2019

NT KONFERENCA

21. - 23. MAJ 2019

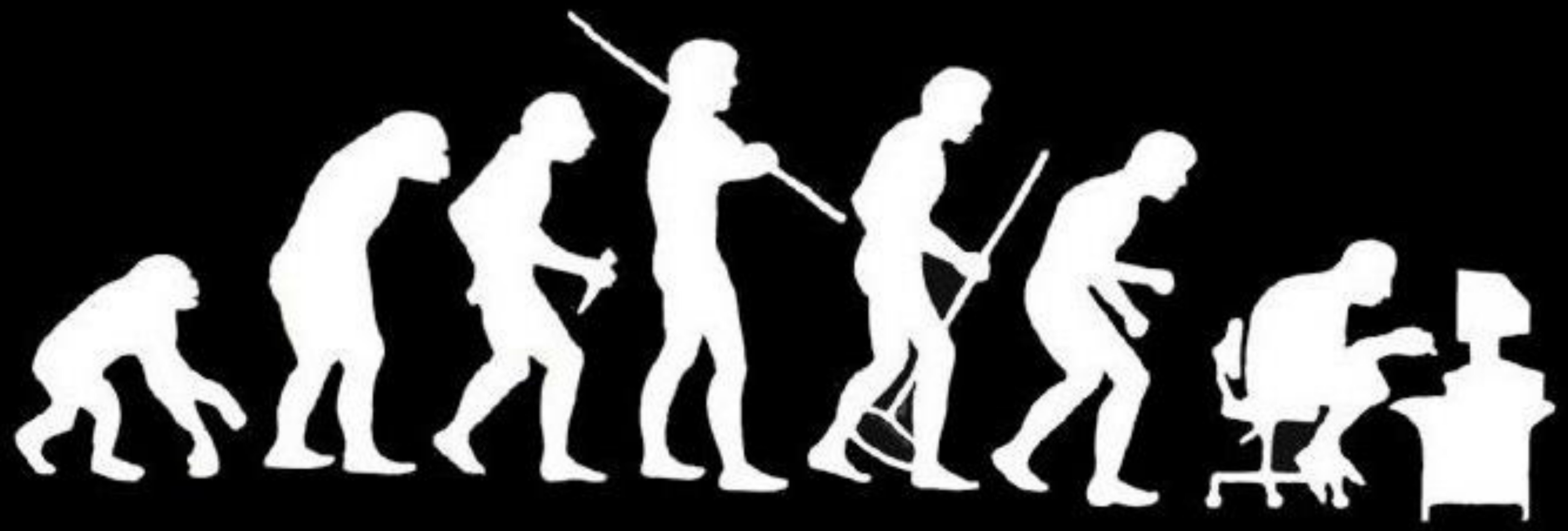
#ntk19

Kaj imajo migracija, modernizacija in Serverless skupnega z DevOps?

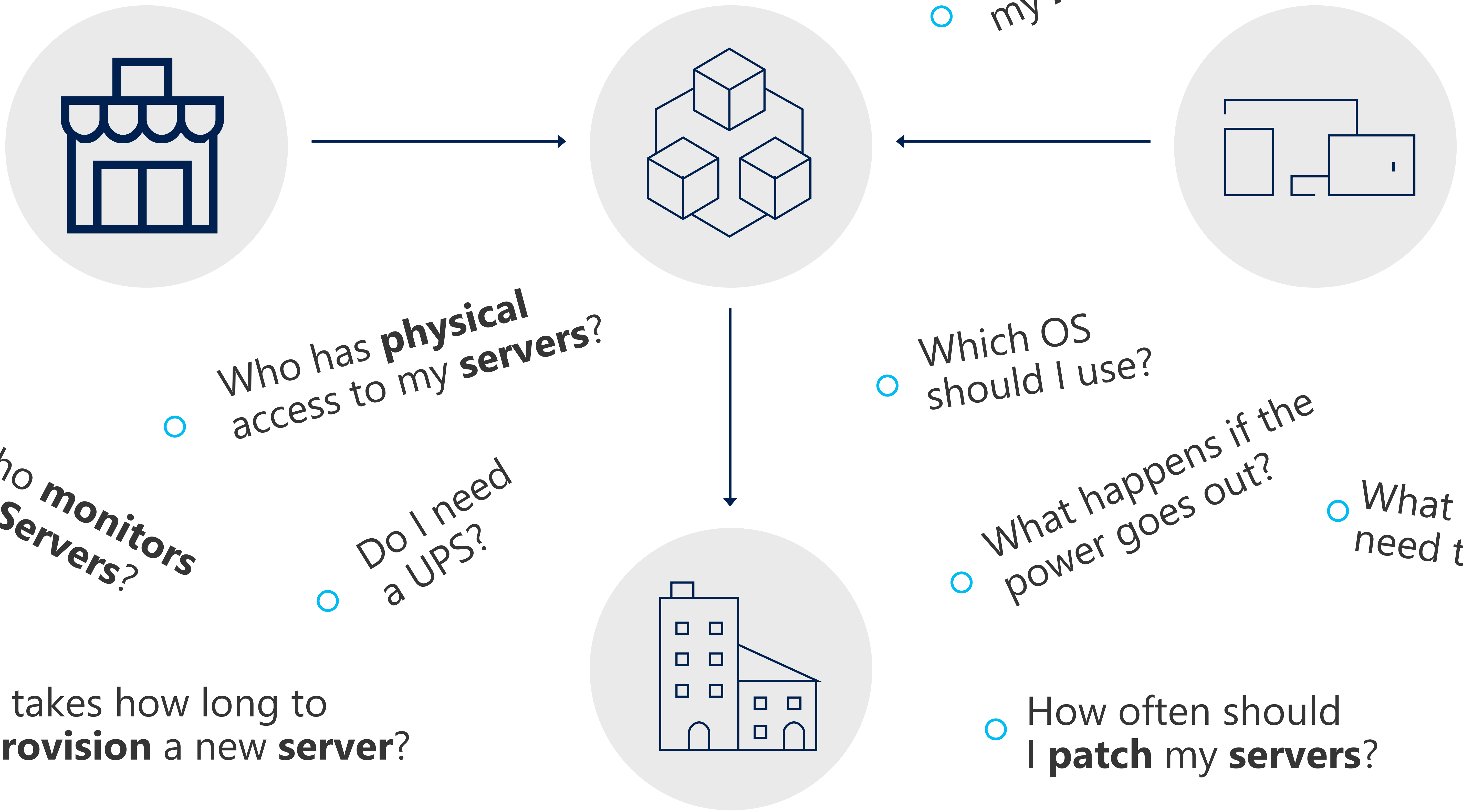
Uroš Kastelic
Technology Solutions Professional - Azure AppDev

#ntk19

The “evolution” of application platforms



- What media should I use to keep backup?
- What is the right **size** of **servers** for my business needs?
- How do I **deploy** new **code** to my **server**?
- What happens in case of **server hardware** failure?
- Which packages should be on my **server**?
- What size of **servers** should I **buy**?
- Who **monitors** my **App**?
- How often should I backup my **server**?
- How can I **scale** my app?



- How can I increase **server** utilization?
- Who has **physical** access to my **servers**?
- Which OS should I use?
- Are my **server** in a secure location?
- Do I need secondary network connection?
- Who **monitors** my **Servers**?
- Do I need a UPS?
- What happens if the power goes out?
- What storage I need to use?
- How many **servers** do I need?
- It takes how long to **provision** a new **server**?
- How often should I **patch** my **servers**?
- How can I dynamically configure my app?

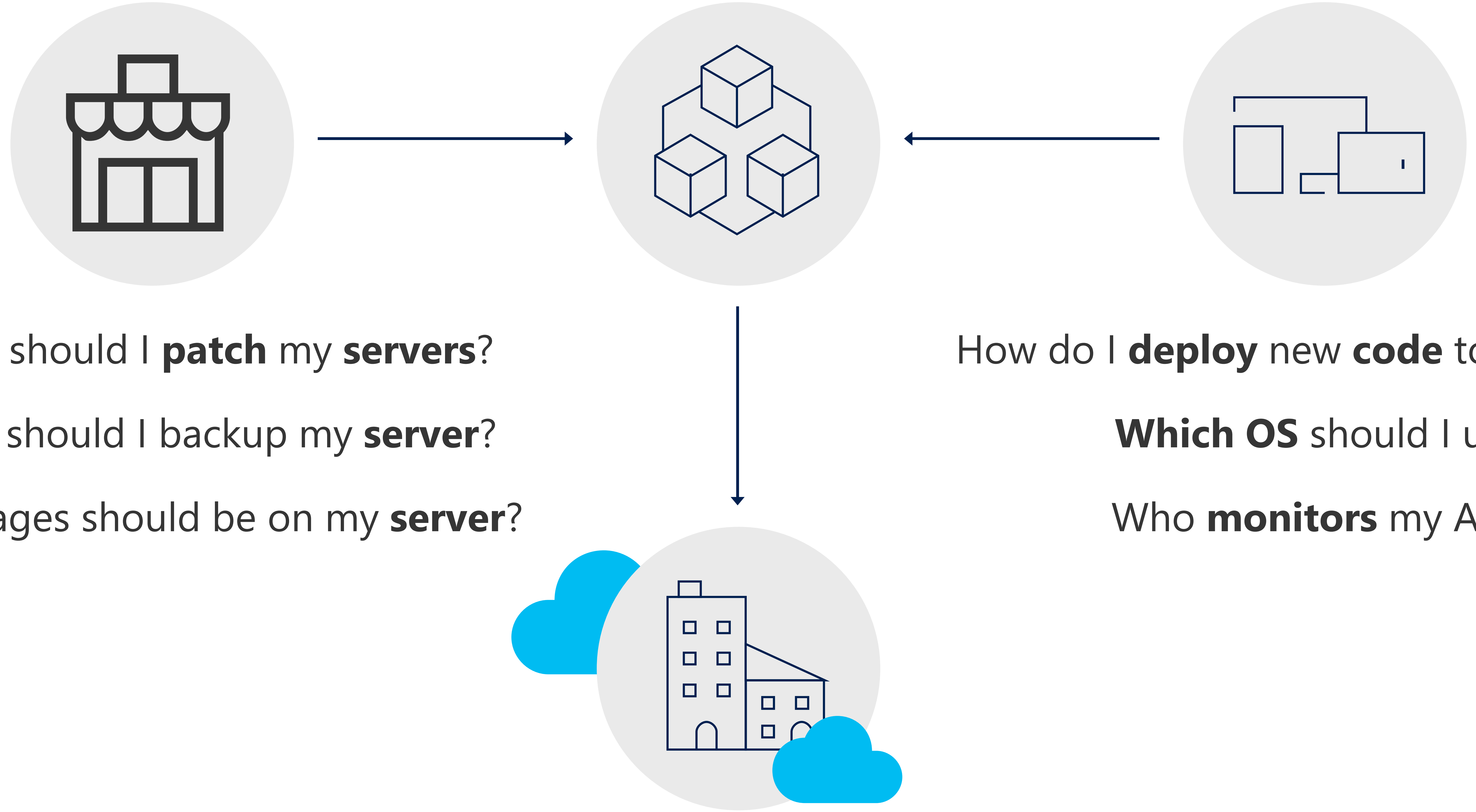
On-Premises

What is the right **size** of **servers** for my business needs?

How can I increase **server** utilization?

How many **servers** do I need?

How can I **scale** my app?



How often should I **patch** my **servers**?

How often should I backup my **server**?

Which packages should be on my **server**?

How do I **deploy** new **code** to my **server**?

Which OS should I use?

Who **monitors** my App?

On-Premises

IaaS

#ntk19

What is the right **size** of **“servers”** for my business needs?

How can I increase **“server”** utilization?

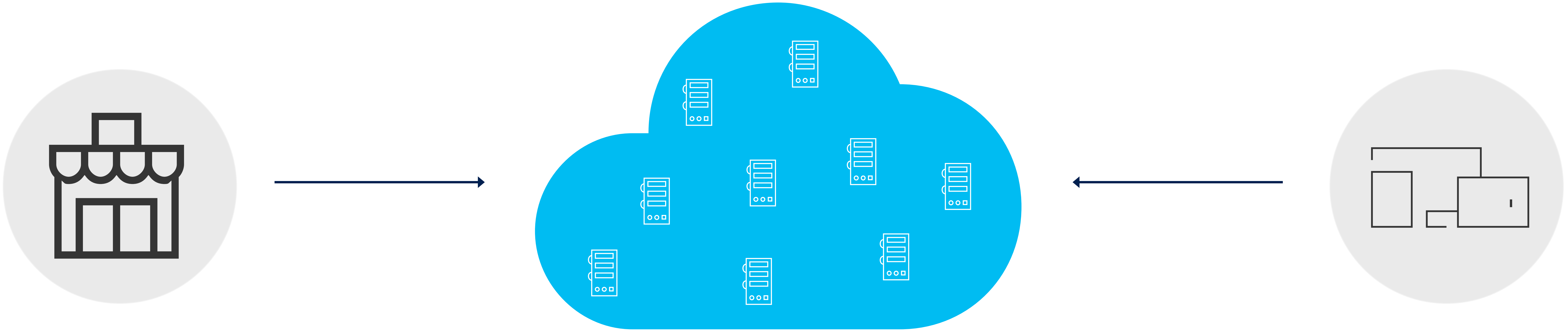
How many **“servers”** do I need?

How can I **scale** my app?

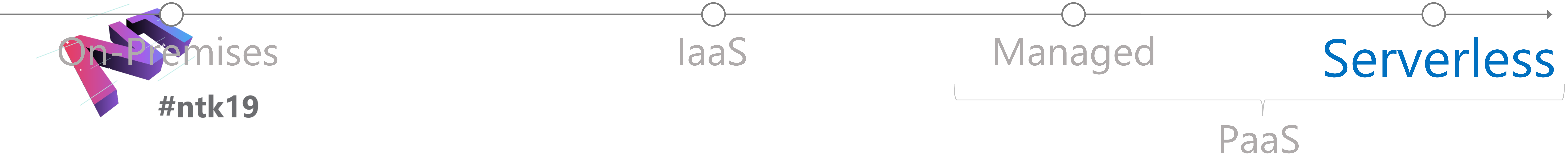
On-Premises
IaaS
Managed

#ntk19

How do I **architect** my app?



Serverless, the architecture for next gen apps



On-Premises

IaaS

Managed

Serverless

PaaS

Challenges

PaaS vs Serverless

Scalability

Ability to scale automatically, without extra configuration from

Pricing

Pay per consumption, when needed.

Launch time

Cold start, fast launch

Deployment

Deploy to edge devices if needed.

What is serverless?

Full abstraction of servers

Developers can just focus on their code—there are no distractions around server management, capacity planning, or availability.

Instant, event-driven scalability

Application components react to events and triggers in near real-time with virtually unlimited scalability; compute resources are used as needed.

Pay-per-use

Only pay for what you use: billing is typically calculated on the number of function calls,
code execution time, and memory used.*

*Supporting services, like storage and networking, may be charged separately.

What are the benefits?

Focus

Solve business problems—not technology problems related to undifferentiated heavy lifting

Efficiency

- Shorter time to market
- Fixed costs converted to variable costs
- Better service stability
- Better development and testing management
- Less waste

Flexibility

- Simplified starting experience
- Easier pivoting means more flexibility
- Easier experimentation
- Scale at your pace—don't bet the farm on Day 1
- Natural fit for microservices

Azure serverless application platform

Development

- IDE support
- Integrated DevOps
- Local development
- Monitoring
- Visual debug history

Platform

Functions

Execute your code based on events you specify

Event Grid

Manage all events that can trigger code or logic

Logic Apps

Design workflows and orchestrate processes

Database

Storage

Analytics

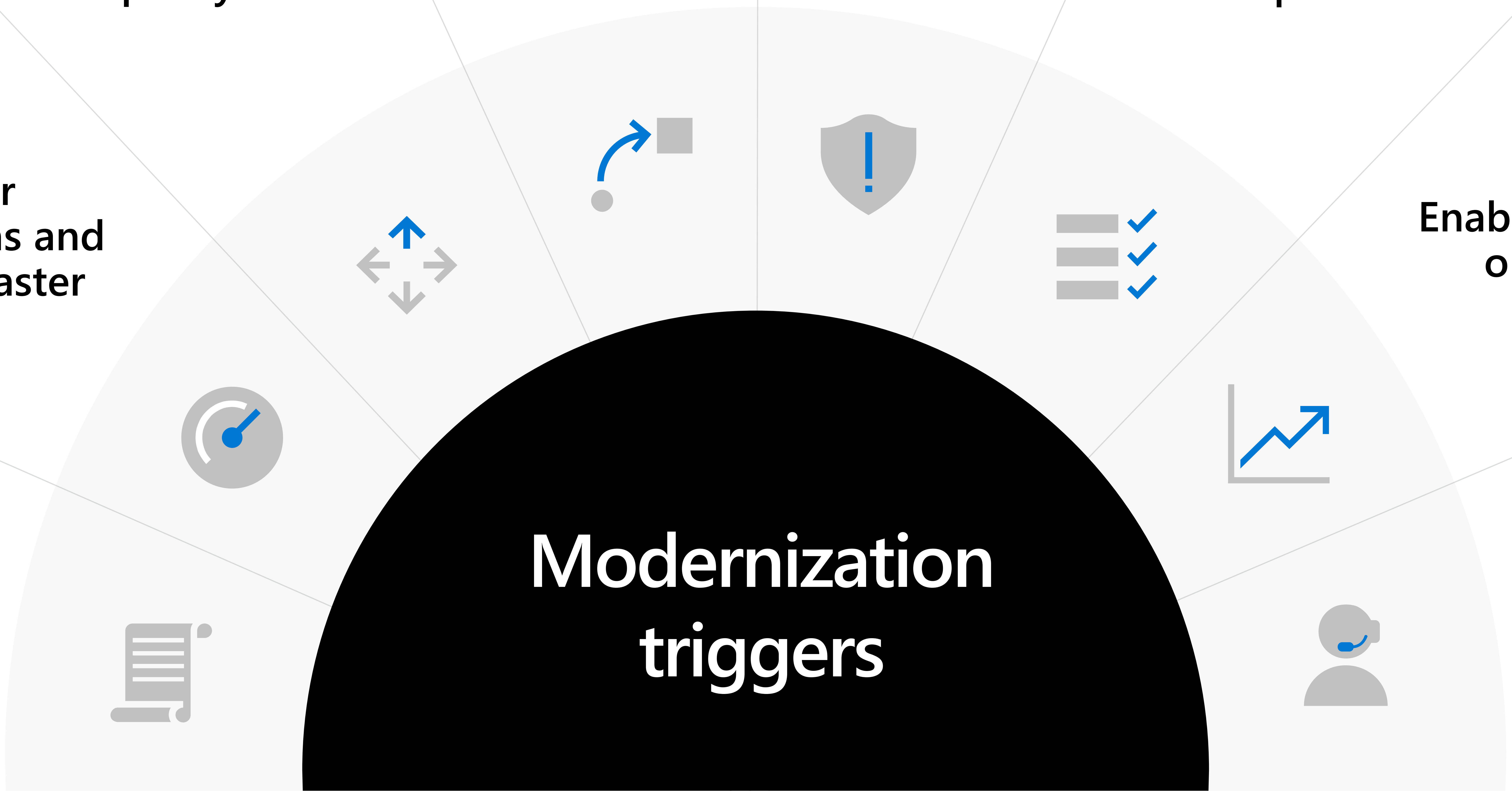
Intelligence

Security

IoT

#ntk19

I understand Serveless, but why you mention migration and modernization…?



Modernization triggers

Datacenter contracts expiring

Deliver applications and features faster

Urgent capacity needs

Software or hardware refresh

Address security threats

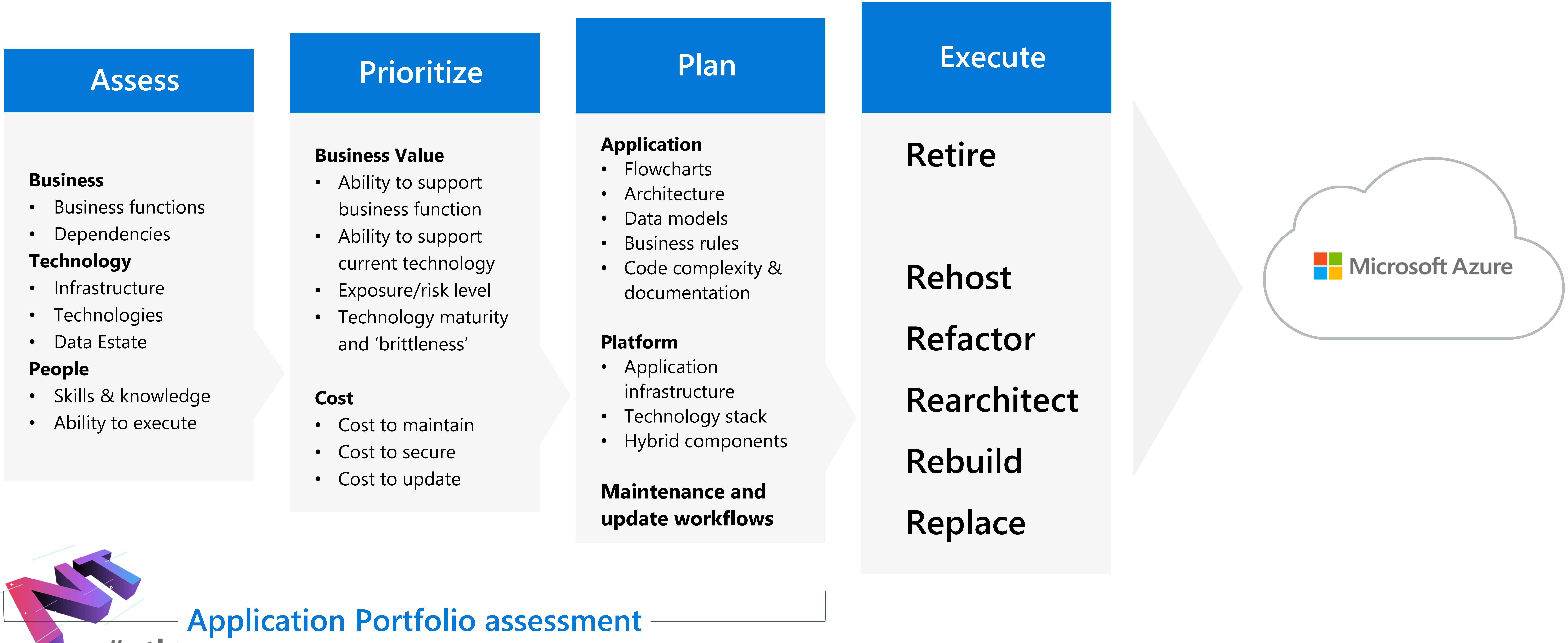
Compliance

Enable new business opportunities

Software end of support

Application portfolio assessment

Creating a migration and modernization roadmap



Assess

Business
- Business functions
- Dependencies

Technology
- Infrastructure
- Technologies
- Data Estate

People
- Skills & knowledge
- Ability to execute

Prioritize

Business Value
- Ability to support business function
- Ability to support current technology
- Exposure/risk level
- Technology maturity and 'brittleness'

Cost
- Cost to maintain
- Cost to secure
- Cost to update

Plan

Application
- Flowcharts
- Architecture
- Data models
- Business rules
- Code complexity & documentation

Platform
- Application infrastructure
- Technology stack
- Hybrid components

Maintenance and update workflows

Execute

Retire

Rehost

Refactor

Rearchitect

Rebuild

Replace

Application Portfolio assessment

#ntk'19

Customer cloud journey

On-premises

App

Data

Infrastructure

Migration & Modernization

Rehost

Refactor

Rearchitect

Cloud-Native

Rebuild/New

SaaS

Replace

Virtual Machines

Containers

Platform & Data Services

On-Premises

Cloud

DevOps

Oh, ok, so here comes DevOps....

A PaaS and Serverless platform for Application Modernization

Web & Mobile development

Containers

Microservices

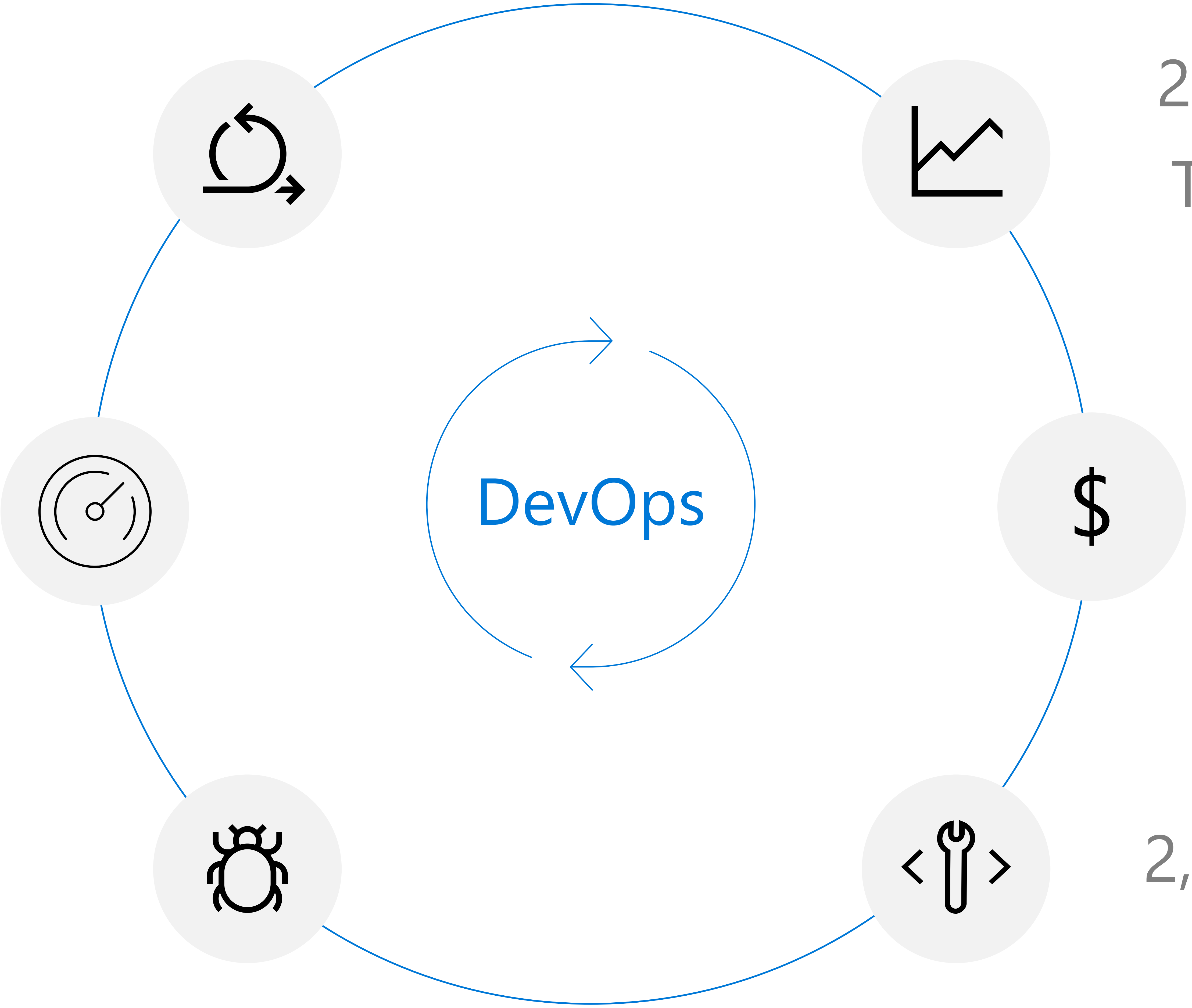
Integration services

Event-driven

AI

Dev Ops

High Performance DevOps Companies Achieve...



46x Deployment Frequency

2,555x Faster Lead Time For Changes

Faster Time to Market

Increased Revenue

7x Lower Change Failure Rate

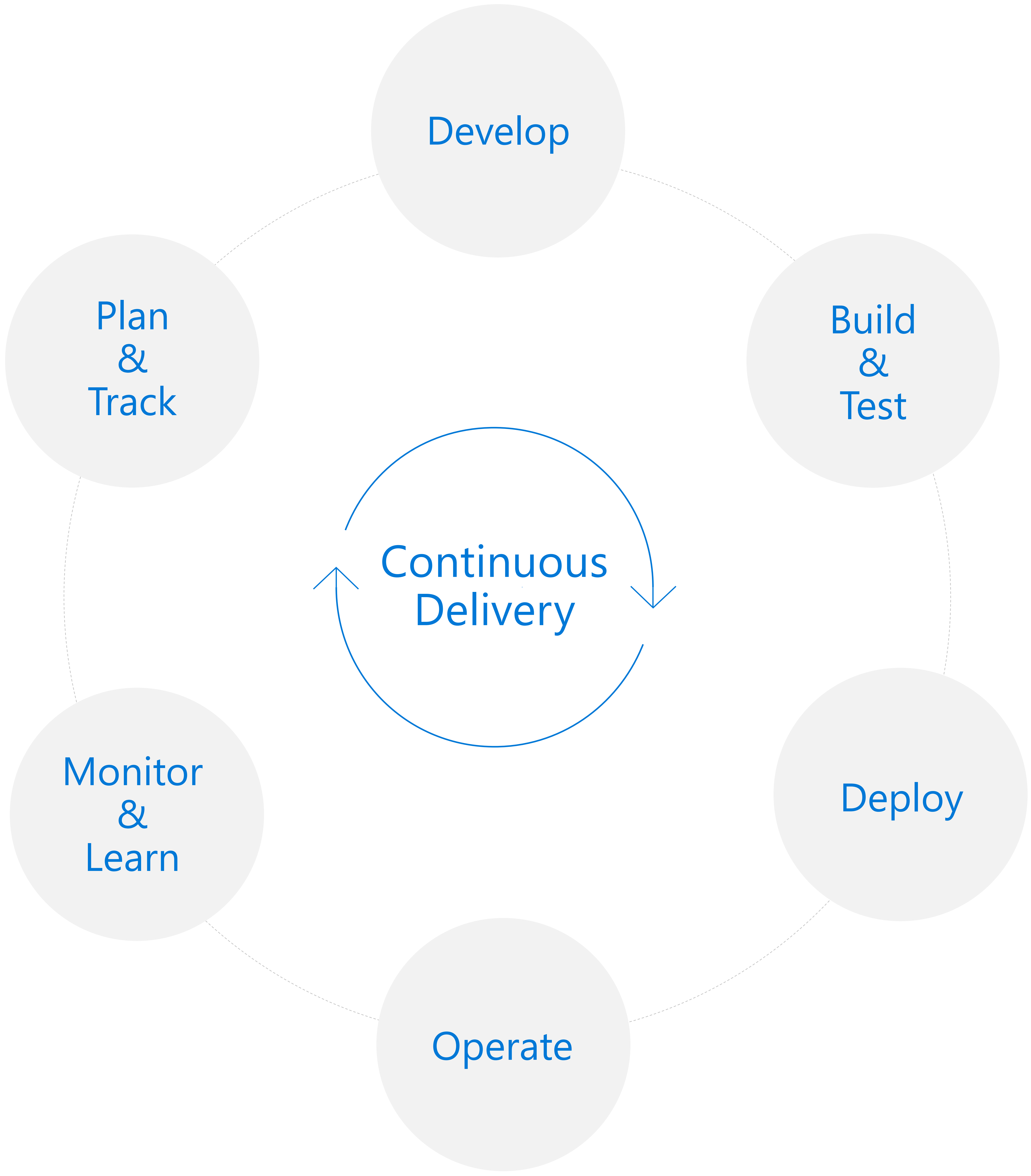
2,604x Faster Mean Time to Recover

Source: 2018 Accelerate: State of DevOps: Strategies for a New Economy." N. Forsgren, J. Humble, G. Kim. DevOps Research and Assessment (DORA)

What is DevOps?

People. Process. Products.

> “DevOps is the union of **people**, **process**, and **products** to enable continuous delivery of value to your end users.”



Introducing Azure DevOps

Azure Boards

Deliver value to your users faster using proven agile tools to plan, track, and discuss work across your teams.

Azure Repos

Get unlimited, cloud-hosted private Git repos and collaborate to build better code with pull requests and advanced file management.

Azure Pipelines

Build, test, and deploy with CI/CD that works with any language, platform, and cloud. Connect to GitHub or any other Git provider and deploy continuously.

➔

https://azure.com/devops

Azure Artifacts

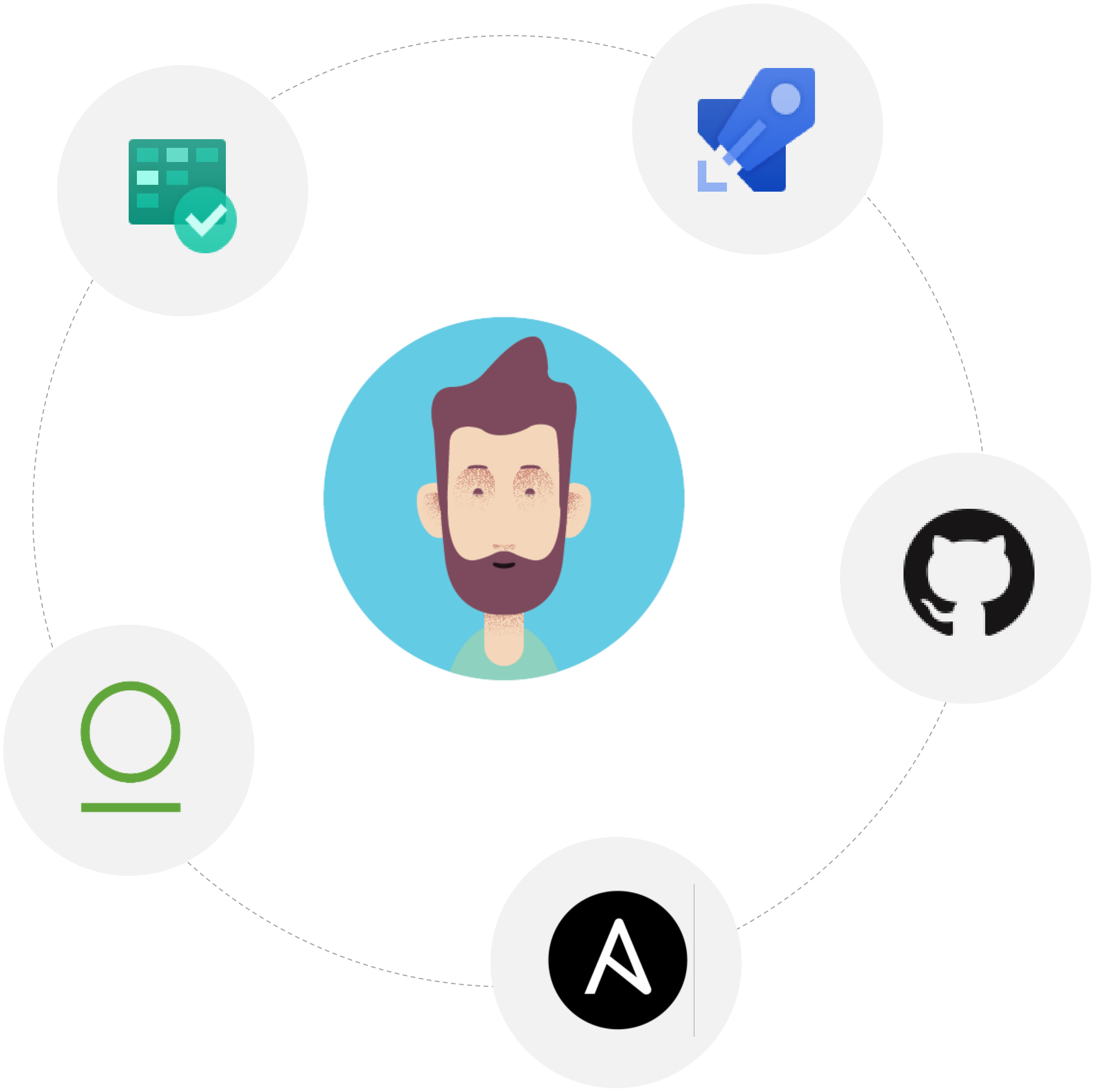
Create, host, and share packages with your team, and add artifacts to your CI/CD pipelines with a single click.

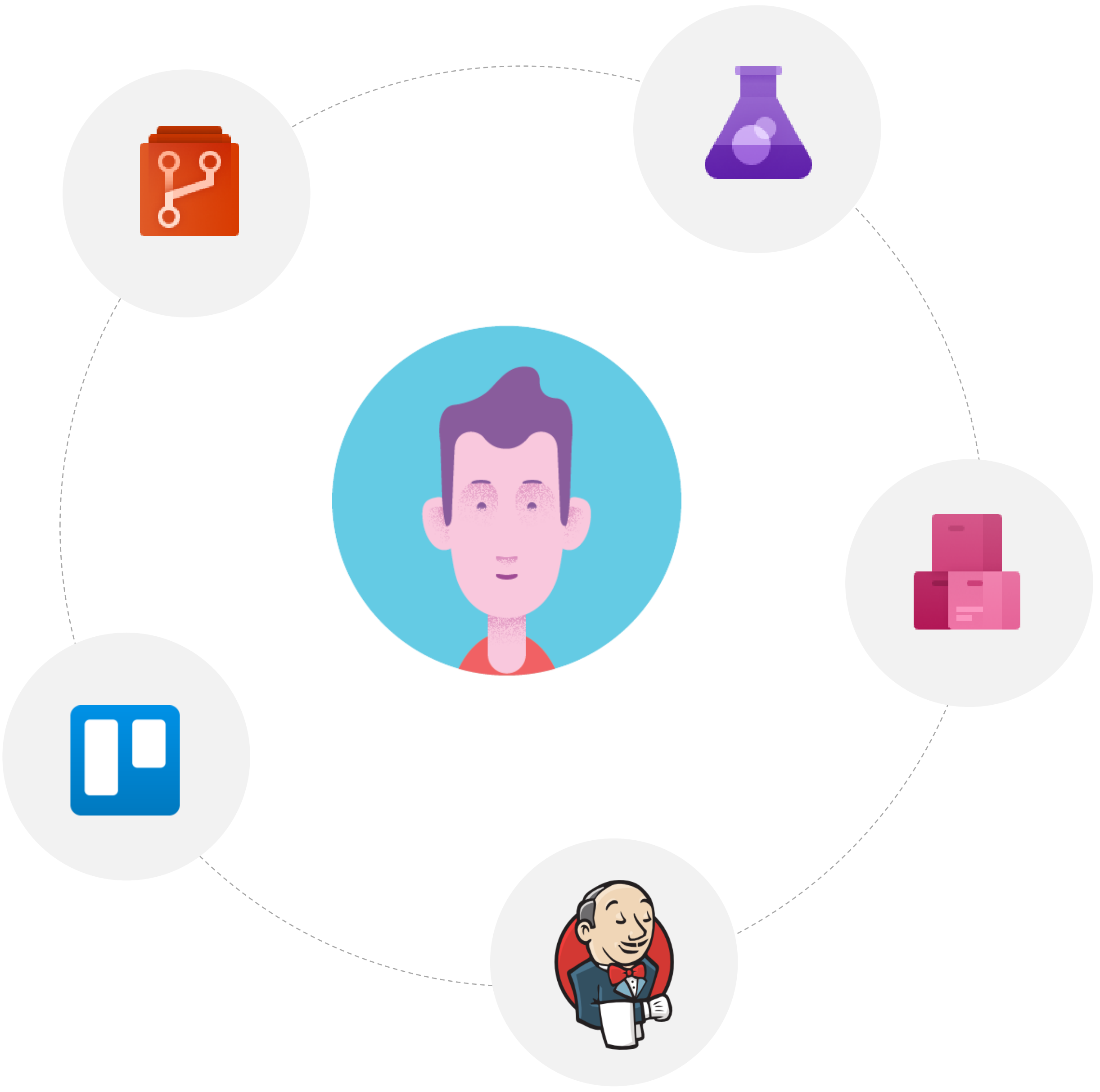
Azure Test Plans

Test and ship with confidence using manual and exploratory testing tools.

Azure DevOps: Choose the tools and clouds you love

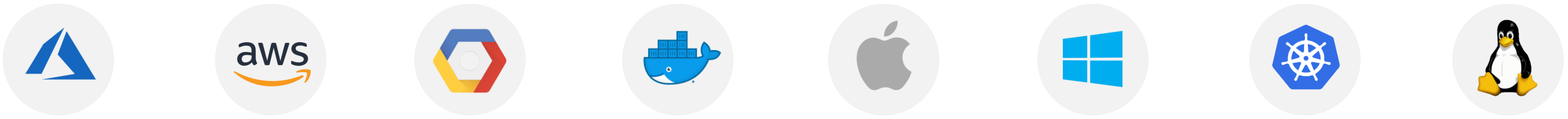
Azure DevOps lets developers choose the tools that are right for them



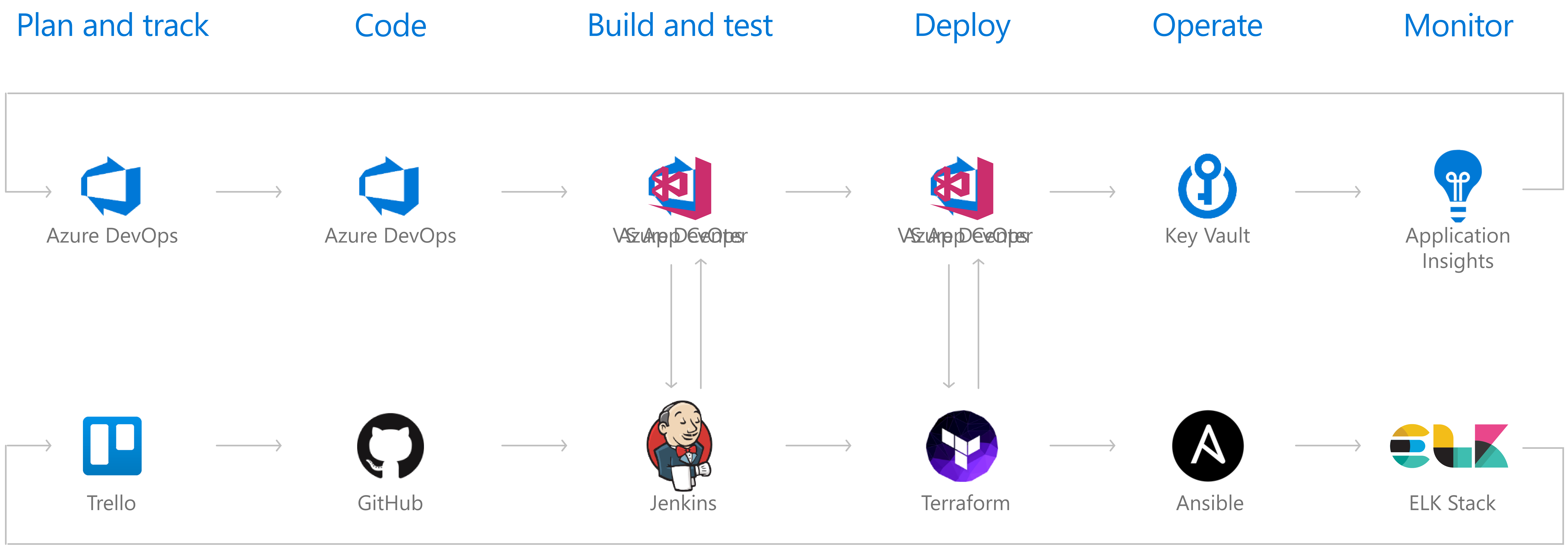


Mix and match to create workflows with tools from Microsoft, open source or your favorite 3rd party tools

Target any cloud, on-prem or both and deploy to the servers you need



Azure DevOps framework



Plan and track

Code

Build and test

Deploy

Operate

Monitor

Azure DevOps → Azure DevOps → VS App Center Azure DevOps → VS App Center Azure DevOps → Key Vault → Application Insights

Trello → GitHub → Jenkins → Terraform → Ansible → ELK Stack

Open source support

DevOps	Nagios	VAGRANT		GRUNT				Xamarin		
Management	CHEF	puppet labs		SALTSTACK			mist.io	libcloud		SCALR CLOUD MANAGEMENT
Applications		Joomla!	Drupal™		APPRENDA		™			
App frameworks and tools	php	nodeJS		JS	Ruby	IJ	eclipse			
Databases and middleware		redis	CLEARDB	cloudera	MySQL™	mongoDB		Couchbase		
Infrastructure			redhat	suse		bitnami	ORACLE LINUX	FreeBSD	docker	

Azure Functions

FaaS is at the center of serverless

Functions-as-a-Service programming model use functions to achieve true serverless compute

Single responsibility

Functions are single-purposed, reusable pieces of code that process an input and return a result

Short lived

Functions don’t stick around when finished executing, freeing up resources for further executions

Stateless

Functions don’t hold any persistent state and don’t rely on the state of any other processes

Event driven & scalable

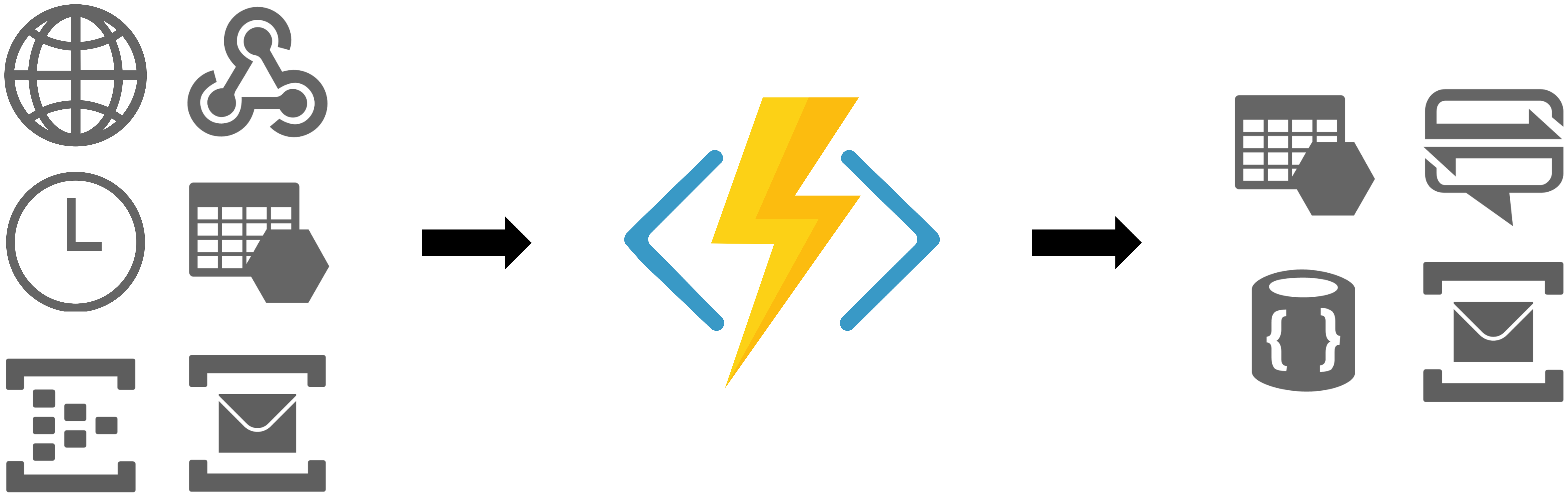
Functions respond to predefined events, and are instantly replicated as many times as needed

Azure Functions

Events

Code

Outputs



React to timers, HTTP, or events from your favorite Azure services, with more on the way

Author functions in C#, F#, Node.JS, Java, Python, PowerShell, and more

Send results to an ever-growing collection of services

Focus on code, not plumbing

No infrastructure management

Auto-scale based on your workload

No wasted resources, pay only for what you use

Boost development efficiency

Triggers

- Use triggers to define how functions are invoked
- Avoid hardcoding with preconfigured JSON files
- Build serverless APIs using HTTP triggers

Proxies

- Define one API surface for multiple function apps
- Create endpoints as reverse proxies to other APIs
- Condition proxies to use variables

CI/CD

- Save time with built-in DevOps
- Deploy functions using App Service for CI
- Leverage Microsoft, partner services for CD

Bindings

- Connect to data with input and output bindings
- Bind to Azure solutions and third-party services
- Use HTTP bindings in tandem with HTTP triggers

Local debugging

- Debug C# and JavaScript functions locally
- Use debugging tools in Azure portal, VS, and VS Code

Monitoring

- Integrate with Azure Application Insights
- Get near real-time details about function apps
- See metrics around failures, executions, etc.

#ntk19

Gain flexibility and develop your way

Multiple languages

Write code in C#, JavaScript, F#, and Java
Continuous investment in new, experimental languages

Durable Functions

Write stateful functions in a serverless environment
Simplify complex, stateful coordination problems
Add the extension to enable advanced scenarios

Hosting options

Choose from six consumption plans to run Functions
Run your first million function executions for free

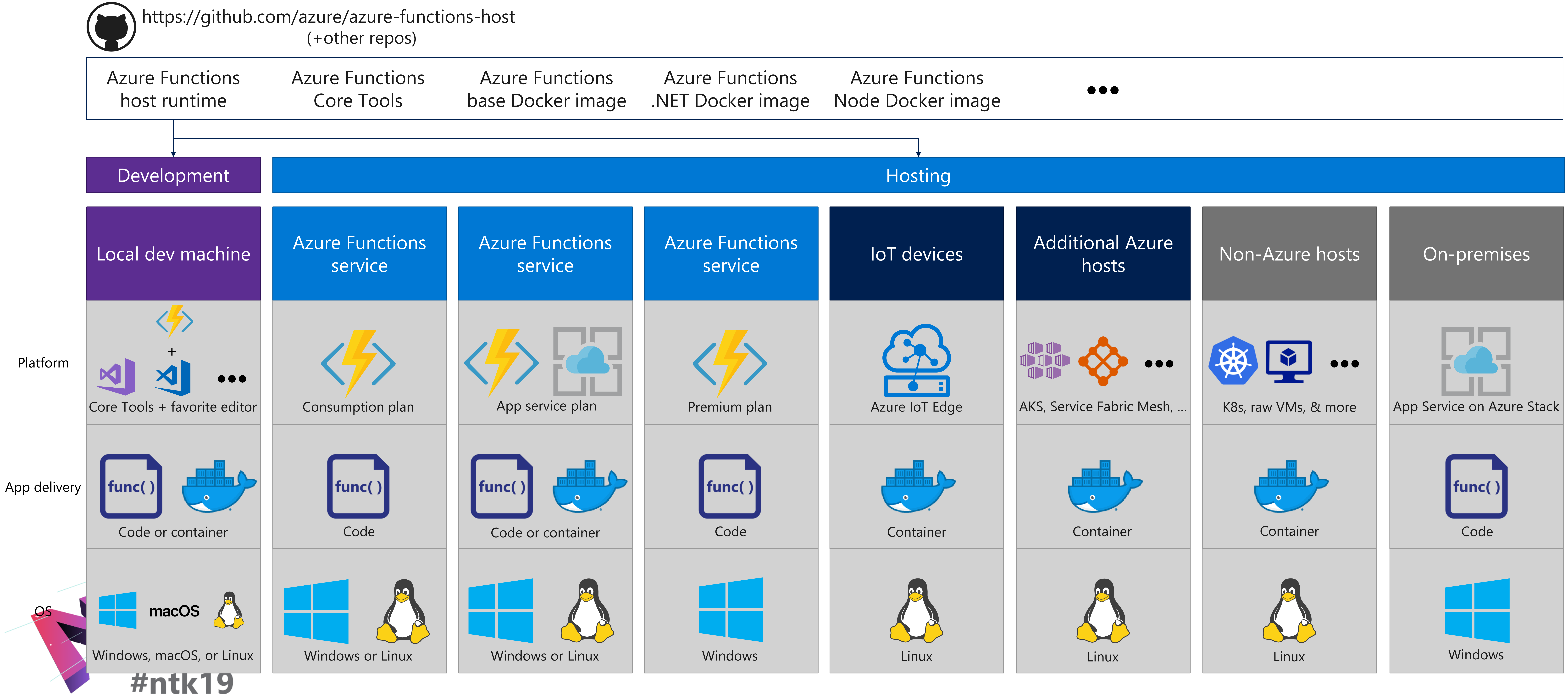
Dev options

Simplify coding for new users with native Azure portal
Select from popular editors, like VS, VS Code, CLI, Maven*

#ntk19

*VS and VS Code only support C#; Maven only supports Java

Functions everywhere

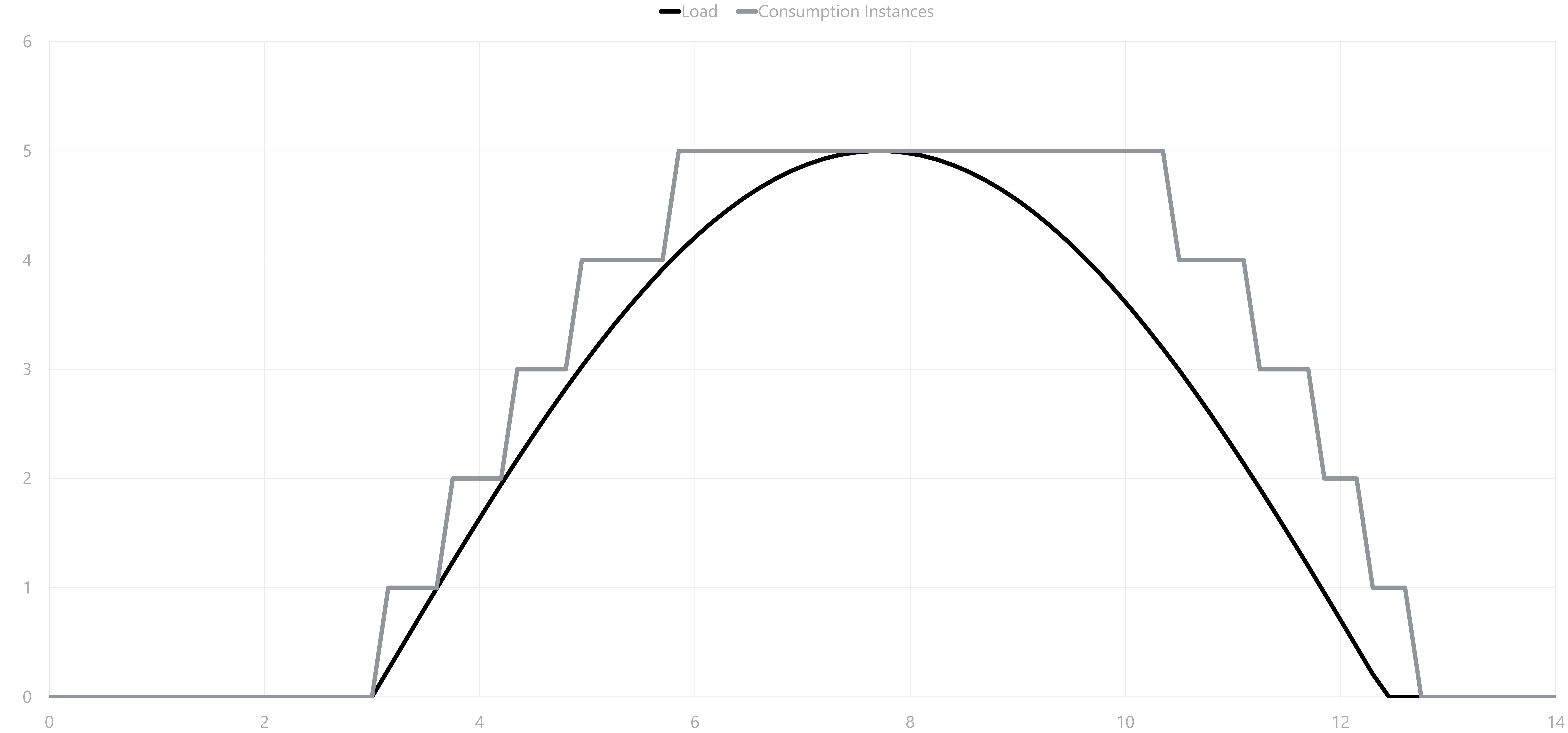


https://github.com/azure/azure-functions-host
(+other repos)

Azure Functions host runtime | Azure Functions Core Tools | Azure Functions base Docker image | Azure Functions .NET Docker image | Azure Functions Node Docker image | •••

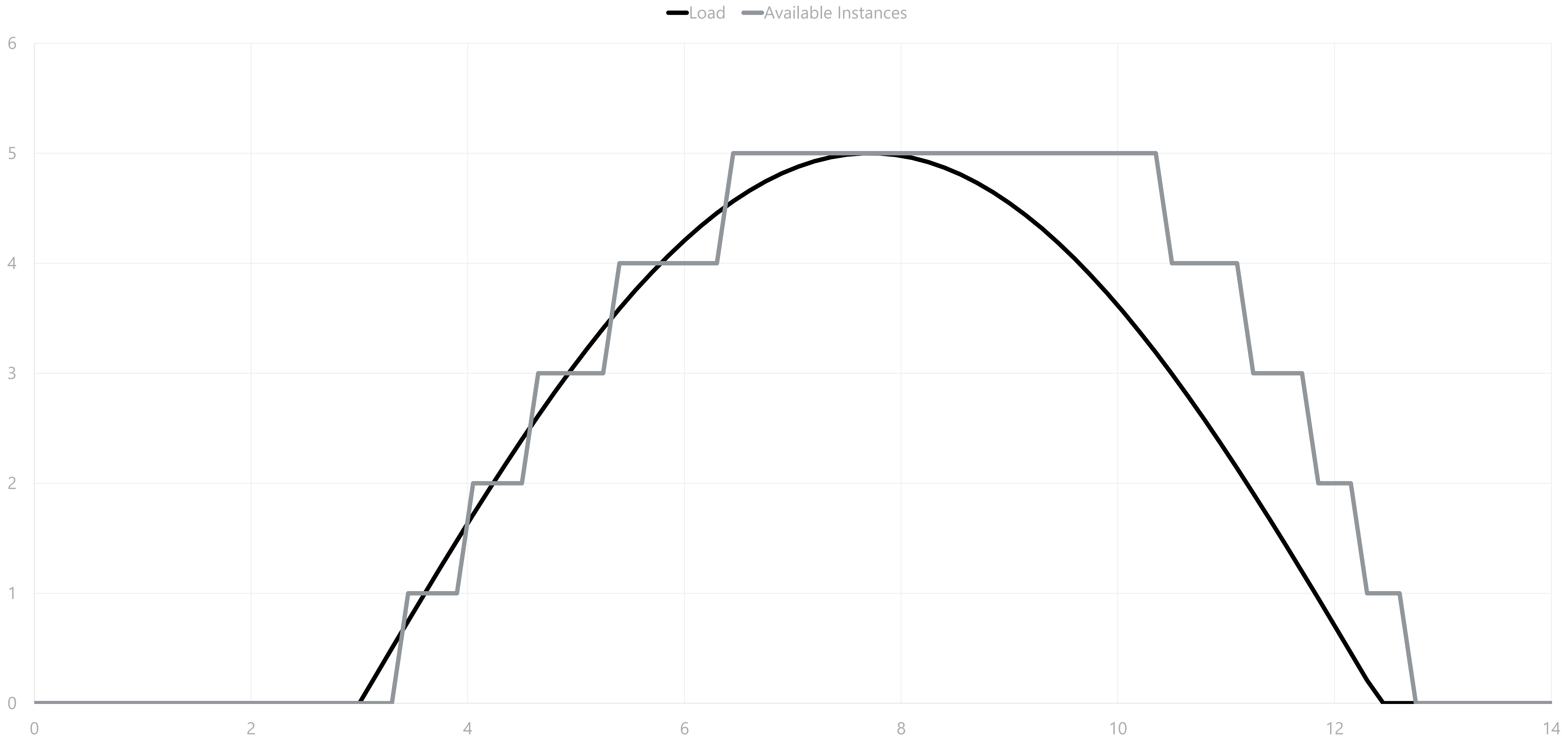
	Development	Hosting						
	Local dev machine	Azure Functions service	Azure Functions service	Azure Functions service	IoT devices	Additional Azure hosts	Non-Azure hosts	On-premises
Platform	+ ••• Core Tools + favorite editor	Consumption plan	App service plan	Premium plan	Azure IoT Edge	••• AKS, Service Fabric Mesh, ...	••• K8s, raw VMs, & more	App Service on Azure Stack
App delivery	func( ) Code or container	func( ) Code	func( ) Code or container	func( ) Code	Container	Container	Container	func( ) Code
OS	macOS Windows, macOS, or Linux	Windows or Linux	Windows or Linux	Windows	Linux	Linux	Linux	Windows

Your app in ✨concept✨



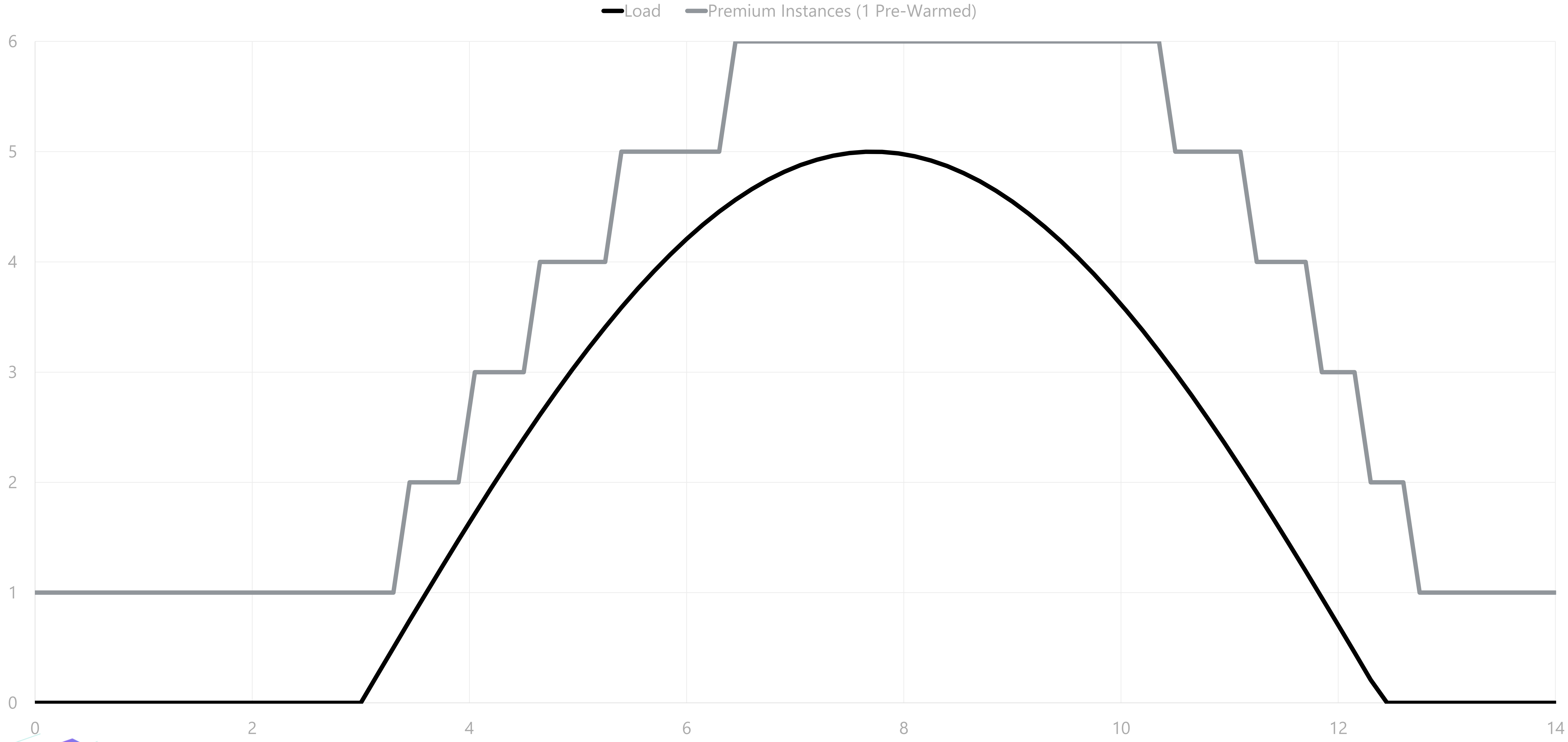
#ntk19

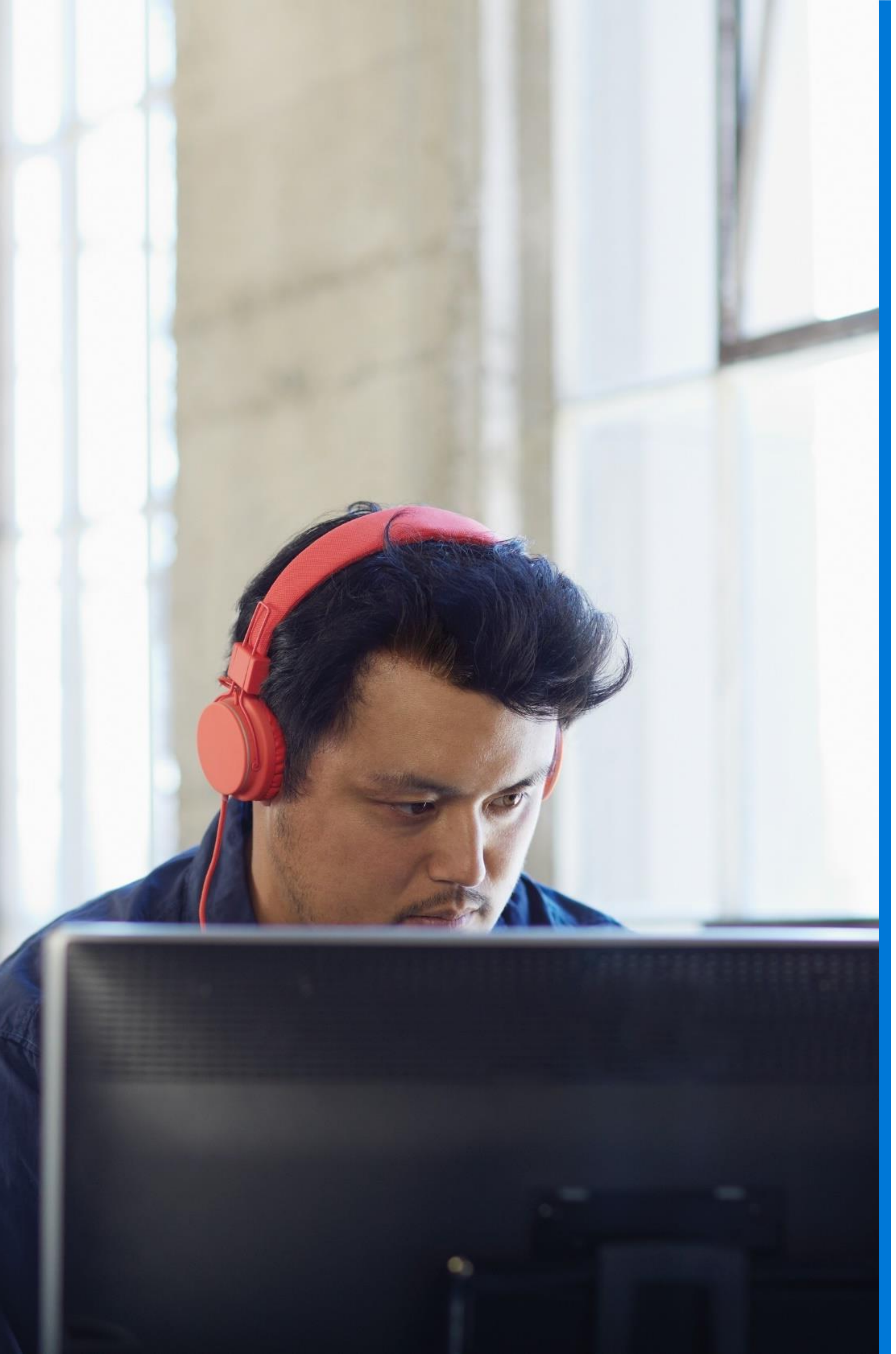
Your app with long cold start



#ntk19

Your app with one pre-warmed instance





Sample scenarios for Functions

Web application backends

Mobile application backends

IoT-connected backends

Conversational bot processing

Real-time file processing

Real-time stream processing

Automation of scheduled tasks

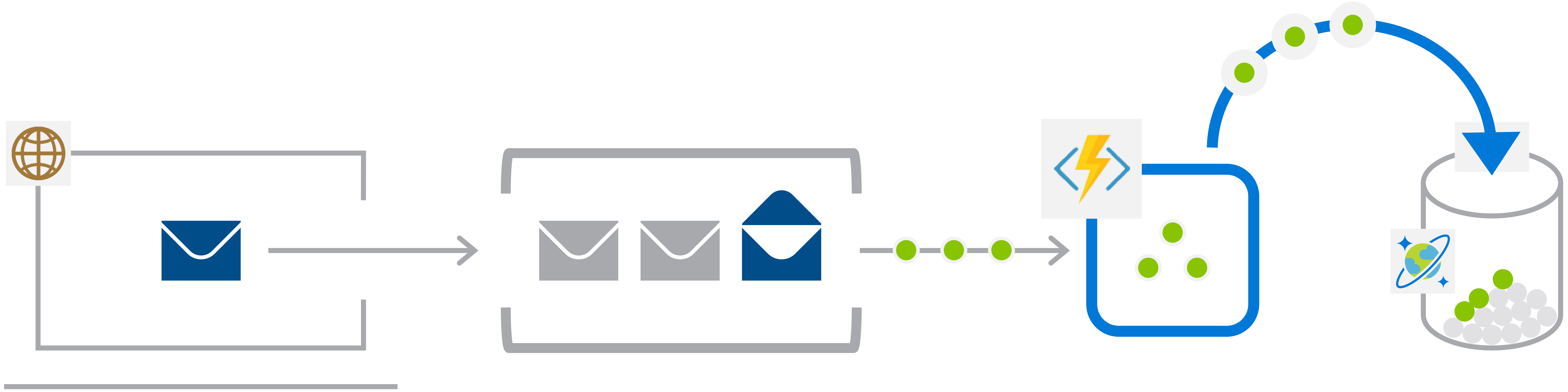
Extending SaaS Applications

Scenario Example

Retail

Online orders are picked up from a queue, processed and the resulting data is stored in a database.

Web application backends



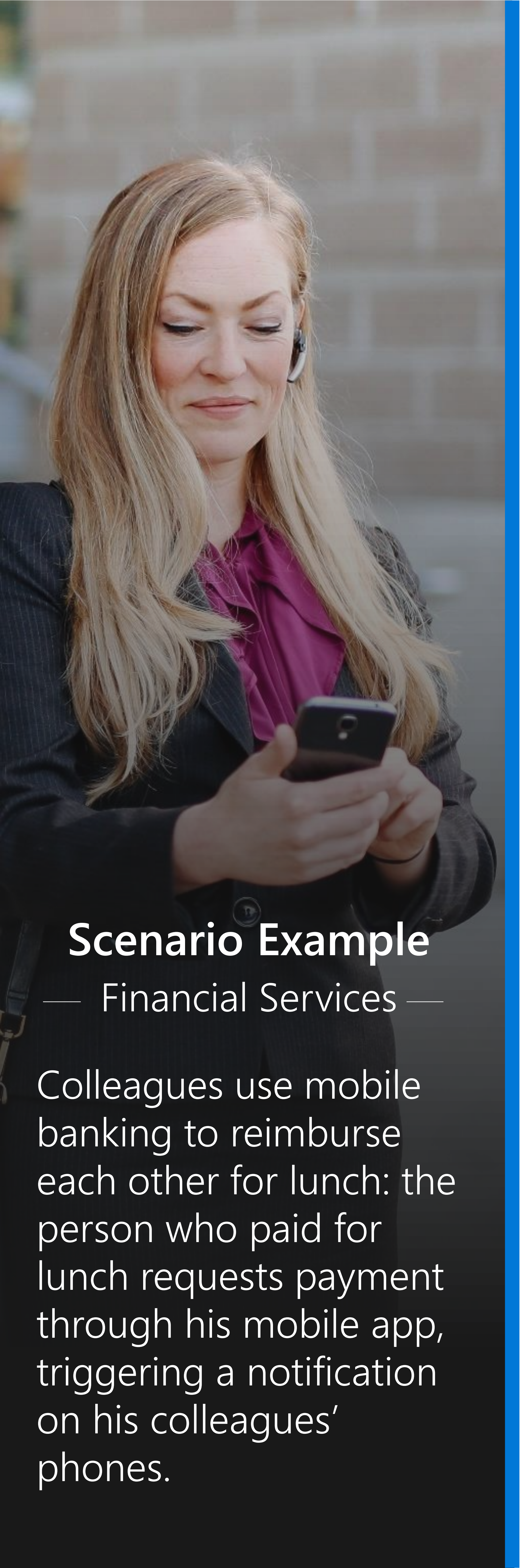
Request made in a web app

Request queued in Service Bus

A function processes the request…

…sends output to Cosmos DB

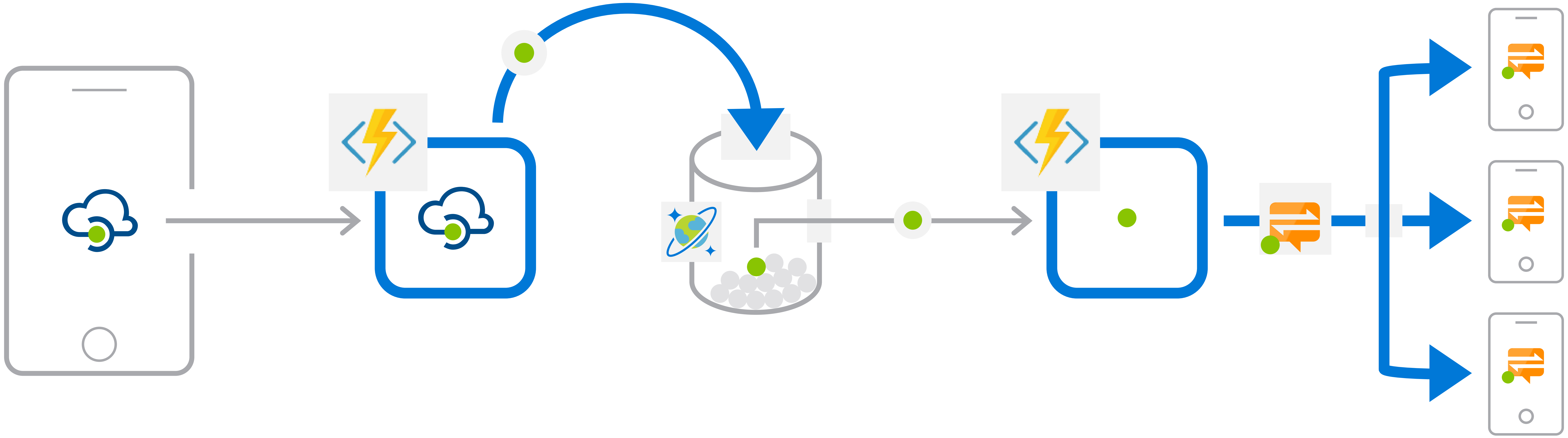
Mobile application backends



Scenario Example

— Financial Services —

Colleagues use mobile banking to reimburse each other for lunch: the person who paid for lunch requests payment through his mobile app, triggering a notification on his colleagues' phones.



HTTP API call from a mobile app

Call processed by a function

Output data stored in Cosmos DB

Data transfer triggers second function…

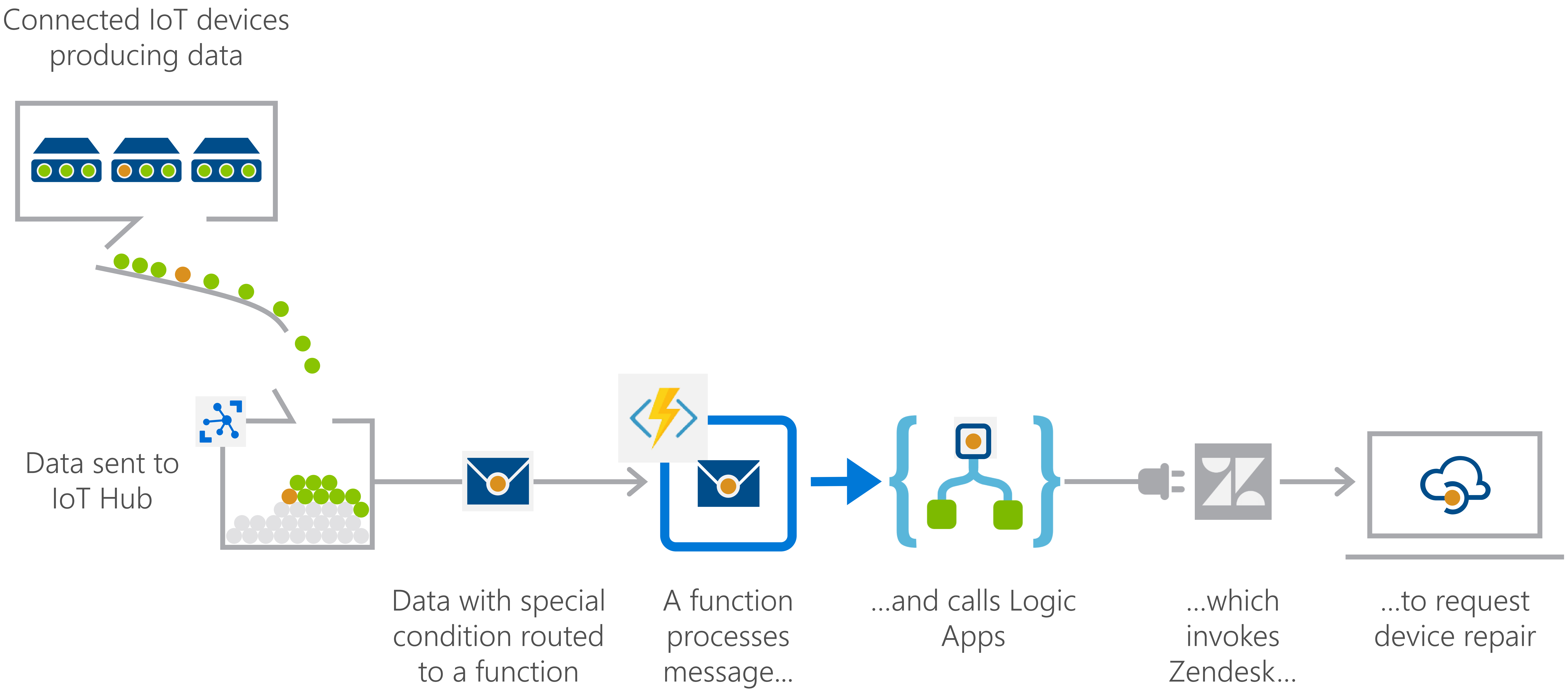
…which sends notifications using Notifications Hub

IoT-connected backends

Scenario Example

Manufacturing

A manufacturing company uses IoT to monitor its machines. Functions detects anomalous data and triggers a message to Service department when repair is required.



Connected IoT devices producing data

Data sent to IoT Hub

Data with special condition routed to a function

A function processes message…

…and calls Logic Apps

…which invokes Zendesk…

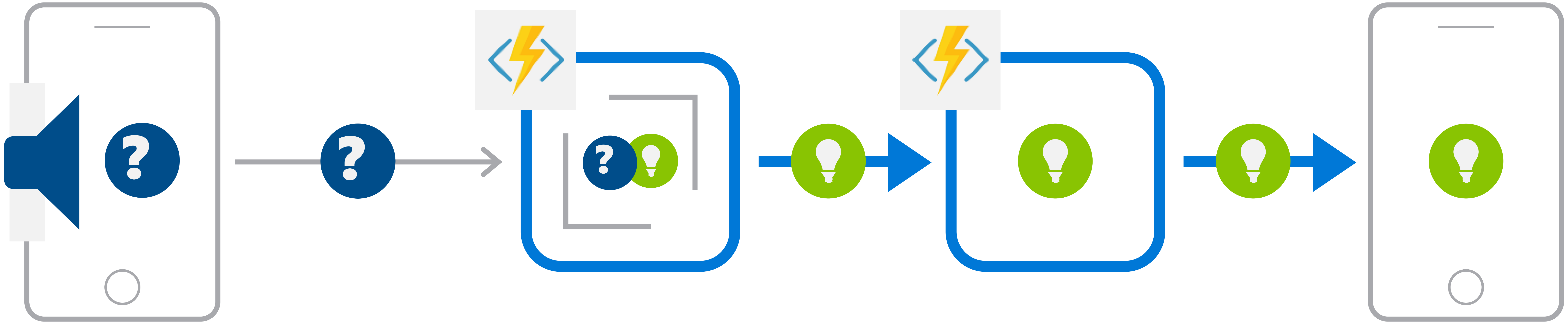
…to request device repair

Scenario Example

Hospitality

Customer asks for available vacation accommodations on her smartphone. A serverless bot deciphers the request and returns vacation options.

Conversational bot processing



User request through conversational interface

Bot running in a function deciphers request using language understanding

Another function processes the request

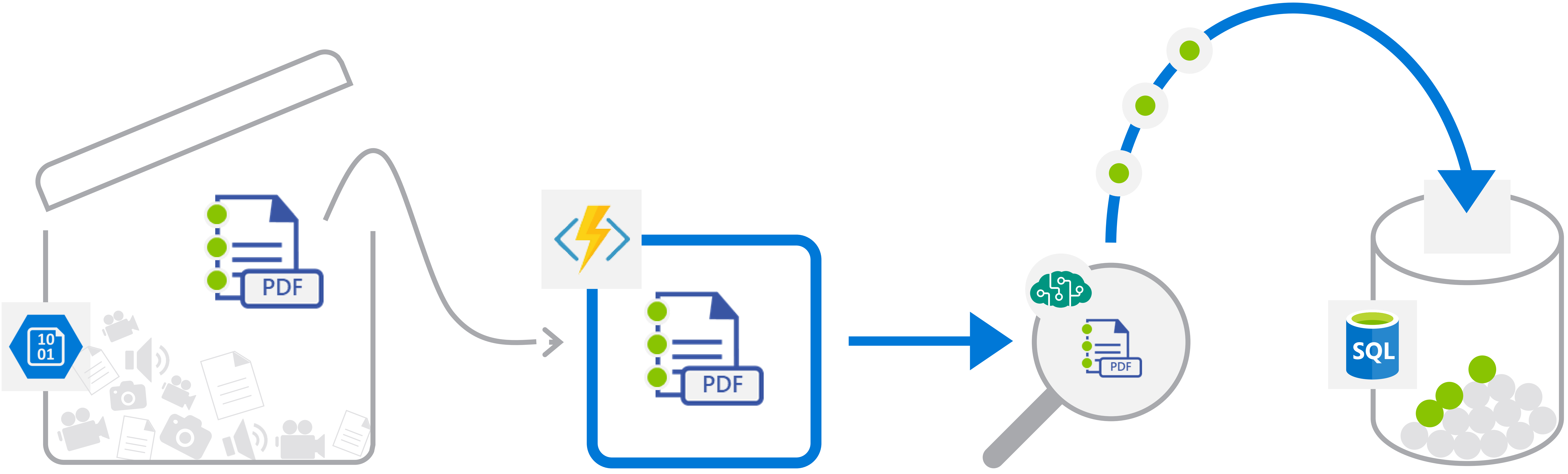
...and sends response to original requester

Real-time file processing

Scenario Example

Healthcare

Patient records are securely uploaded as PDF files. That data is then decomposed, processed using OCR detection, and added to a database for easy queries.



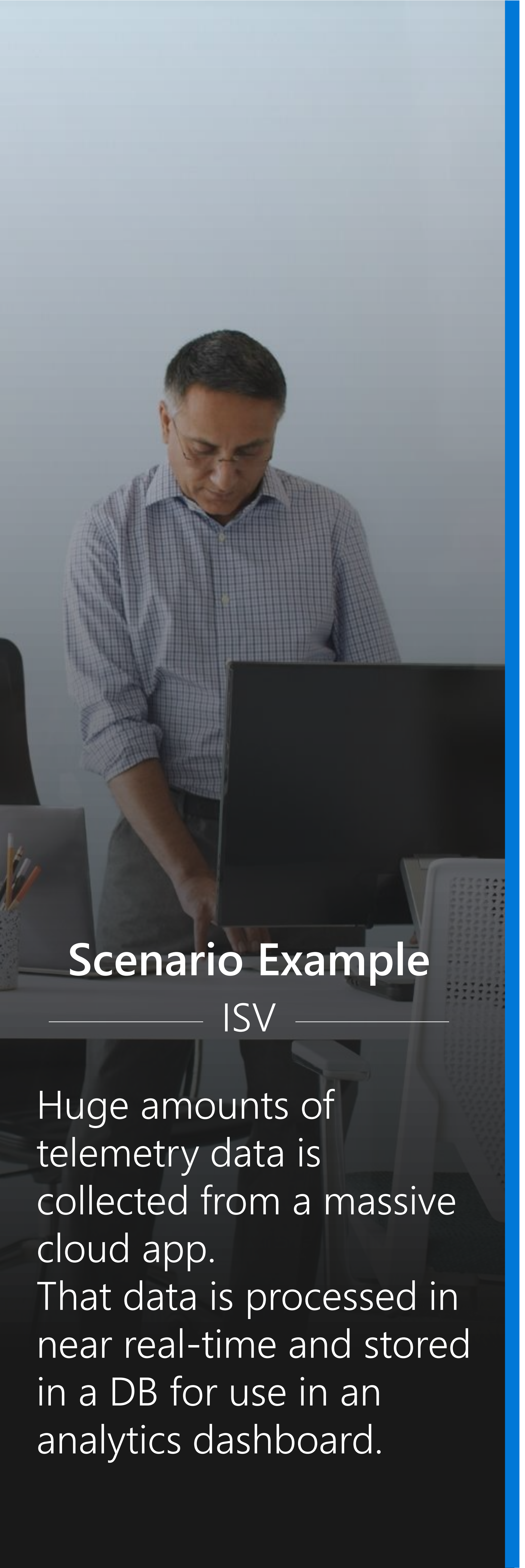
PDF file added to Blob Storage

A function decomposes PDF file…

…and sends it to Cognitive Services for OCR detection

Structured data from file sent to SQL DB

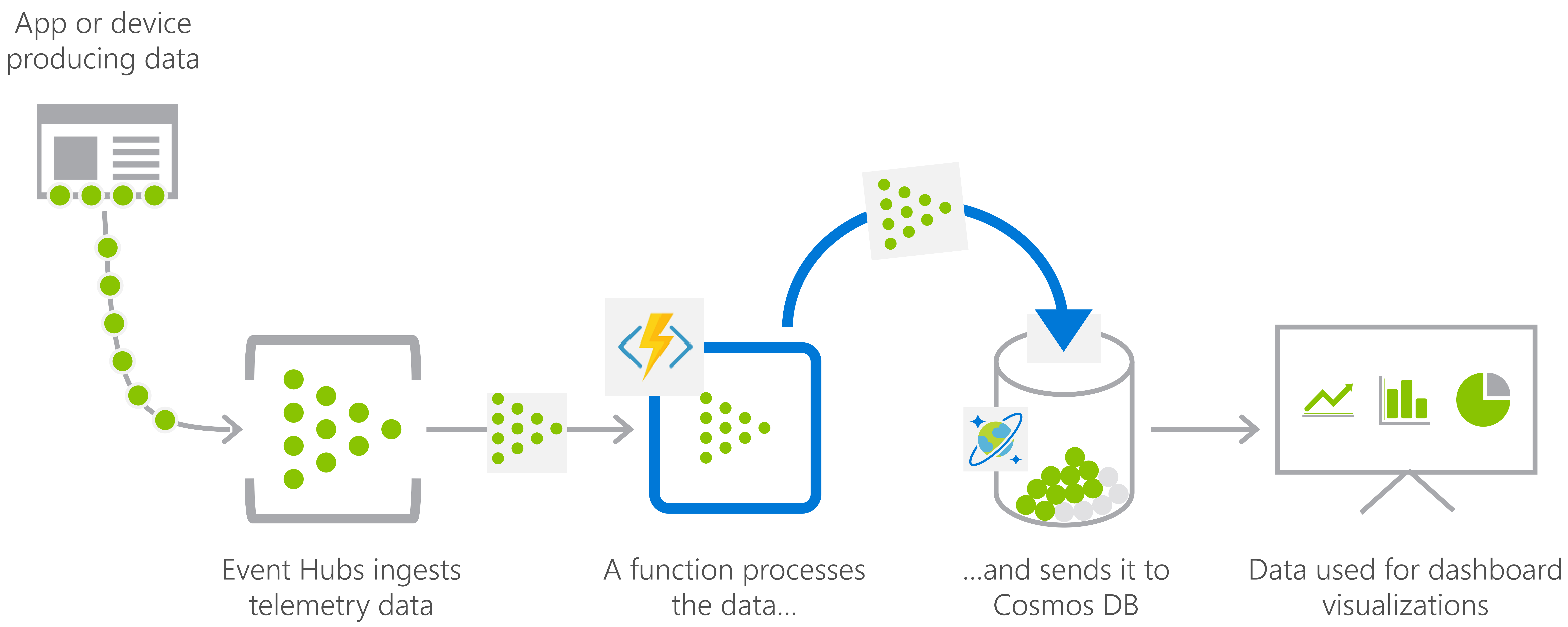
Real-time **stream** processing



Scenario Example

ISV

Huge amounts of telemetry data is collected from a massive cloud app.
That data is processed in near real-time and stored in a DB for use in an analytics dashboard.



App or device producing data

Event Hubs ingests telemetry data

A function processes the data…

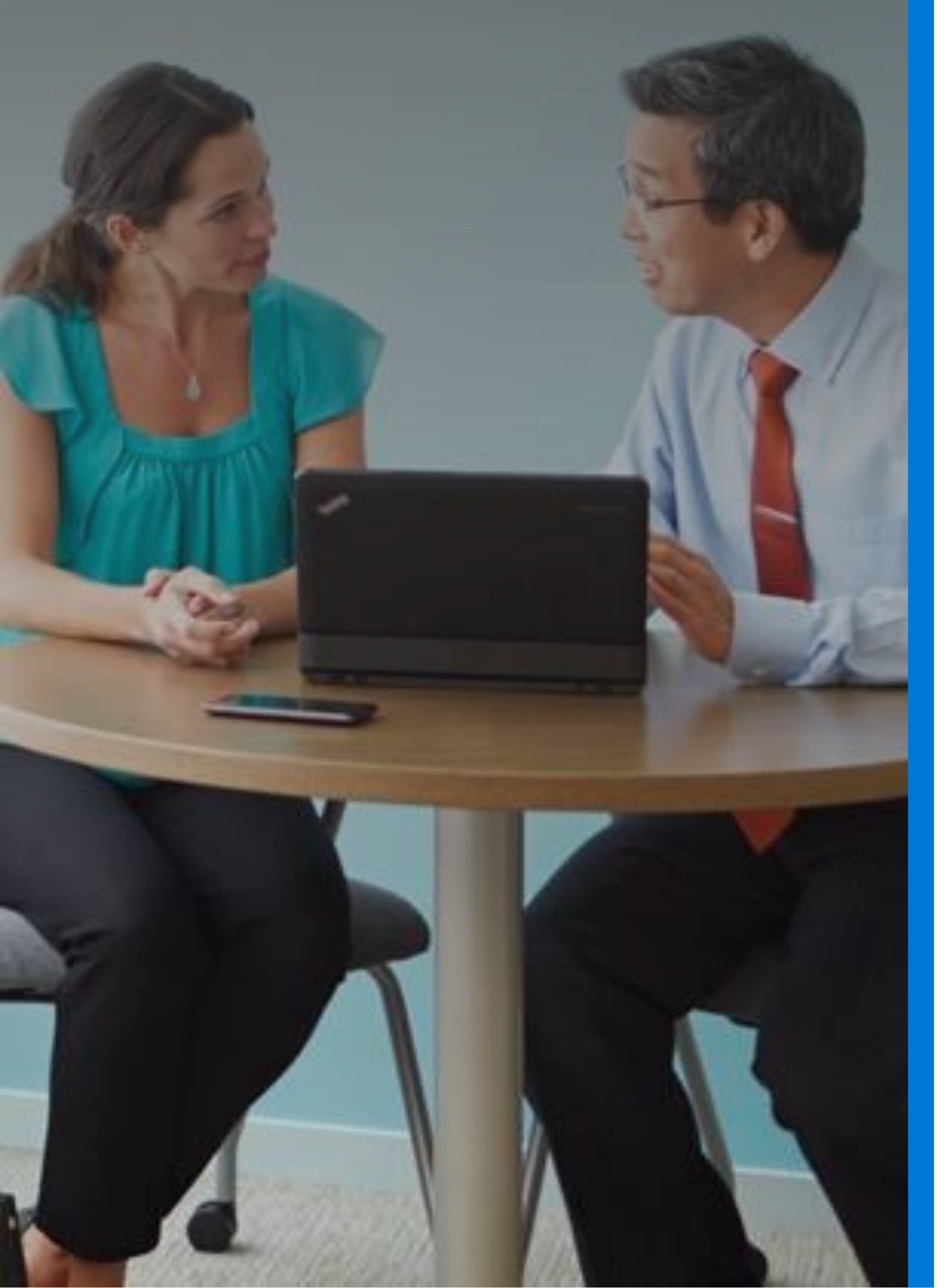
…and sends it to Cosmos DB

Data used for dashboard visualizations

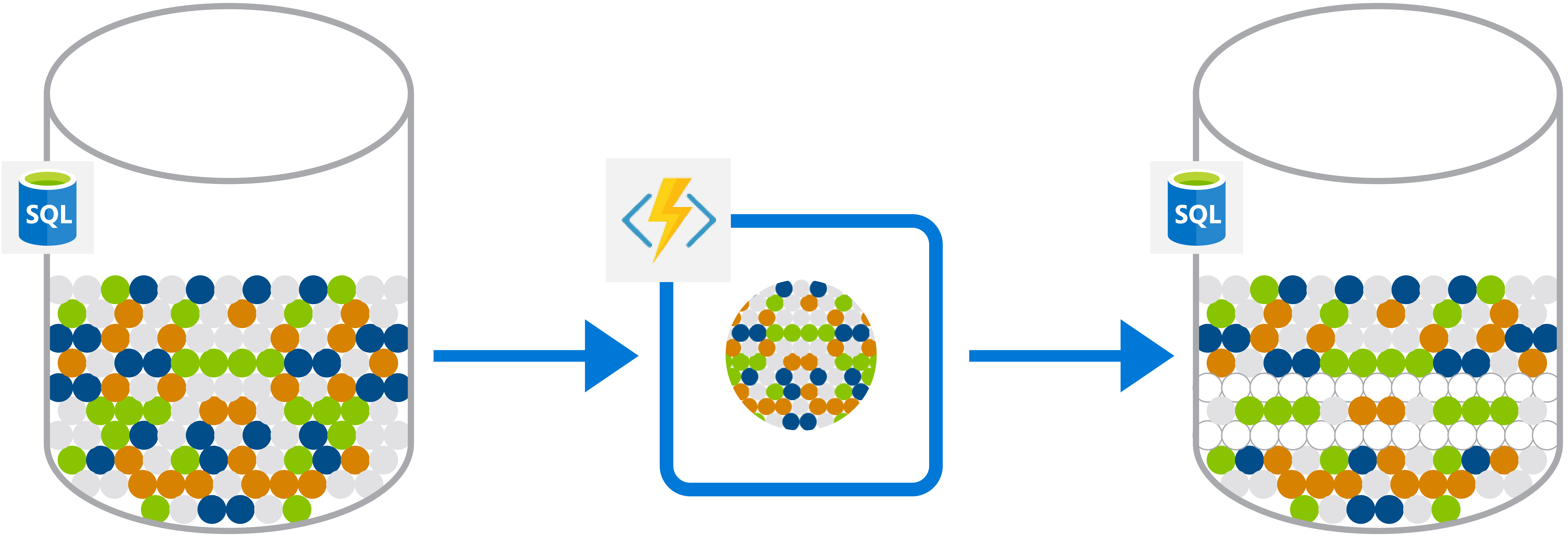
Scenario Example

— Financial Services —

A customer database is analyzed for duplicate entries every 15 minutes, to avoid multiple communications being sent out to same customers.



Automation of scheduled tasks



A function cleans a database every 15 minutes...

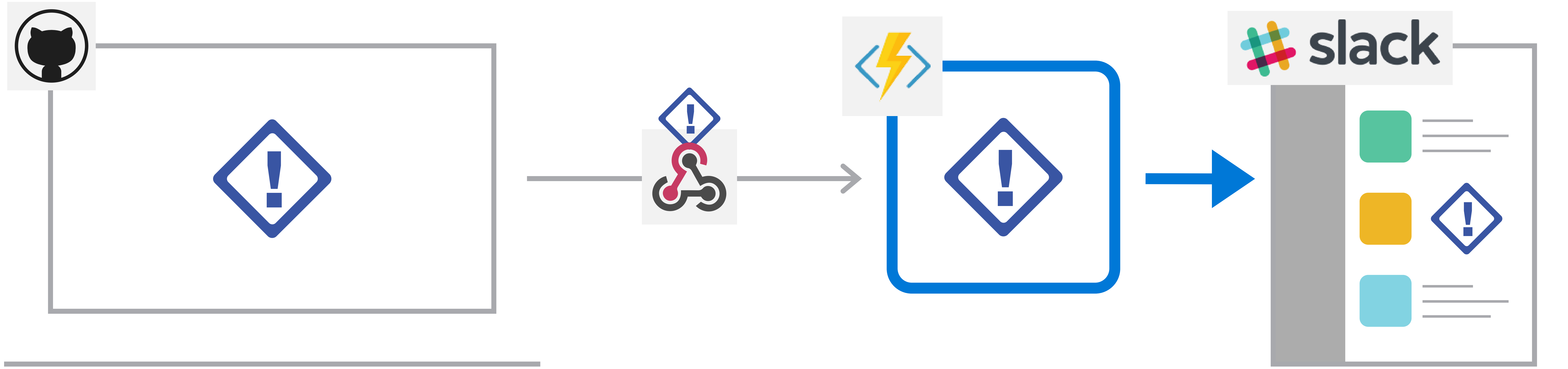
...deduplicating entries based on business logic

Scenario Example

— Professional Services —

A SaaS solution provides extensibility through webhooks, which can be implemented through Functions, to automate certain workflows.

Extending SaaS applications



Issue created in GitHub…

…which triggers a webhook call

…which is processed by a function…

…by posting the issue details to Slack

Azure Logic Apps and Azure Event Grid

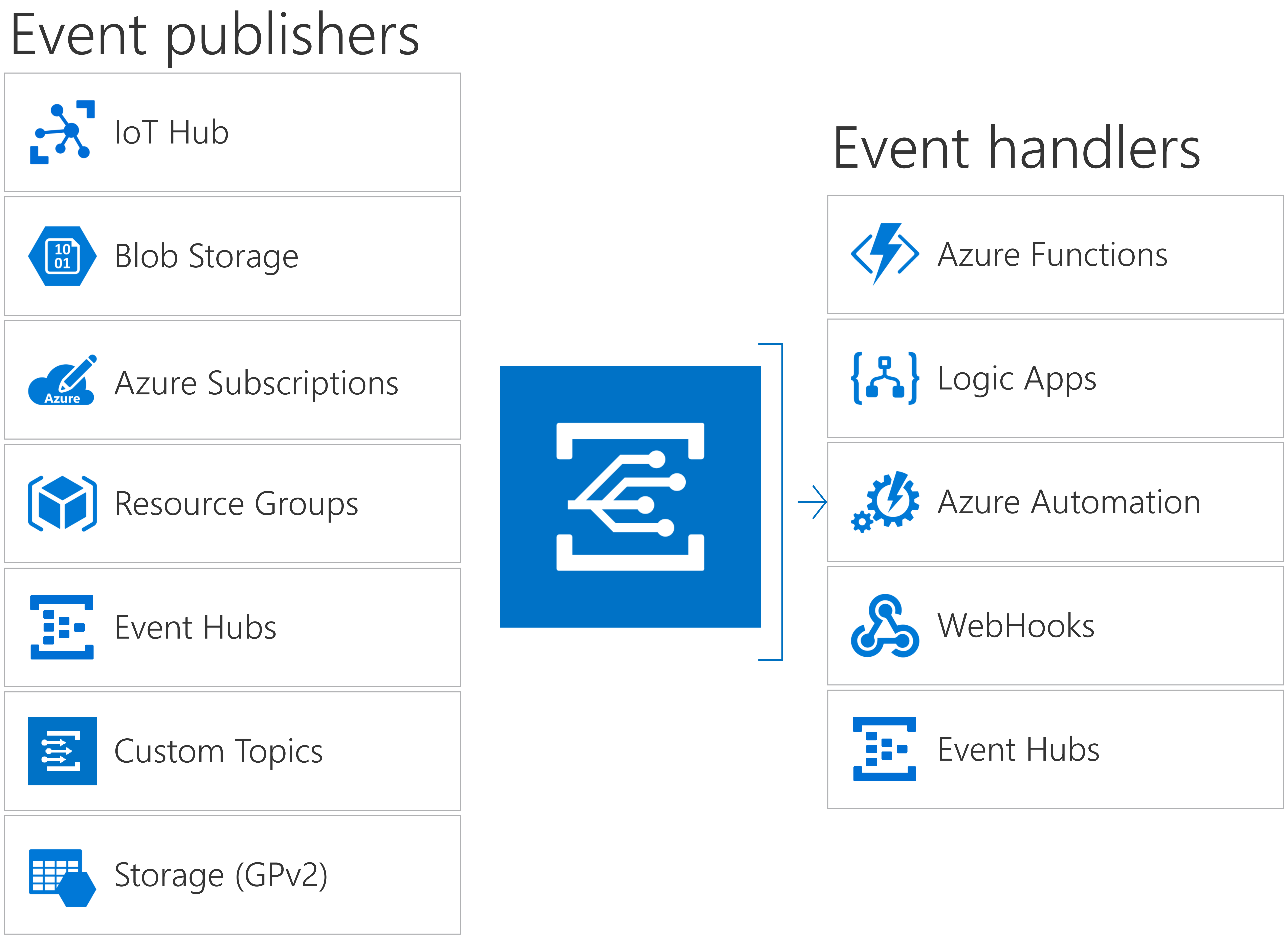
Event Grid

Eliminate polling—and the associated cost and latency

Build reliable apps and services through reactive programming

Enable richer scenarios by connecting multiple event sources and destinations

Support for open CloudEvent standard



Event publishers

- IoT Hub
- Blob Storage
- Azure Subscriptions
- Resource Groups
- Event Hubs
- Custom Topics
- Storage (GPv2)

Event handlers

- Azure Functions
- Logic Apps
- Azure Automation
- WebHooks
- Event Hubs

Logic Apps

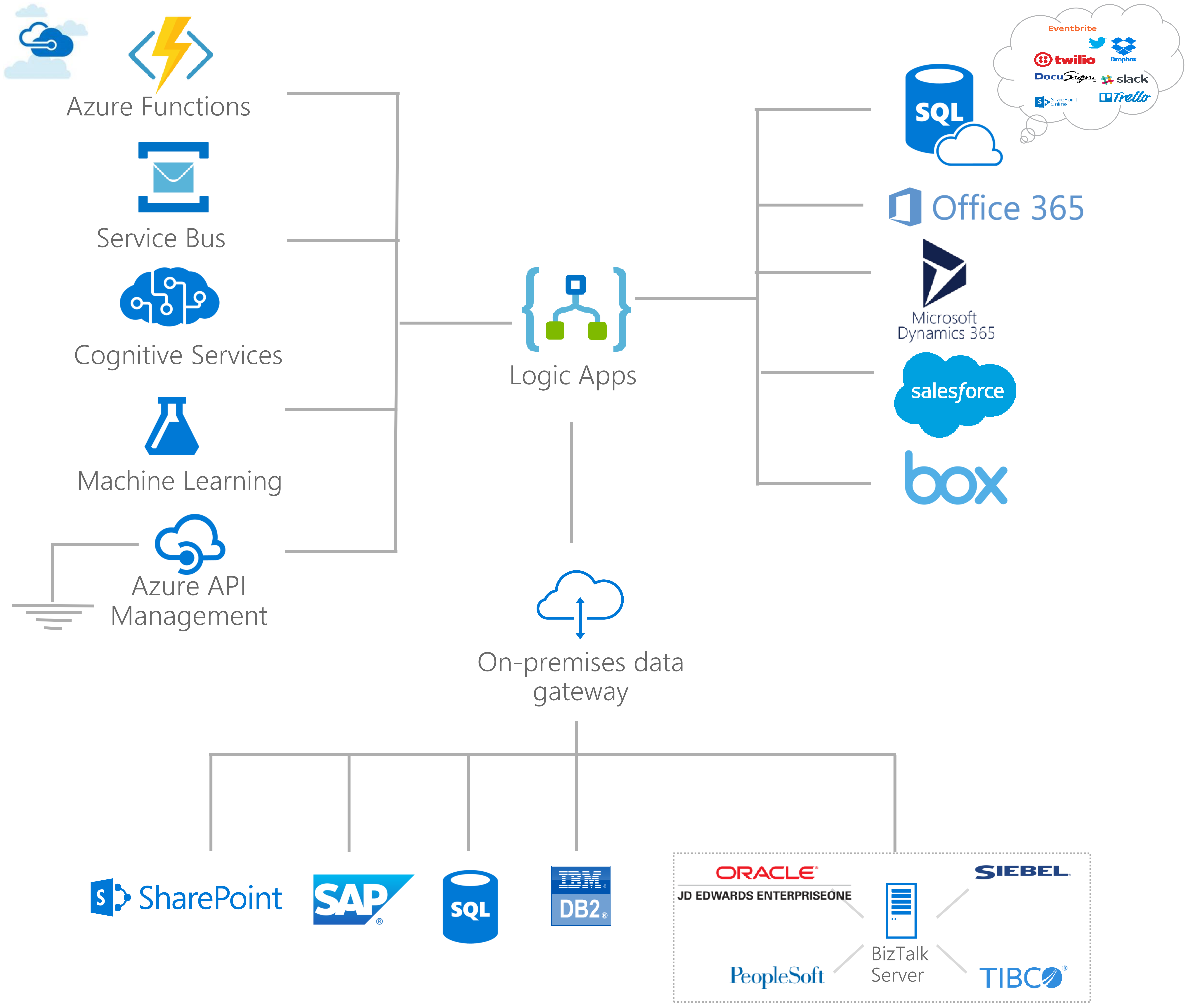
Visually design workflows in the cloud

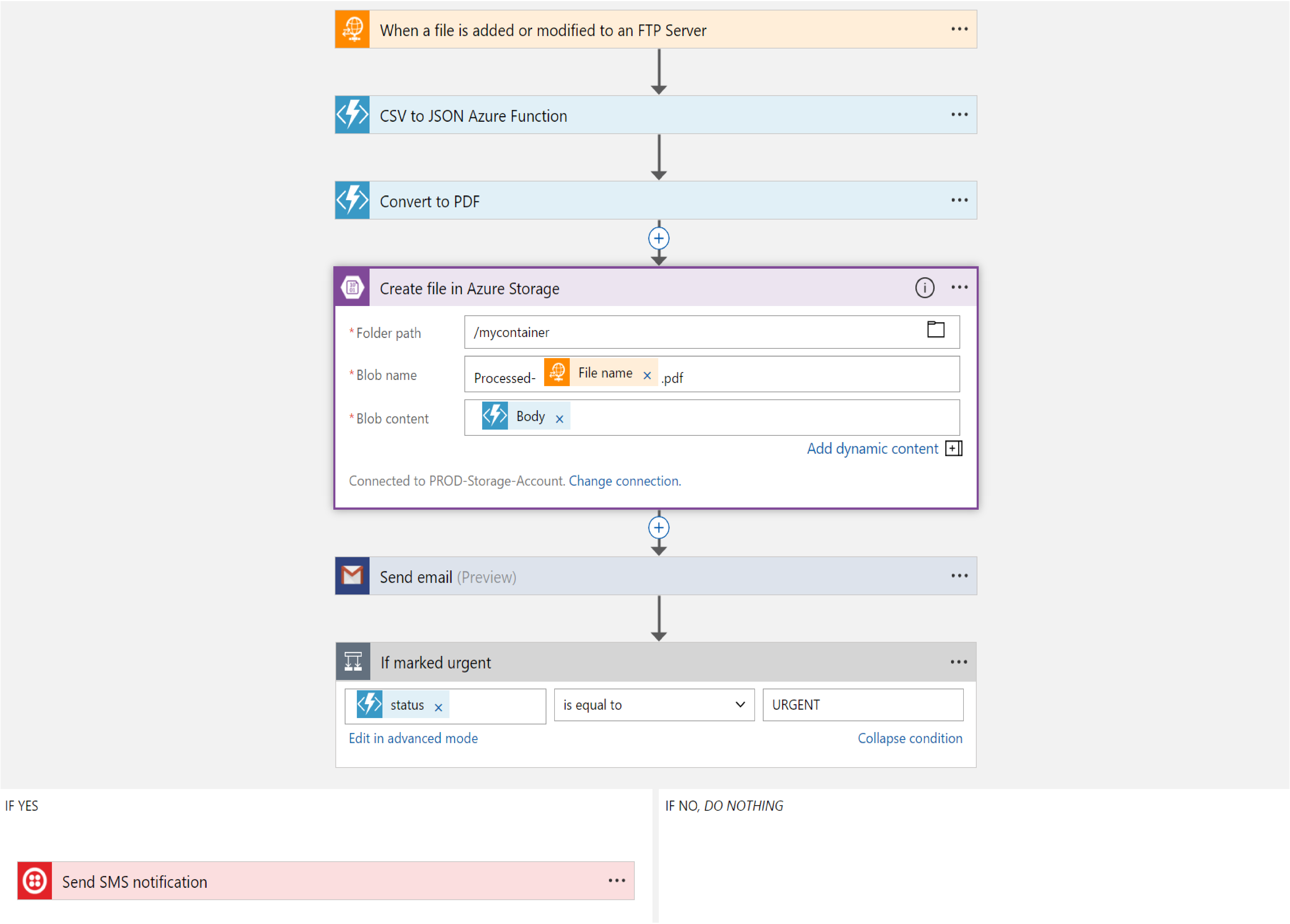
Express logic through powerful control flow

Connect disparate functions and APIs

Utilize declarative definition to work with CI/CD

Connect applications, data and services





When a file is added or modified to an FTP Server

CSV to JSON Azure Function

Convert to PDF

Create file in Azure Storage

- *Folder path: /mycontainer
- *Blob name: Processed- File name × .pdf
- *Blob content: Body ×

Add dynamic content

Connected to PROD-Storage-Account. Change connection.

Send email (Preview)

If marked urgent

status × | is equal to | URGENT

Edit in advanced mode

Collapse condition

IF YES

Send SMS notification

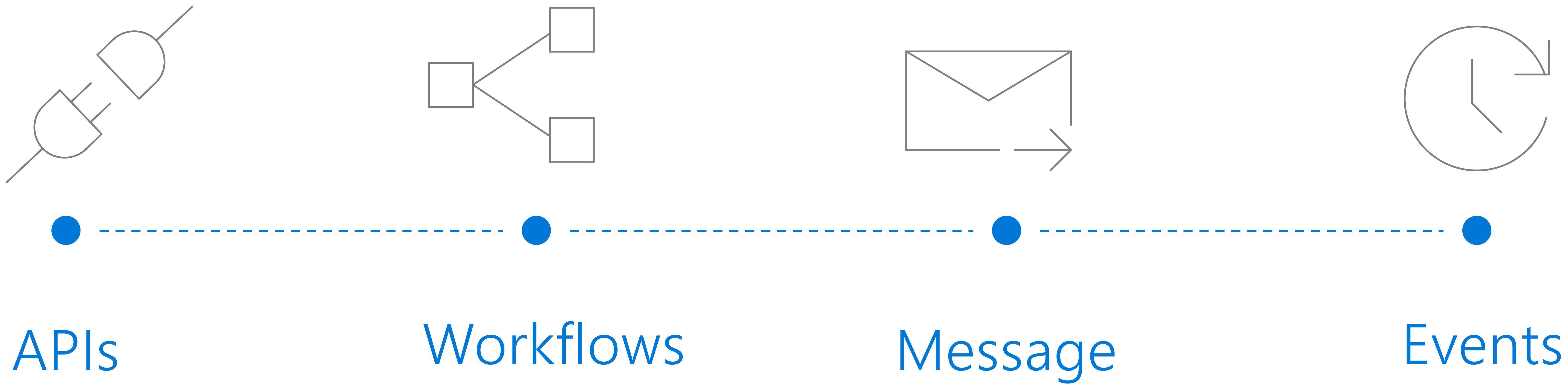
IF NO, *DO NOTHING*

Logic Apps connectors—Over 200 and growing

Dropbox
GitHub
facebook®
box
twitter
WordPress
slack
salesforce

....and more!

Integration is key

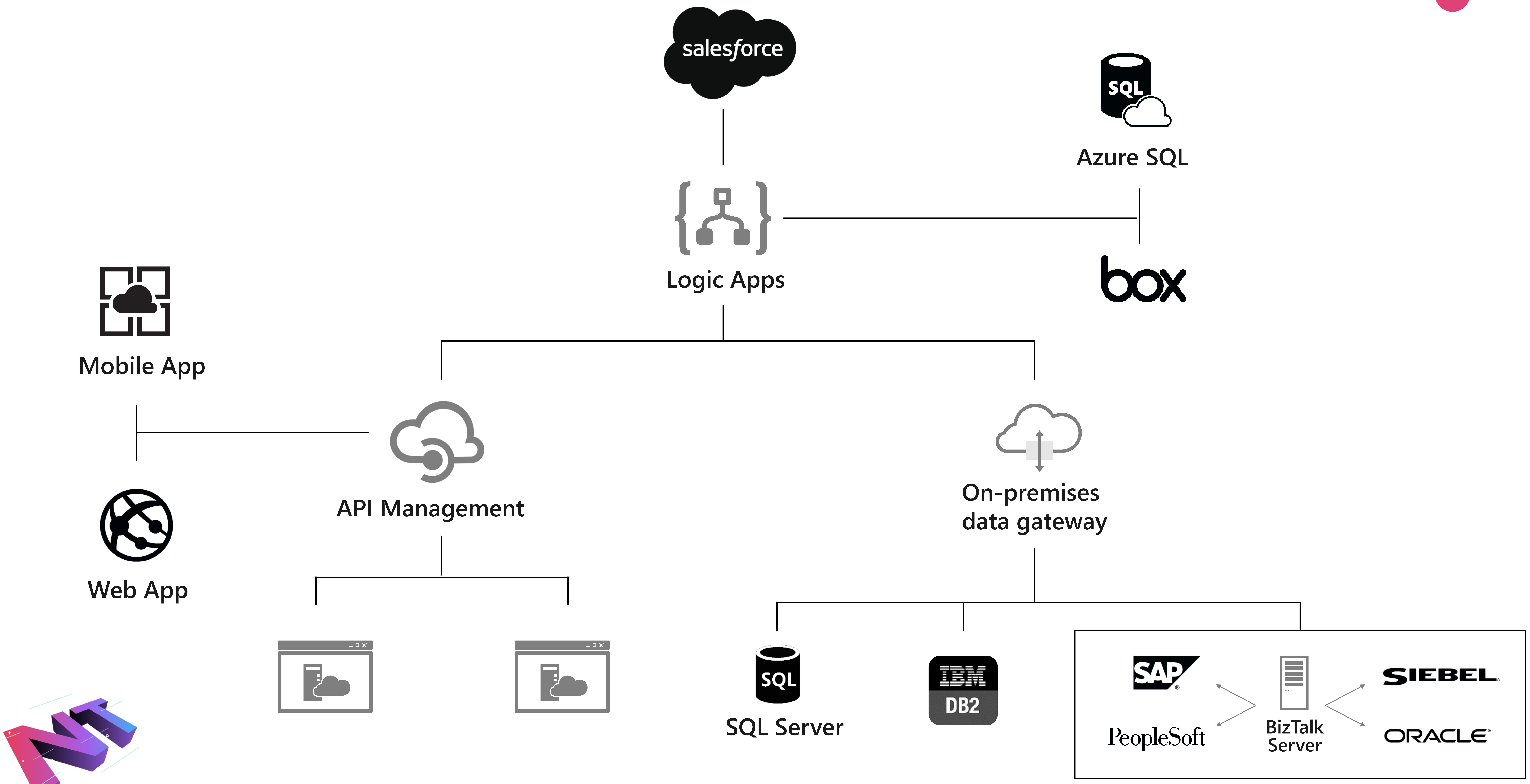


APIs

Workflows

Message

Events

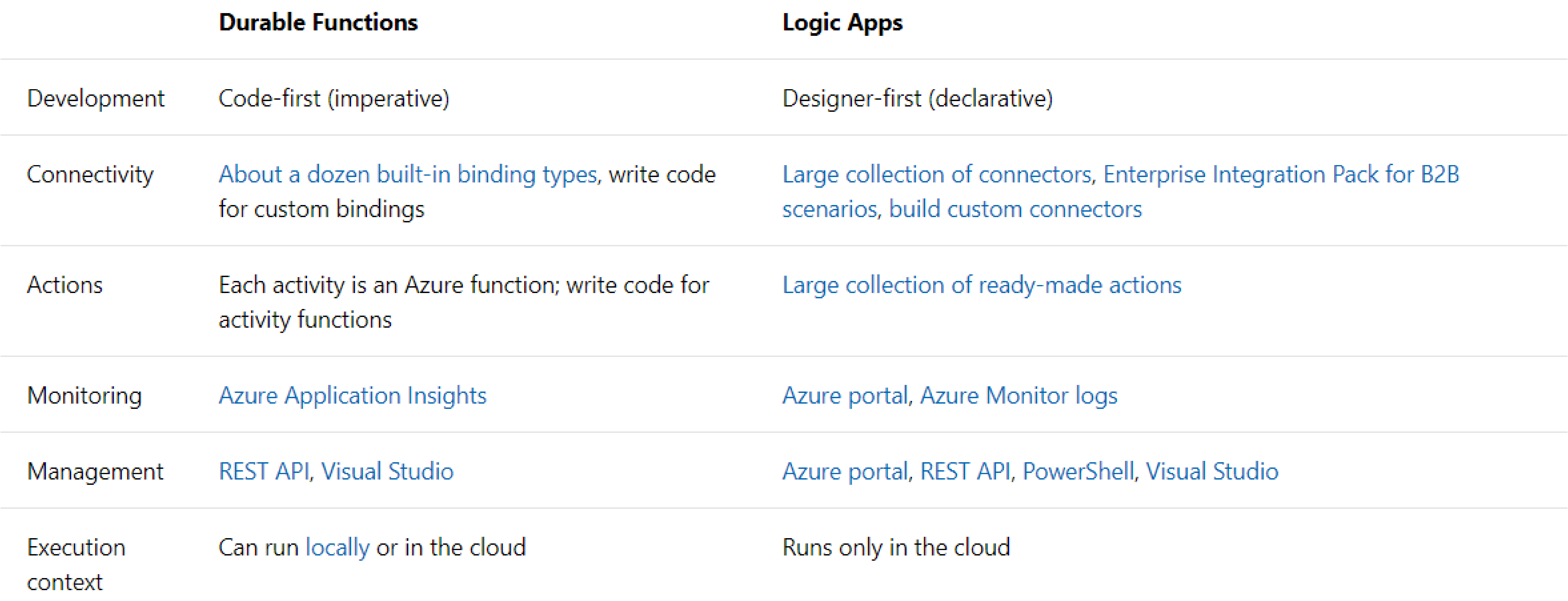

salesforce
Azure SQL
SQL
Logic Apps
box
Mobile App
Web App
API Management
On-premises data gateway
SQL
SQL Server
IBM DB2
SAP
PeopleSoft
BizTalk Server
SIEBEL
ORACLE

#ntk19

Service comparison

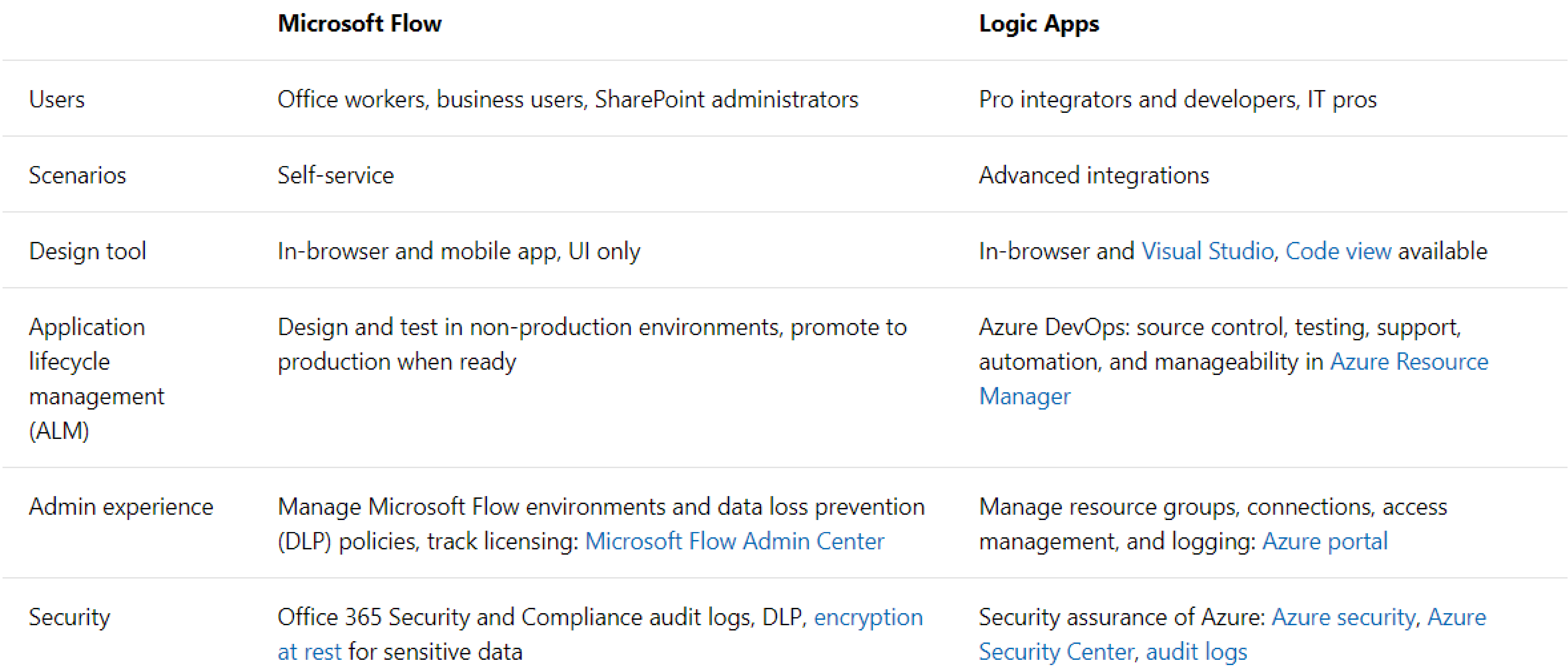
#ntk19

„Azure Functions is a serverless compute service, whereas Azure Logic Apps provides serverless workflows. Both can create complex orchestrations."



	Durable Functions	Logic Apps
Development	Code-first (imperative)	Designer-first (declarative)
Connectivity	About a dozen built-in binding types, write code for custom bindings	Large collection of connectors, Enterprise Integration Pack for B2B scenarios, build custom connectors
Actions	Each activity is an Azure function; write code for activity functions	Large collection of ready-made actions
Monitoring	Azure Application Insights	Azure portal, Azure Monitor logs
Management	REST API, Visual Studio	Azure portal, REST API, PowerShell, Visual Studio
Execution context	Can run locally or in the cloud	Runs only in the cloud

Source: https://docs.microsoft.com/en-us/azure/azure-functions/functions-compare-logic-apps-ms-flow-webjobs



	Microsoft Flow	Logic Apps
Users	Office workers, business users, SharePoint administrators	Pro integrators and developers, IT pros
Scenarios	Self-service	Advanced integrations
Design tool	In-browser and mobile app, UI only	In-browser and Visual Studio, Code view available
Application lifecycle management (ALM)	Design and test in non-production environments, promote to production when ready	Azure DevOps: source control, testing, support, automation, and manageability in Azure Resource Manager
Admin experience	Manage Microsoft Flow environments and data loss prevention (DLP) policies, track licensing: Microsoft Flow Admin Center	Manage resource groups, connections, access management, and logging: Azure portal
Security	Office 365 Security and Compliance audit logs, DLP, encryption at rest for sensitive data	Security assurance of Azure: Azure security, Azure Security Center, audit logs

Source: https://docs.microsoft.com/en-us/azure/azure-functions/functions-compare-logic-apps-ms-flow-webjobs

Hvala.

Microsoft