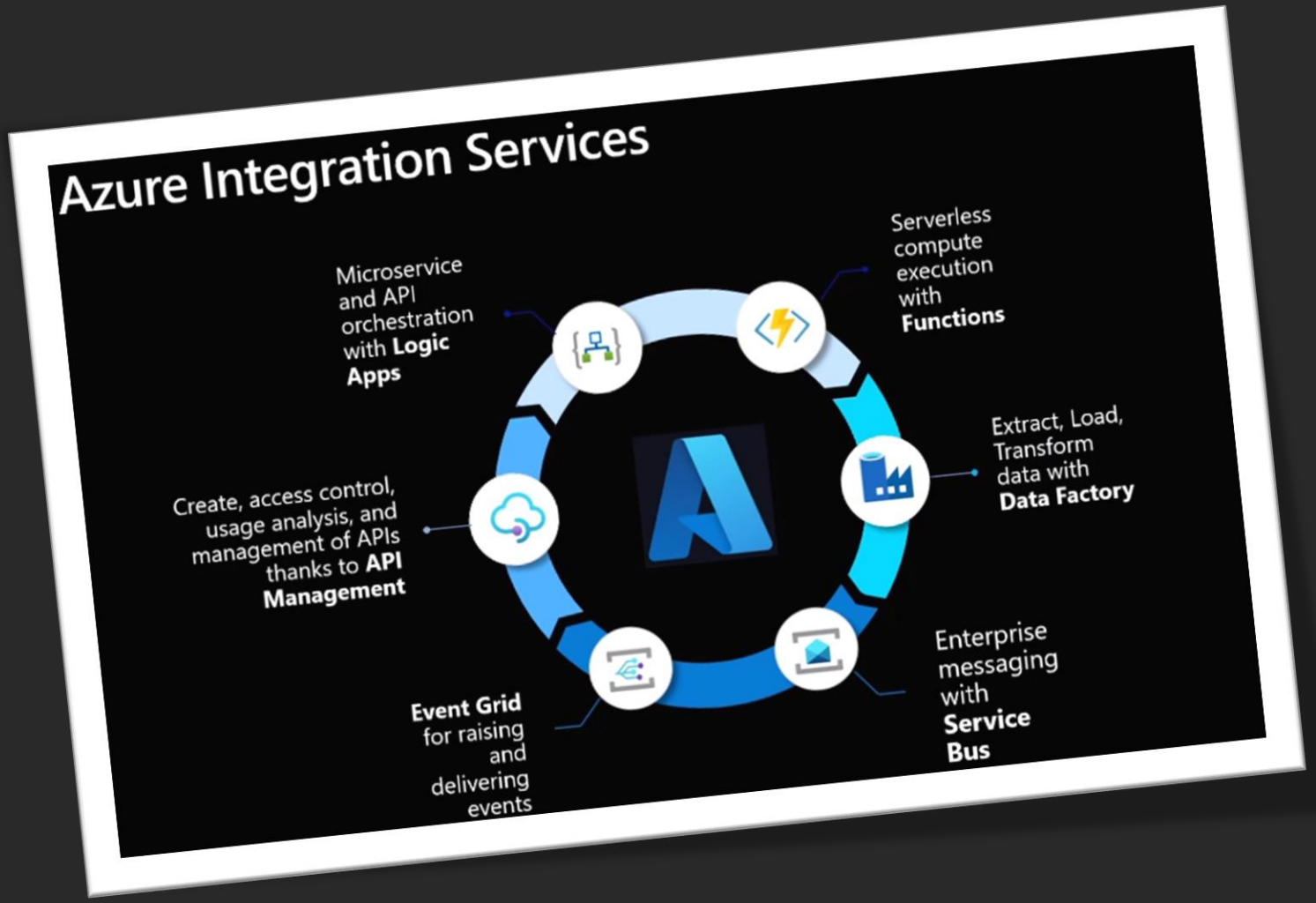
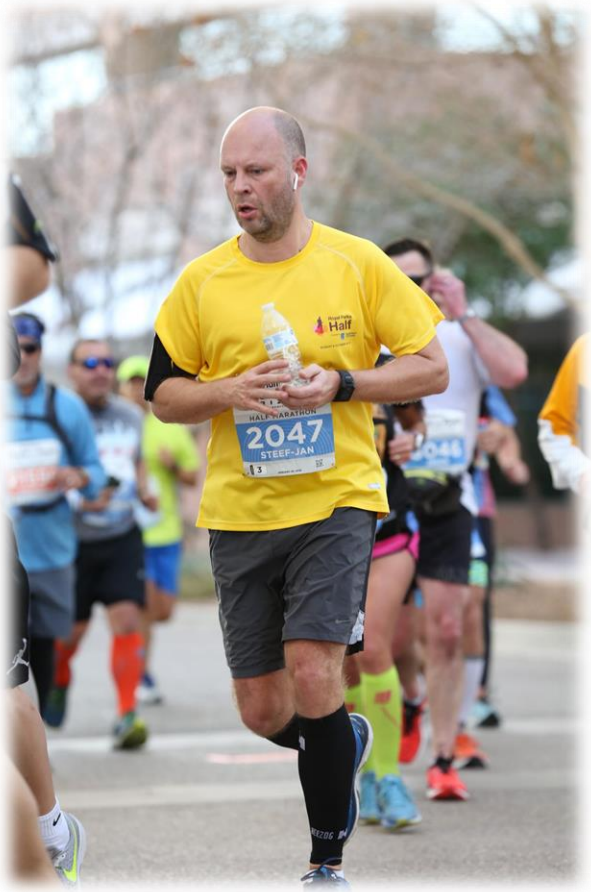
Let Systems Communicate using Azure Integration Services (AIS)

CODE EUROPE



@SteefJan

Me



Technical Integration Architect

Microsoft MVP – Azure

InfoQ Cloud Editor

WAZUG NL board member

Azure Lowlands Organizer

Writer

The Story Outline

- Integration
- Architecture
- Technology
- Real-world cases
- Best practices
- Market
- Take-aways


Tower of Babel

Systems instead of people

paycom

splunk>

twilio

salesforce

SAP Business One

AFAS software

ATLASSIAN

SAP HANA

Microsoft Dynamics 365

servicenow

zendesk

workday

slack

zoom

ORACLE

HubSpot

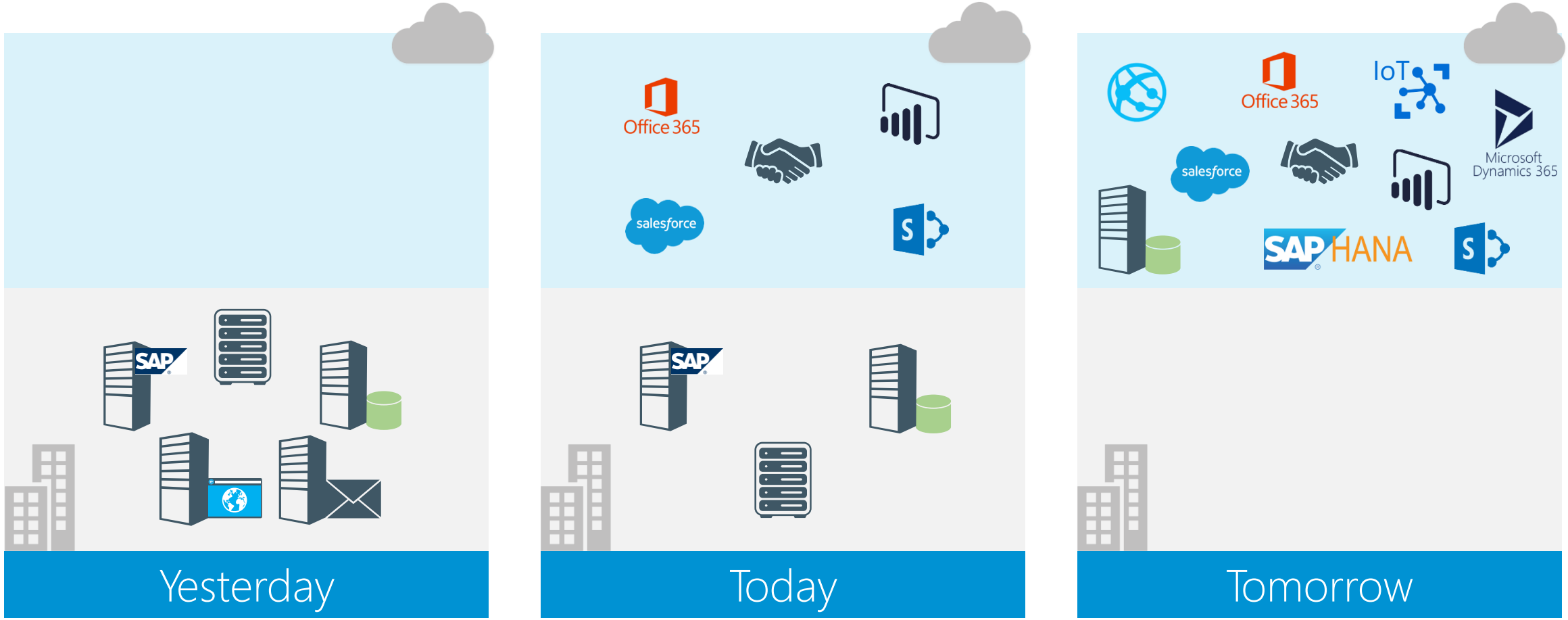
MathWorks

Dropbox

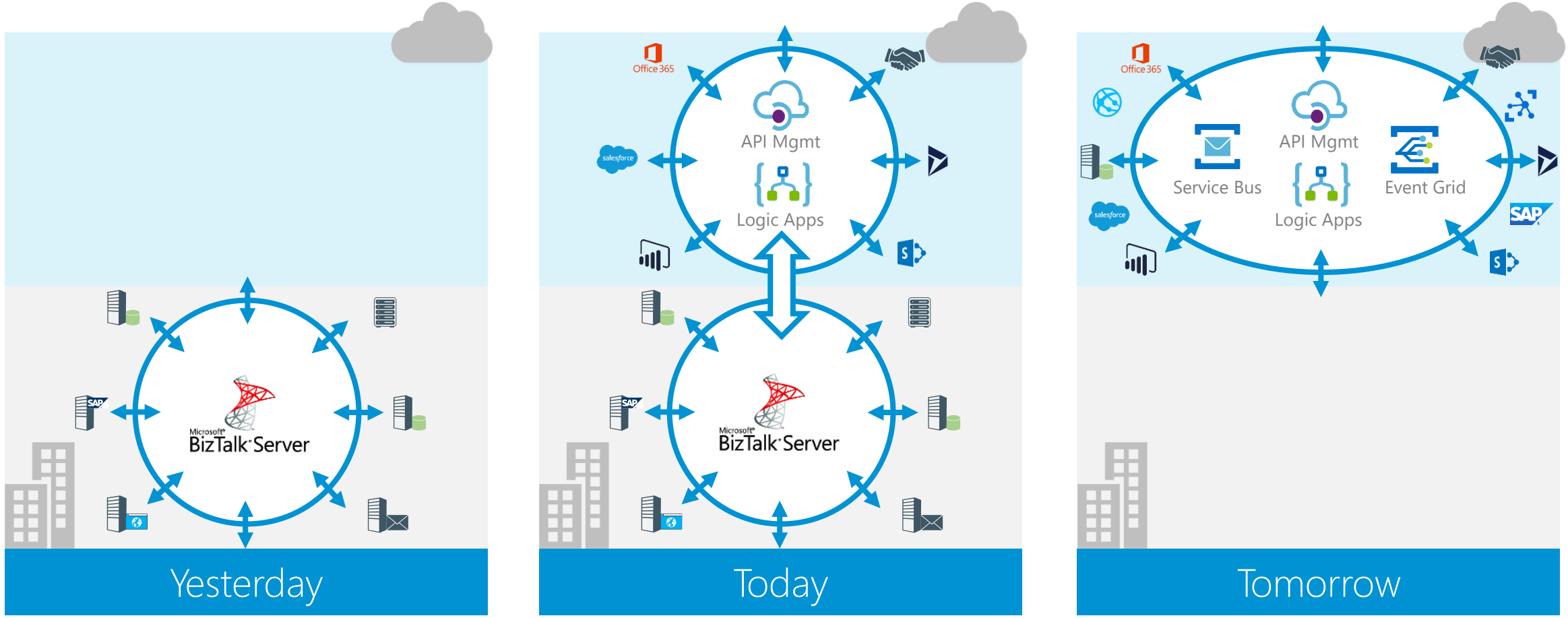
Systems in the cloud

- Finance
- HR
- Planning
- Logistics
- Field Services
- Storage
- Services
- Ticketing

Application Landscape



Integration Landscape




Integration Tech

Azure Integration Services (AIS)

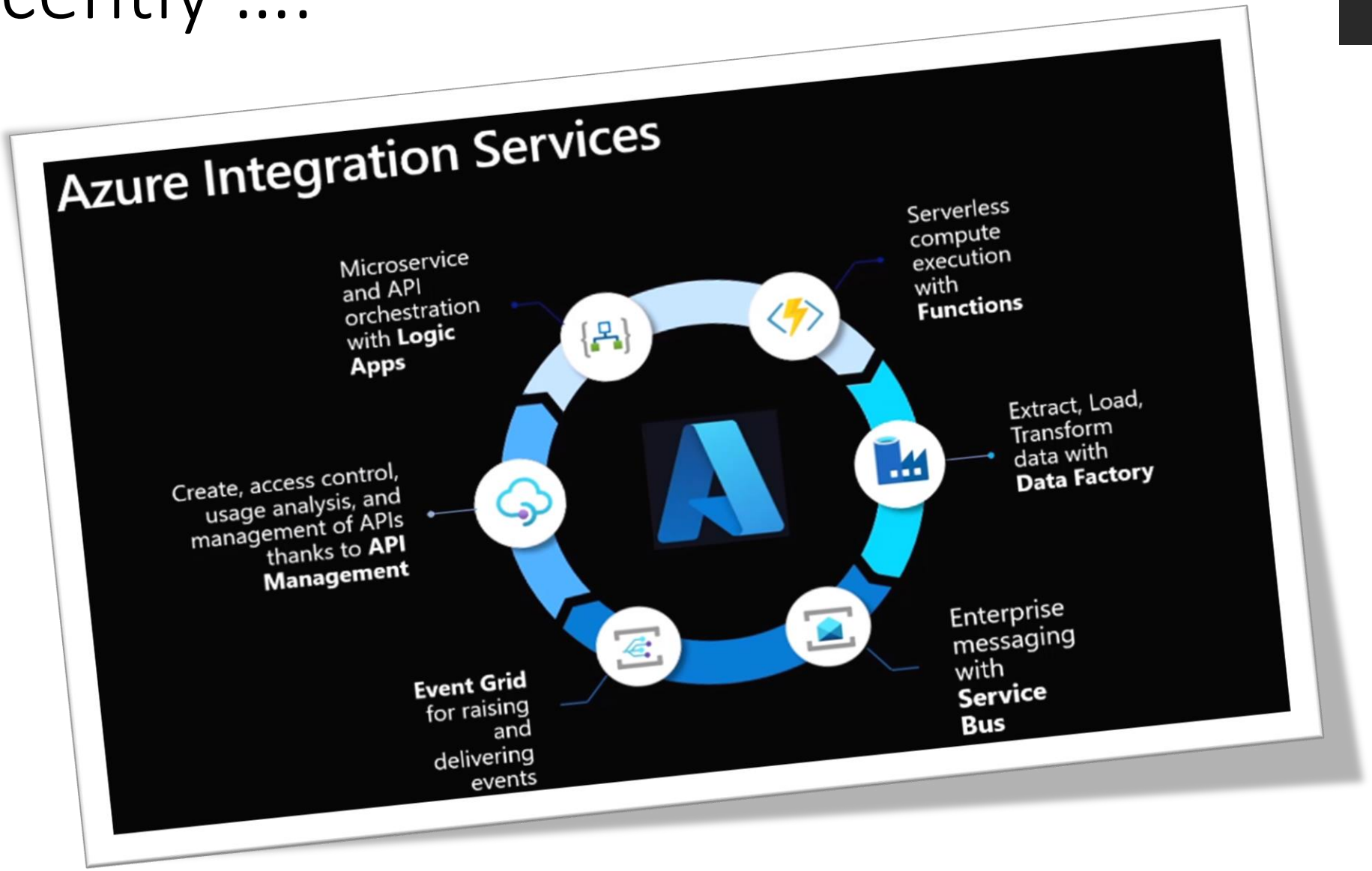
Logic Apps: create workflows and orchestrate business processes to connect hundreds of services in the cloud and on-premises.

Service Bus: connect on-premises and cloud-based applications and services to implement highly secure messaging workflows.

API Management: publish your APIs securely for internal and external developers to use when connecting to backend systems hosted anywhere.

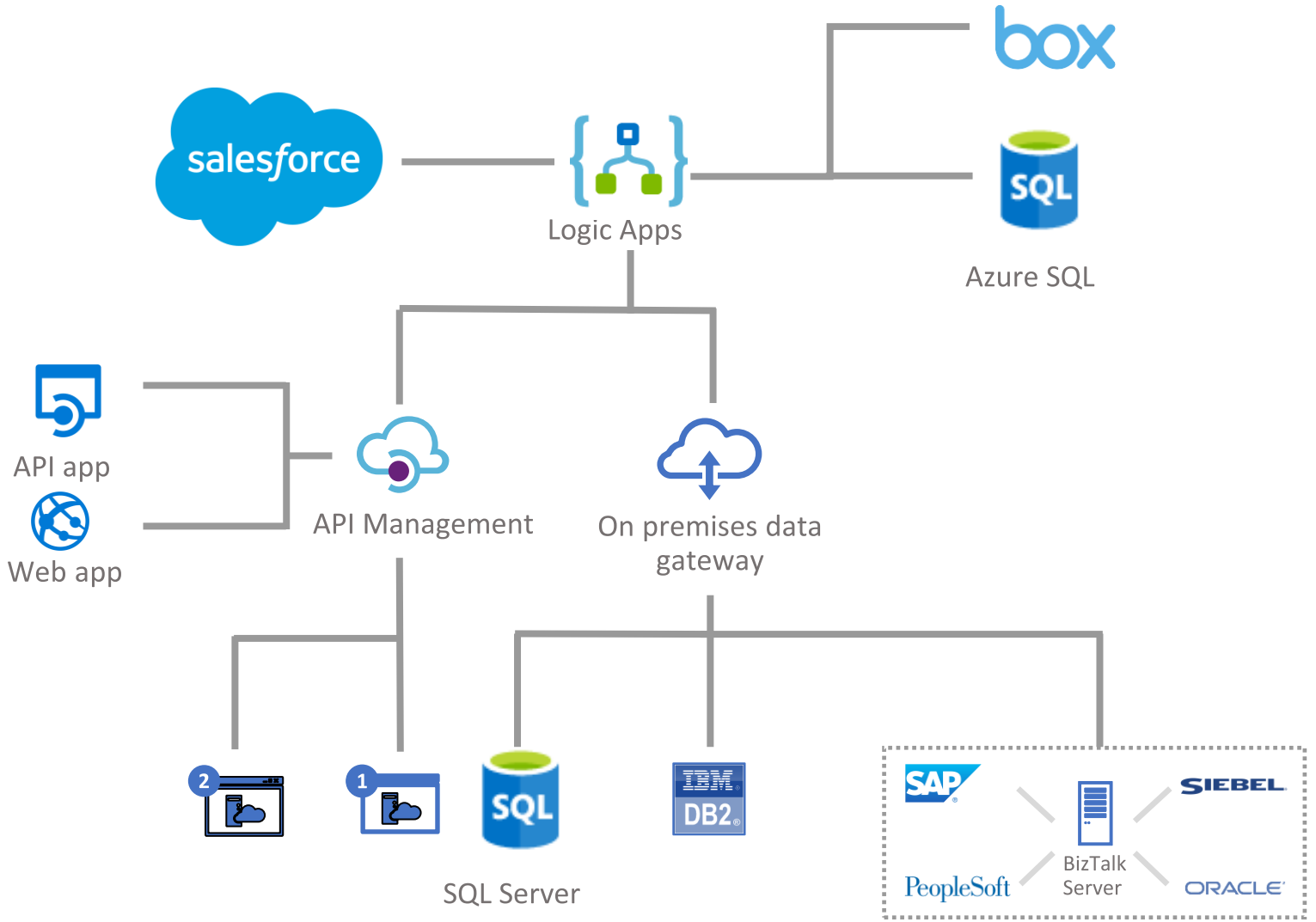
Event Grid: connect supported Azure and third-party services using a fully managed event-routing service with a publish-subscribe model that simplifies event-based app development

Recently ….


Azure Integration Services
Microservice and API orchestration with **Logic Apps**
Serverless compute execution with **Functions**
Extract, Load, Transform data with **Data Factory**
Create, access control, usage analysis, and management of APIs thanks to **API Management**
Enterprise messaging with **Service Bus**
Event Grid for raising and delivering events

What can **Logic Apps** do?

- Connect to on-premises, hybrid, and cloud applications
- Run mission critical, complex integration scenarios with ease
- Build smart integrations leveraging machine learning, cognitive services



Connectors

Cloud APIs and platform functionality

- Hundreds of out of box connectors
  - SaaS, on-prem, protocols, B2B and message manipulation
- Hybrid connectivity
- Hosted and managed within the platform
- Scales to meet your needs
- First class designer experience

Custom Connectors

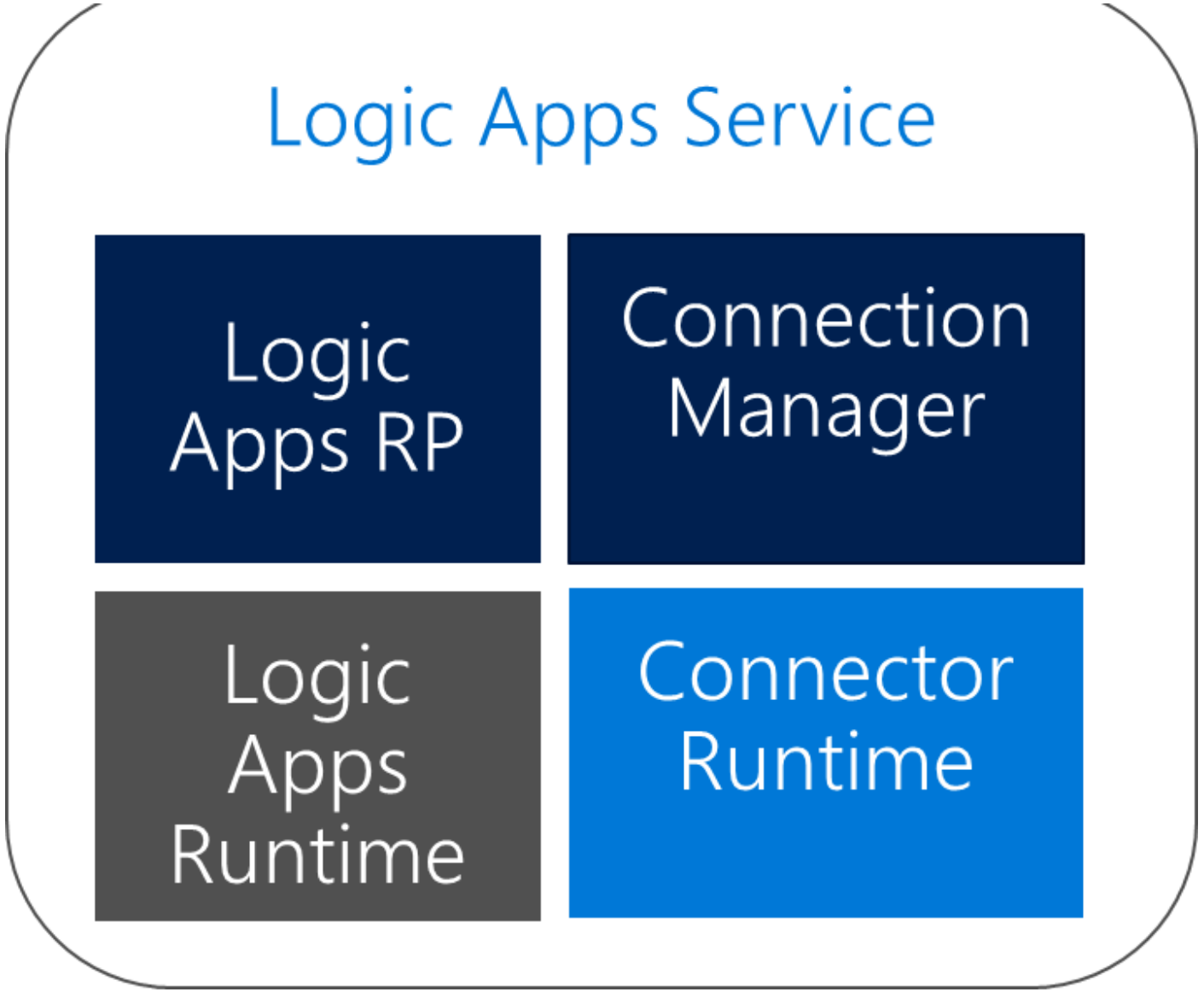
- Access any REST/SOAP API
- Cloud or on-premises
- Simple creation wizard
- Connections and managed secrets
- First class designer experience

API connections

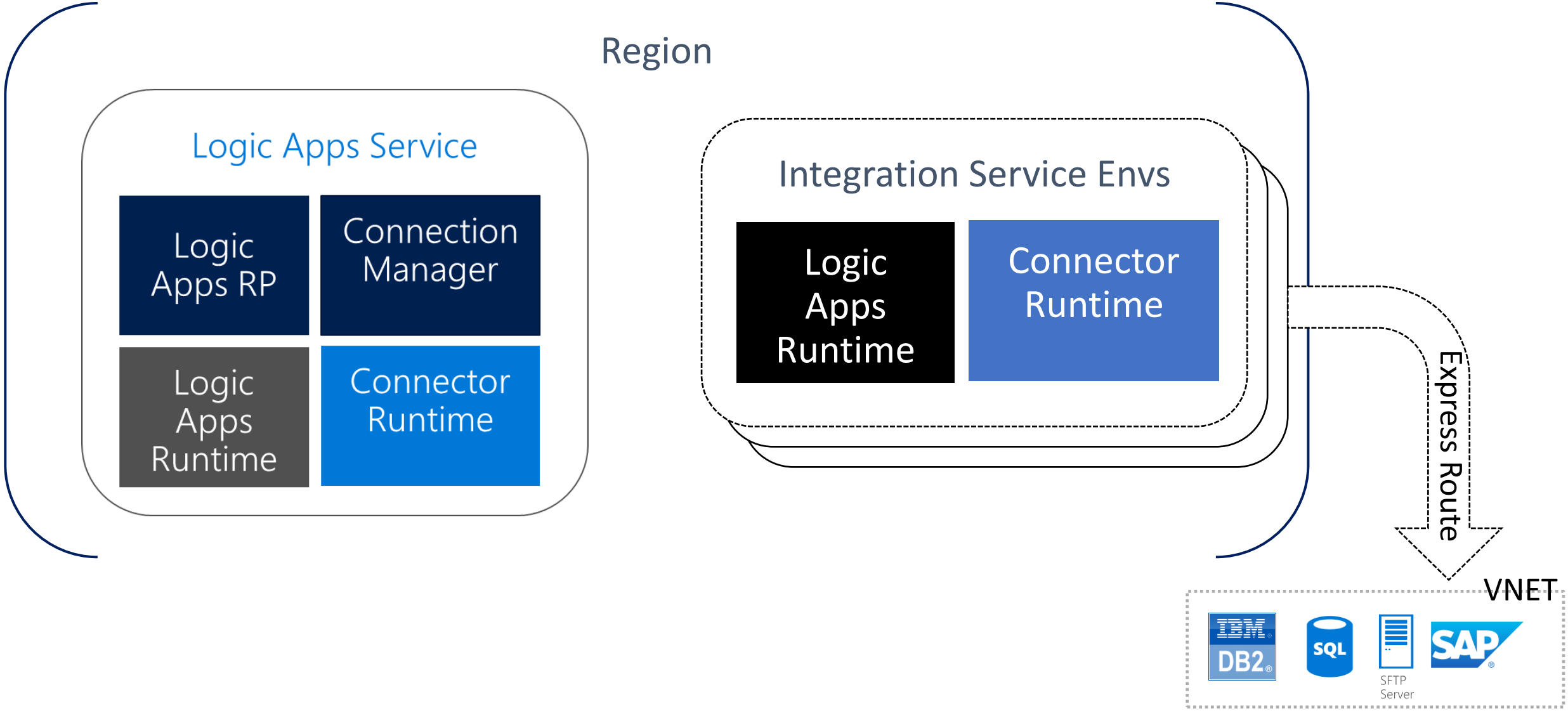
- Authenticate once and reuse
- Differentiate connection configuration
- Simple to deploy
- Portal experience for managing API Connections

Component Architecture

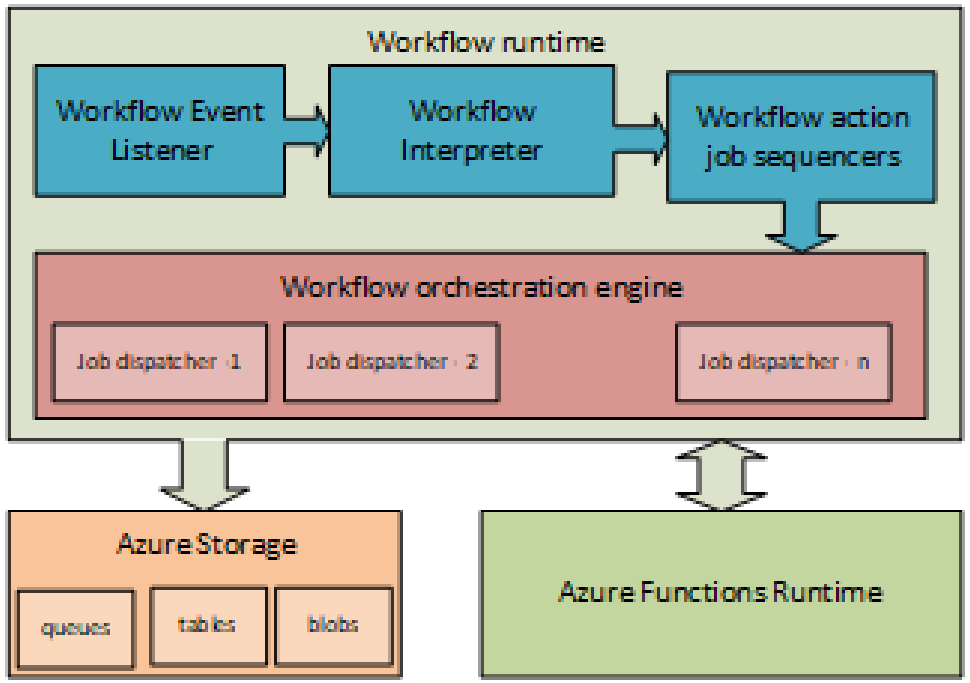
- **Logic Apps RP**
  Reads the workflow definition and breaks down into a composition of tasks with dependencies
- **Logic Apps Runtime**
  Distributed compute/workers are coordinated to complete tasks on-demand
- **Connection Manager**
  Manages connection configuration, credentials and token refreshment
- **Connector Runtime**
  API abstraction via Open API descriptions



Integrated Service Environment (ISE)

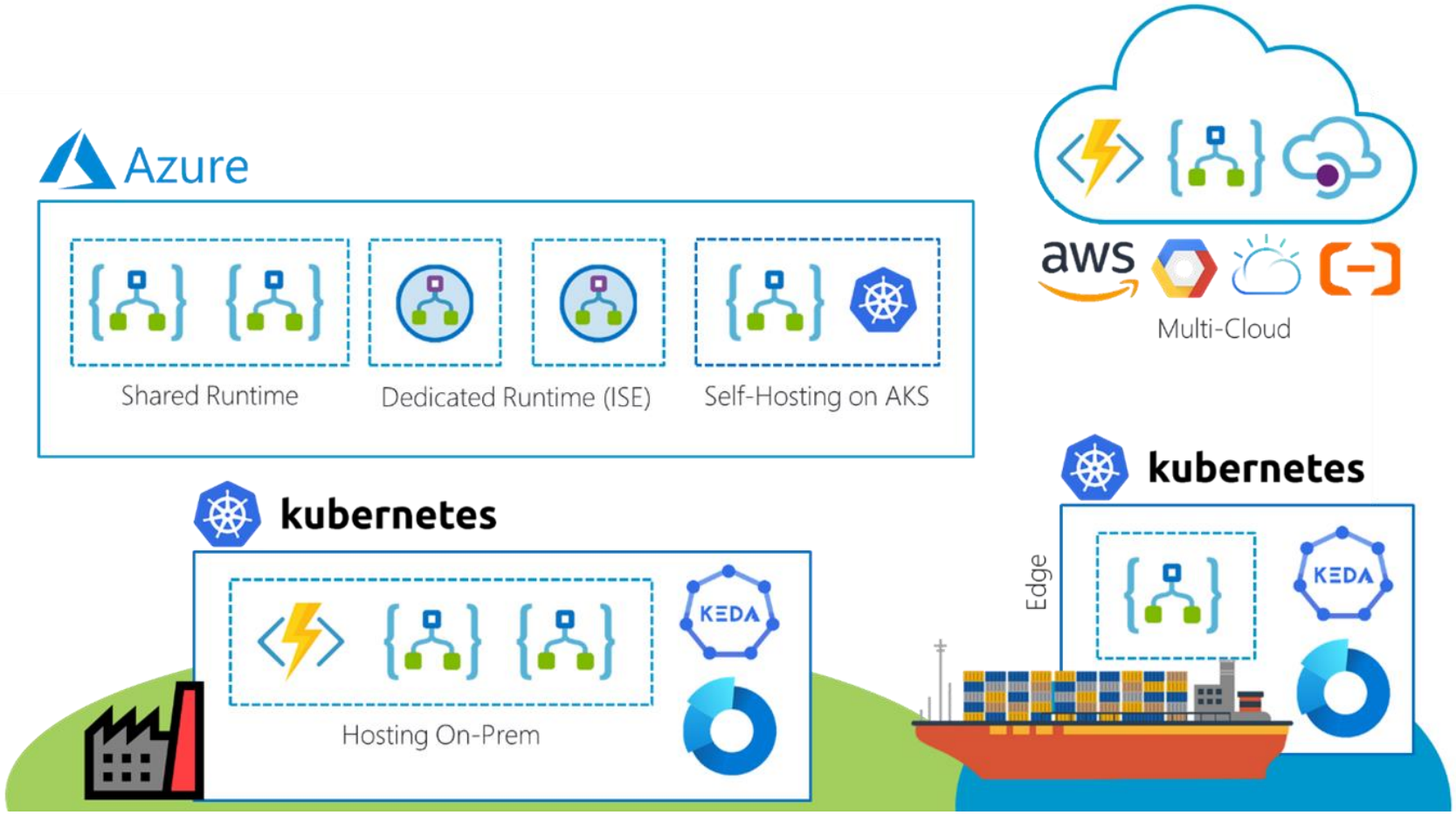


Logic App Standard



- Workflow definition is based upon the **same DSL** as for consumption Logic Apps.
- The Logic Apps runtime implements each action type in a workflow **definition as a job** that is run by the underlying **Logic Apps job orchestration engine**.
- Every workflow definition contains a sequence of actions that can be mapped to a **directed acyclic graph (DAG) of jobs** with various complexity.
- When a workflow is triggered, the Logic Apps runtime looks up the workflow definition and generates the corresponding **jobs** that are organized as a **DAG**, which Logic Apps calls a **job-sequencer**.
- Each workflow definition is **compiled** into a job-sequencer that **orchestrates** the running of jobs for that workflow definition.
- The job orchestration engine is a **horizontally scalable distributed service** that can run DAGs with large numbers of jobs.

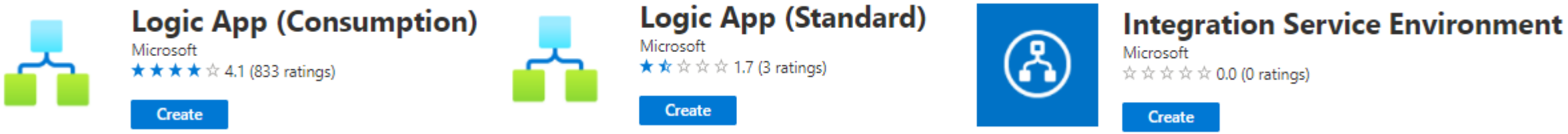
Overview Hosting Options



- Logic App Standard allows to containerize your workflow and extend the deployment to different environments.

Hosting Options

We have choices:



- Logic App – Consumption Plan (Serverless Option)
- Logic App – Standard supported by an Workflow Service Plan – WS1, WS2, WS3
- Integrated Service Environment (ISE) – Isolated environment

Note: Workload, and non-functionals are drivers for the hosting option.

What does **service bus** deliver?

Fully **managed** service in Azure offering a **reliable** and **secure** platform for asynchronous transfer of data and state

- Communication backbone for many sophisticated cloud solutions
- Entities like relay, queues, topics and subscriptions
- Enables hybrid cloud solutions
- Compliance with AMQP
- Richer application semantics like
  - FIFO
  - Deduplication
  - Transactional behavior and atomicity

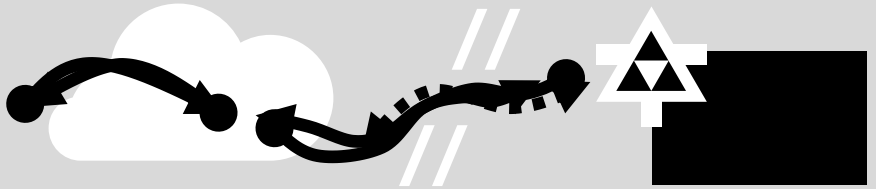
Azure Service Bus

Service Bus Relay:

Scenario: You have on-premise systems that you need to communicate with directly from outside your organization…

Solution: Internal Web Services are exposed securely via the Relay which passes calls into the on-premise service and back to the calling clients

Relay: Two Way Call into On-Premise Service



Service Bus Queues:

Scenario: Multiple systems and remote clients need to send business events to head office which processes these messages under varying load.

Solution: Queues decouple senders from receivers, multiple receivers can handle varying load, simple to add new senders without impact.

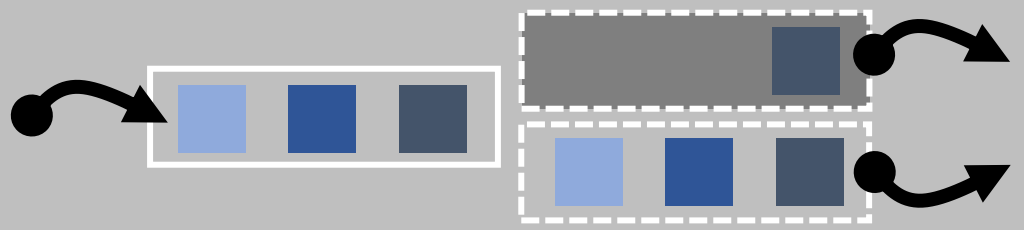
Message Queue: FIFO Resilient Queue

Service Bus Topics:

Scenario: Multiple actions must be taken as a result of incoming messages from external systems, but these actions frequently change.

Solution: Topics are special queues that have subscriptions which contain rules to determine which messages a subscription will contain.

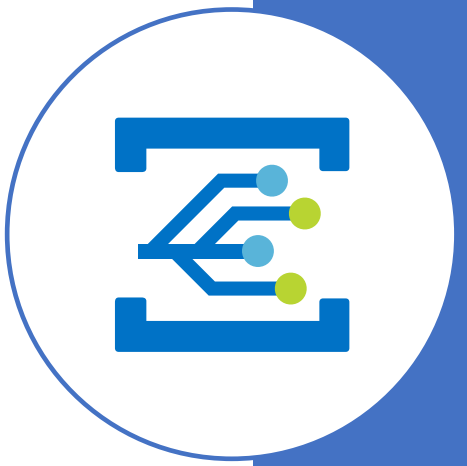
Topic: Queue with 1:n rule-based subscriptions



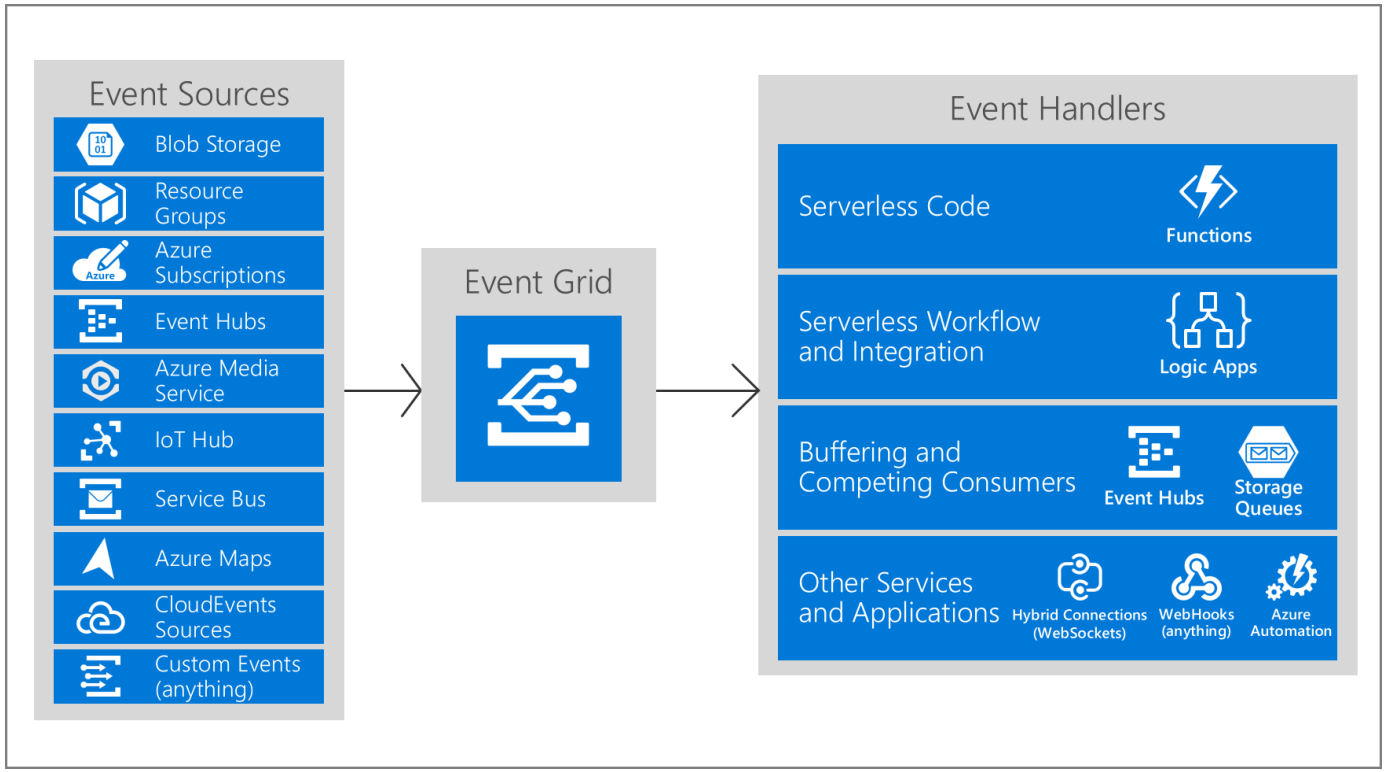
What is **Azure Event Grid**?

Eventing **backplane** that enables event-driven reactive programming

- Event consumption of Azure and Non-azure resources
- Allows reactive programming, and events are discrete facts
- The event message has the information you need to react to changes in services and applications
- Light weight
- Low cost



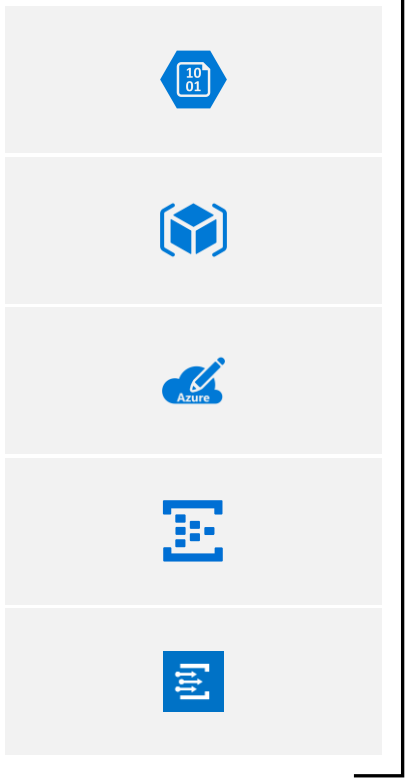
Azure Event Grid



Manage all events in **one place**

CODE EUROPE

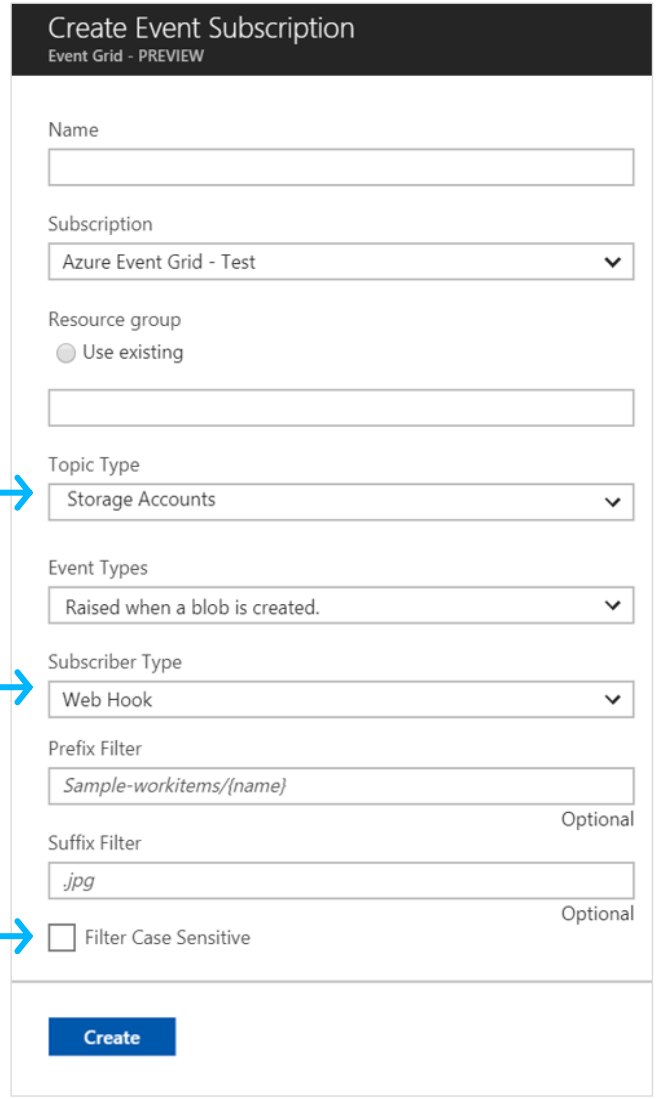
Event publishers



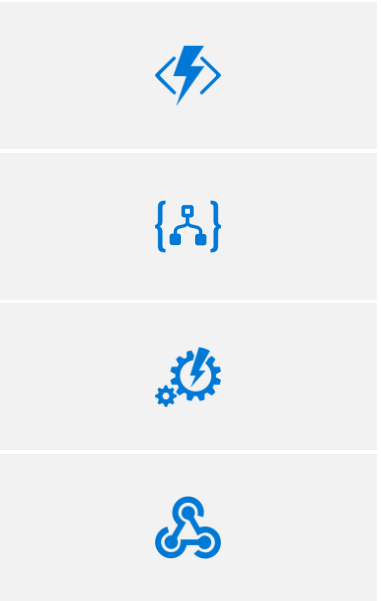
Subscribe to pre-defined system events in Azure or create your own custom topics

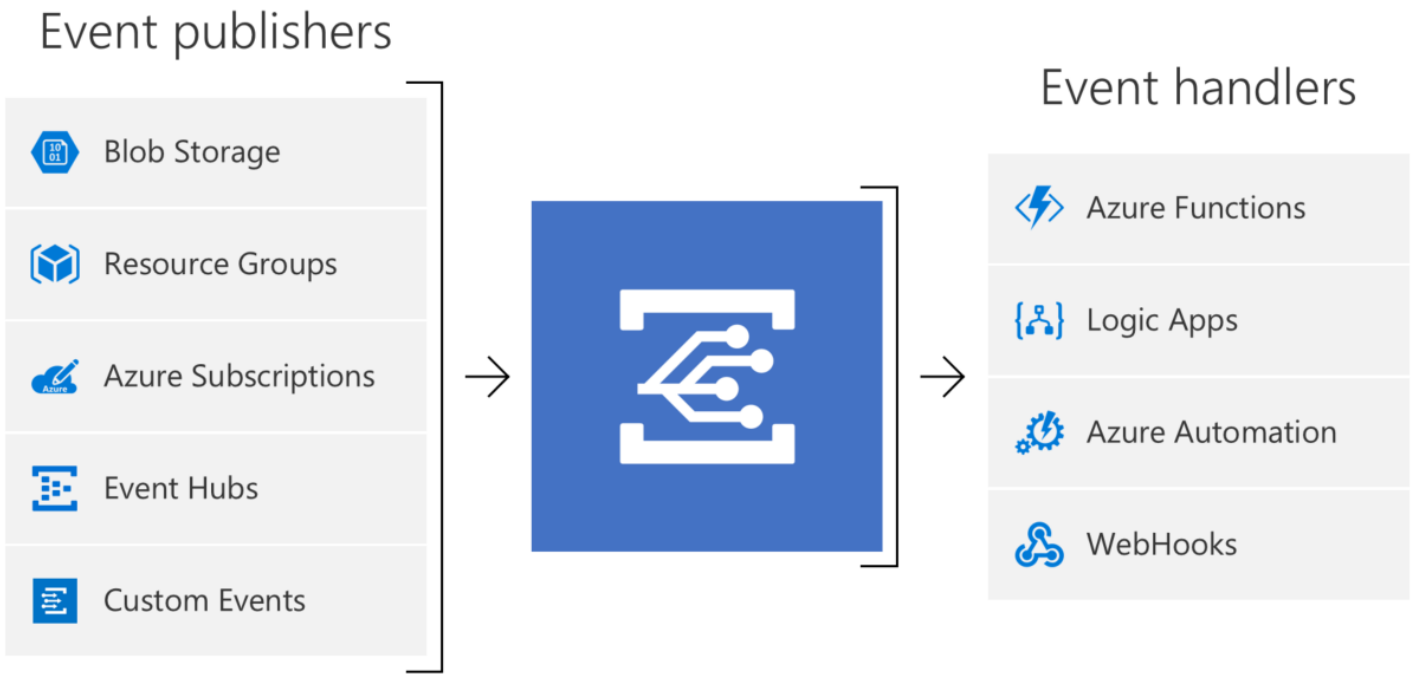
Route events to any end-points, Azure or even beyond

Enable filtering and efficient routing of events



Event handlers





Concepts

- Events: what happened
- Event Publishers: where it took place
- Topics: where publishers send events
- Event Subscriptions: how you receive events
- Event Handlers: the app or service reacting to the event

Event Grid Schema

Proprietary Schema:

```
[
 {
   "topic": string,
   "subject": string,
   "id": string,
   "eventType": string,
   "eventTime": string,
   "data":{
    object-unique-to-each-publisher
   },
   "dataVersion": string,
   "metadataVersion": string
 }
]
```

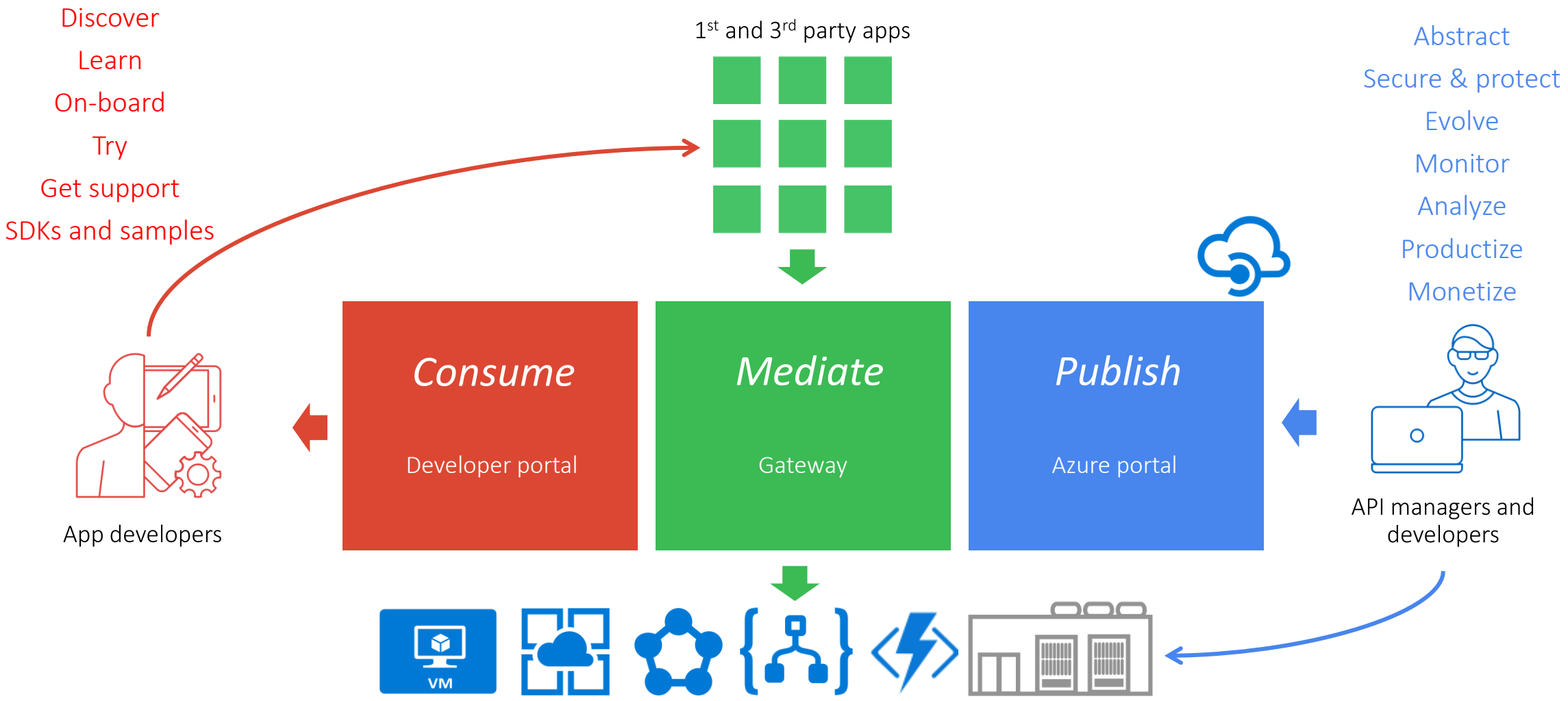
CloudEvent Schema:

```
[
 {
   "specversion":  string,
   "type": string,
   "source": string,
   "id": string,
   "time": string,
   "subject": string,
   "dataschema": string
   "data":{
    object-unique-to-each-publisher
   },
}
]
```

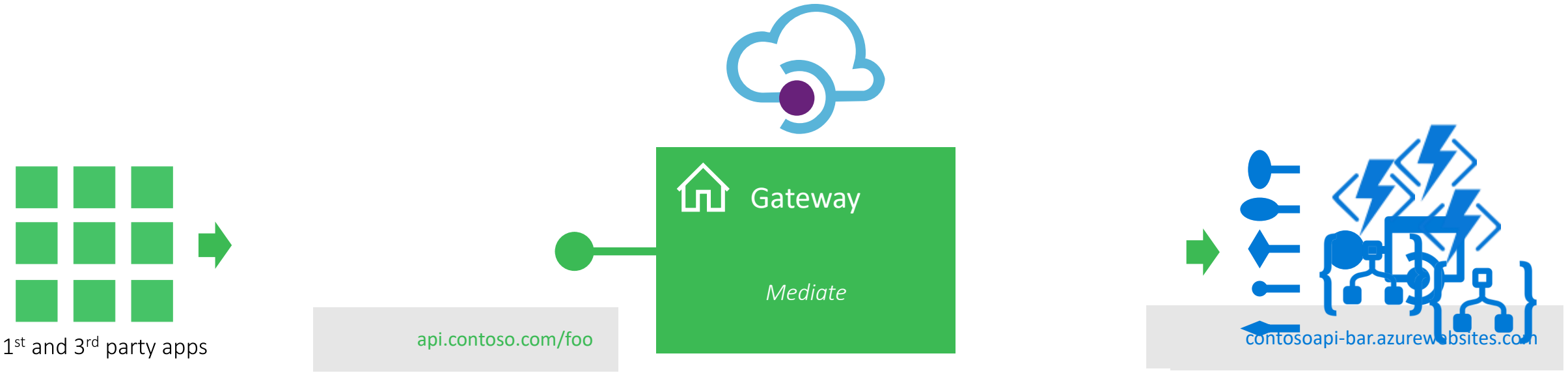
What value does **API Management** bring?

- A fully **managed** service that enables customers to publish, secure, transform, maintain, and monitor APIs.
- **API gateway** providing a simplified and secure façade for serverless Azure resources such as Logic Apps, APIs, and Functions.

API Management - a **hub** for enterprise APIs

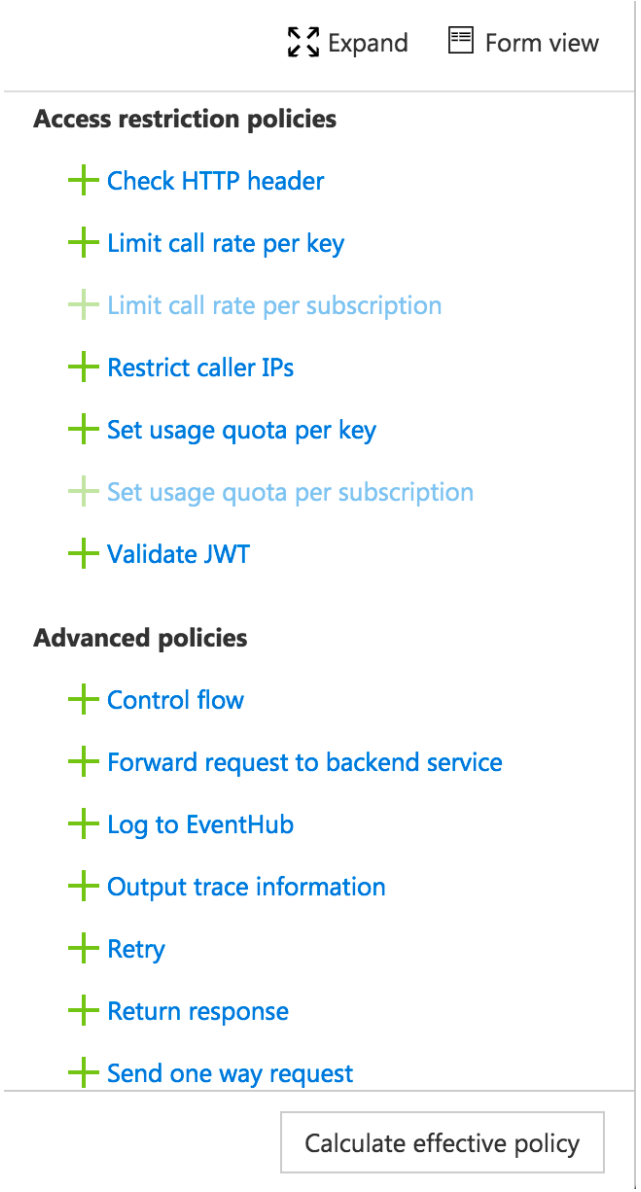


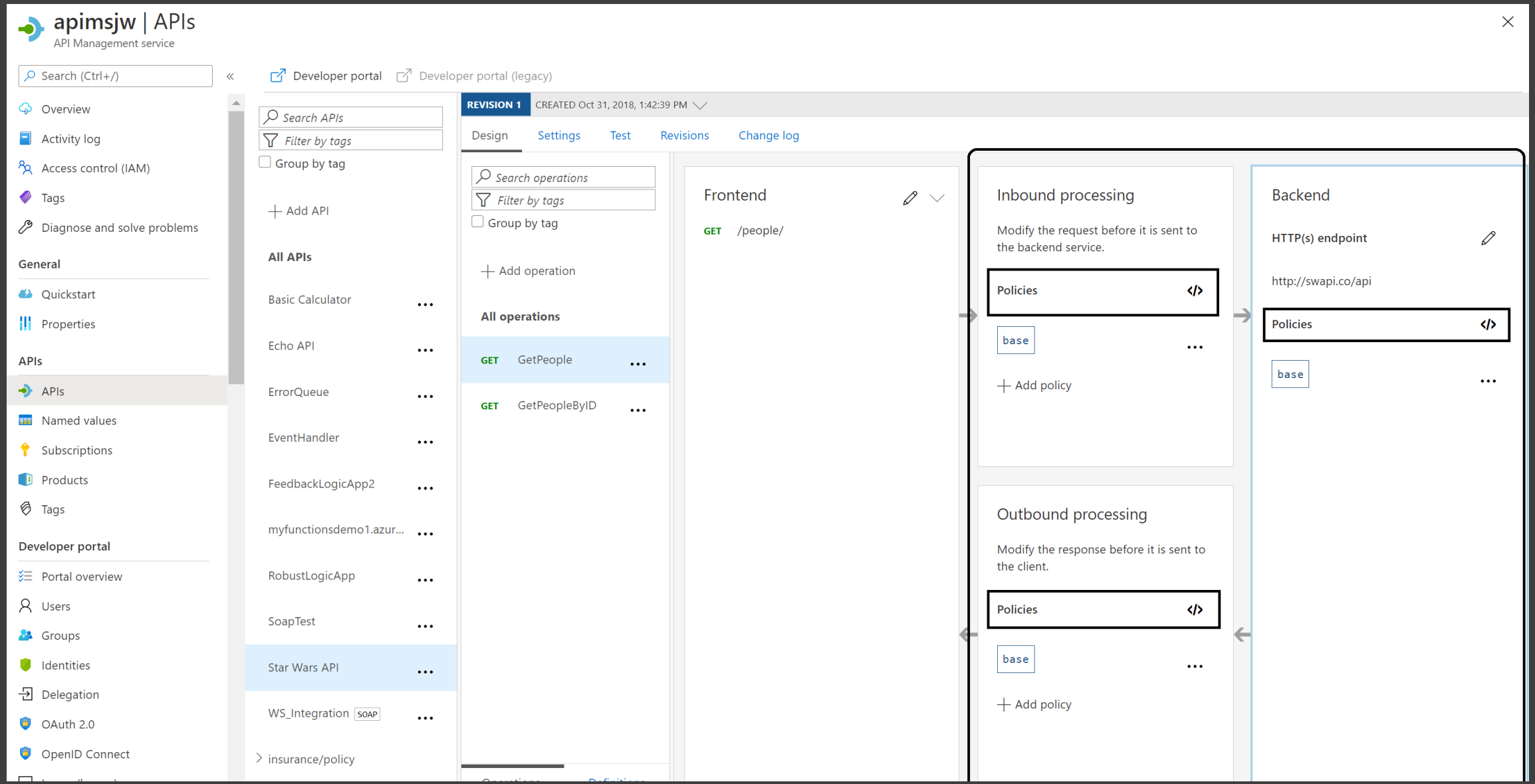
Façade and **front** door



Policies

- Encapsulate common API management functions
  - Access control, Protection, Transformation, Caching, …
- Chained together into a pipeline
- Mutate request context or change API behavior
- Set in the inbound and outbound directions
- Can be triggered on error
- Applied at a variety of scopes



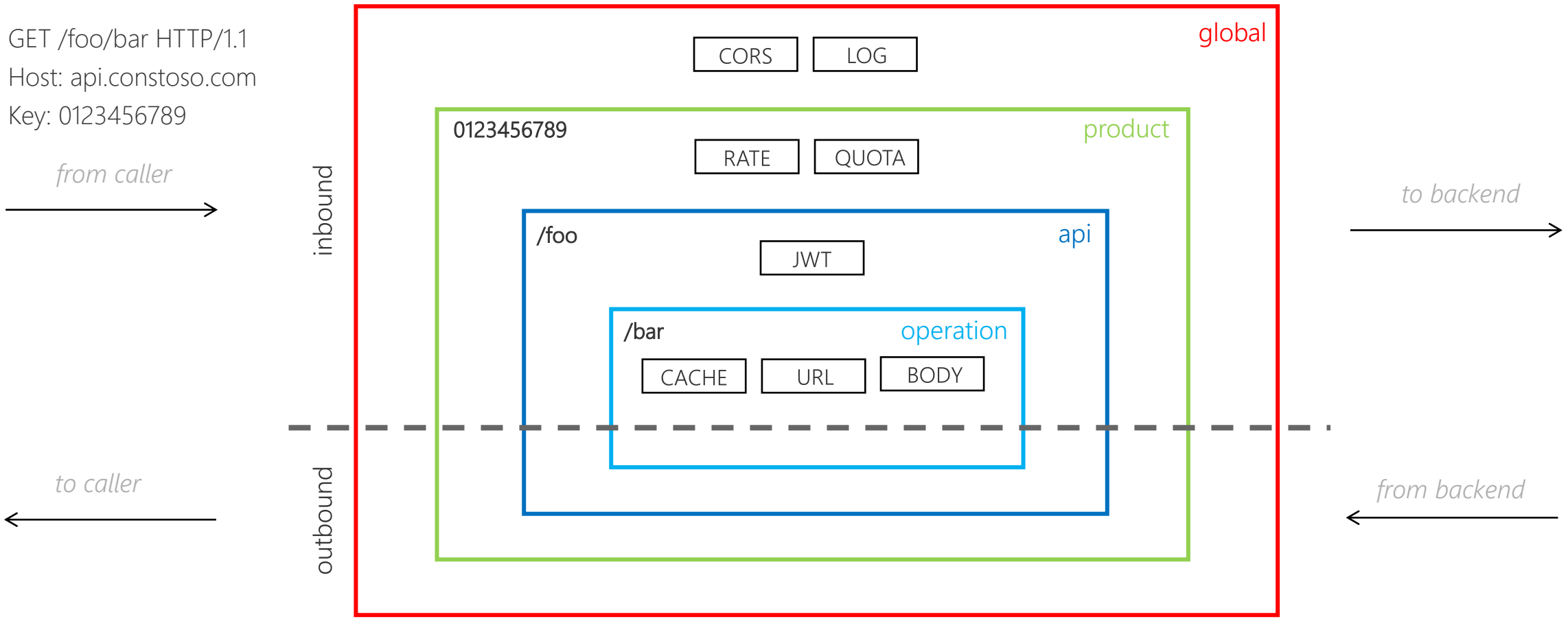

apimsjw | APIs
API Management service
Search (Ctrl+/)
Developer portal
Developer portal (legacy)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
General
Quickstart
Properties
APIs
APIs
Named values
Subscriptions
Products
Tags
Developer portal
Portal overview
Users
Groups
Identities
Delegation
OAuth 2.0
OpenID Connect
Search APIs
Filter by tags
Group by tag
Add API
All APIs
Basic Calculator
Echo API
ErrorQueue
EventHandler
FeedbackLogicApp2
myfunctionsdemo1.azur...
RobustLogicApp
SoapTest
Star Wars API
WS_Integration
SOAP
insurance/policy
REVISION 1
CREATED Oct 31, 2018, 1:42:39 PM
Design
Settings
Test
Revisions
Change log
Search operations
Filter by tags
Group by tag
Add operation
All operations
GET GetPeople
GET GetPeopleByID
Frontend
GET /people/
Inbound processing
Modify the request before it is sent to the backend service.
Policies
base
Add policy
Outbound processing
Modify the response before it is sent to the client.
Policies
base
Add policy
Backend
HTTP(s) endpoint
http://swapi.co/api
Policies
base

Star Wars API > GetPeople > Policies

Show snippets Expand

```
<!--
    IMPORTANT:
    - Policy elements can appear only within the <inbound>, <outbound>, <backend> section elements.
    - To apply a policy to the incoming request (before it is forwarded to the backend service), place a corresp
    - To apply a policy to the outgoing response (before it is sent back to the caller), place a corresponding p
    - To add a policy, place the cursor at the desired insertion point and select a policy from the sidebar.
    - To remove a policy, delete the corresponding policy statement from the policy document.
    - Position the <base> element within a section element to inherit all policies from the corresponding sectio
    - Remove the <base> element to prevent inheriting policies from the corresponding section element in the enc
    - Policies are applied in the order of their appearance, from the top down.
    - Comments within policy elements are not supported and may disappear. Place your comments between policy el
-->
<policies>
    <inbound>
        <base />
    </inbound>
    <backend>
        <base />
    </backend>
    <outbound>
        <base />
    </outbound>
    <on-error>
        <base />
    </on-error>
</policies>
```

Policy scopes



Azure Functions

CODE EUROPE

Events

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

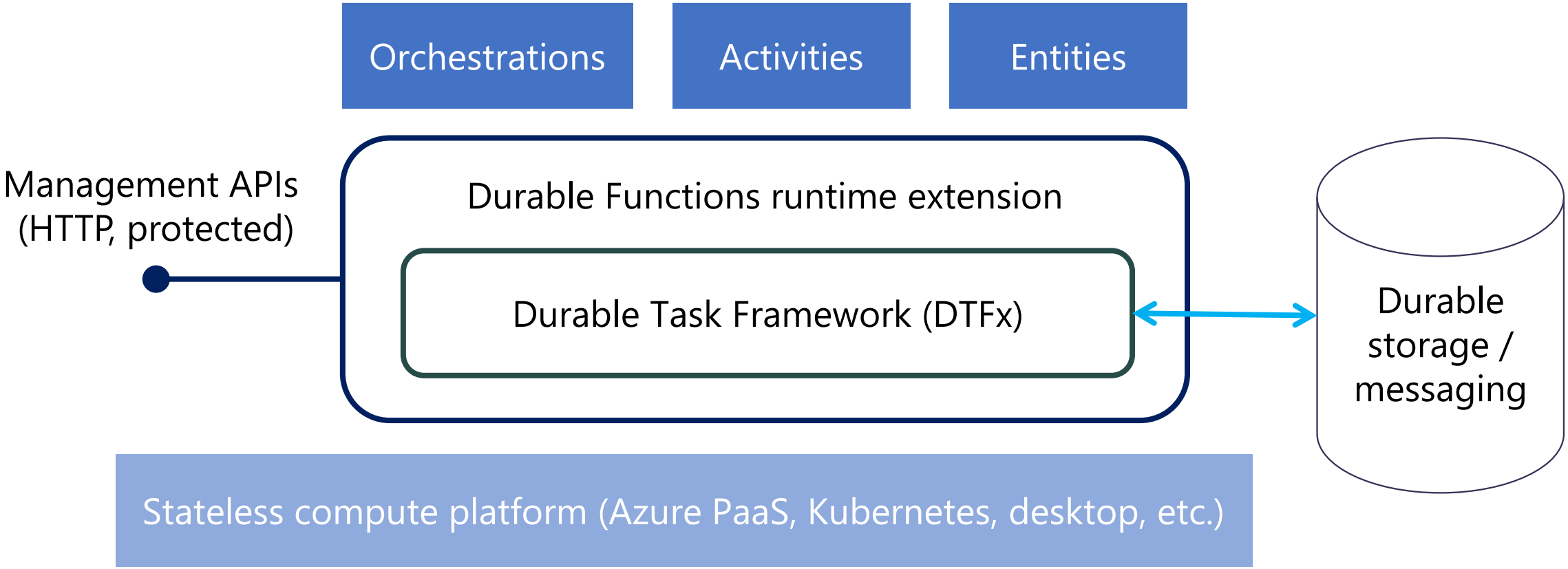
Code

Author functions in C#, F#, JavaScript, TypeScript, Java, Python, PowerShell, and more

Outputs

Send results to an ever-growing collection of services

Durable Functions

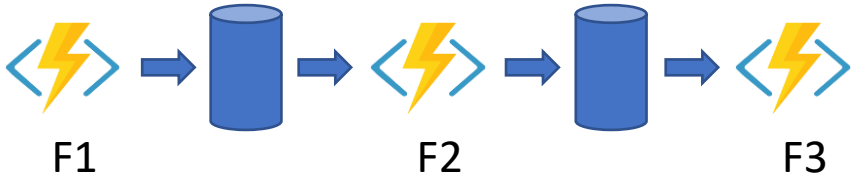


How it looks in code

Orchestrator Function

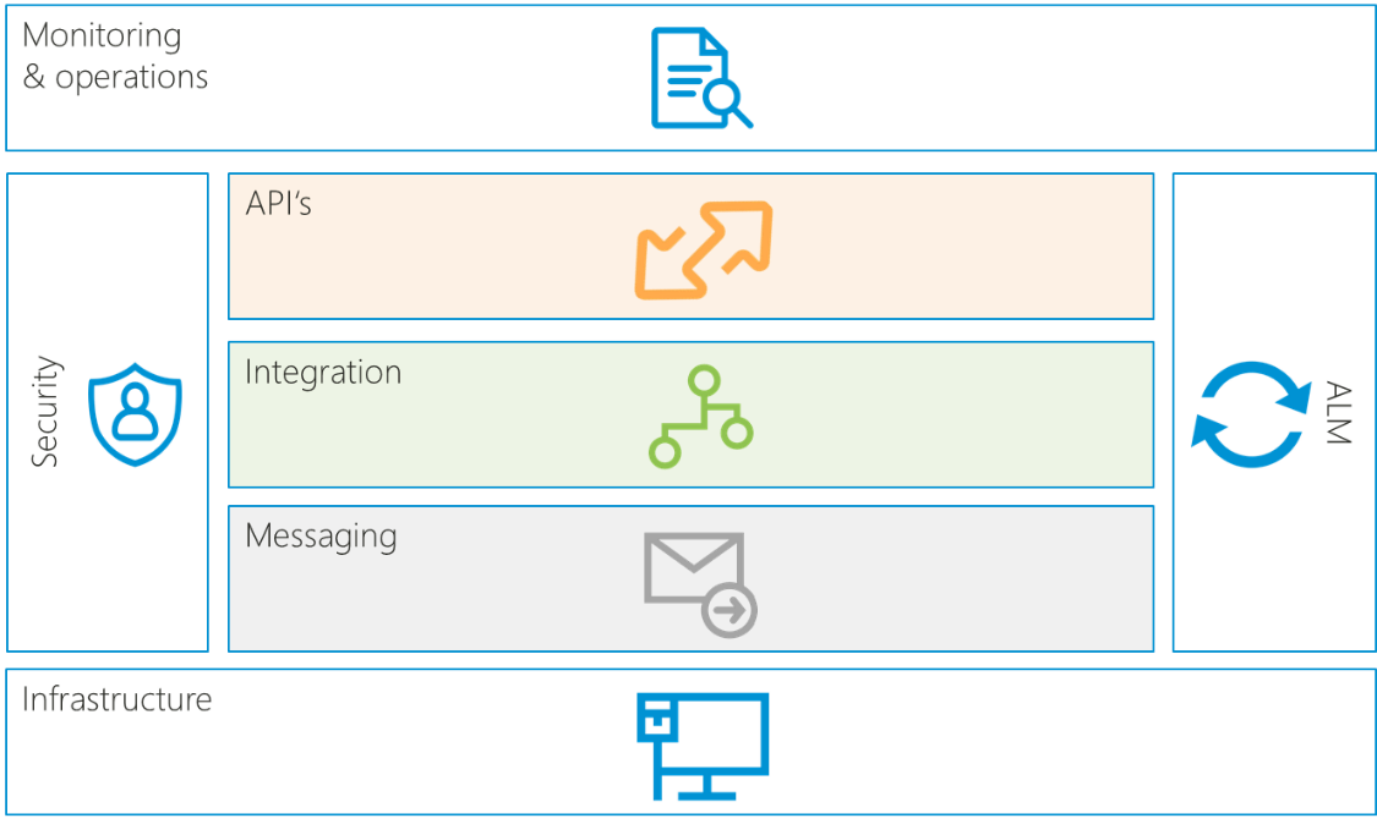
```
// calls functions in sequence
public static async Task<object> Run(IDurableOrchestrationContext ctx)
{
    try
    {
        var x = await ctx.CallActivityAsync("F1");
        var y = await ctx.CallActivityAsync("F2", x);
        return await ctx.CallActivityAsync("F3", y);
    }
    catch (Exception)
    {
        // error handling/compensation can go here (or anywhere)
    }
}
```

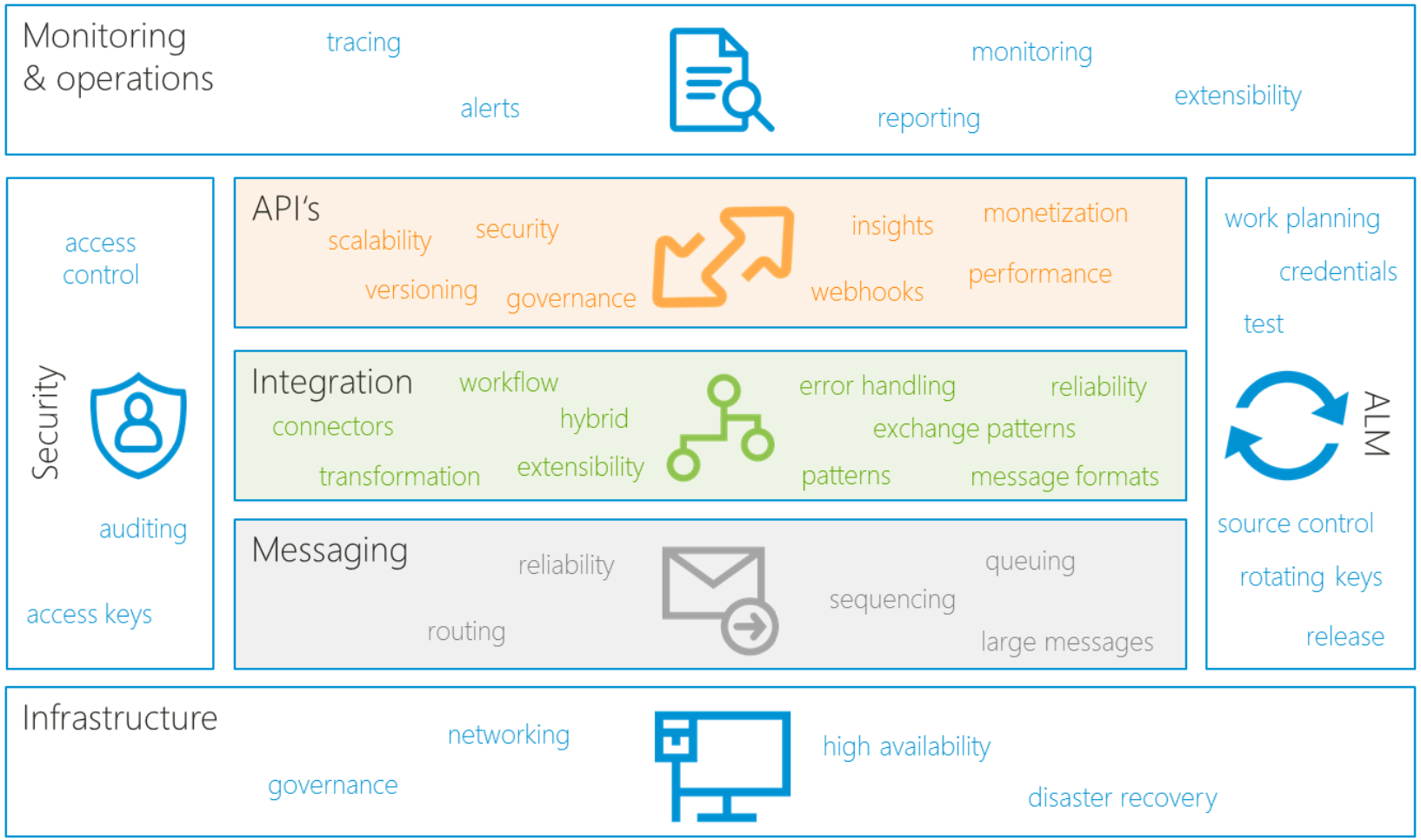
Activity Functions



Solution Architecture

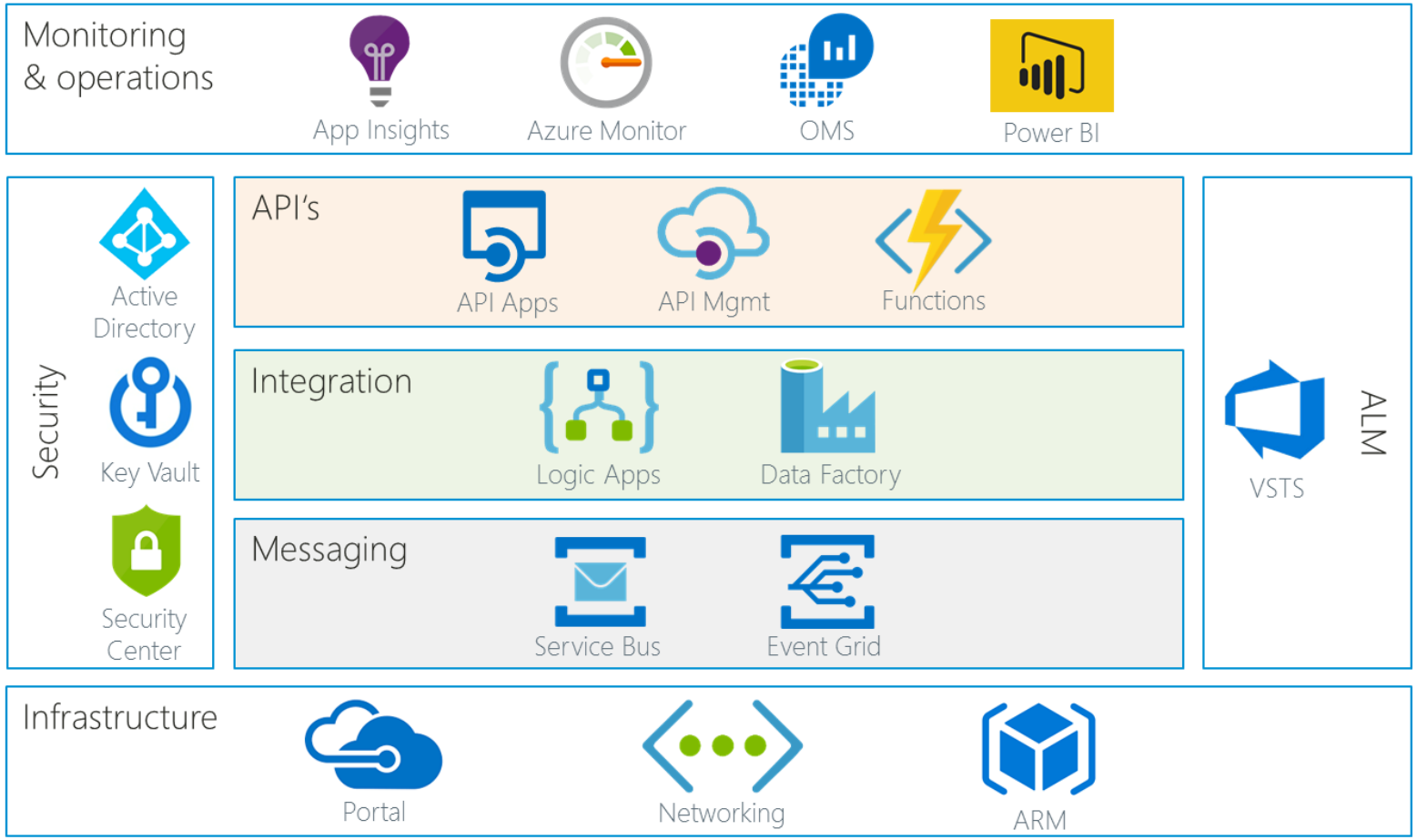
Architectural Building Blocks



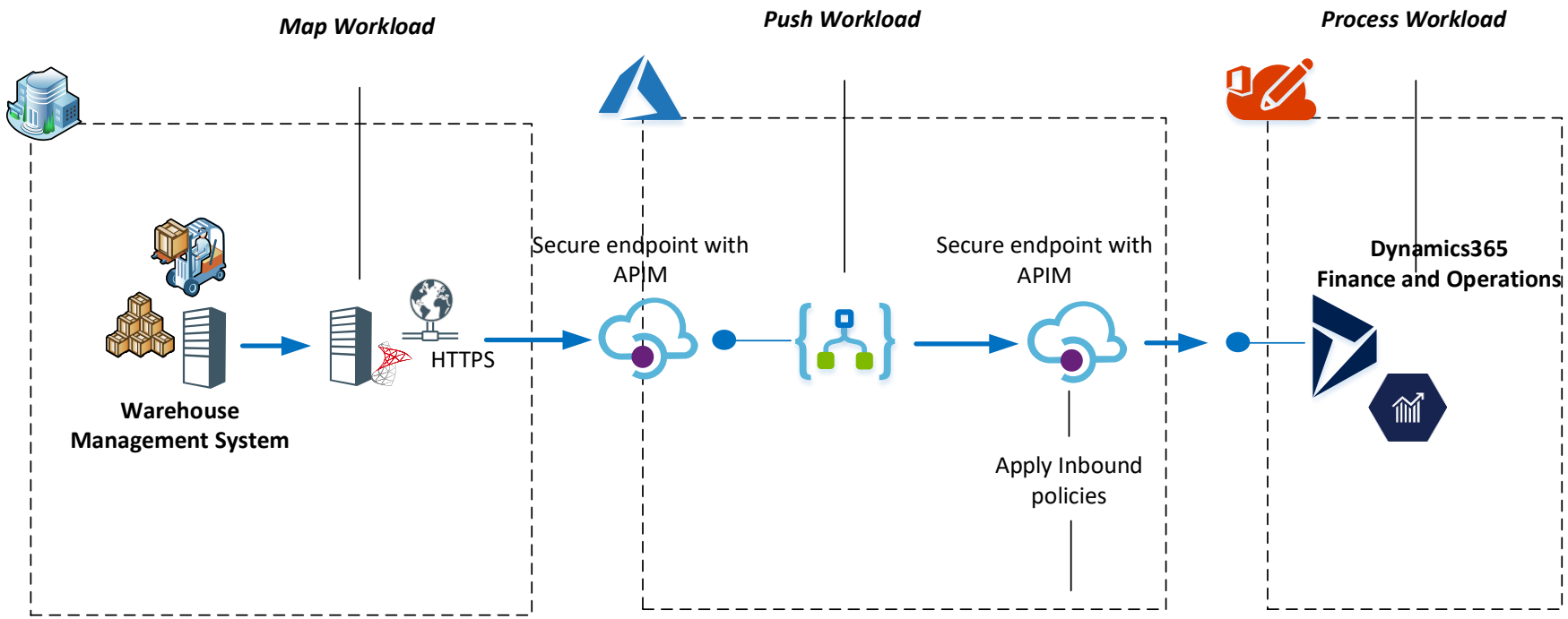


Responsibilities

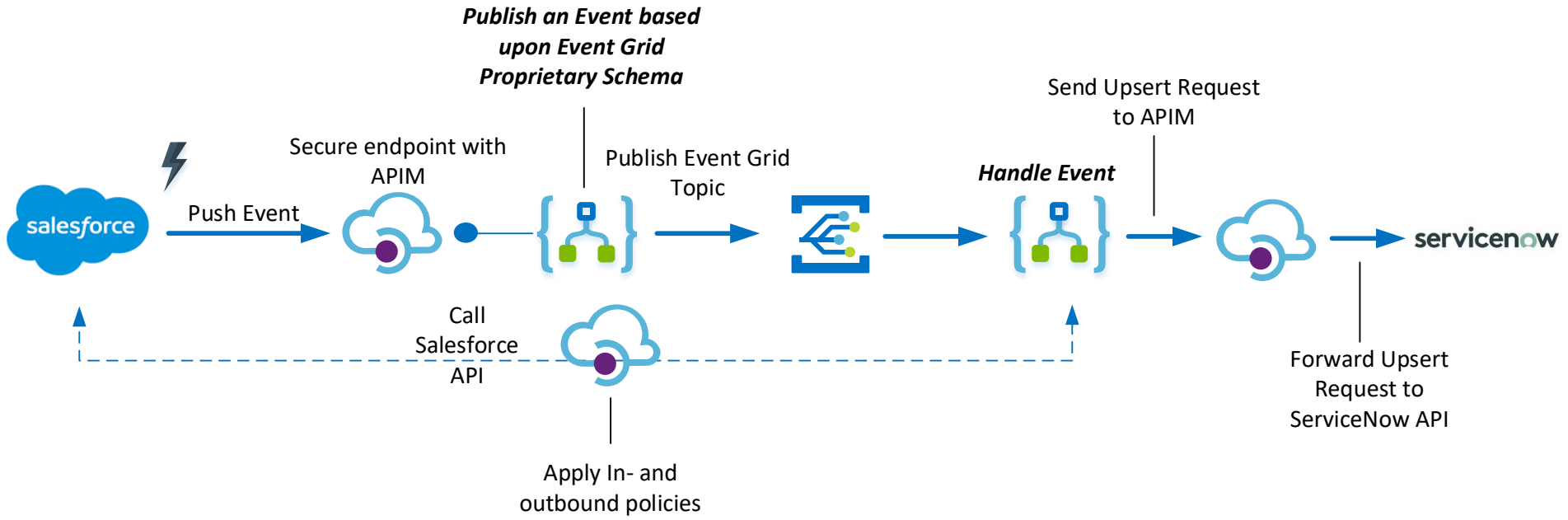
Azure Integration Solution Building Blocks



Real World Integration Cases

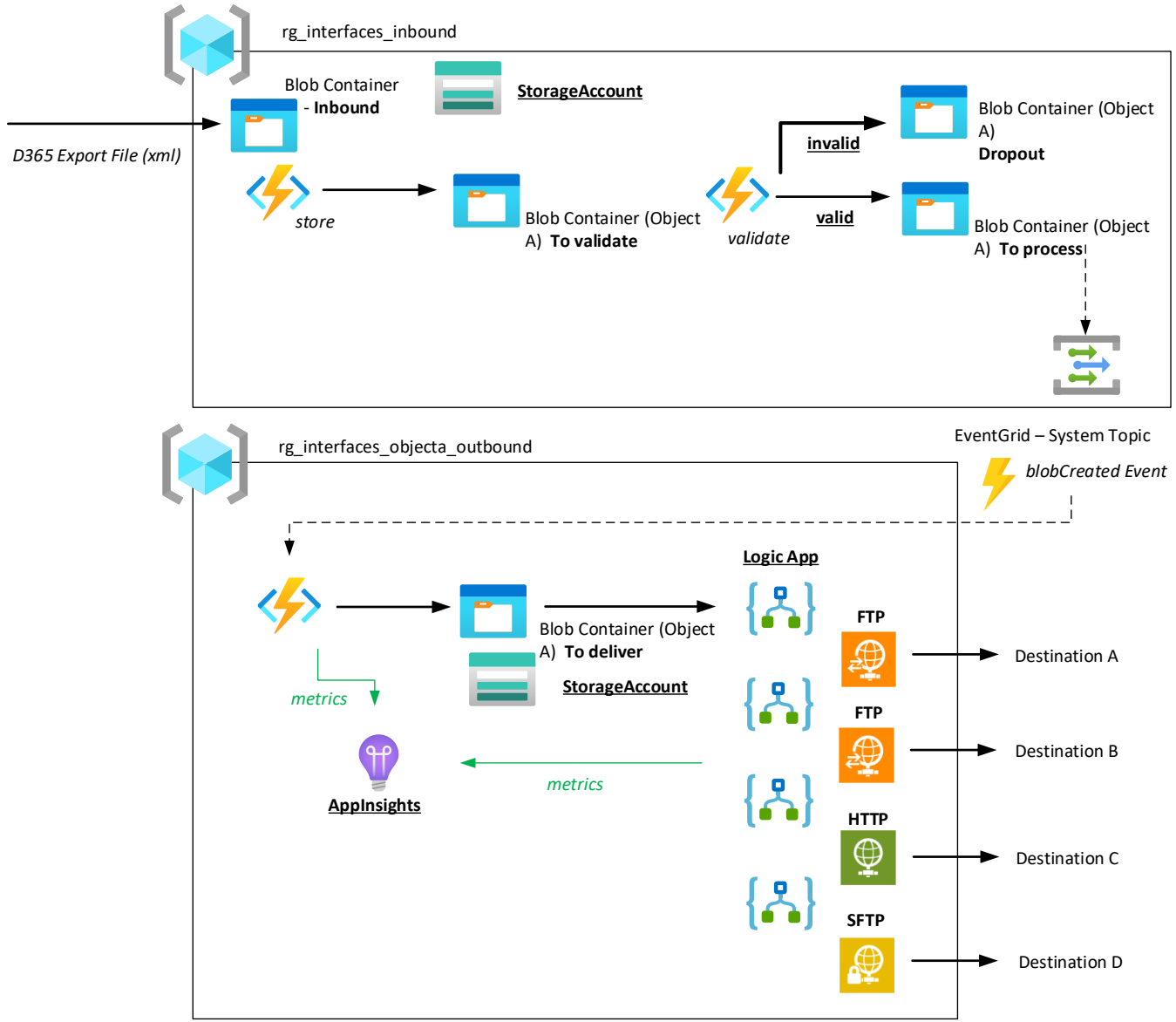


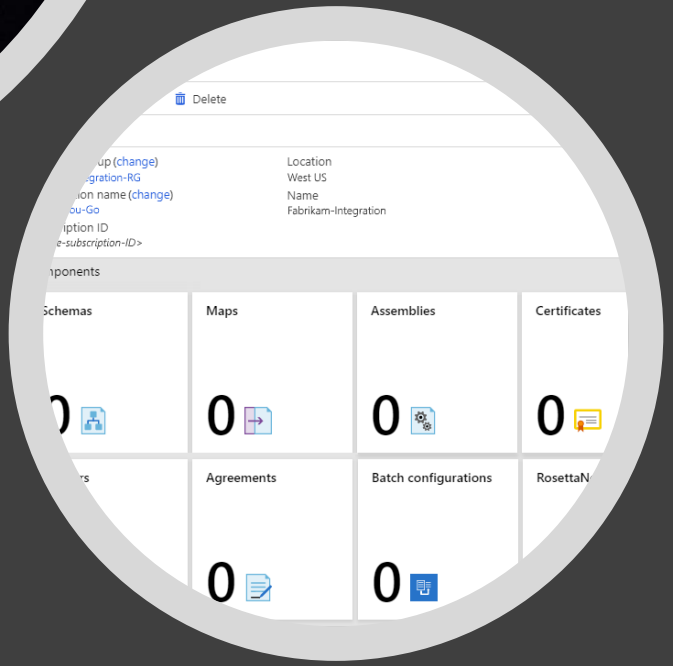

```
<policies>
  <inbound>
      <base />
  </inbound>
  <outbound>
      <base />
      <xsl-transform>
        <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
            <xsl:output omit-xml-declaration="yes" method="xml" indent="yes" />
            <!-- Copy all nodes directly-->
            <xsl:template match="node()| @*|*">
                <xsl:copy>
                    <xsl:apply-templates select="@* | node()|*" />
                </xsl:copy>
            </xsl:template>
        </xsl:stylesheet>
    </xsl-transform>
  </outbound>
</policies>
```

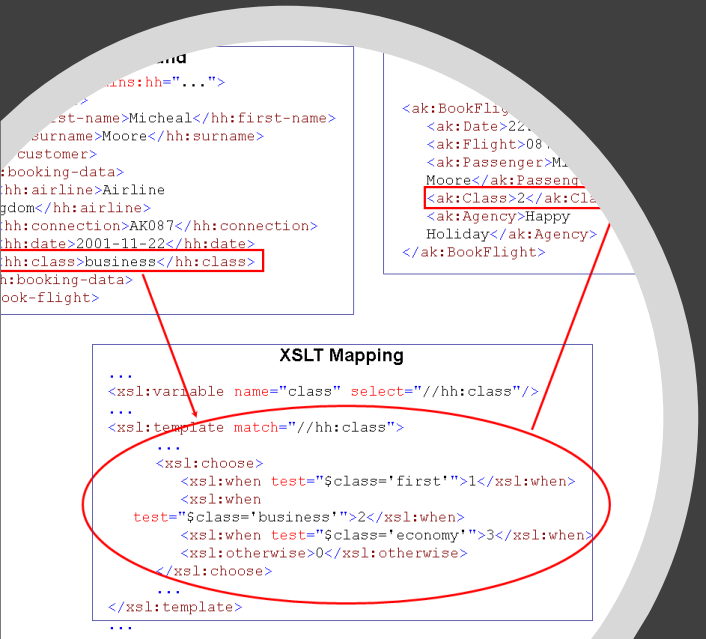



```
    <outbound>
        <base />
        <set-body template="liquid">{
    "Project": {
        "Description": "{{body.Description__c}}",
        "Account_ID": "{{body.Account__c}}",
        "Project_Placeholder_ID": "{{body.Id}}"
    }
}</set-body>
    </outbound>
```

```
<inbound>
    <base />
    <!-- Same for all operations. Should be part of "All operations" -->
    <send-request ignore-error="true" timeout="20" response-variable-name="bearerToken"
        <set-url>{{salesForceAuthorizationServer}}</set-url>
        <set-method>POST</set-method>
        <set-header name="Content-Type" exists-action="override">
            <value>application/x-www-form-urlencoded</value>
        </set-header>
        <set-body>@{
          return "grant_type=password&client_id={{salesForceClientId}}&client_secret={{
      }</set-body>
    </send-request>
    <set-header name="Authorization" exists-action="override">
        <value>@("Bearer " + (String)((IResponse)context.Variables["bearerToken"]).Body
    </set-header>
    <!--  Don't expose APIM subscription key to the backend. -->
    <set-header name="Ocp-Apim-Subscription-Key" exists-action="delete" />
</inbound>
```


rg_interfaces_inbound
Blob Container
- Inbound
StorageAccount
D365 Export File (xml)
store
Blob Container (Object A) To validate
validate
invalid
valid
Blob Container (Object A) Dropout
Blob Container (Object A) To process
rg_interfaces_objecta_outbound
EventGrid – System Topic
blobCreated Event
Logic App
Blob Container (Object A) To deliver
StorageAccount
metrics
AppInsights
metrics
FTP
Destination A
FTP
Destination B
HTTP
Destination C
SFTP
Destination D





The Magic - Transformation

Tech:

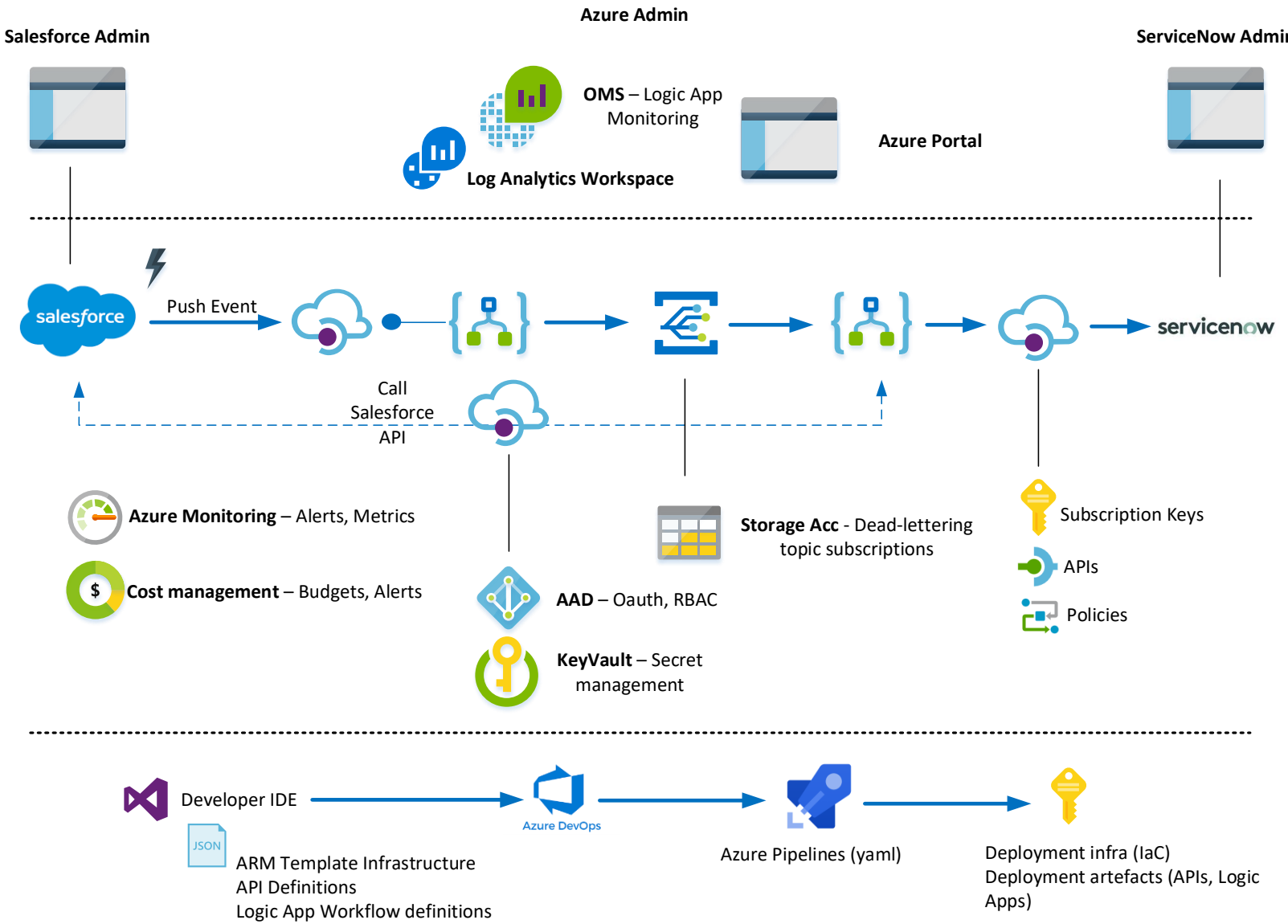
- Integration Account
- API (Function)
- API Management Policy

Format:

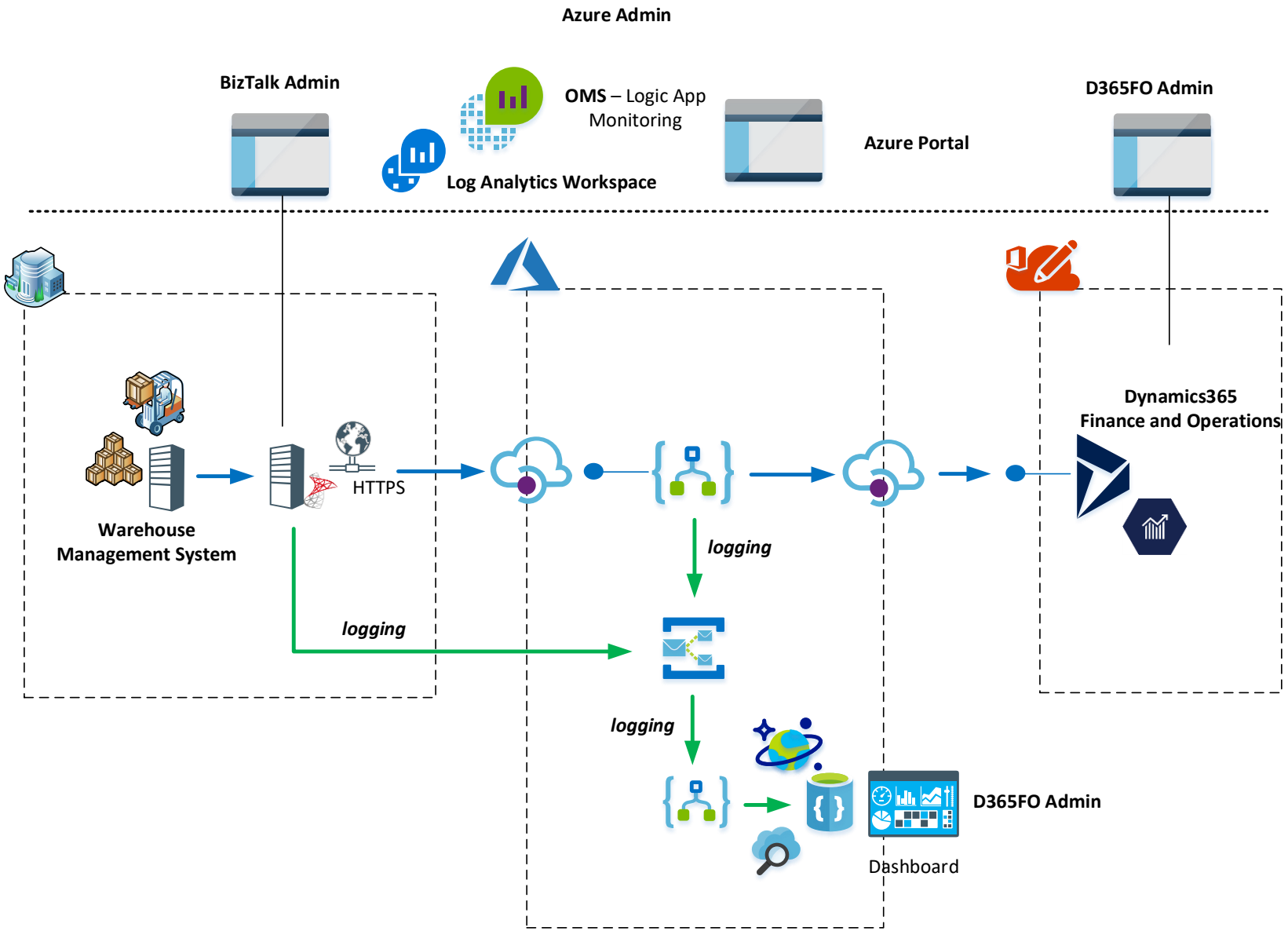
- XML (XLST)
- JSON (Liquid)
- Custom (.NET)

Best Practices

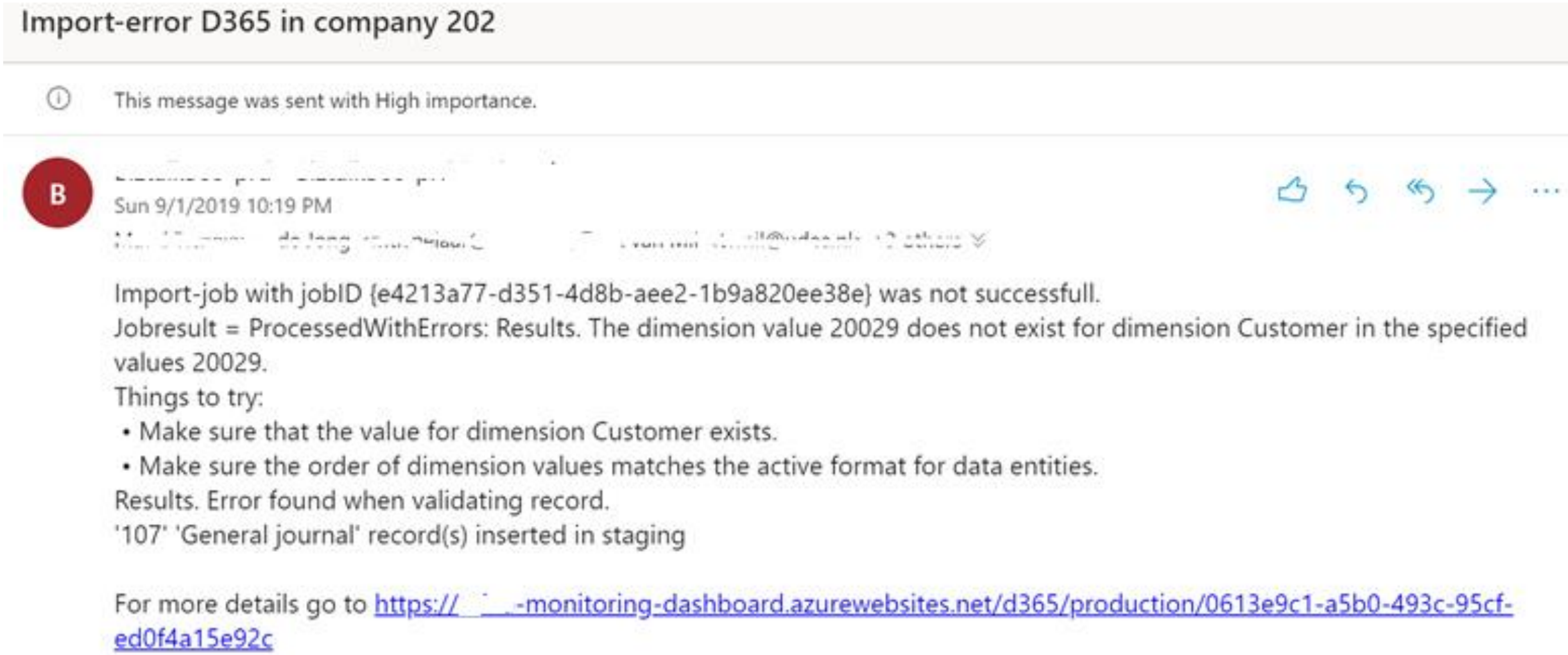
- Loose coupling
- Monitoring
- Event driven
- Cost management
- Versioning
- Governance
- DevOps (automation)


Salesforce Admin
Azure Admin
ServiceNow Admin
OMS – Logic App Monitoring
Log Analytics Workspace
Azure Portal
salesforce
Push Event
servicenow
Call Salesforce API
Azure Monitoring – Alerts, Metrics
Cost management – Budgets, Alerts
AAD – Oauth, RBAC
KeyVault – Secret management
Storage Acc - Dead-lettering topic subscriptions
Subscription Keys
APIs
Policies
Developer IDE
Azure DevOps
JSON
ARM Template Infrastructure
API Definitions
Logic App Workflow definitions
Azure Pipelines (yaml)
Deployment infra (IaC)
Deployment artefacts (APIs, Logic Apps)

Monitoring


Azure Admin
BizTalk Admin
OMS – Logic App Monitoring
Log Analytics Workspace
Azure Portal
D365FO Admin
Dynamics365 Finance and Operations
Warehouse Management System
HTTPS
logging
logging
logging
D365FO Admin
Dashboard

Functional Monitoring



Import-error D365 in company 202

This message was sent with High importance.

B

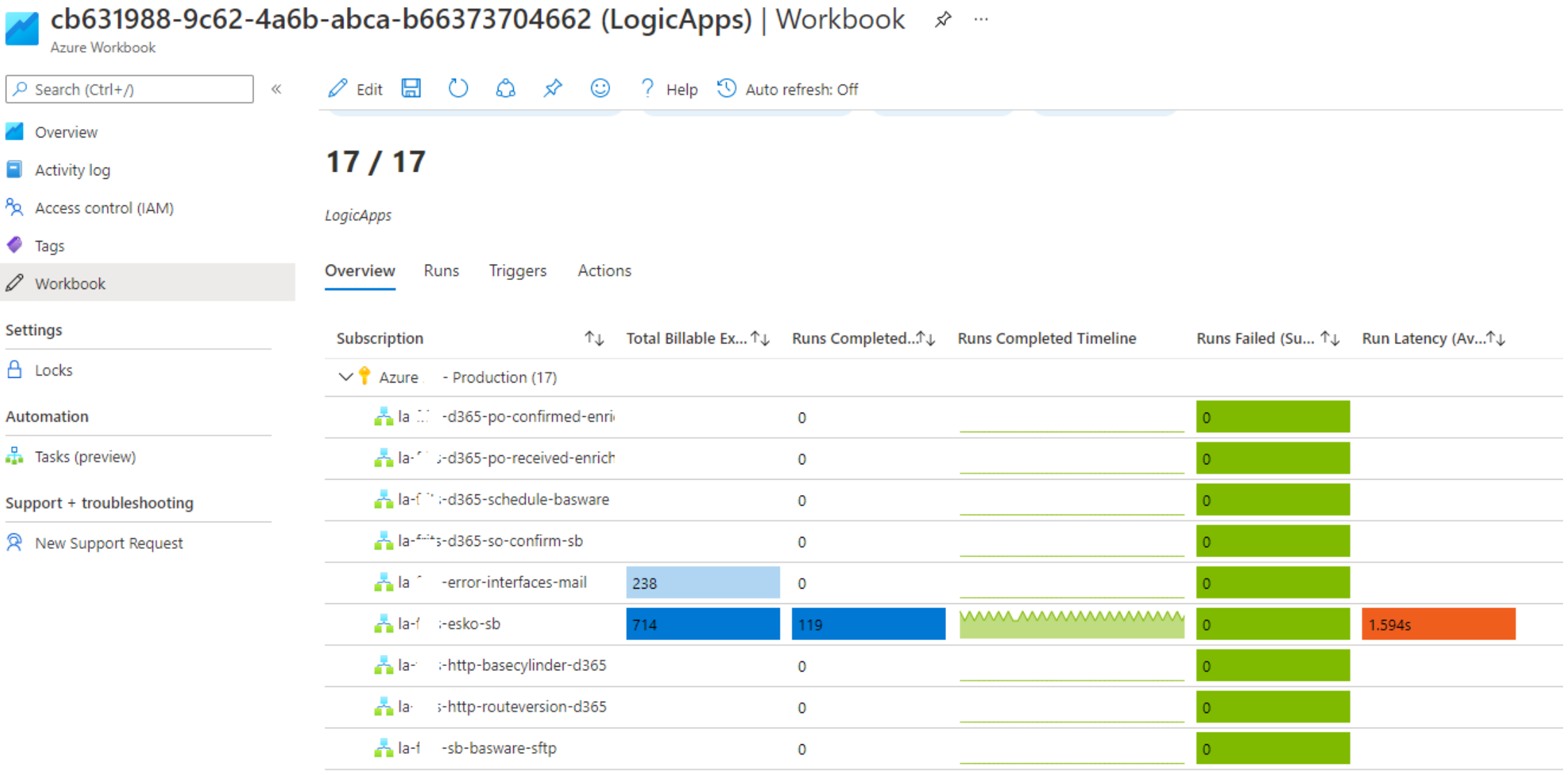
Sun 9/1/2019 10:19 PM

Import-job with jobID {e4213a77-d351-4d8b-aee2-1b9a820ee38e} was not successfull.
Jobresult = ProcessedWithErrors: Results. The dimension value 20029 does not exist for dimension Customer in the specified values 20029.
Things to try:
- Make sure that the value for dimension Customer exists.
- Make sure the order of dimension values matches the active format for data entities.

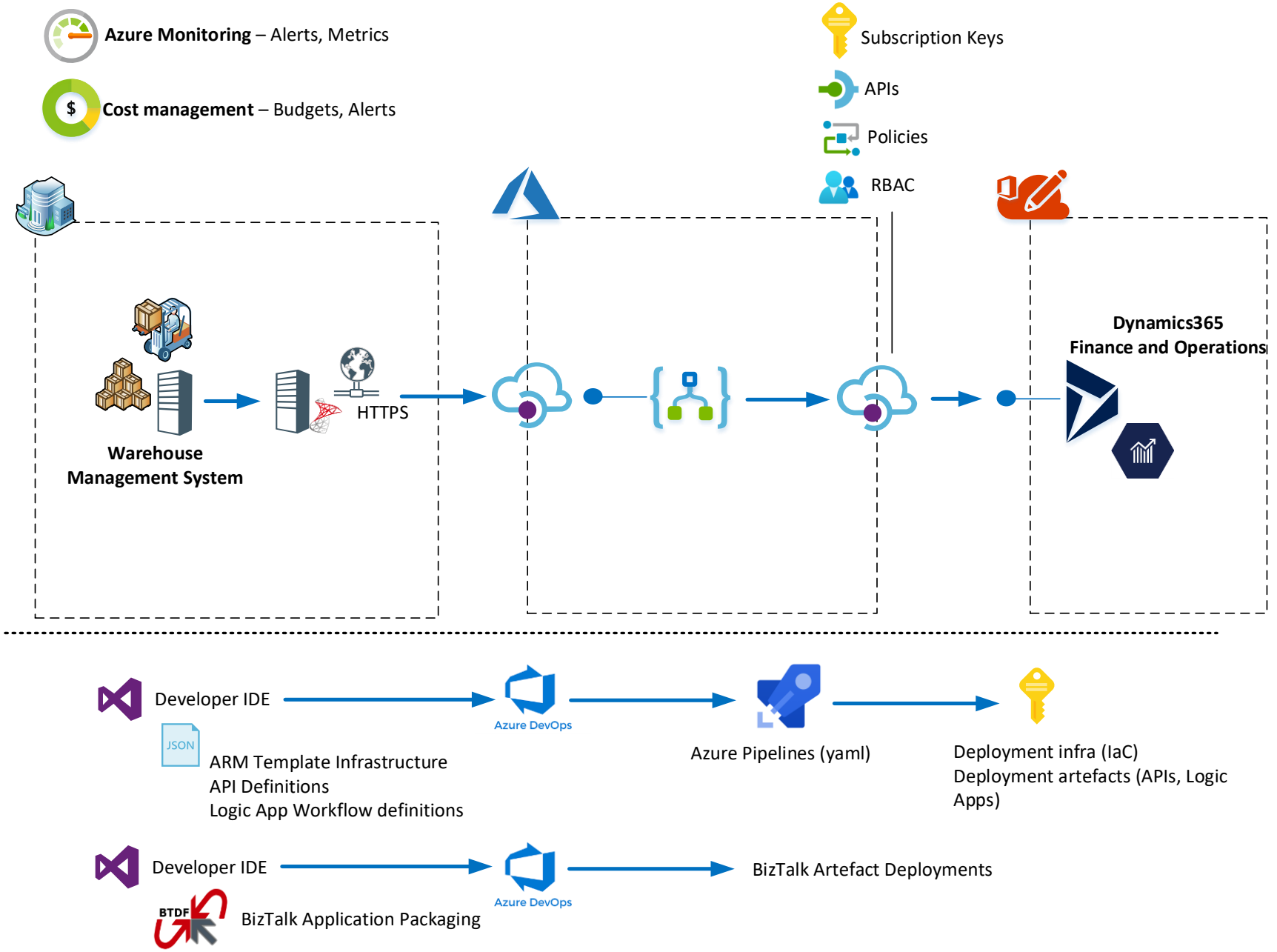
Results. Error found when validating record.
'107' 'General journal' record(s) inserted in staging

For more details go to https:// -monitoring-dashboard.azurewebsites.net/d365/production/0613e9c1-a5b0-493c-95cf-ed0f4a15e92c

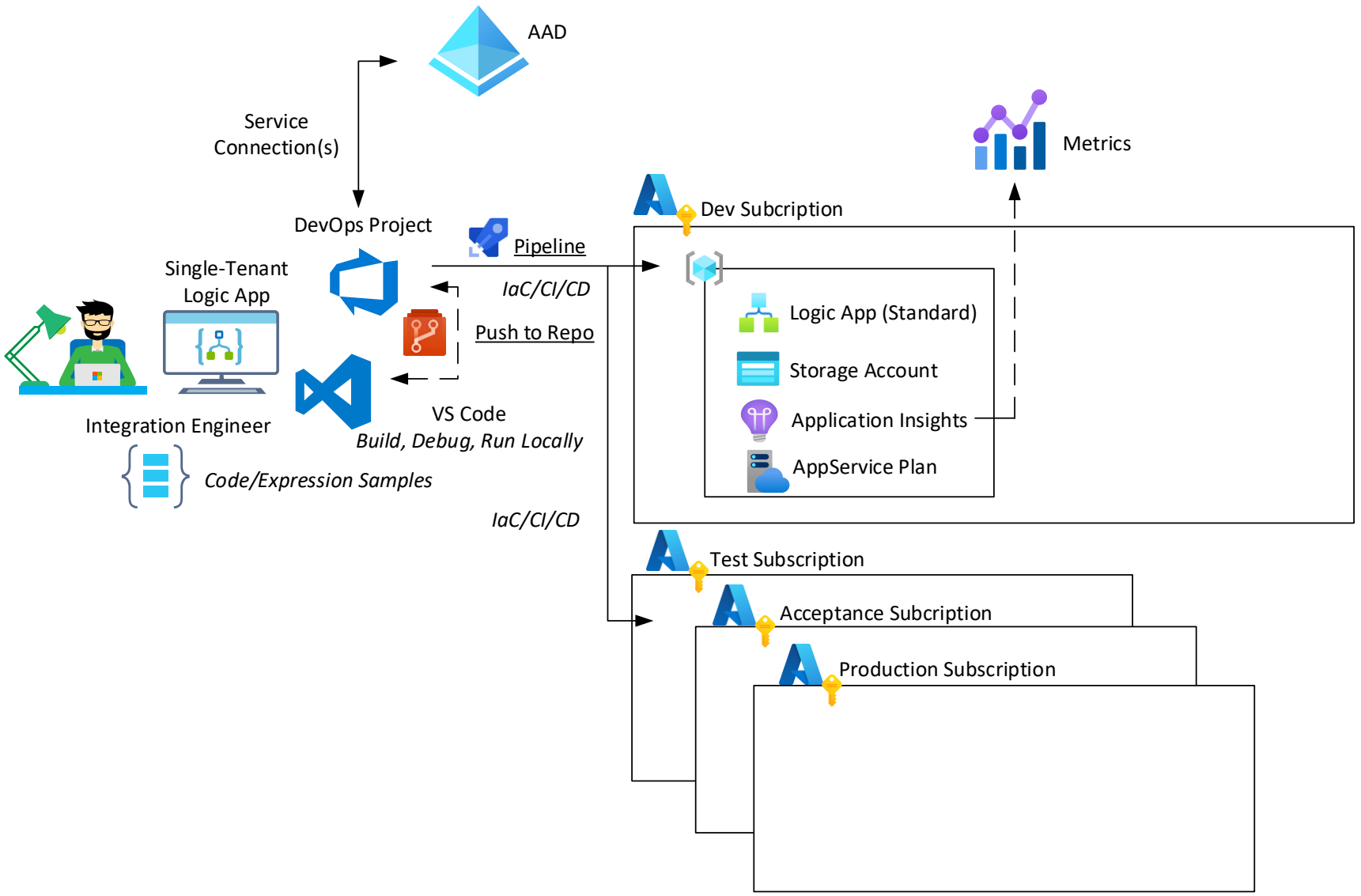
Technical Monitoring



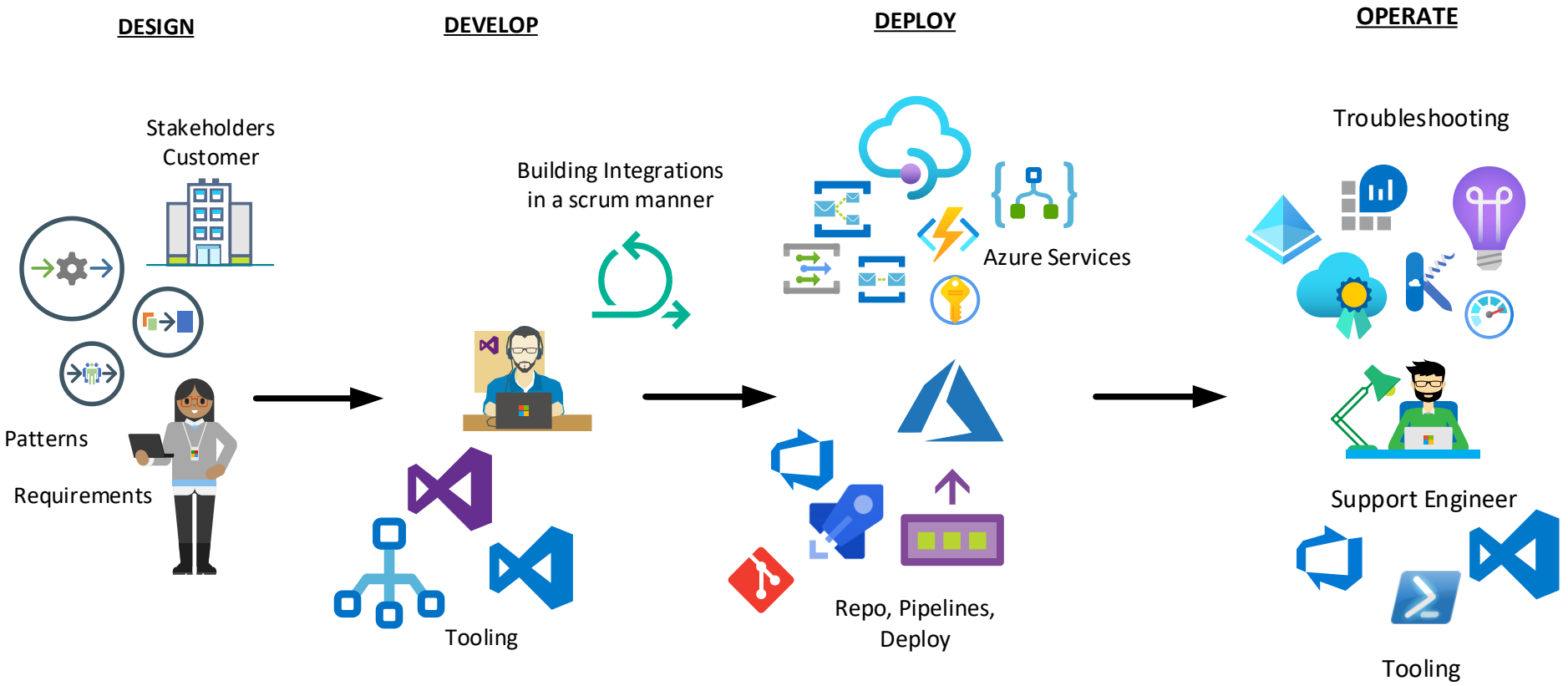
Other aspects



Developer Setup



All things considered ….



What do the analysts say?

Gartner®

August 2020

A Leader in Enterprise Integration Application Platform as a Service*

Magic Quadrant

Figure 1. Magic Quadrant for Enterprise Integration Platform as a Service

Source: Gartner (August 2020)

***Gartner "Magic Quadrant for Enterprise Integration Platform as a Service," by Eric Thoo, Bindi Bhullar, Massimo Pezzini, Keith Guttridge Abhishek Singh Shaheem Pillai 21 September 2020**

Gartner®

September 2020

A Leader in Full Life Cycle API Management*

Figure 1. Magic Quadrant for Full Life Cycle API Management

Source: Gartner (September 2020)

***Gartner "Magic Quadrant for Full Life Cycle API Management," by Paolo Malinverno, Kimihiko Iijima, Mark O'Neill, John Santoro, Shameen Pillai, Akash Jain 22 September 2020**

Key Takeaways

- Choose technology based upon your requirements
- Logic Apps and Azure (Durable) Functions are NOT competing technologies. Same applies for Event Grid and Service Bus!
- Understanding the tech, processes and non-functionals is key!
- Apply the best practices and leverage Azure Well-Architected Framework

Sources to explore!

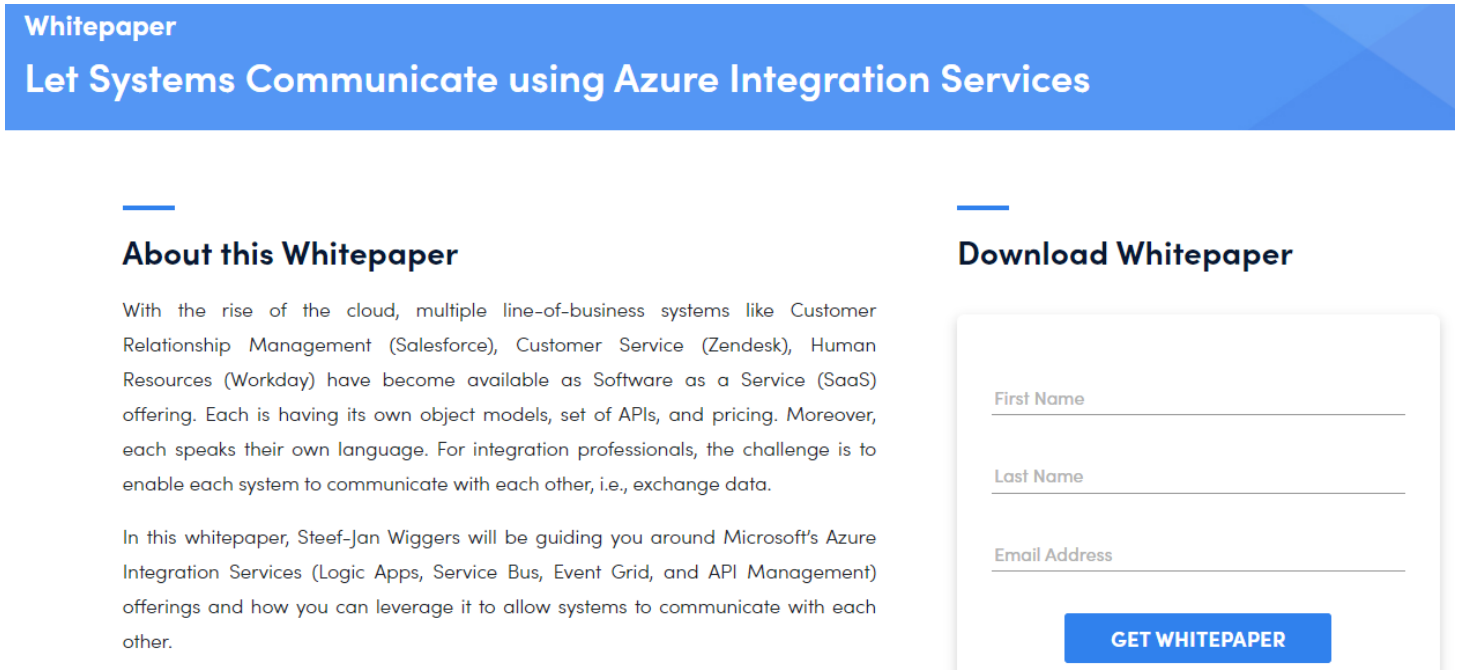
- Azure Integration Services
- Serverless Notes
- Azure Well-Architected Framework
- Codit Blog
- Power Automate
- Enterprise Integration Patterns
- Integration Usergroup
- Microsoft Learn

And more …

Discover about the missing piece in the Azure Integration puzzle

https://www.serverless360.com/ebook/managing-events-in-the-cloud-with-azure-event-grid

And even more …



https://www.serverless360.com/whitepaper/let-systems-communicate-using-azure-integration-services

CODE EUROPE

Thanks for attending!

https://www.linkedin.com/in/steefjan/

https://twitter.com/SteefJan

sj.wiggers@gmail.com

Microsoft® MVP Most Valuable Professional

InfoQ Editor