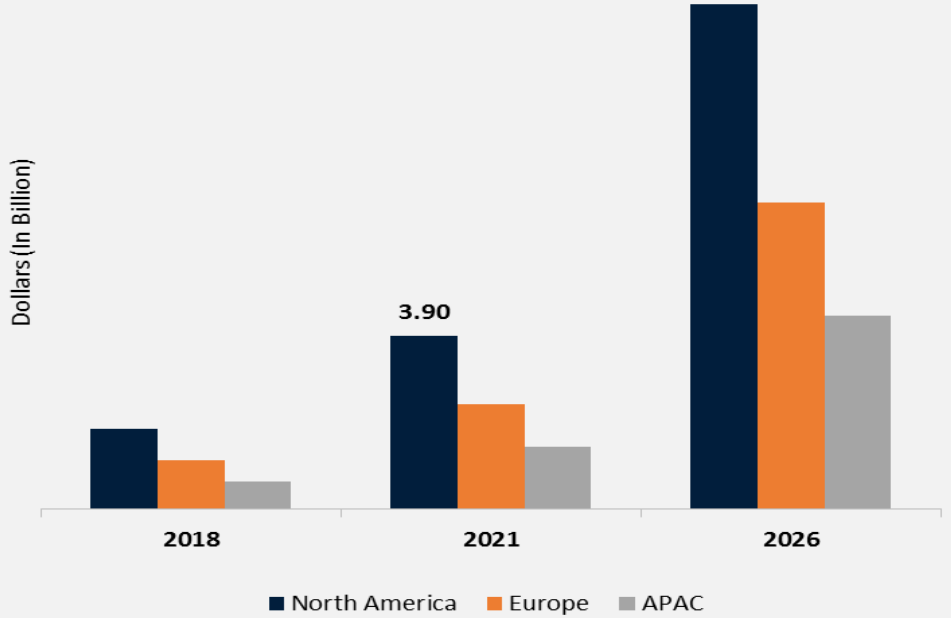


21.3%
of the total serverless architecture market is occupied by the Automation & Integration segment.

The rise in awareness regarding the benefits of serverless architecture, such as increased process agility and reduced operational cost, are fueling the growth of the market.



- Large enterprises are adopting serverless architecture for critical tasks, such as essential web applications and data processing.
- Small and medium enterprises are forecasted to grow with a CAGR of 24.1% during the forecast period owing to the costs and reduced infrastructure advantage provided by the serverless architecture.



High adoption of cloud infrastructure solutions and increasing penetration of IoT devices are fuelling the demand for serverless architecture in North America.

Europe is forecasted to hold a market share of 27.1% in the year 2026. The region is witnessing high growth owing to the rise in cloud computing technologies.

Asia Pacific is forecasted to have a CAGR of 26.4% during the forecast period. Sectors such as IT, manufacturing, BFSI, and retail are rapidly shifting towards automation.

REPORTS AND DATA

WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM

Serverless Economy

SERVERLESS: MAJOR PLAYERS

The usual suspects….

Amazon Lambda / S3

IBM Cloud Functions

Microsoft Azure Functions

Google Cloud Functions

And others…

Platform9 Systems Fission

Joyent (Samsung) Manta

Syncano

GitHub Lever OS

Iron.io

Nstack

Serverless Framework

etc.

SERVERLESS: ECONOMICS

Table 1: Comparing hostings price for one hour of operation, assuming 200 ms of runtime, executing every five minutes.

Service instance	Billable unit	Unit cost (USD)	Fail-over costs (%)	Cost of 12 x 200ms exec'ns	% reference price
Lambda (128 MB)	100 ms	$0.000000208	included	$0.000004992	24.94%
Lambda (512 MB)	100 ms	$0.000000834	included	$0.000020016	100.00%
Heroku Hobby (512 MB)	1 month	$7.00	100%	$0.0097222222	48572.25%
AWS EC2 t2.nano (512 MB)	1 hour	$0.0059	100%	$0.0118	58952.84%
AppEngine B1 (128MB)	1 hour	$0.05	100%	$0.1	499600.32%
AppEngine B4 (512MB)	1 hour	$0.20	100%	$0.4	1998401.28%

For a service task of 200ms to execute:

Standard VMs billable for a unit hour

FaaS only billable for a 100ms unit

This represents a cost reduction of over 99.8%

So should we always use serverless to save money?

Source: Gojko Adzic, Robert Chatley Serverless Computing: Economic and Architectural Impact

SERVERLESS ECONOMICS

Heavily dependant on execution and volume

Major players charge on 3 variables:

Duration of code execution

Resources assigned to that code during execution

Number of times the code is executed

Prices shift, and difficult to predict OpEx. Any other problems?

http://serverlesscalc.com/

SOME RESEARCH CHALLENGES WITH SERVERLESS

How to reduce *cold start* latency	How to design past limited function runtimes	How to deal with charging for I/O waiting	How to ease function management?
How to migrate to Serverless from a traditional architecture	How to migrate between vendors without lock-in	How to predict OpEx with unpredictable workloads (IoT etc)	Can Serverless work with long-running Edge AI tasks?
How to integrate serverless with location-aware Edge?	How to reduce communication complexity?	How to handle fault-tolerance effectively	GPU support? Simulation?

ENOUGH!

Hope you enjoyed (or at least learned something on) day one!

Day two will look at:

Cloud Orchestration
Keynote from Ericsson Research
Learn how to use the ER DC
Cloud Economics
(extra special) Keynote from Google
Plus assignment details

Always feel free to email me: paul.townend@umu.se