Azure Service Fabric

Build always-on, hyper-scalable, microservice-based cloud applications

Ross Smith
Technical Evangelist
@ross_p_smith

Microsoft

Why a microservices approach?

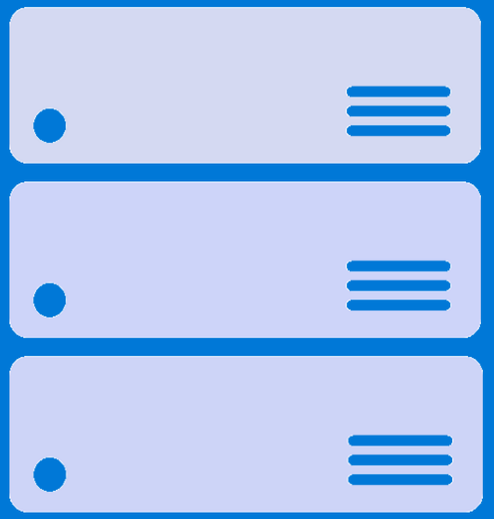
- Build and operate a service at scale
- Continually evolving applications
- Faster delivery of customer features
- Improved resource utilization to reduce costs

Monolithic application approach

- A monolith app contains domain specific functionality and is normally divided by functional layers such as web, business and data

App 1

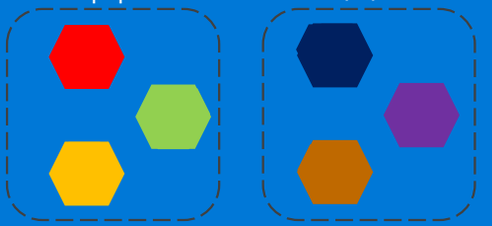
- Scales by cloning the app on multiple servers/VMs/Containers



Microservices application approach

- A microservice application separates functionality into separate smaller services.

App 1 App 2



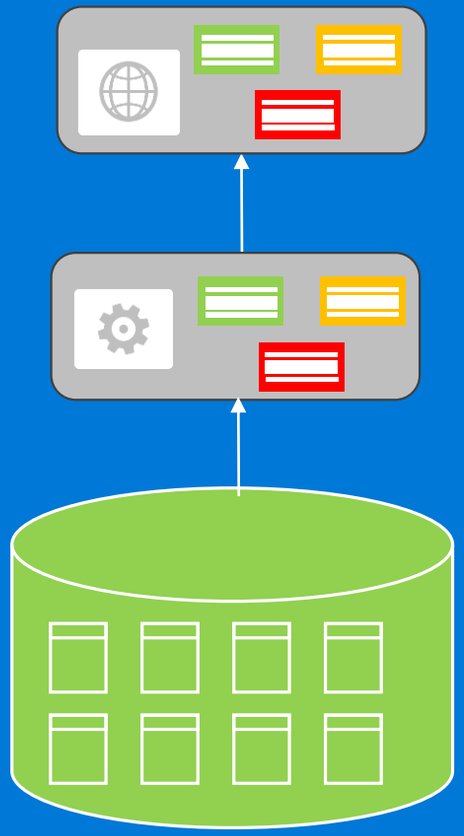
- Scales out by deploying each service independently creating instances of these services across servers/VMs/containers

What is a microservice?

- Encapsulates a scenario
- Are developed by a small engineering team
- Can be written in any language and framework
- Contain code plus state that is independently versioned, deployed, and scaled
- Interact with other microservices over well defined interfaces and protocols such as http
- Have a unique name (URL) that can be resolved
- Remains consistent and available in the presence of failures

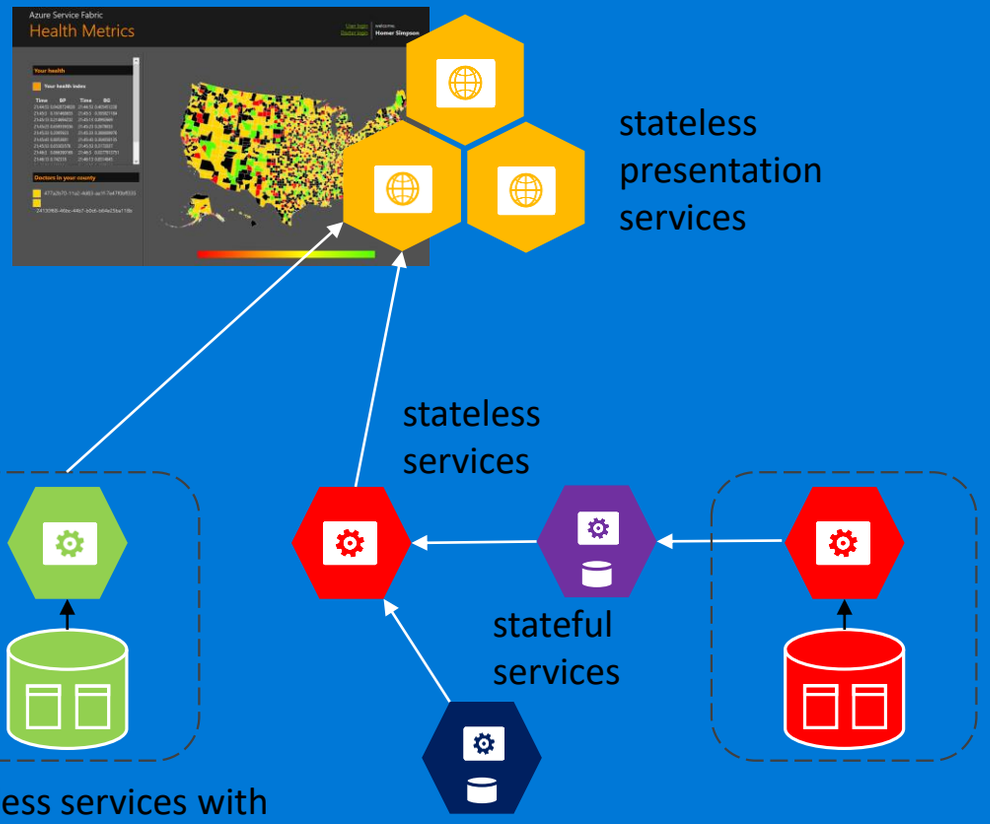
State in Monolithic approach

- Single monolithic database
- Tiers of specific technologies



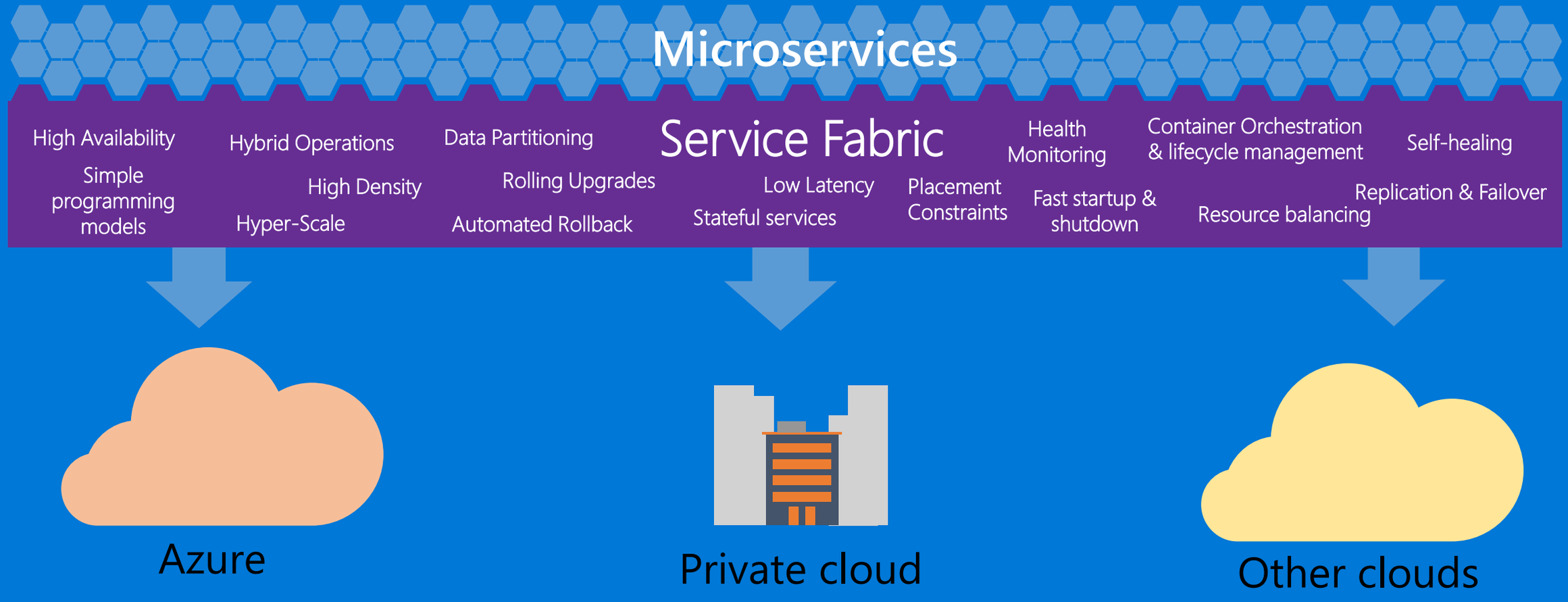
State in Microservices approach

- Graph of interconnected microservices
- State typically scoped to the microservice
- Variety of technologies used



Microsoft Azure Service Fabric

A platform for reliable, hyperscale, microservice-based applications



Services Powered by Service Fabric

SQL Database

2.1 million DBs

Cosmos DB

Billions transactions/day

IoT Hub

Millions of messages

Event Hubs

60bn events/day

30% of Azure cores run Service Fabric

Skype

Cortana

Intune

Dynamics

Power BI

Designed for mission critical tier 1 workloads

To monolith or to Microservice?

5 stages in a continuum…

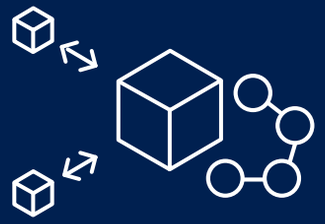
1

Traditional app

2

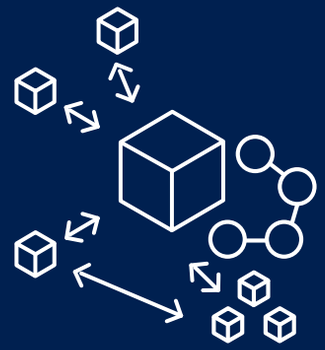
Monolith Hosted as guest executable or container

3



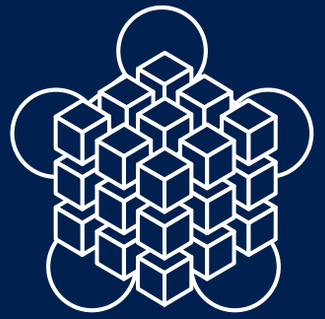
Existing Monolith + new microservices

4



Parts of existing monolith extracted

5



New or transformed microservices app

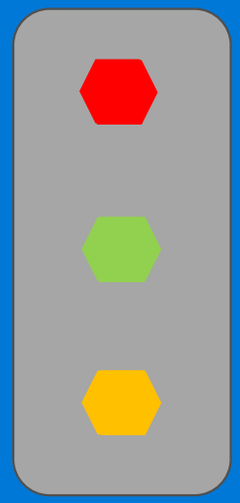
… we support any stage you choose

Types of microservices
from a Service Fabric perspective

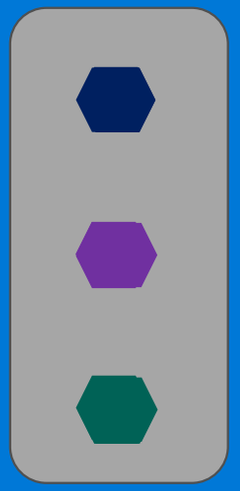
- Stateless microservice
  - Has either no state or it can be retrieved from an external store
  - There can be N instances
  - e.g. web frontends, protocol gateways, Azure Cloud Services etc.

- Stateful microservice
  - Maintain hard, authoritative state
  - N consistent copies achieved through replication and local persistence
  - e.g. database, documents, workflow, user profile, shopping cart etc.

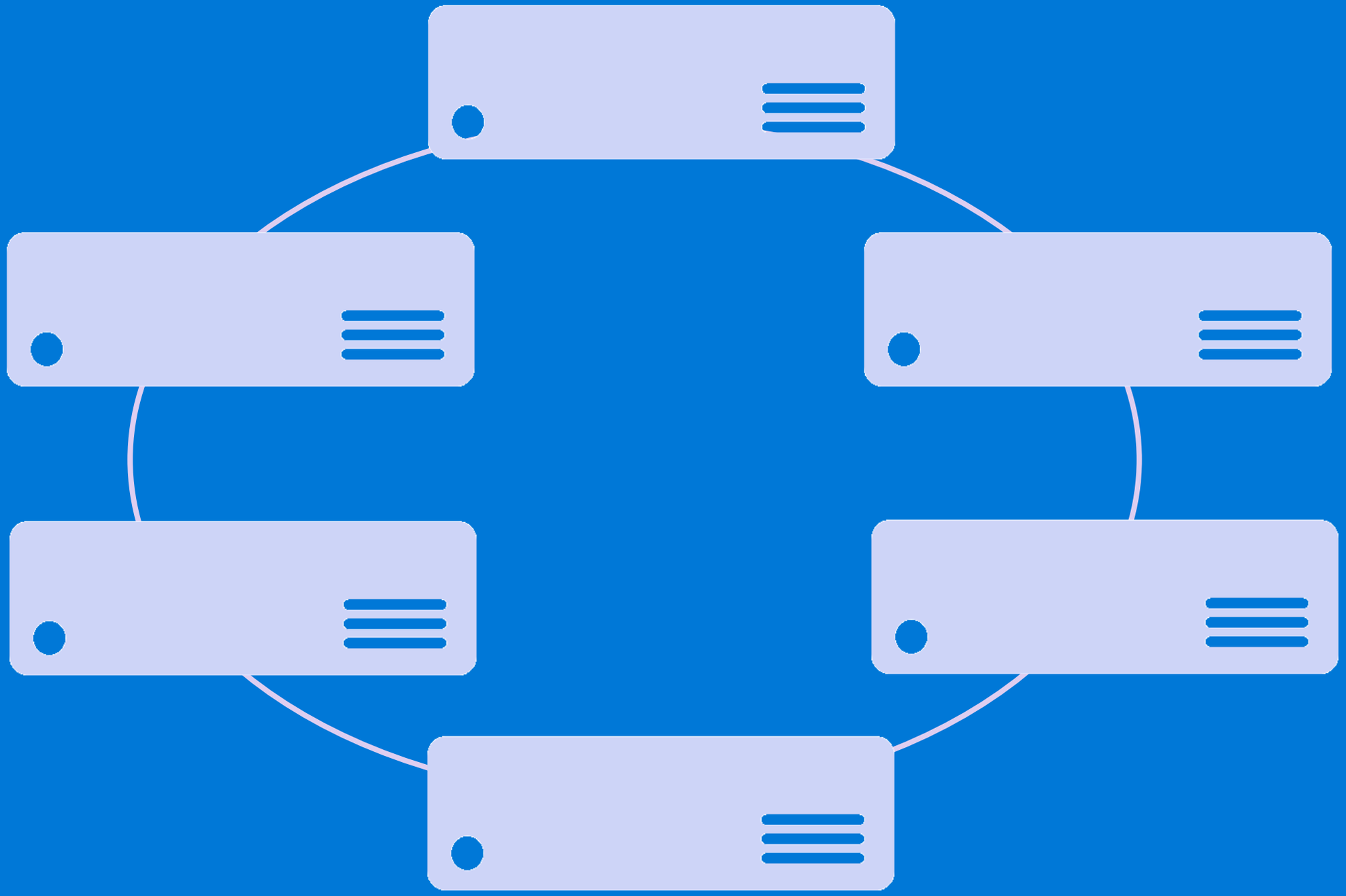
Service Fabric cluster with microservices



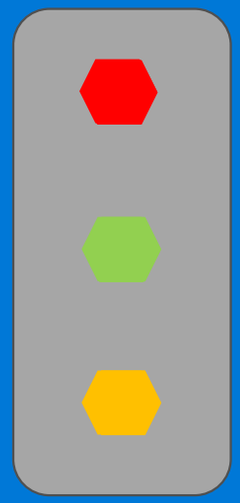
App1



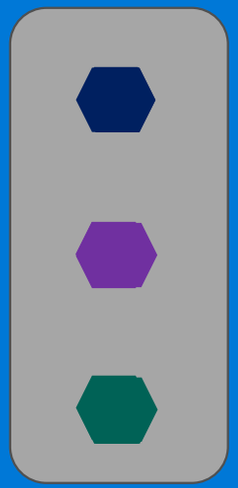
App2



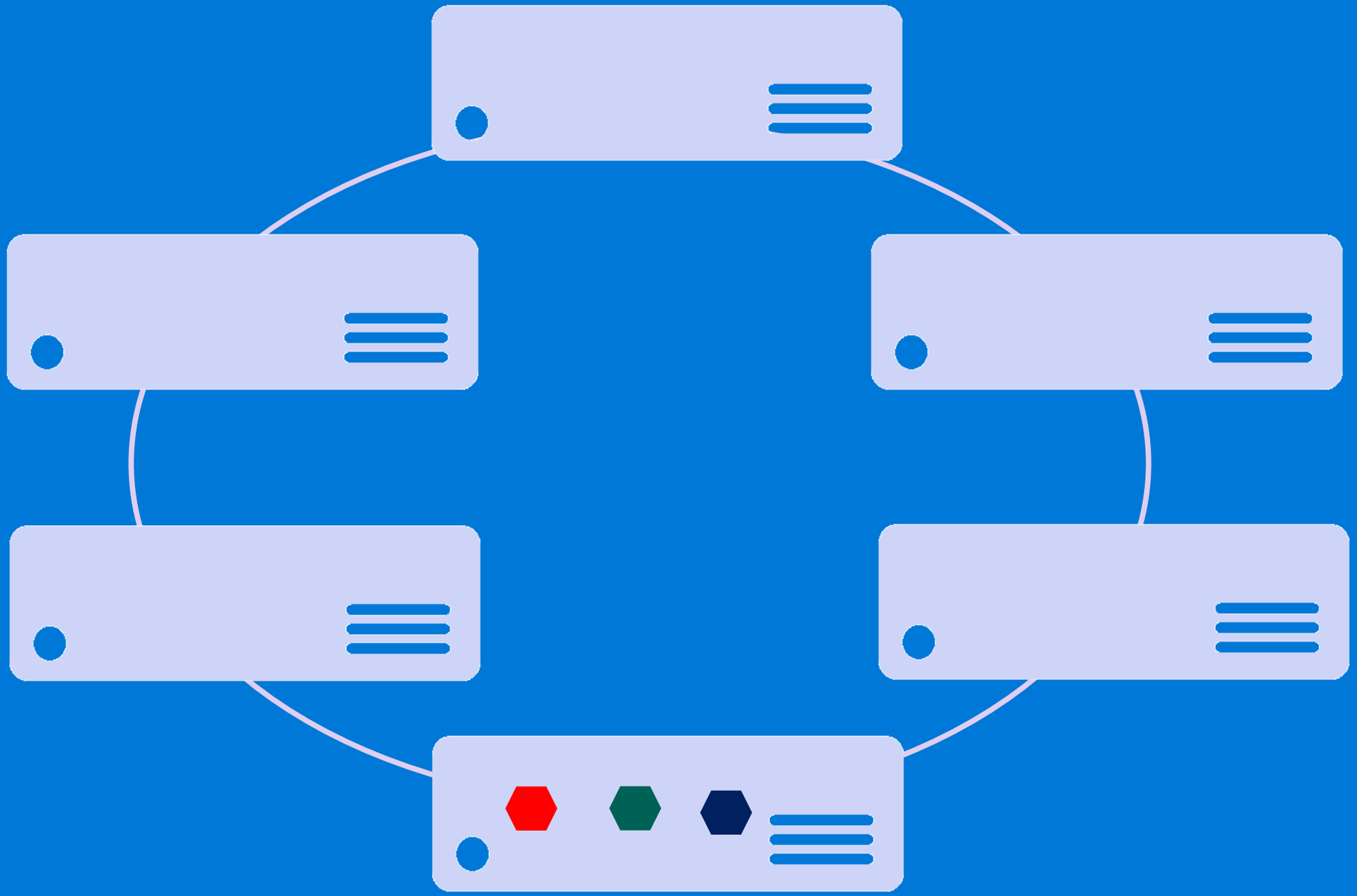
Handling machine failures



App1

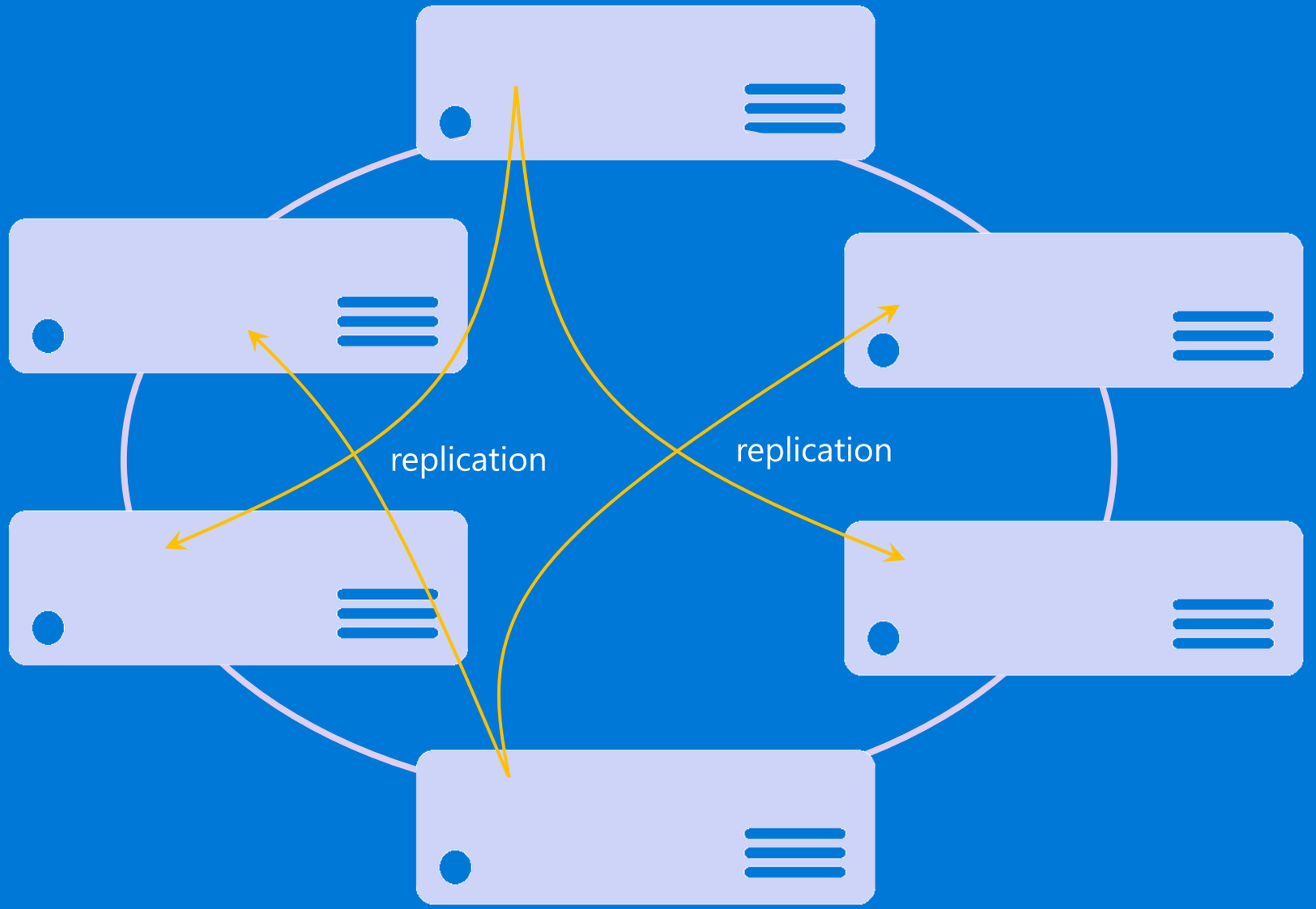


App2



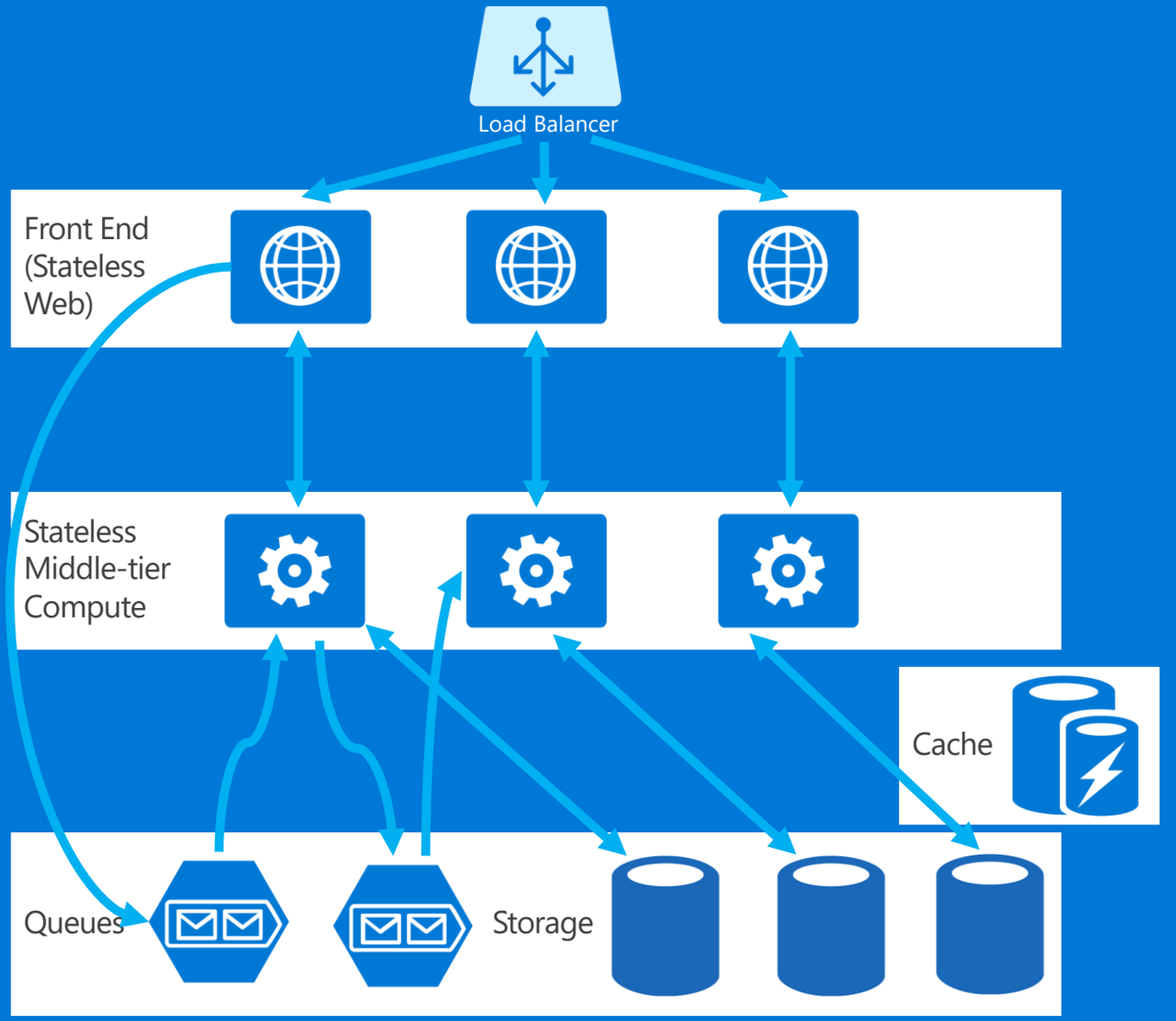
Stateful microservice

Application
Package



Stateless Services Pattern

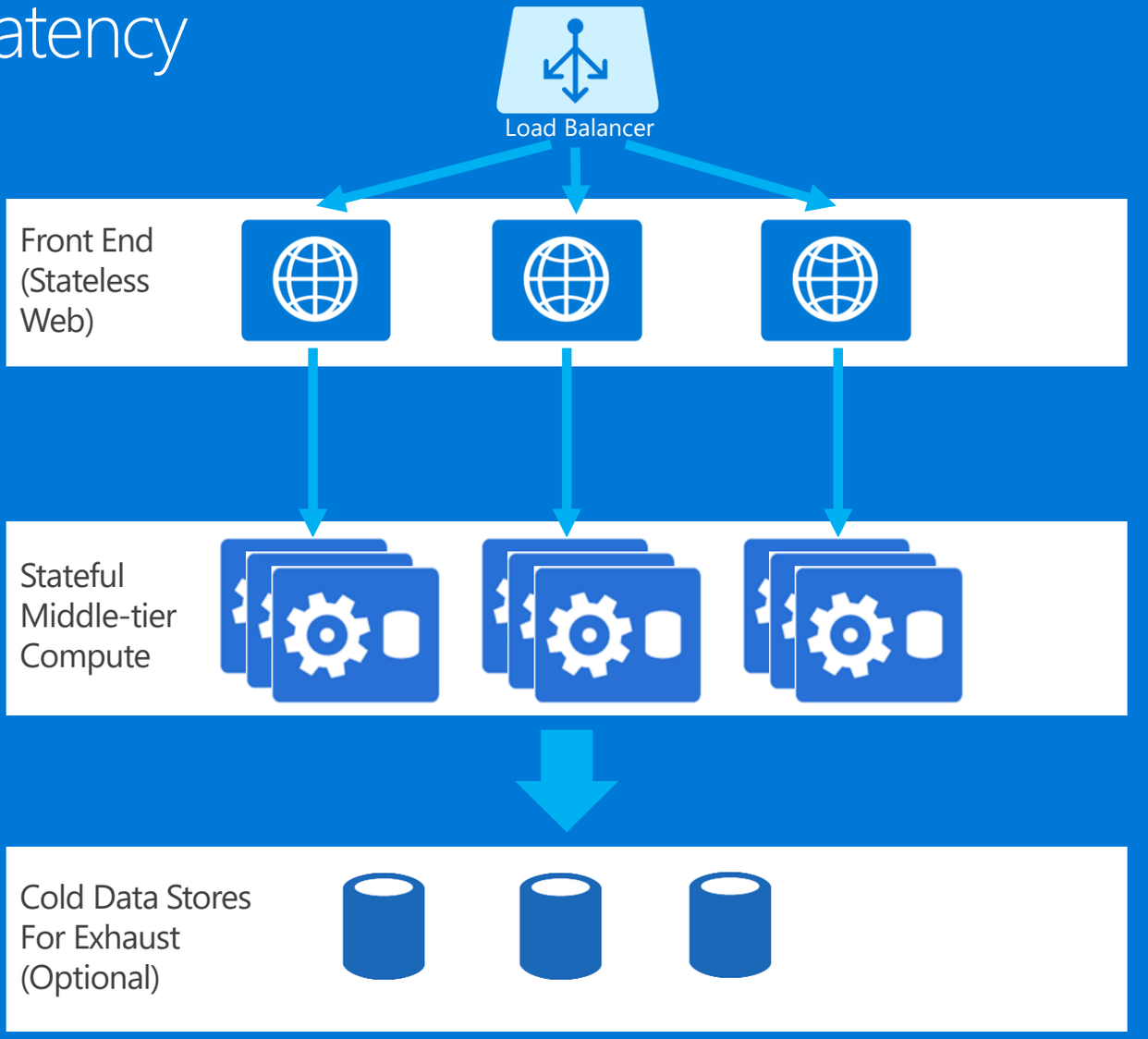
- Scale stateless services backed by partitioned storage
- Increase reliability and ordering with queues
- Reduce read latency with caches
- Manage your own transactions for state consistency
- More moving parts each managed differently



Stateful Services Pattern

Simplify design, reduce latency

- Application state lives in the compute tier
- Low Latency reads and writes
- Partitions are first class at the service layer for scale-out
- Built in transactions
- Fewer moving parts
- External stores for exhaust and offline analytics

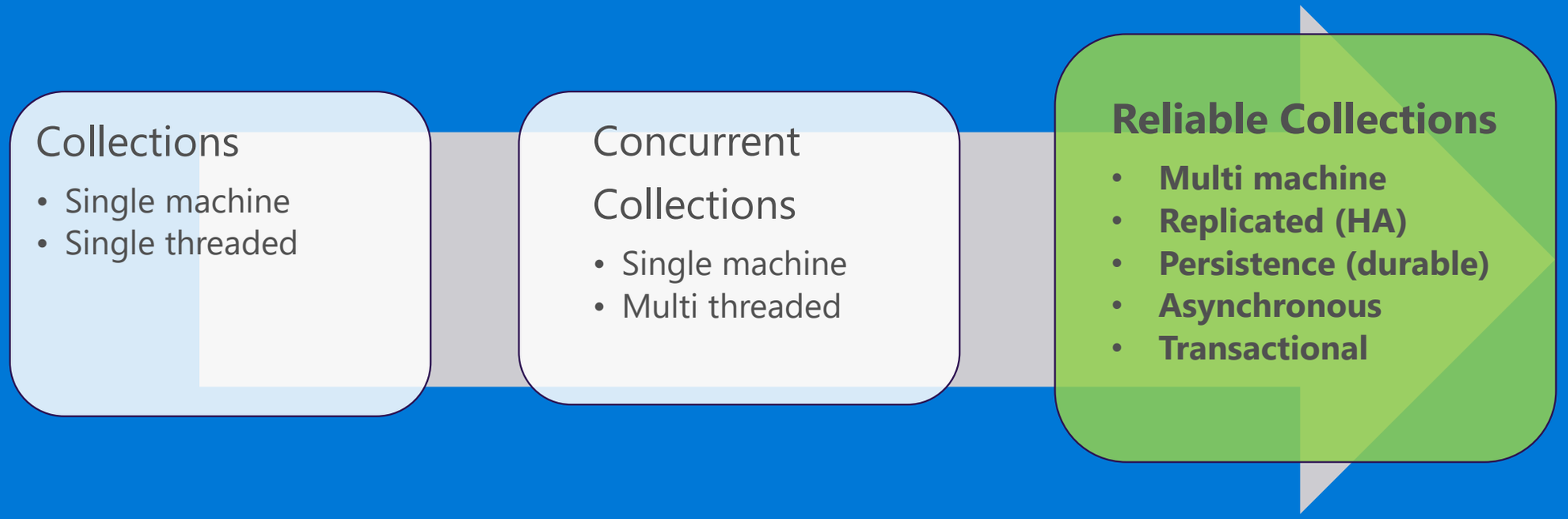


Reliable Services API

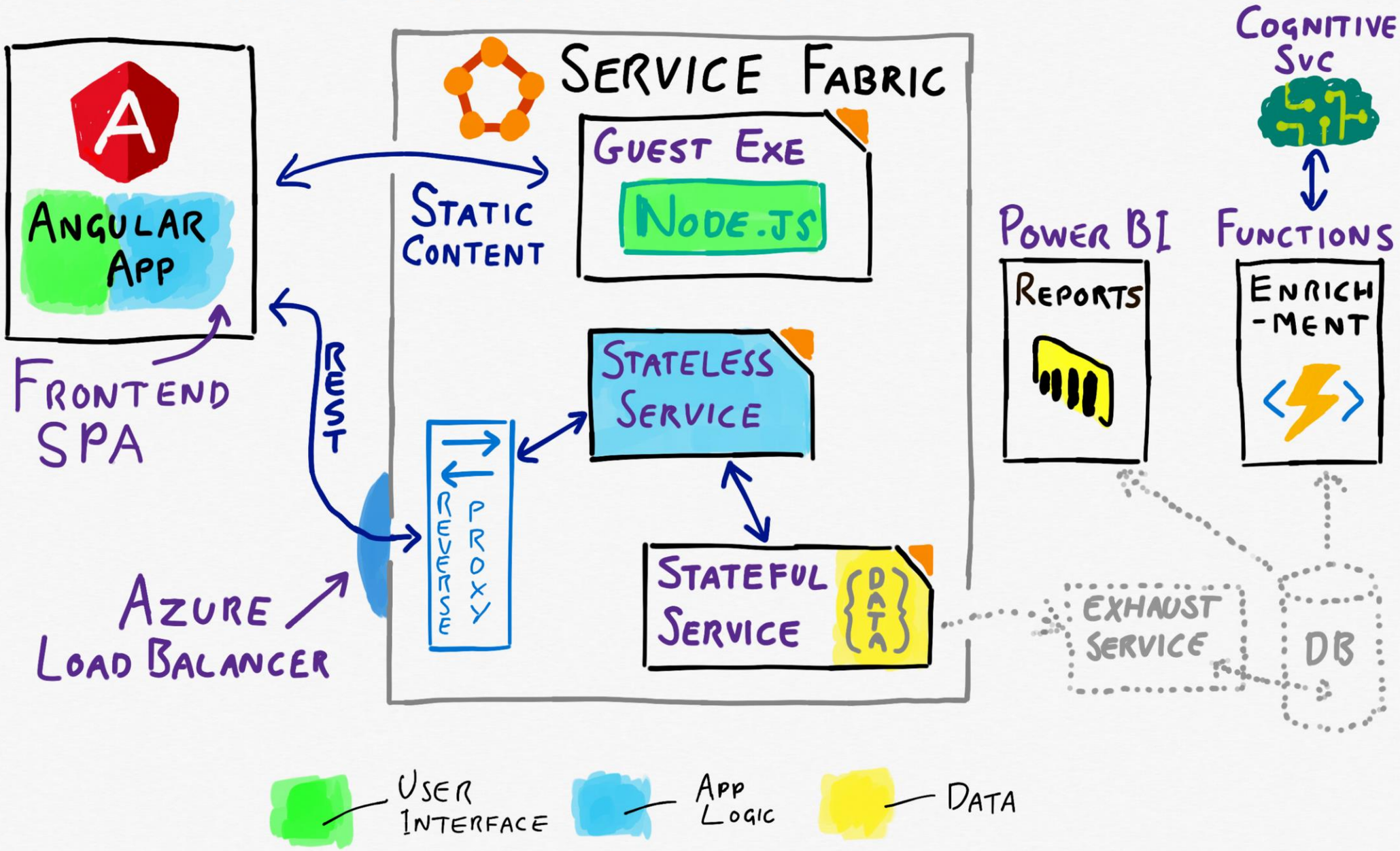
- Build stateless services using existing technologies such as ASP.NET, node.js, EXEs etc
- Manage concurrency and granularity of state changes with transactions in stateful services.
- Communicate with services using the technology of your choice (e.g Web API, WCF, [web]sockets, etc).

Reliable Collections

- Reliable collections make it easy to build stateful services.
- An evolution of .NET collections for the cloud.



Service Fabric Architecture

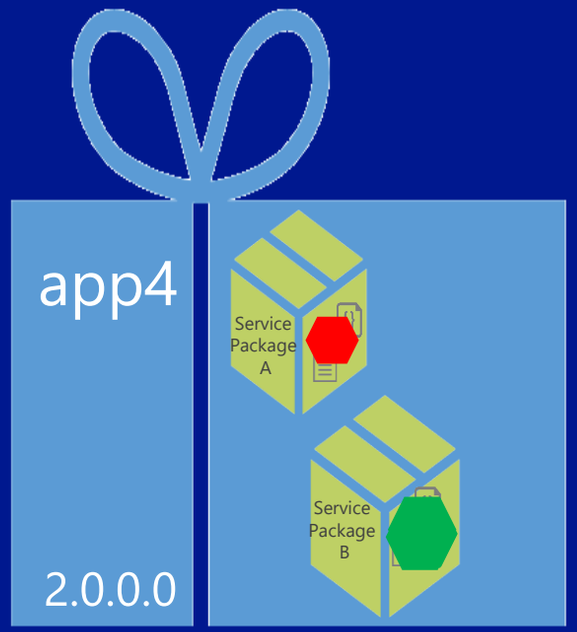


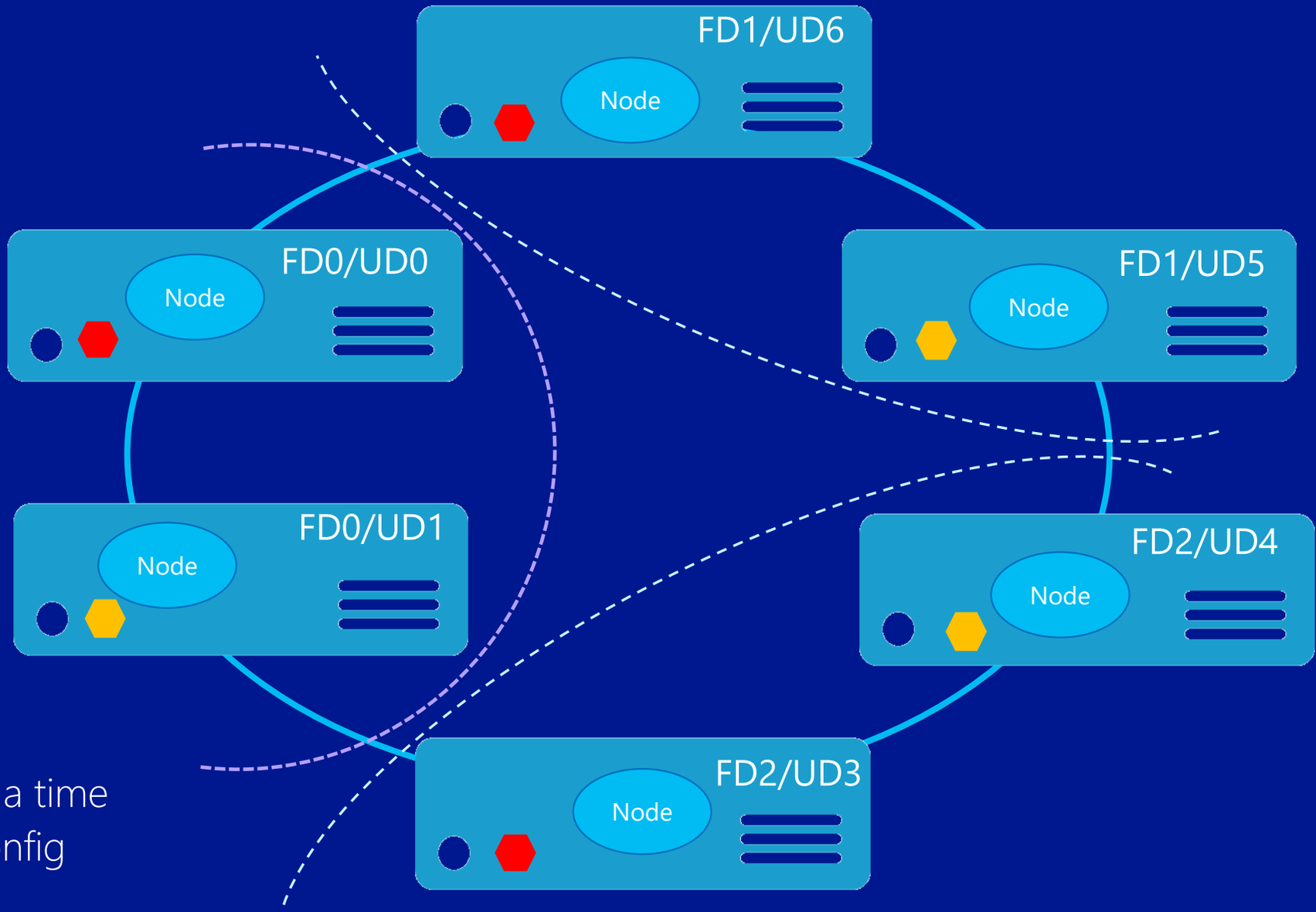
DEMO: Smilr in Azure Service Fabric

Service Fabric management capabilities

- Reliable optics into application health
- Automatic repair action based on policies you set
- Scales up/down based on service demand
- Integrated alerting and notification system
- Tools to effectively test a service for robustness
- Tools for easy deployments and config management
- Tools to perform service upgrades without downtime

Upgrading Services with zero downtime

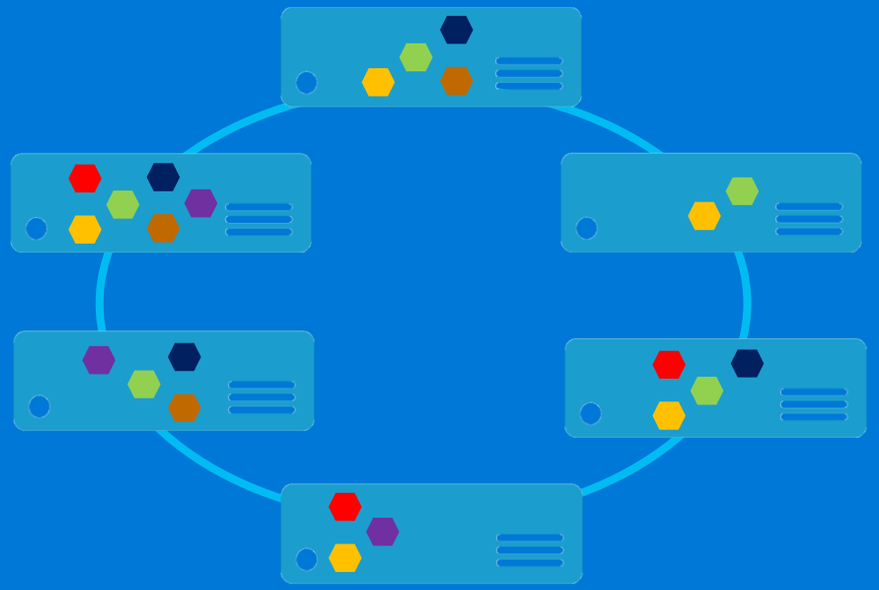




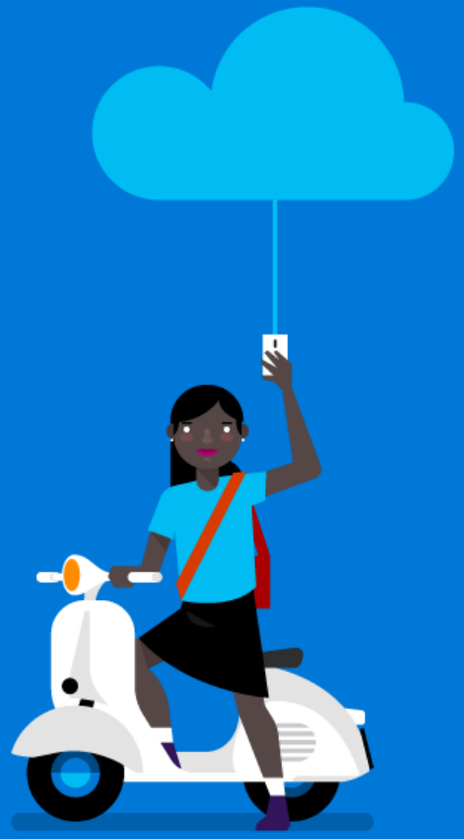
- Upgrade progresses one UD at a time
- Upgrade limited to the code/config package that changed

Other important topics in Service Fabric

- Cluster scale-up and down and different VM sizes
- Cluster placement constraints
- Application rolling upgrades and rollback
- Application health monitoring and reporting
- Application operational insights
- Advance application resource balancing and scheduling
- Chaos and scenario testing in production



Example Customer Solutions



Microsoft

TalkTalk, a UK video-on-demand service delivering TV and movie content across multiple-devices

Benefits

Microservices workflow for content encoding and resolution

Agility - Ability to upgrade microservices independently and without downtime. No need to coordinate DB schema with app upgrades

Programming API - Using actors and reliable collections to easily orchestrate the encoding and resolution of the on-demand content

Scalability - Real time résolution for 30K titles, designed to scale for growth of users, devices and content

- Replacing existing IaaS/DB backed system with microservices solution
- 1.5 PB of for streaming content delivered to millions of customers using Azure Media Services

TalkTalk

Microservices workflow for content encoding

TalkTalk

Service Fabric Cluster

Content Master App

Content Profile App

Encoding Workflow App

Azure Media Services Job Management App

KeyStore App

Resource Resolution App

Encryption Job Management App

Management API

DocumentDB

(Ad-hoc searching, long term data storage)

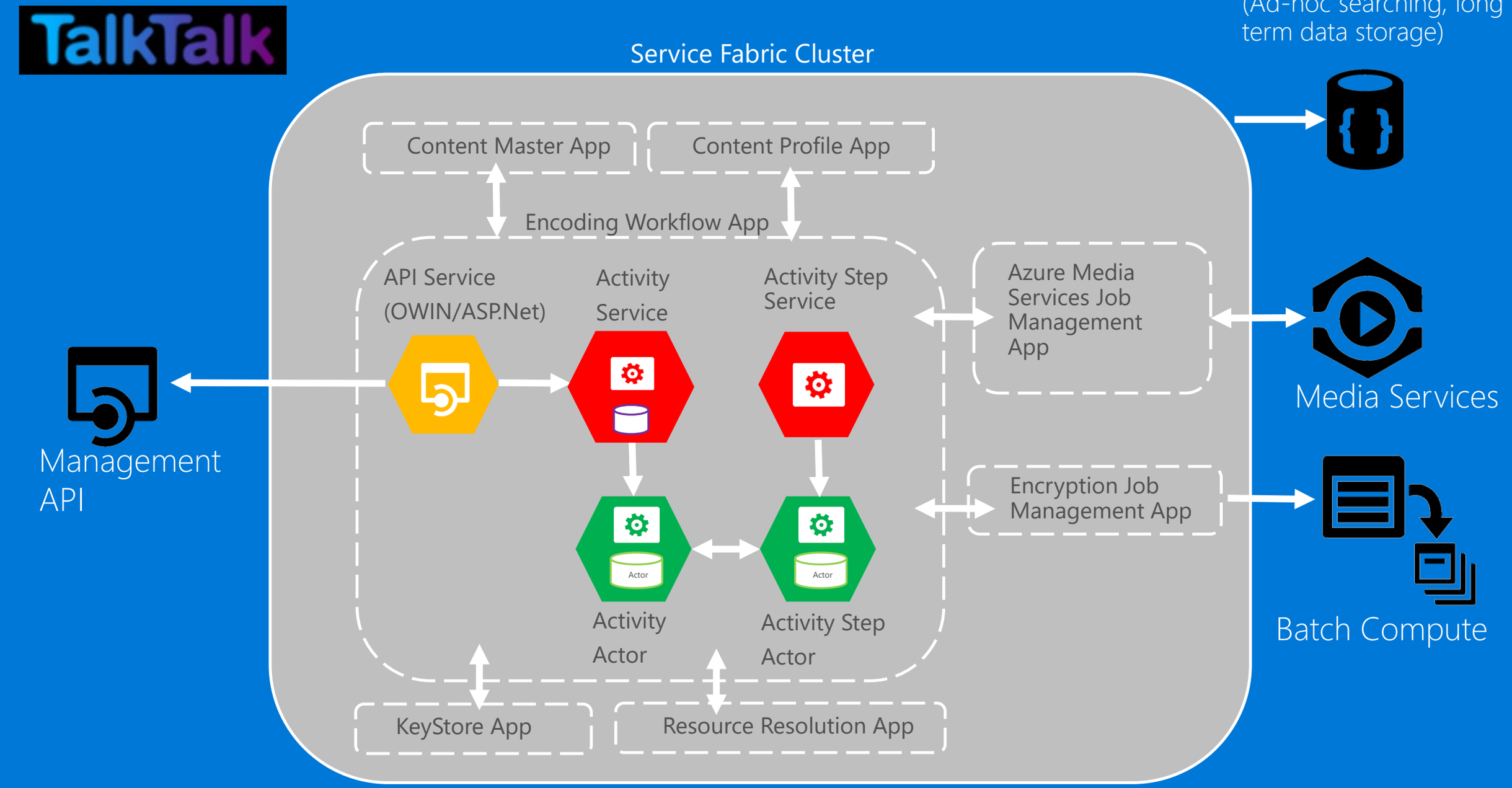
Media Services

Batch Compute

Microservices workflow for content encoding

TalkTalk

DocumentDB
(Ad-hoc searching, long term data storage)



Schneider Electric develops connected technologies and solutions to manage energy and process in ways that are safe, reliable, efficient and sustainable

Benefits

Microservices IoT solution to manage uninterruptable power supplies (UPS)

Scale – Service Fabric simplifies scale. With millions of devices we need partitioning and resource balancing that makes this transparent

Actor Programming API – Service Fabric has the simplest-to-use actor model implementation in the market

Density and Availability – VM utilization enables managing millions of devices with automatic failover

- Management & operation of devices. Query and execute commands, send commands from device to LOB apps
- Communicate with devices securely and in multiple protocols
- Processing and latency need to be sub-second
- Integration Azure services such as Event Hubs and storage.

Schneider Electric

Microservices IoT solution to manage devices