AGILE INTEGRATION

BY

JOHN SYLVESTER

Architecture & Design

→ EAI

→ ESB

→ SOA

→ REST

→ Microservices

→ APIs

People & Process

→ Decentralization

→ Autonomy

→ Multi Skilled Teams (Agile)

Infrastructure & Technology

→ Virtualization

→ Containerization

→ Cloud

Architecture & Design

→ EAI ... 1990

→ ESB ... 2008

→ SOA ... 2008

→ REST ... 2000 ... 2012 to current

→ Microservices ... 2014 to current

→ APIs ... 2014 to current

People & Process

→ Decentralization ... 2012

→ Autonomy ... 2012

→ Multi Skilled Teams (Agile) ... 2012 to current

Infrastructure & Technology

→ Virtualization .. 1990 to current

→ Containerization ...Popular in 2014

→ Cloud ... 2012 to current

Architecture & Design

→ EAI

→ ESB

→ SOA

→ REST

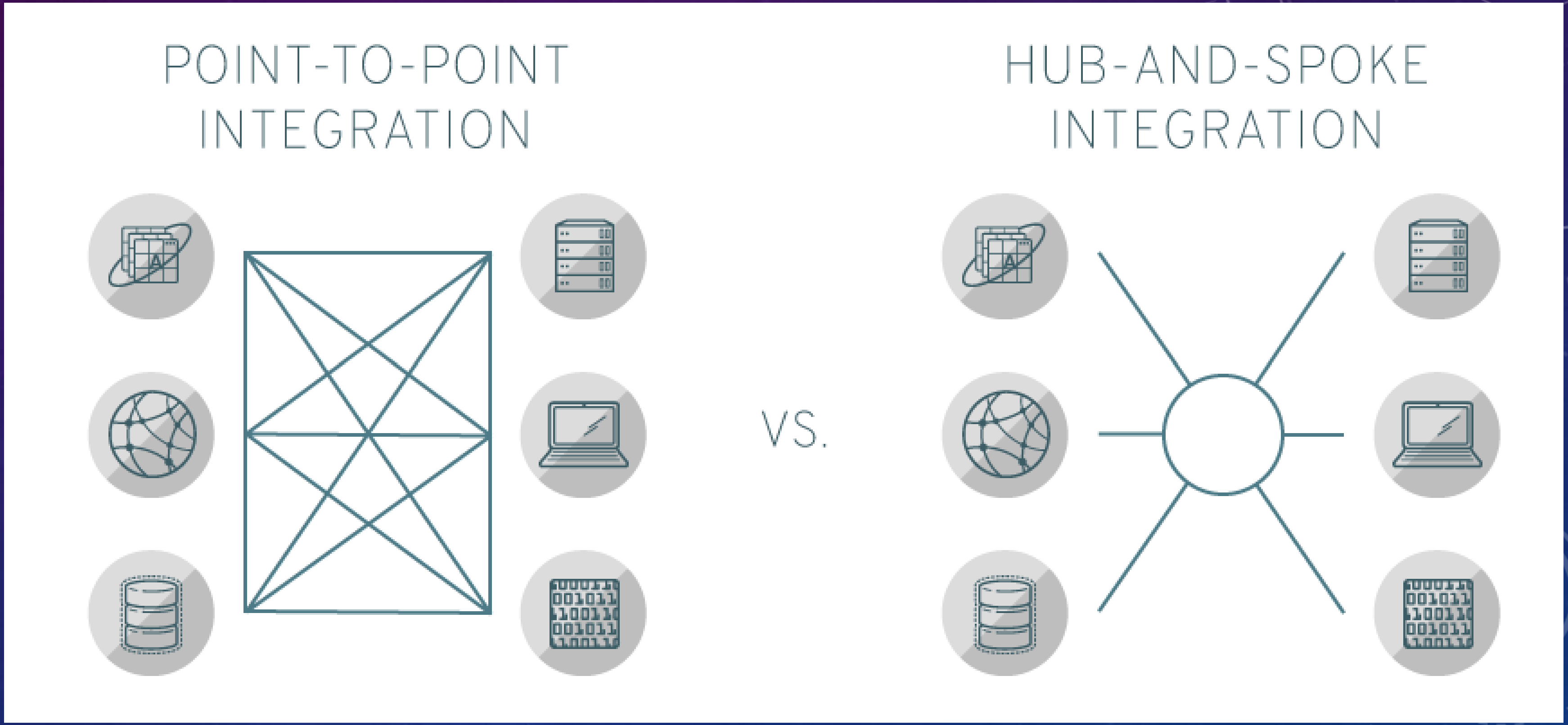
→ Microservices

→ APIs

Architecture & Design

- → **EAI**
- → ESB
- → SOA
- → REST
- → Microservices
- → APIs

Enterprise Application Integration (EAI)



Architecture & Design

→ EAI

→ ESB

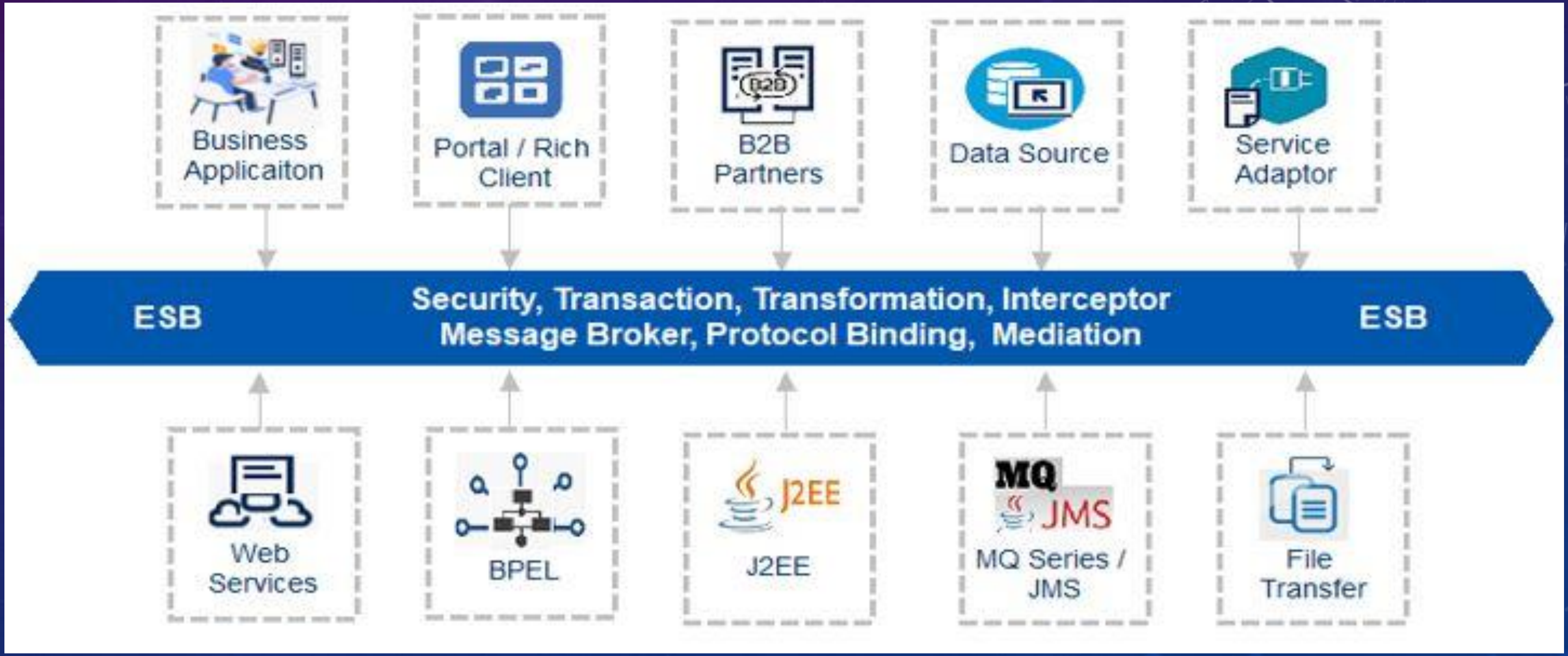
→ SOA

→ REST

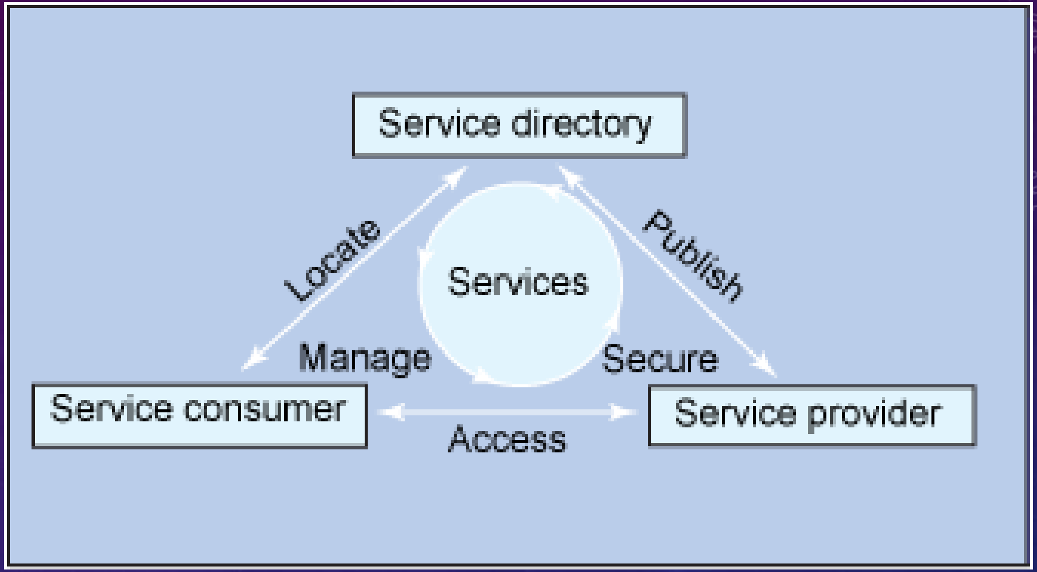
→ Microservices

→ APIs

Enterprise Service Bus (ESB)

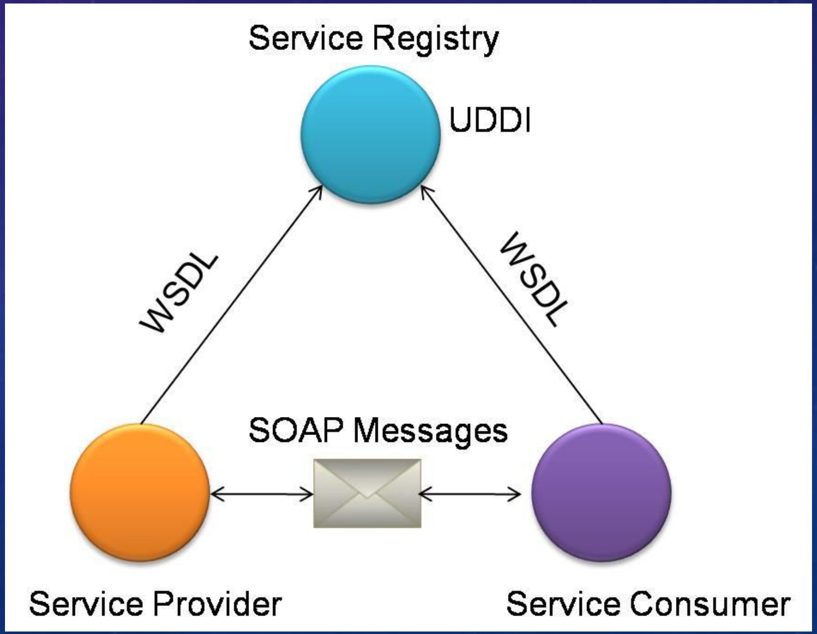


Service Oriented Architecture (SOA)

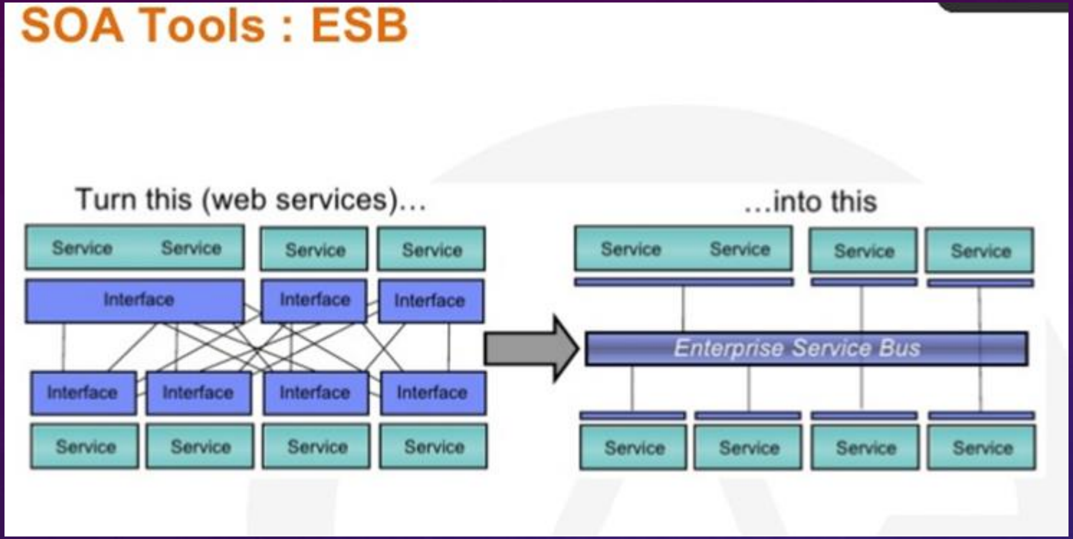


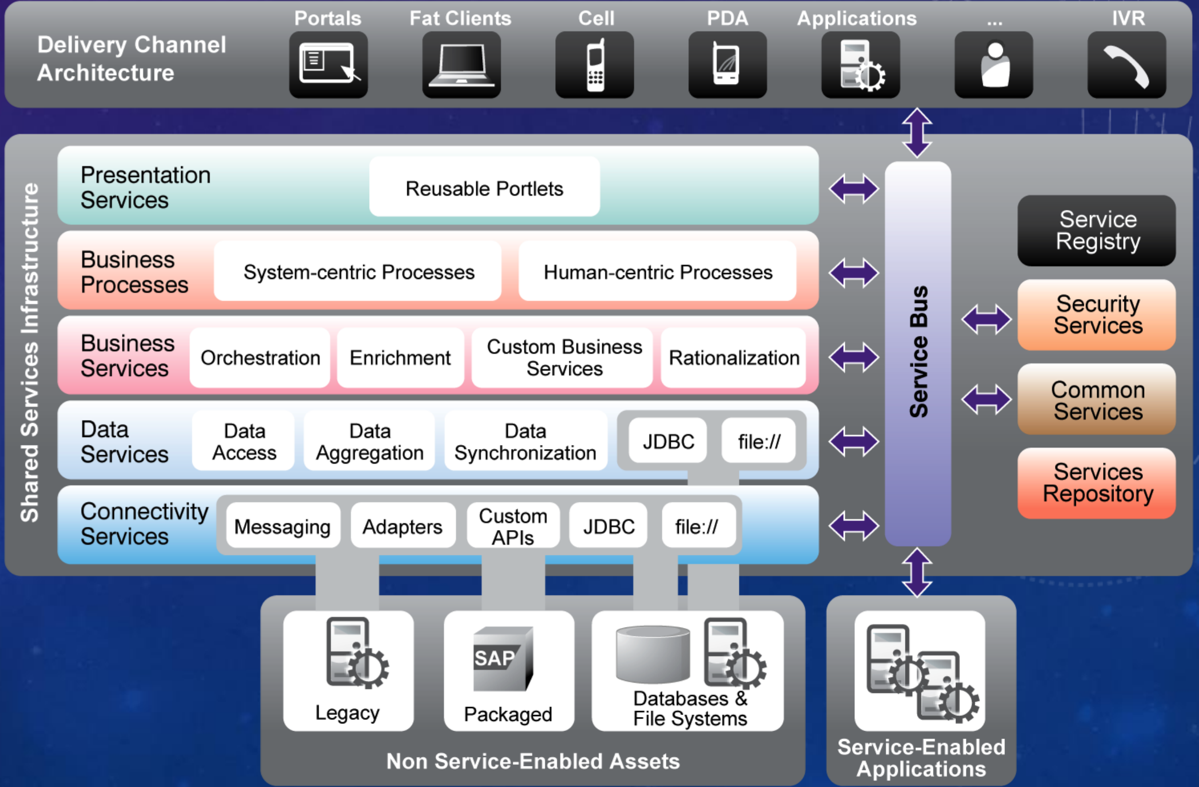
Web services :

HTTP/S, SOAP

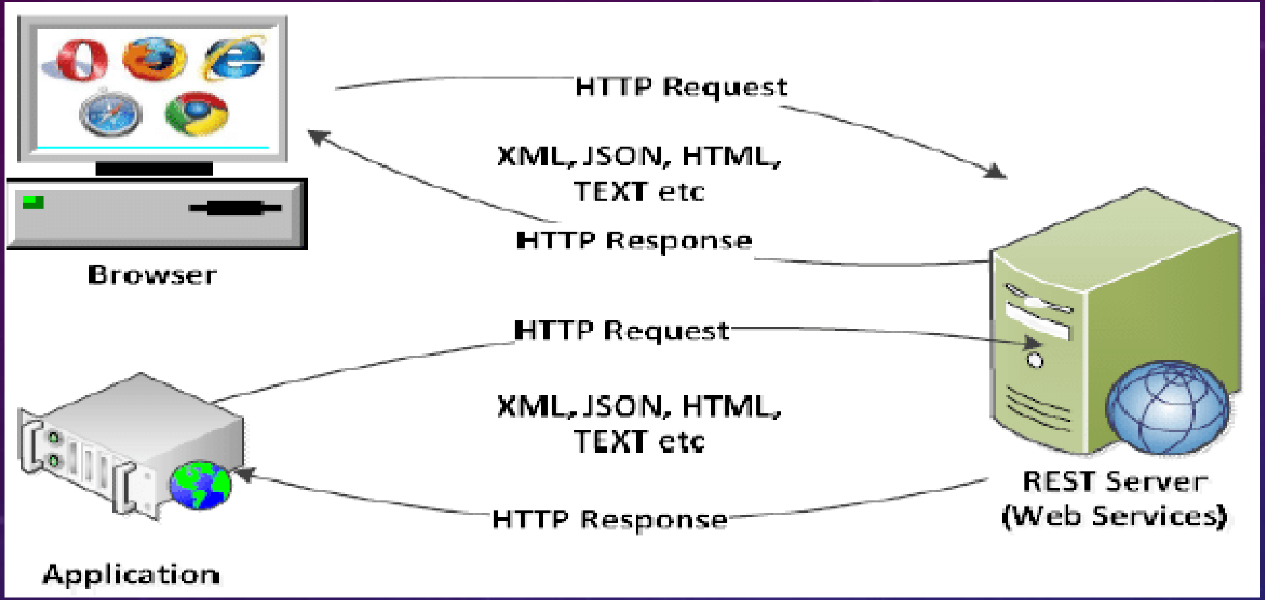


Service Oriented Architecture (SOA) contd..





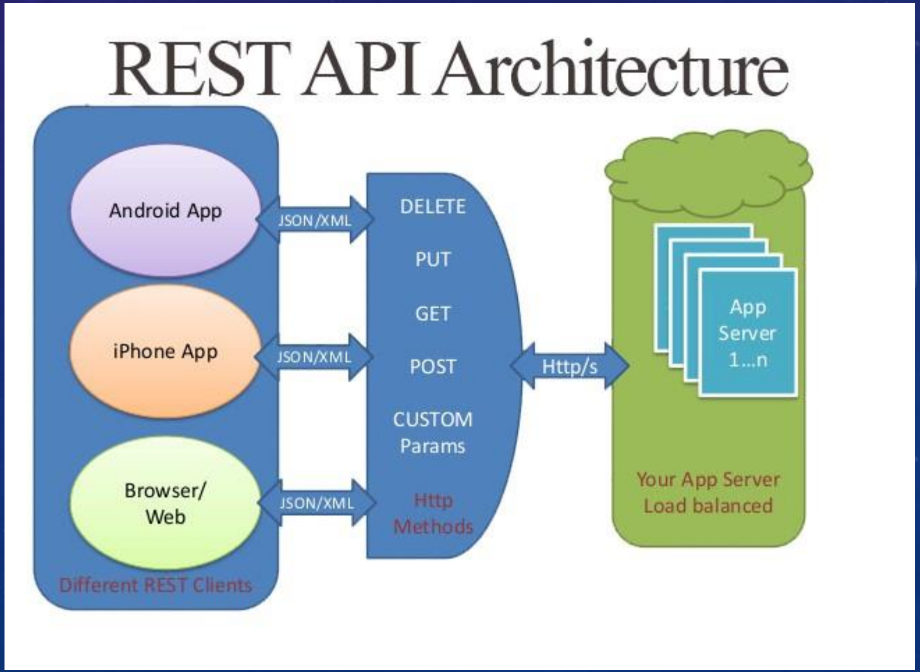
Representational State Transfer (REST)


HTTP Request
XML, JSON, HTML, TEXT etc
HTTP Response
Browser
HTTP Request
XML, JSON, HTML, TEXT etc
HTTP Response
Application
REST Server (Web Services)

REST :
Architecture which concentrates on a single service

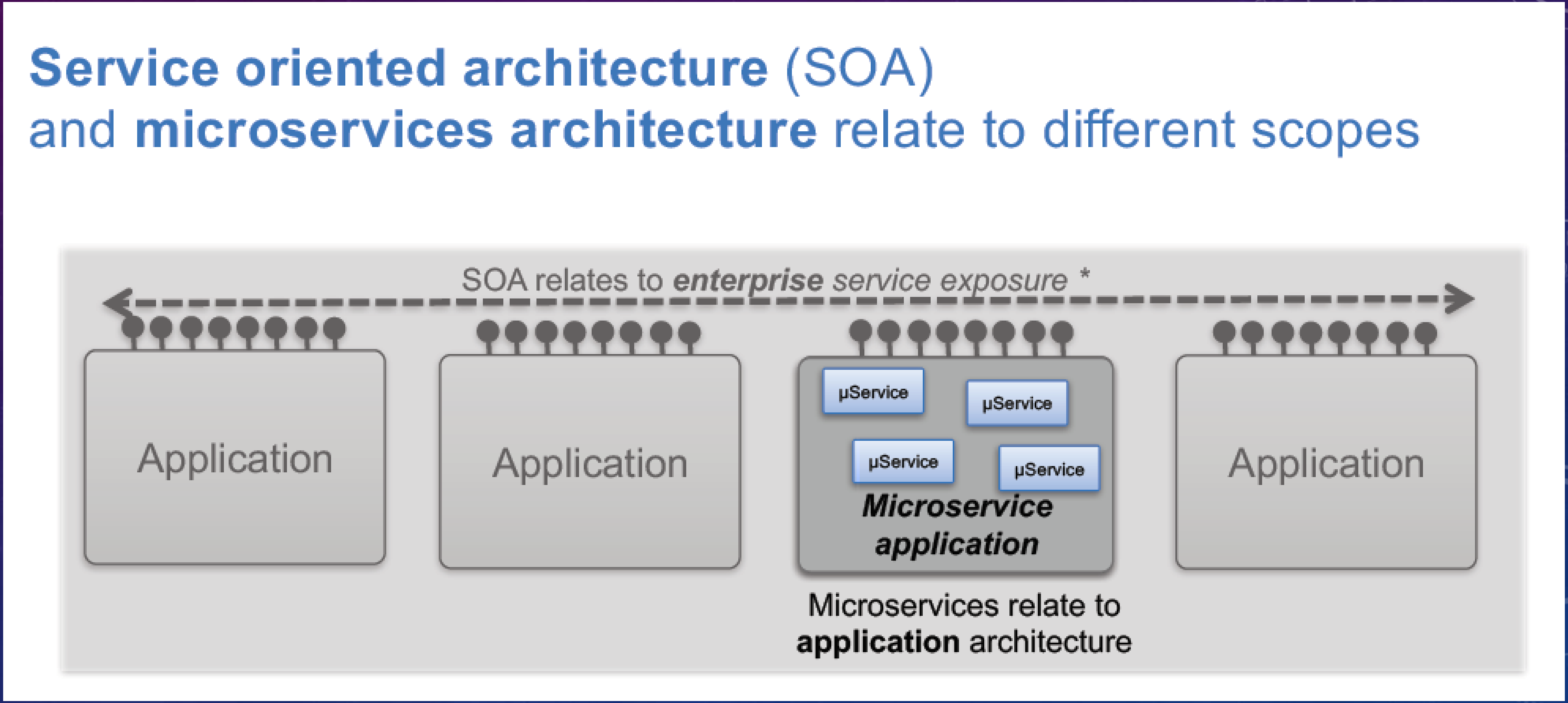
Implementation using : HTTP, JSON, SOAP

Common terms used : RESTful APIs, RESTful Webservices



Architecture & Design

Microservices


Service oriented architecture (SOA)
and **microservices architecture** relate to different scopes

SOA relates to ***enterprise*** *service exposure* *
Application
Application
µService
µService
µService
µService
Microservice application
Application
Microservices relate to **application** architecture

API = Application Programming Interface

APIs are present everywhere

→ OS

→ Programming Language

→ Webservices

→ Hardware APIs

etc…

APIs of two types :

→ Technical APIs

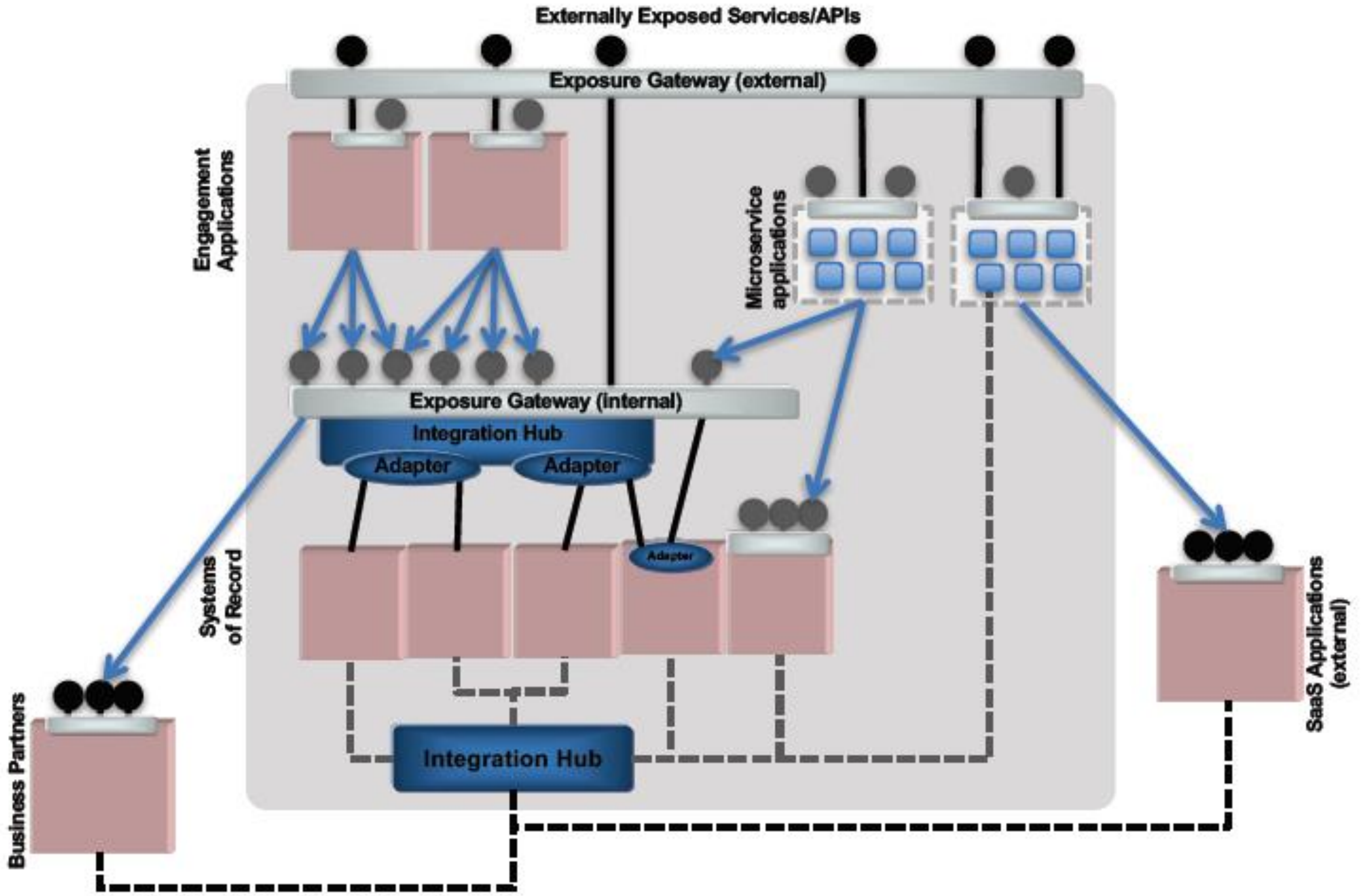
→ Functional APIs

APIs can be :

→ Simple

→ Composite

What does a real integration architecture look like?


Externally Exposed Services/APIs
Exposure Gateway (external)
Engagement Applications
Microservice applications
Exposure Gateway (internal)
Integration Hub
Adapter
Adapter
Adapter
Systems of Record
SaaS Applications (external)
Business Partners
Integration Hub

People & Process

→ Decentralization

→ Autonomy

→ Multi Skilled Teams (Agile)

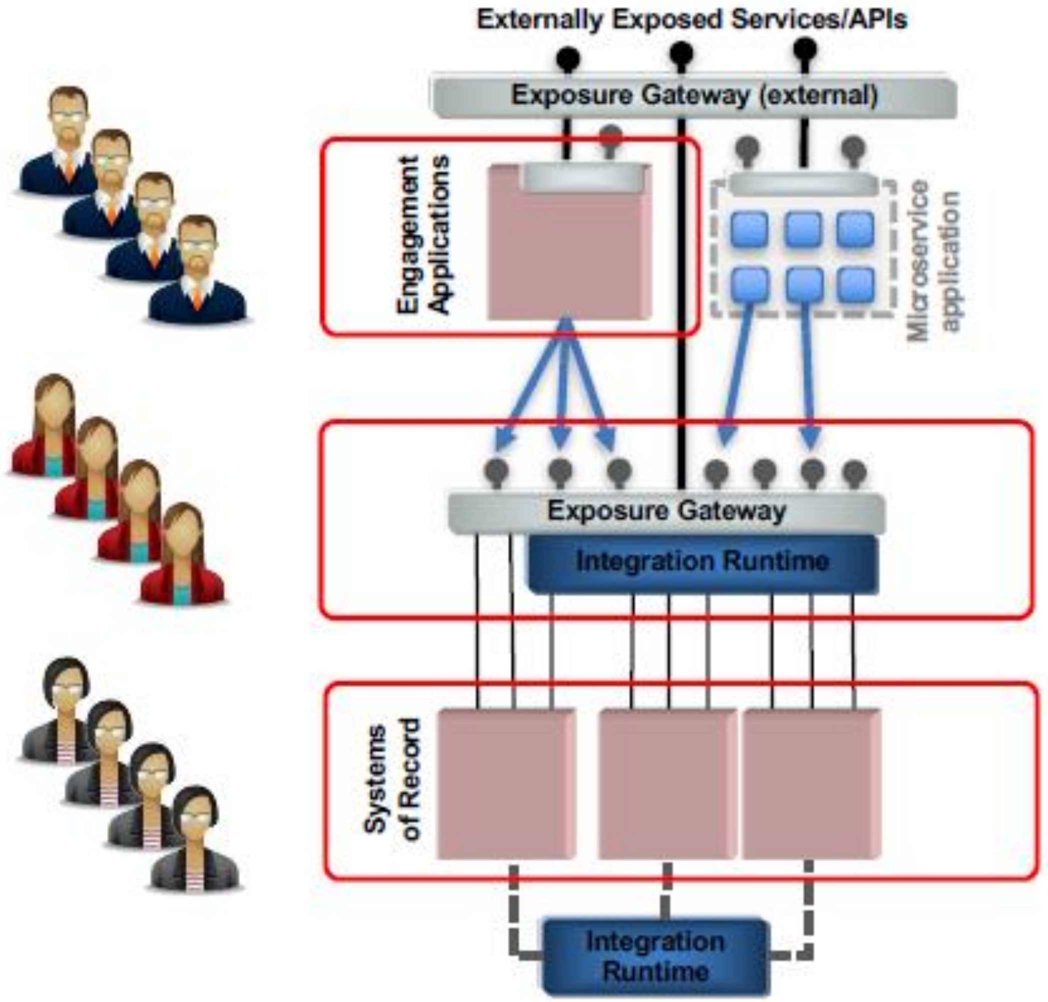
People & Process

→ Decentralization

→ Autonomy

→ Multi Skilled Teams (Agile)

The people aspect of decentralization

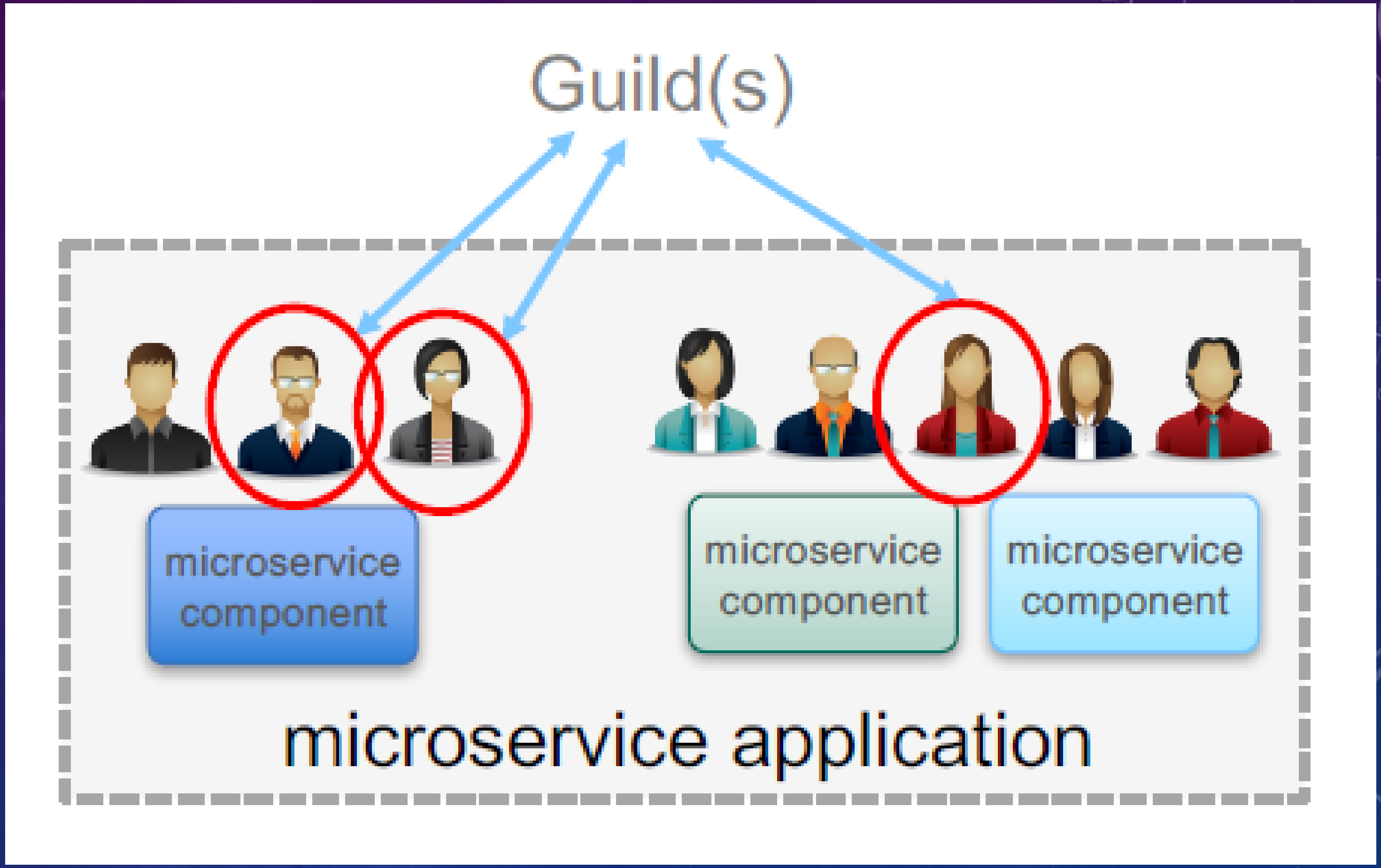


People & Process

→ Decentralization

→ Autonomy

→ Multi Skilled Teams (Agile)

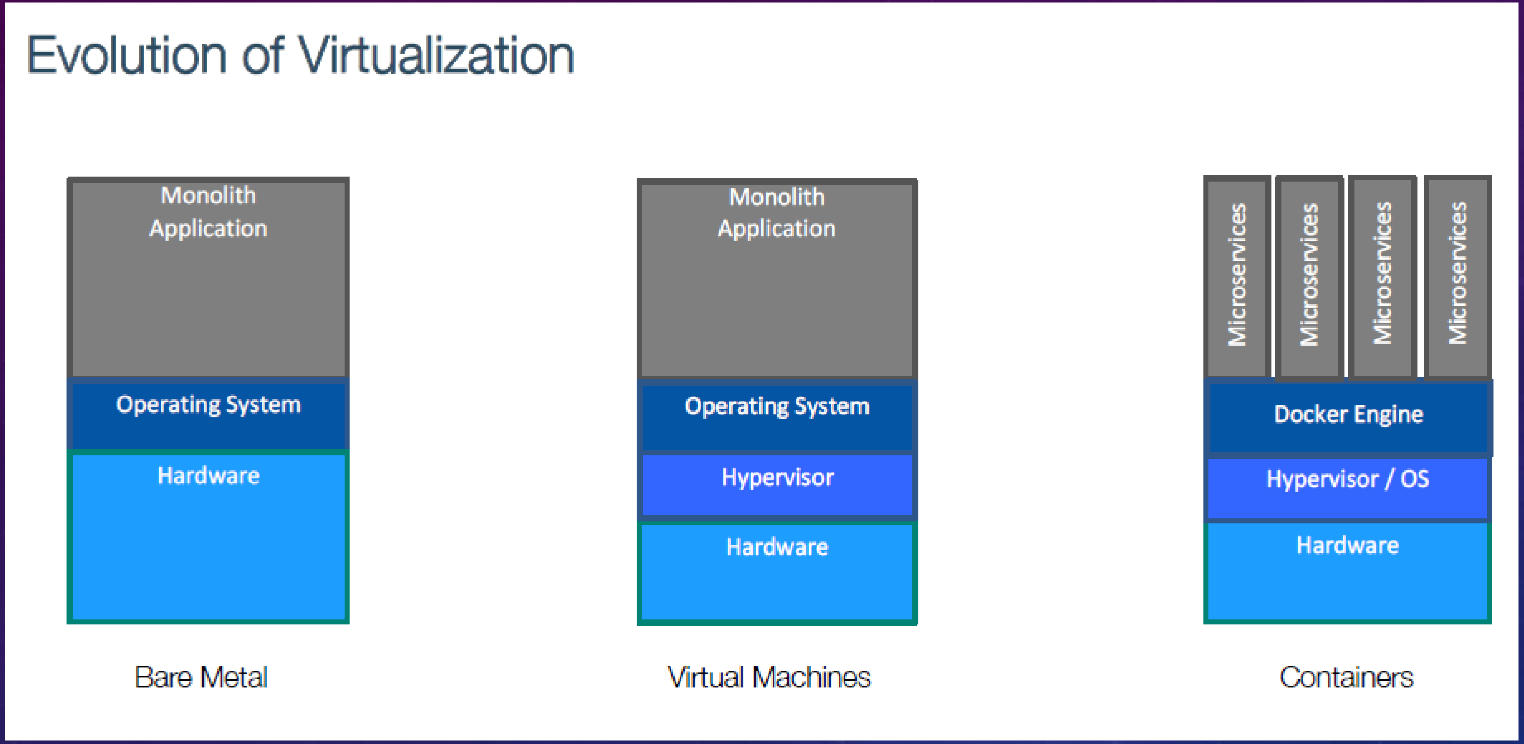


Infrastructure & Technology

→ Virtualization

→ Containerization

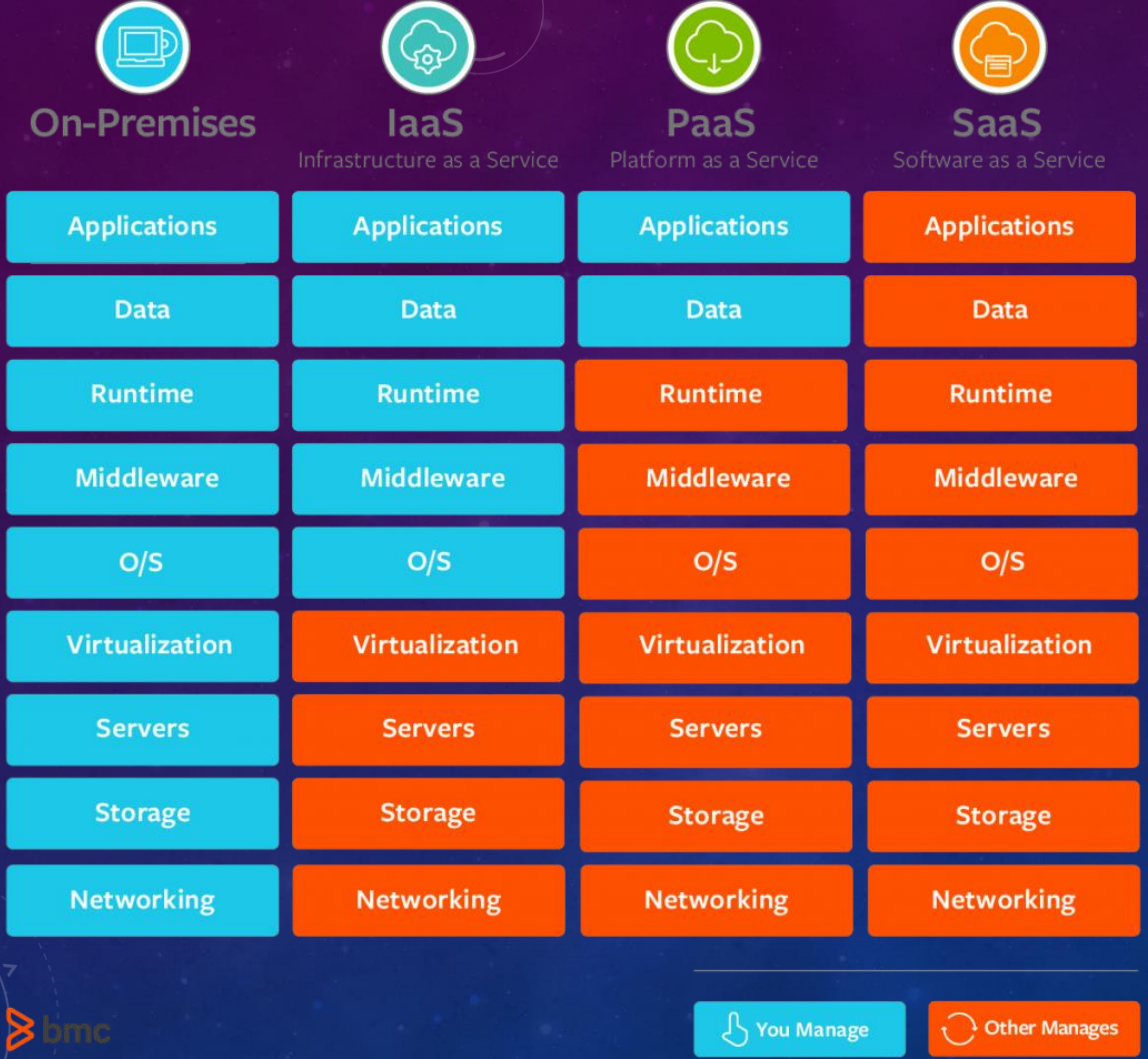
→ Cloud



→ **Virtualization**

→ **Containerization**

→ Cloud



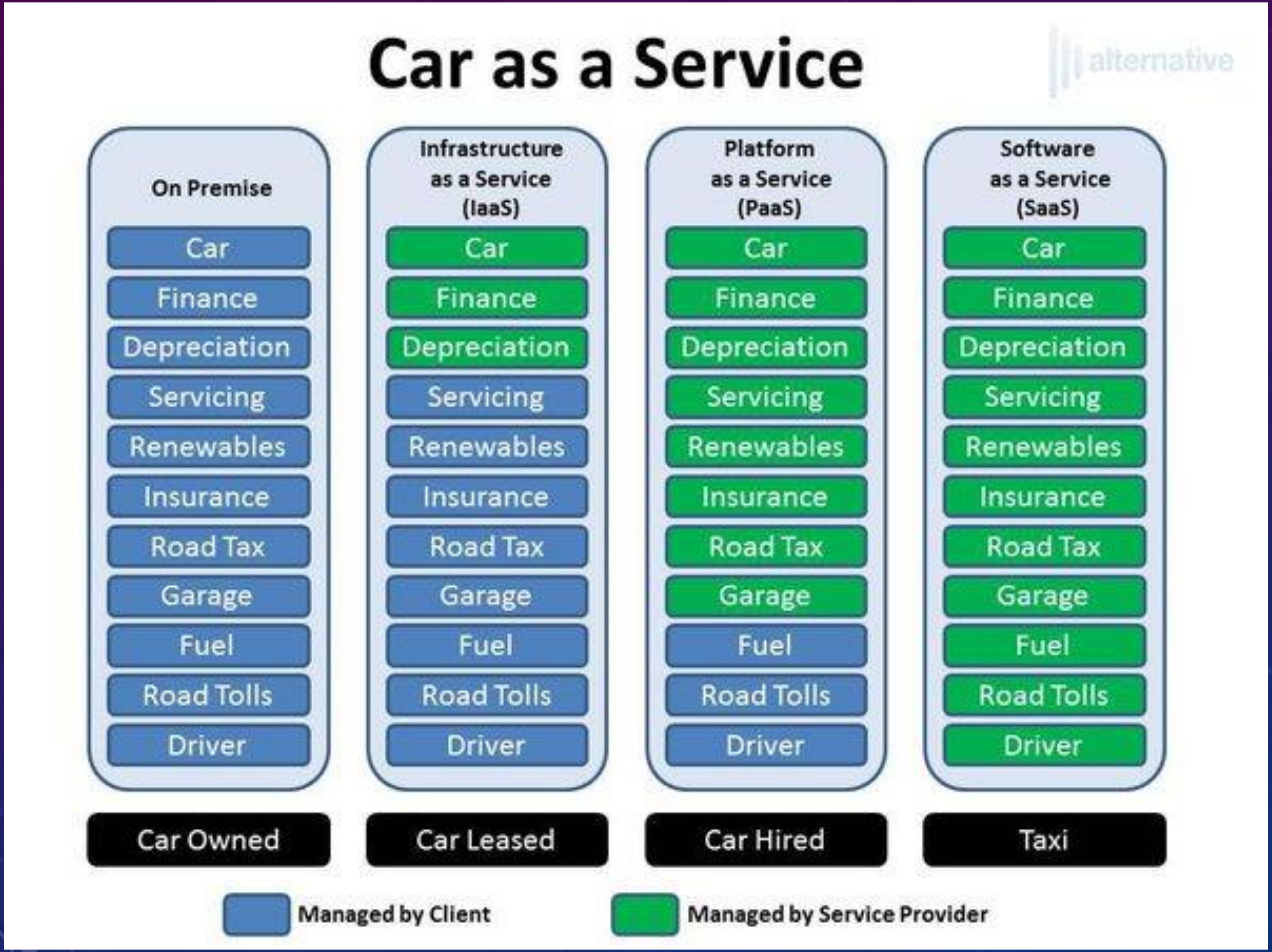
On-Premises	IaaS Infrastructure as a Service	PaaS Platform as a Service	SaaS Software as a Service
🟦 Applications	🟦 Applications	🟦 Applications	🟧 Applications
🟦 Data	🟦 Data	🟦 Data	🟧 Data
🟦 Runtime	🟦 Runtime	🟧 Runtime	🟧 Runtime
🟦 Middleware	🟦 Middleware	🟧 Middleware	🟧 Middleware
🟦 O/S	🟦 O/S	🟧 O/S	🟧 O/S
🟦 Virtualization	🟧 Virtualization	🟧 Virtualization	🟧 Virtualization
🟦 Servers	🟧 Servers	🟧 Servers	🟧 Servers
🟦 Storage	🟧 Storage	🟧 Storage	🟧 Storage
🟦 Networking	🟧 Networking	🟧 Networking	🟧 Networking

🟦 You Manage
🟧 Other Manages

bmc

Infrastructure & Technology

- → Virtualization
- → Containerization
- → **Cloud**



Car as a Service

alternative

On Premise	Infrastructure as a Service (IaaS)	Platform as a Service (PaaS)	Software as a Service (SaaS)
🟦 Car	🟩 Car	🟩 Car	🟩 Car
🟦 Finance	🟩 Finance	🟩 Finance	🟩 Finance
🟦 Depreciation	🟩 Depreciation	🟩 Depreciation	🟩 Depreciation
🟦 Servicing	🟦 Servicing	🟩 Servicing	🟩 Servicing
🟦 Renewables	🟦 Renewables	🟩 Renewables	🟩 Renewables
🟦 Insurance	🟦 Insurance	🟩 Insurance	🟩 Insurance
🟦 Road Tax	🟦 Road Tax	🟩 Road Tax	🟩 Road Tax
🟦 Garage	🟦 Garage	🟩 Garage	🟩 Garage
🟦 Fuel	🟦 Fuel	🟦 Fuel	🟩 Fuel
🟦 Road Tolls	🟦 Road Tolls	🟦 Road Tolls	🟩 Road Tolls
🟦 Driver	🟦 Driver	🟦 Driver	🟩 Driver
Car Owned	Car Leased	Car Hired	Taxi

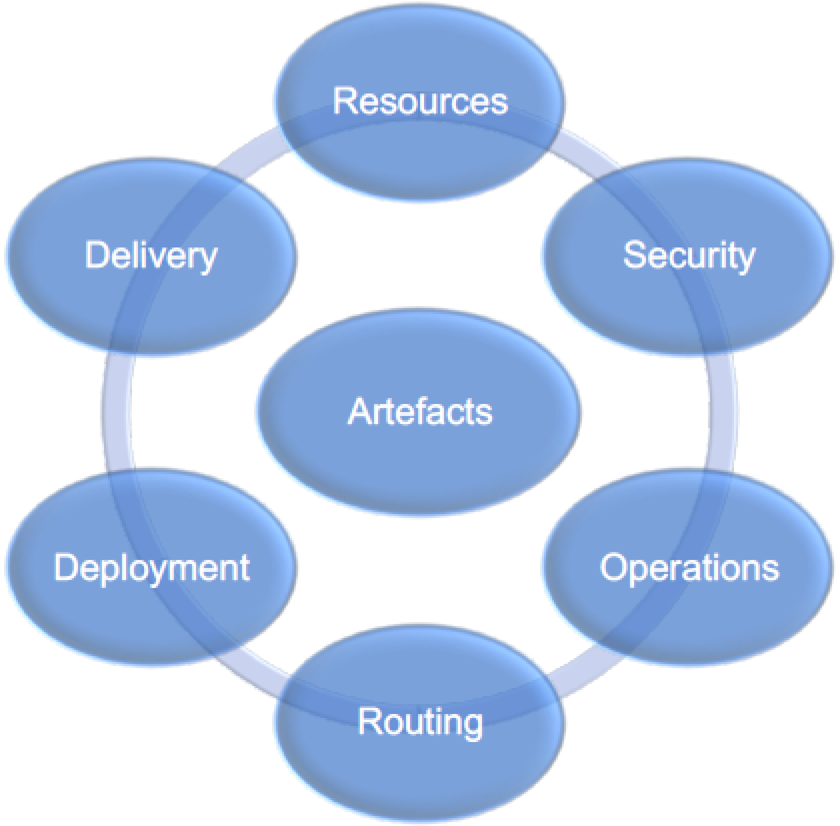
🟦 Managed by Client 🟩 Managed by Service Provider

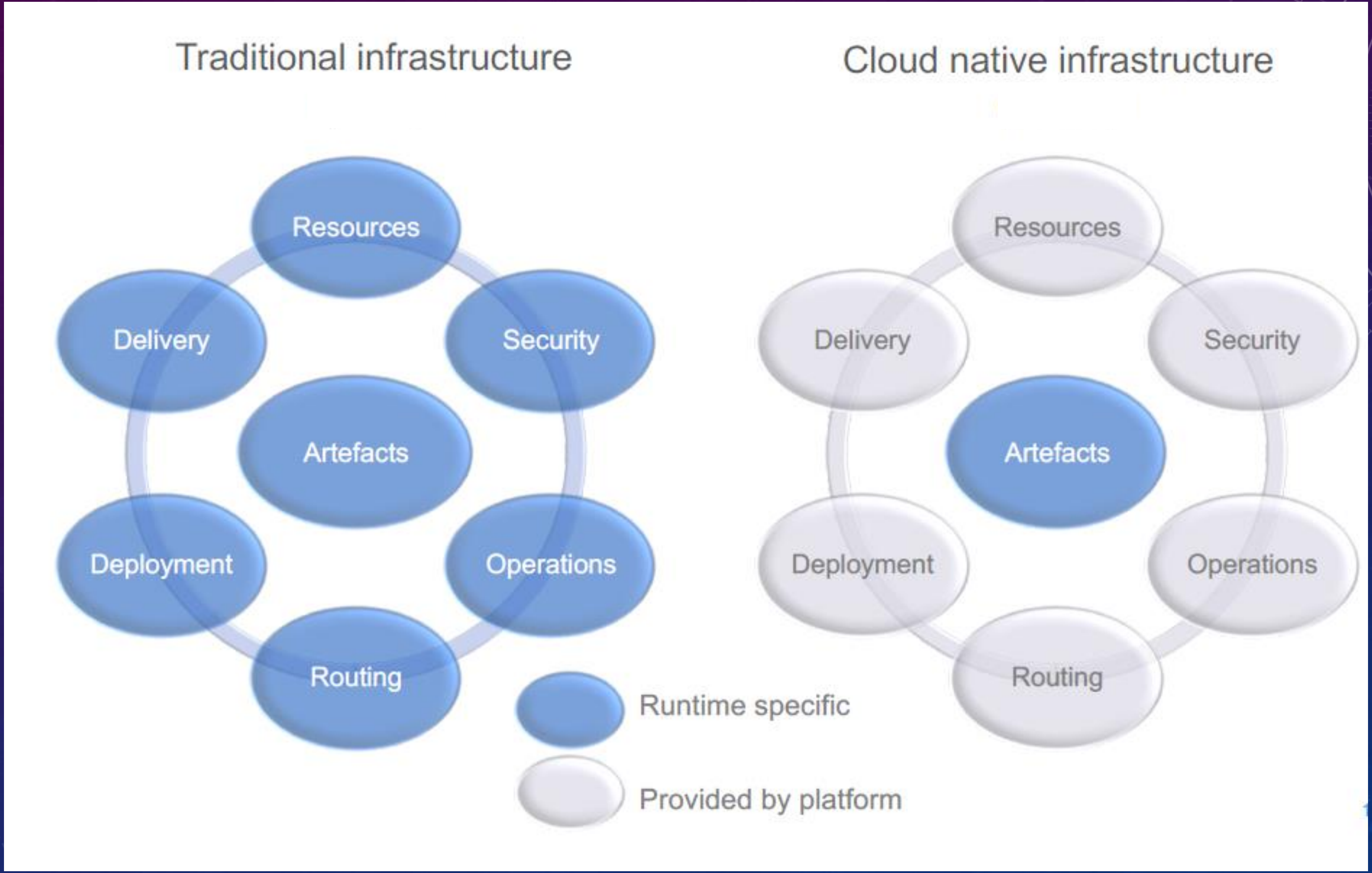
Infrastructure & Technology

- → Virtualization
- → Containerization
- → **Cloud**

What skills and capabilities are required to get something into production?

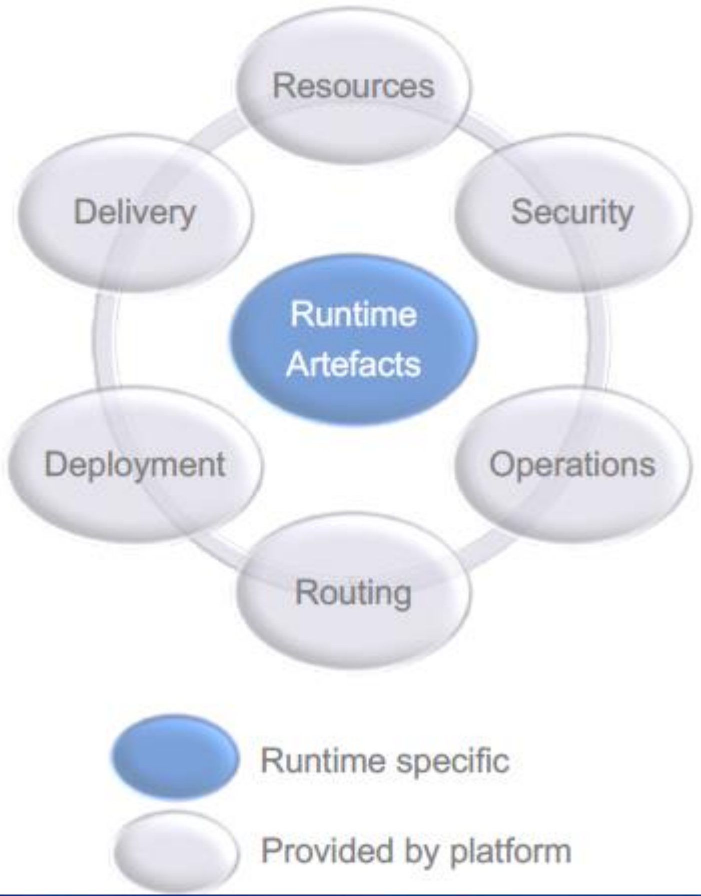
- Author **artefacts**
  (design, implement, maintain)
- Arrange **delivery**
  (source control, install, compile, build, verify, test)
- Allocate **resources**
  (cpu, memory, storage, connections)
- Configure **routing**
  (load balancing, failover, traffic control, re-try, timeout, load shedding/shaping, request tracing)
- Enforce **security**
  (authentication, access control, certificates, encryption, port provisioning)
- Perform **deployment**
  (scaling, distributed deploy, rolling update, A/B testing, blue/green deployment, canary releases)
- Manage **operations**
  (health checks, monitoring, tracing, log aggregation, quotas)



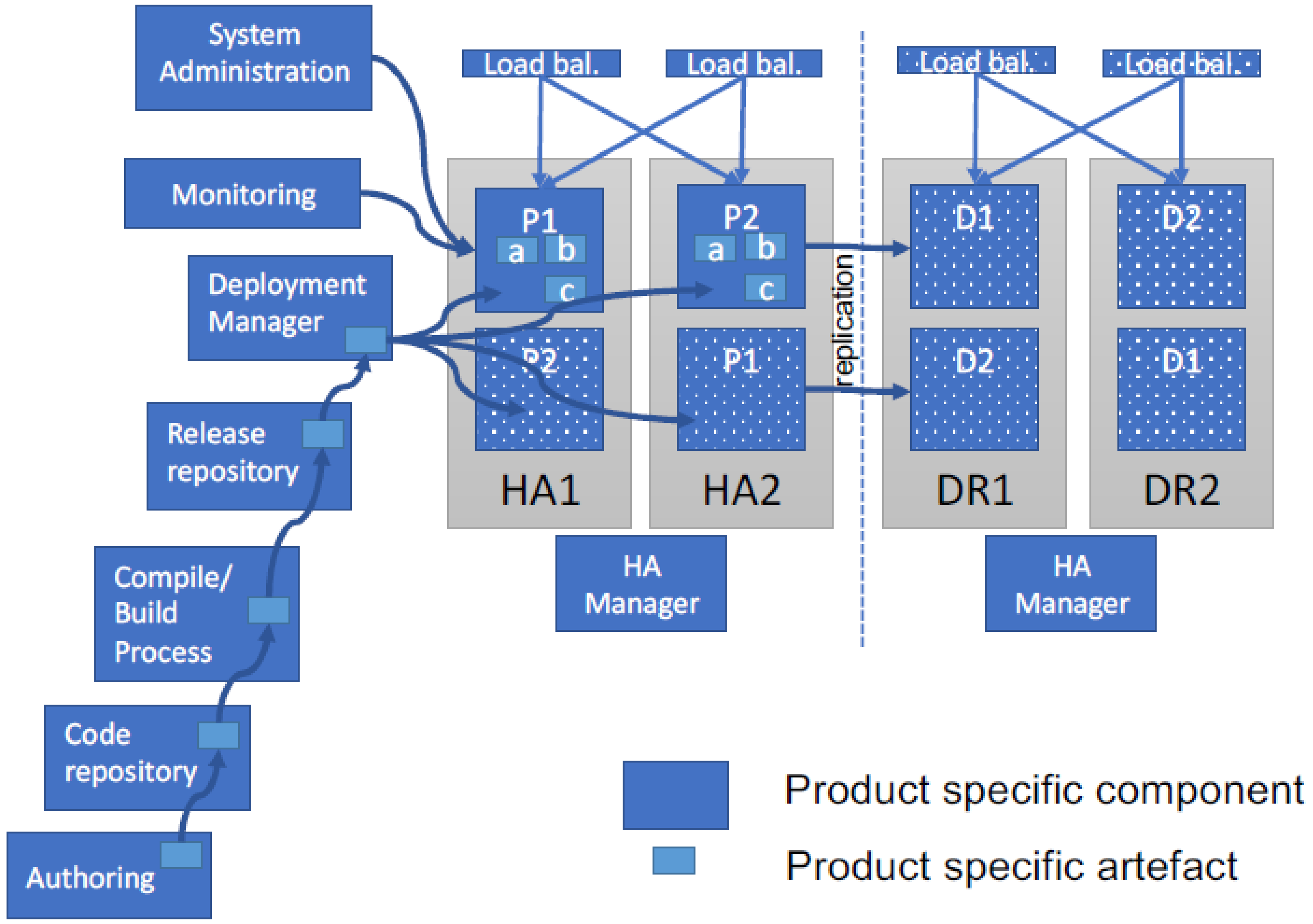

Traditional infrastructure
Resources
Delivery
Security
Artefacts
Deployment
Operations
Routing
Cloud native infrastructure
Resources
Delivery
Security
Artefacts
Deployment
Operations
Routing
Runtime specific
Provided by platform

What capabilities does the cloud platform provide to all runtimes

- Delivery
  - e.g. Docker build, Jenkins, Git
- Resource allocation
  - e.g. Kubernetes, Mesos
- Deployment
  - e.g. Kubernetes, Helm
- Routing
  - e.g. Istio, Linkerd
- Security
  - e.g. Kubernetes/Istio/SPIFFE
- Operations
  - e.g. ELK stack



Challenges of traditional deployment topologies



Characteristics

HA pairs

Scaling manual and vertical

Defined nodes

Explicit install and configure

Explicit cold/warm HA & DR

Peak CPU licensing

Dedicated OS instances/HW

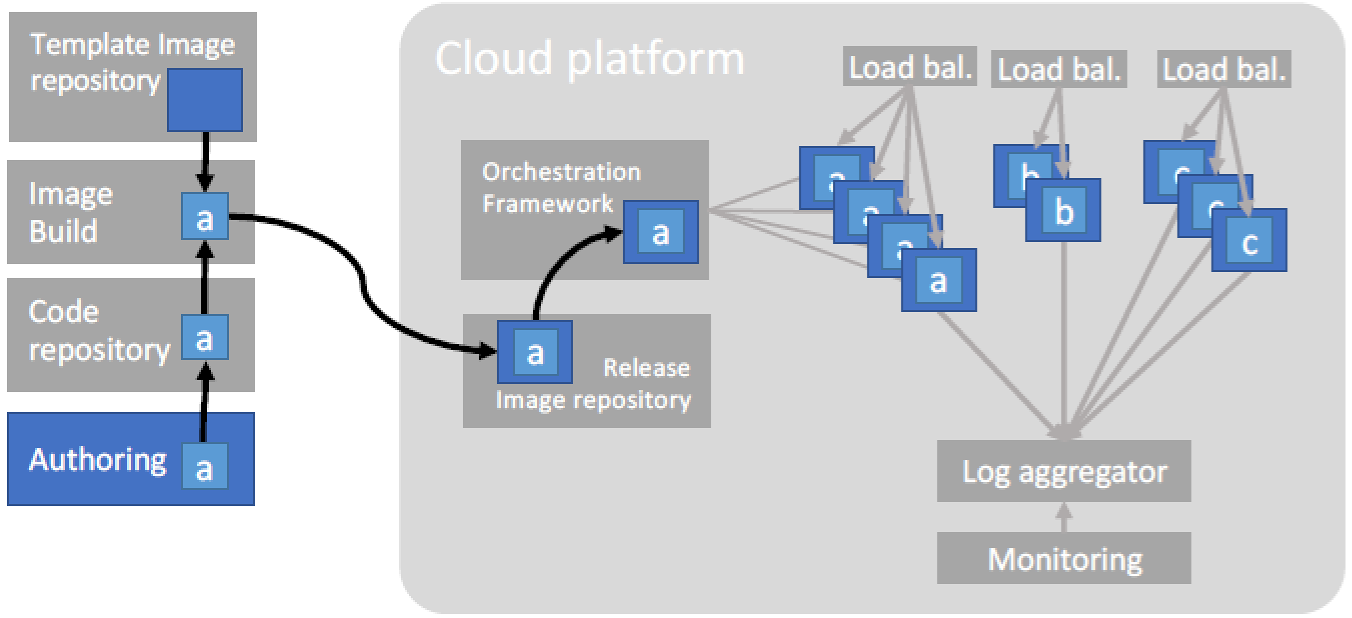
Deploy to running shared servers

Replication across DCs

Administer live shared servers

Code is only joined with the servers at deployment.

Simplicity and scaling benefits of cloud native platforms



Elastically scaled containers

Pooled shared underlying resources, but decoupled containers

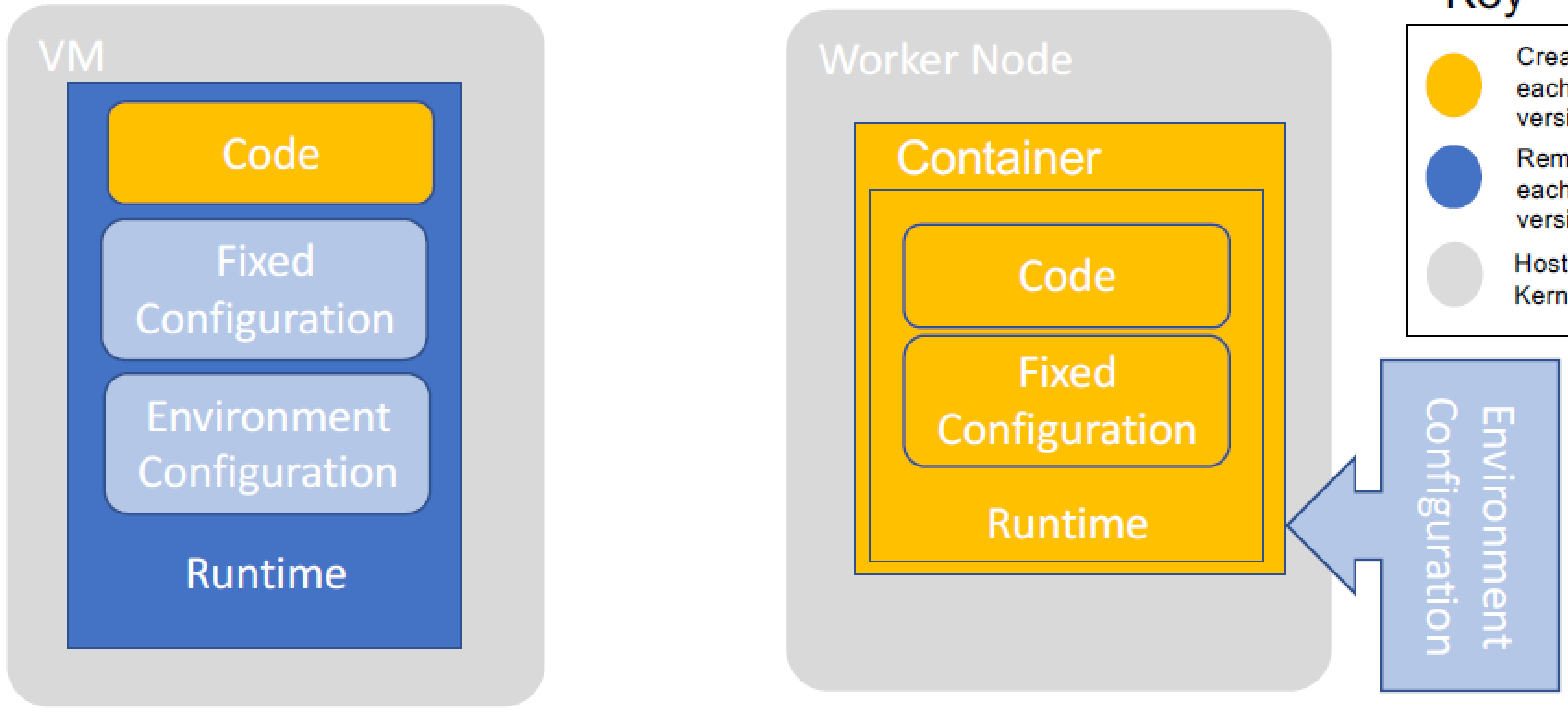
Implicit HA/DR

Deploy by image combining artefacts and infrastructure

Administer image then redeploy, not hot fixing.

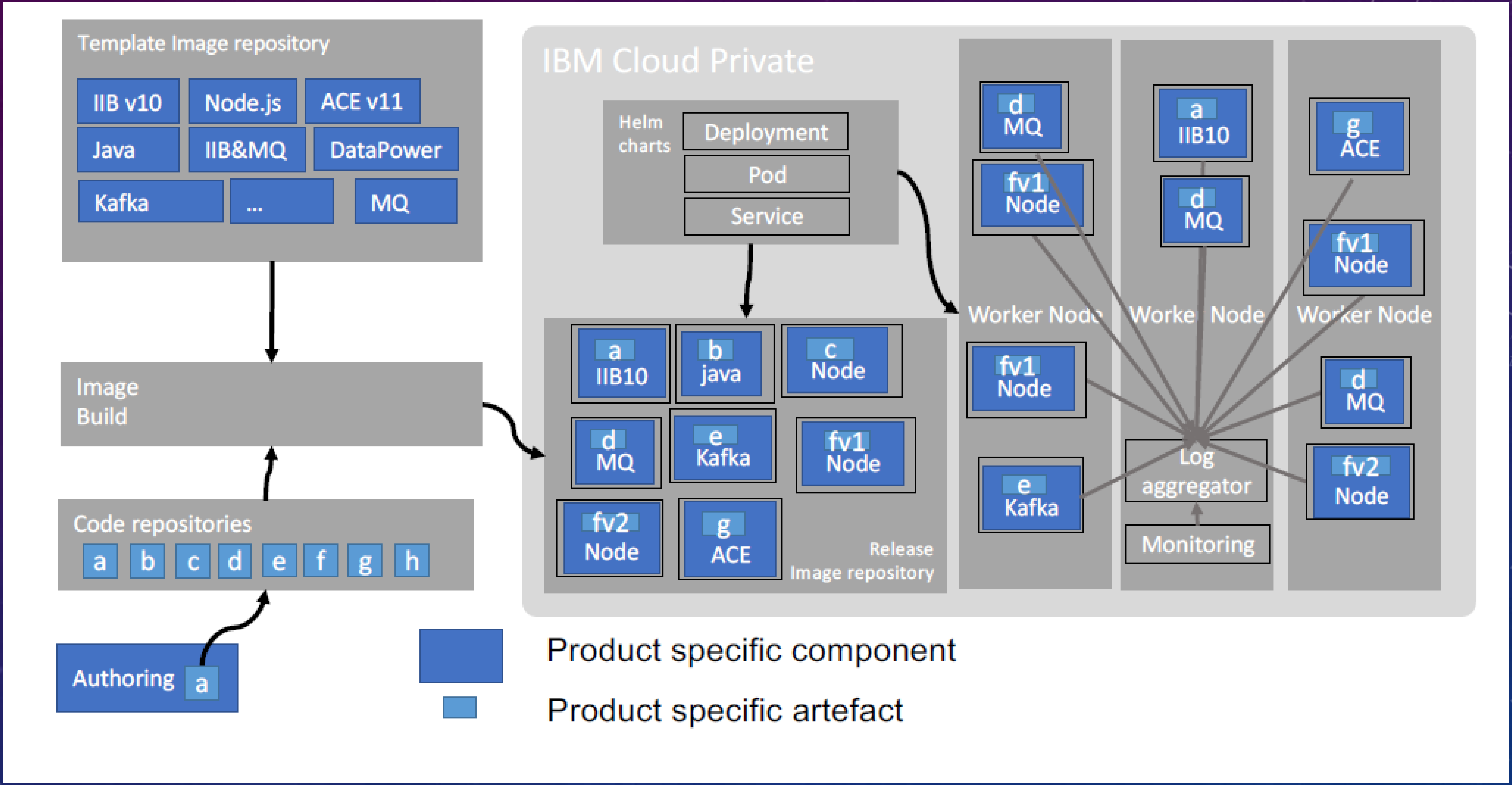
🟦	Product specific component
🟦	Product specific artefact

What Moves Per Release

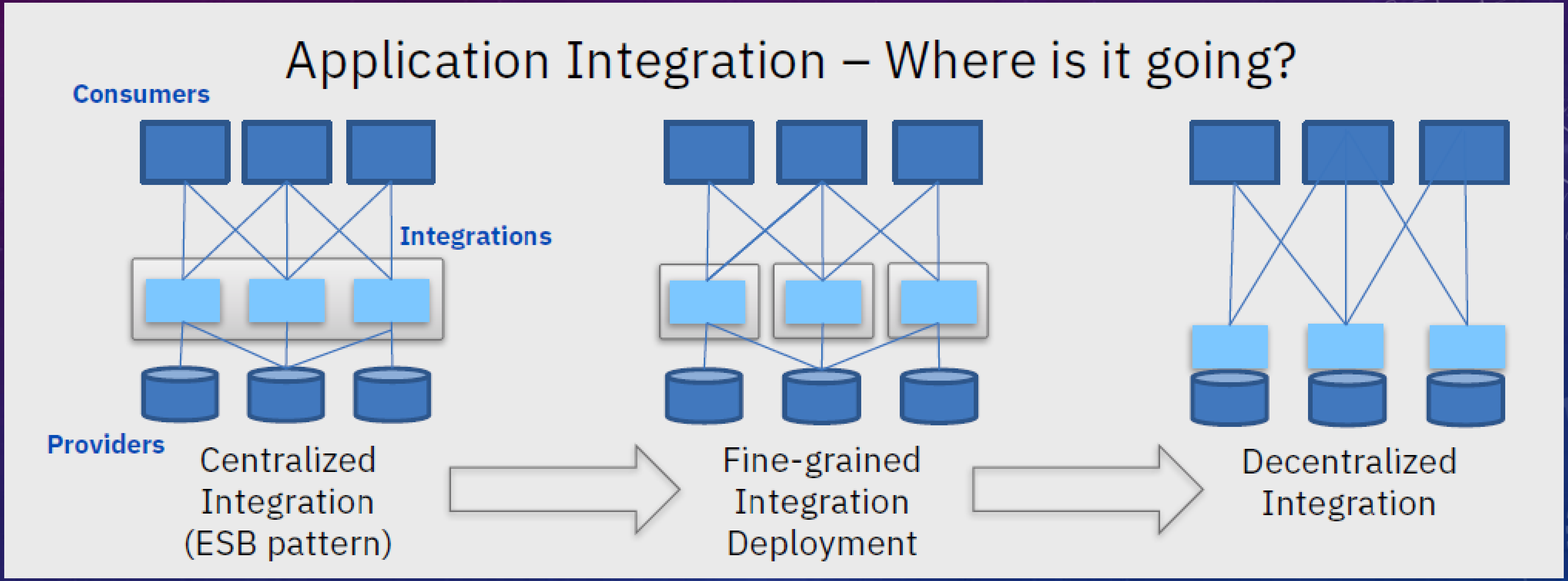


Key

●	Created new for each new code version
●	Remains same for each new code version
●	Host – including Kernel

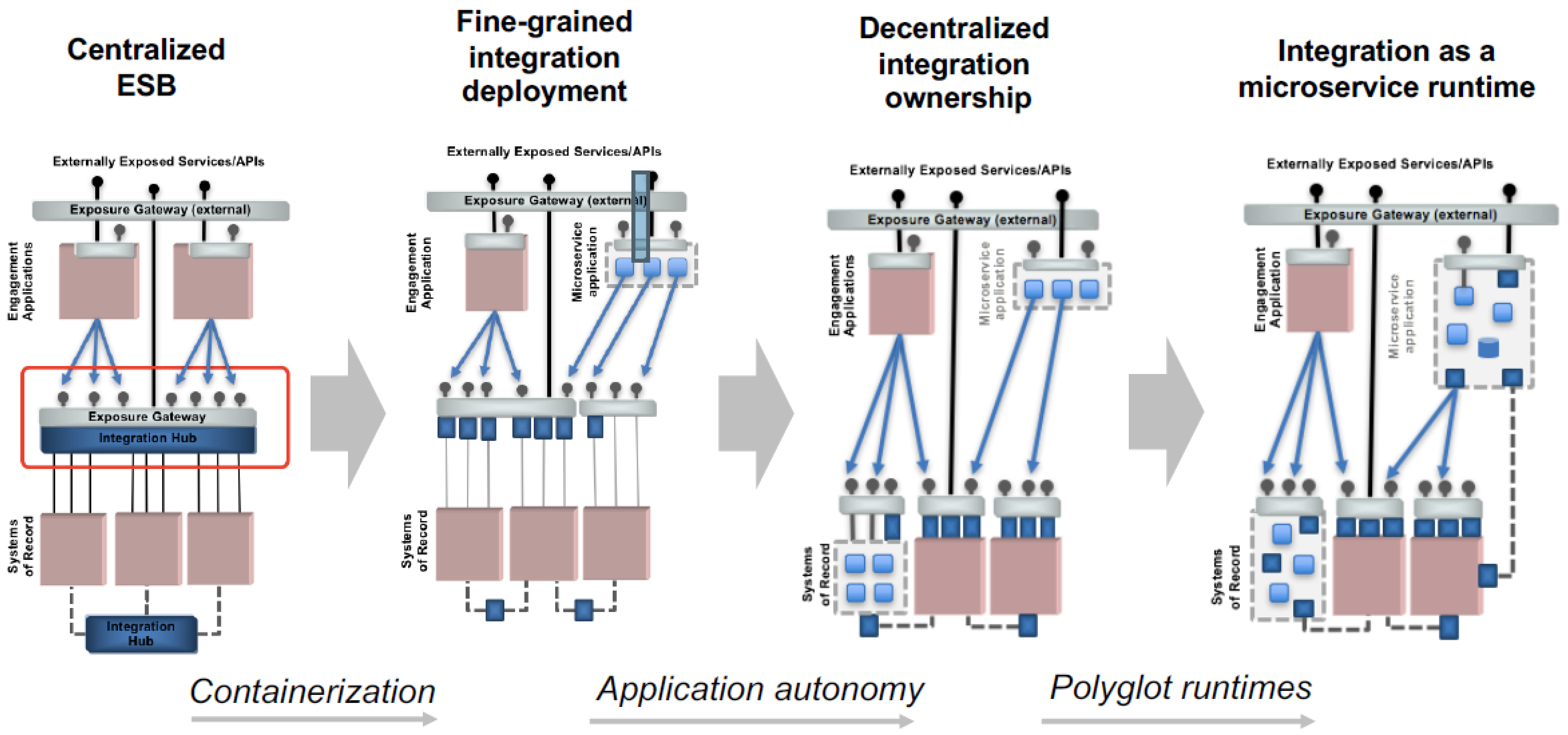

Template Image repository
IIB v10
Node.js
ACE v11
Java
IIB&MQ
DataPower
Kafka
...
MQ
Image Build
Code repositories
a b c d e f g h
Authoring a
IBM Cloud Private
Helm charts
Deployment
Pod
Service
a IIB10
b java
c Node
d MQ
e Kafka
fv1 Node
fv2 Node
g ACE
Release Image repository
d MQ
fv1 Node
Worker Node
fv1 Node
e Kafka
a IIB10
d MQ
Worker Node
Log aggregator
Monitoring
g ACE
fv1 Node
Worker Node
d MQ
fv2 Node
Product specific component
Product specific artefact

Application Integration also needs change…


Application Integration – Where is it going?
Consumers
Integrations
Providers
Centralized Integration (ESB pattern)
Fine-grained Integration Deployment
Decentralized Integration

Application Integration transformation to AGILE INTEGRATION

The fate of the ESB Pattern: Moving to agile integration



References :

1. http://ibm.biz/AgileIntegArchLinks
2. The fate of the ESB : http://ibm.biz/FateOfTheESBPaper
3. Moving to lightweight, agile integration : http://ibm.biz/AgileIntegArchPaper