GIGASPACES
innovate with confidence

Real-Time Analytics Meets Kubernetes

Tal Doron
Director, Technology Innovation

ABOUT ME

@taldoron

in taldoron84

tald@gigaspaces.com

Tal Doron

Director, Technology Innovation

About GigaSpaces

We provide one of the leading in-memory computing platforms for real-time insight to action and extreme transactional processing. With GigaSpaces, enterprises can operationalize machine learning and transactional processing to gain real-time insights on their data and act upon them in the moment.

INSIGHTEDGE

InsightEdge is an in-memory real-time analytics platform for instant insights to action; analyzing data as it's born, enriching it with historical context, for smarter, faster decisions

XAP

In-Memory Computing Platform for microsecond scale transactional processing, data scalability, and powerful event-driven workflows

300+
Direct customers

50+ / 500+
Fortune / Organizations

5,000+
Large installations in production (OEM)

25+
ISVs

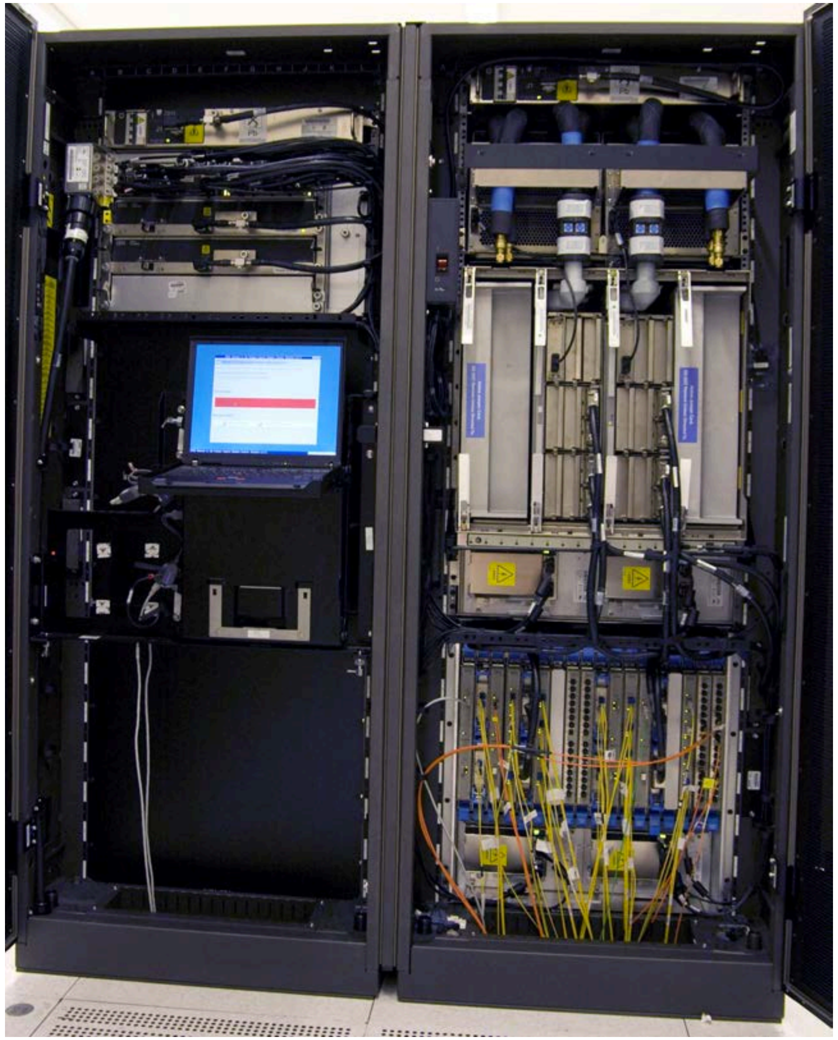
Why

* Intro pictures from Wikipedia

Dinosaurs

Dinosaurs

Dinosaurs



We’ve looked up to the stars

Not without first passing through the clouds

It’s the smallest of opponents that are gamechangers

We needed to find a way to **ship** man there…

The first flight of an airplane, the Wright Flyer on December 17, 1903

How do we become cloud native?

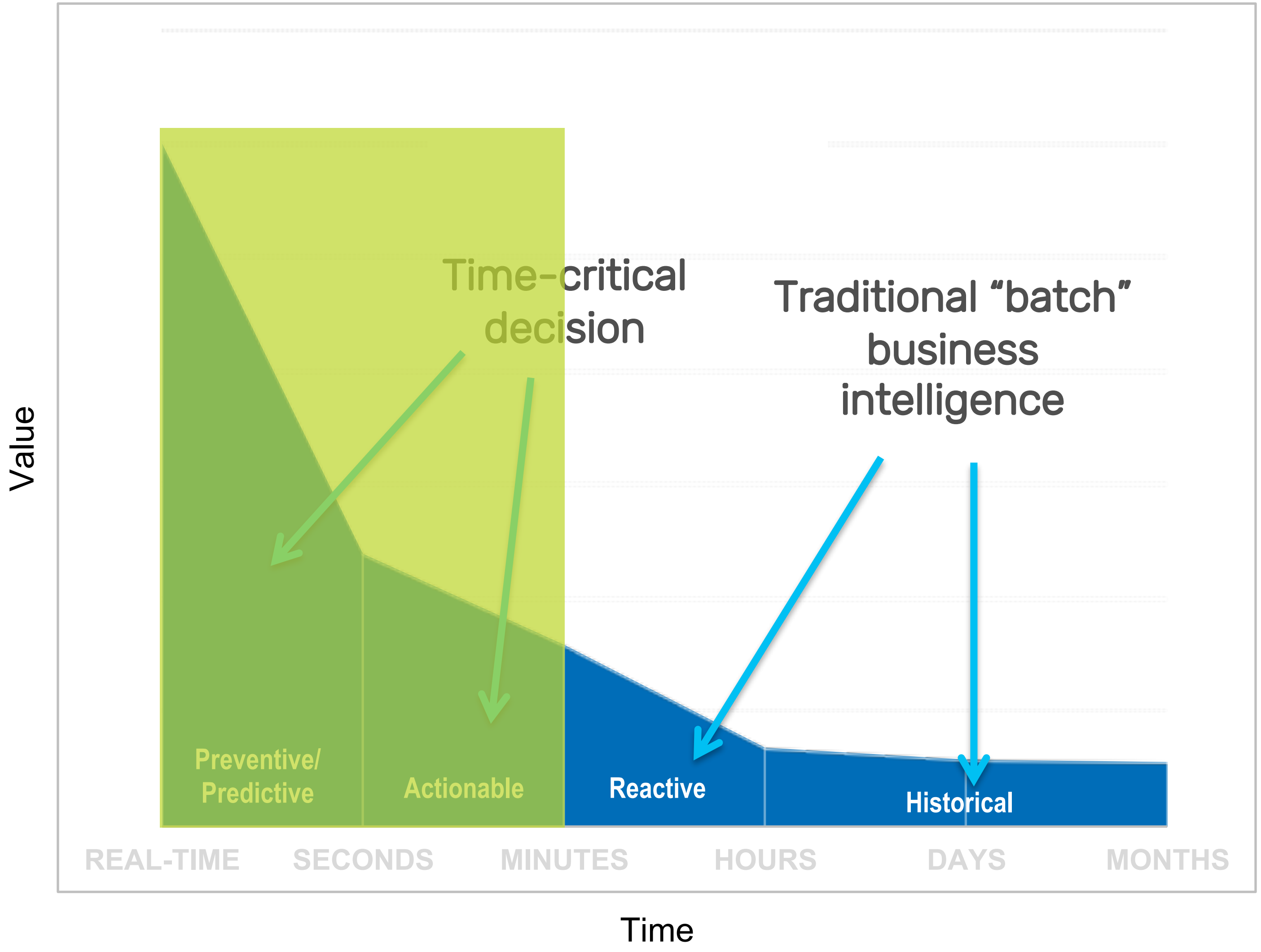
- **Manage Large Deployments**
  - Cloud-ready, ZooKeeper based for large-scale and federated deployments
- **REST API Management**
  - Standards-based, utilizing
- **Containerization and Orchestration**
  - Docker, Kubernetes, OpenShift etc.
- **Application-driven Deployment**
  - Serverless-like user experience
- **Pluggable Elastic Resource Balancing**
  - Scheduling for dynamic re-partitioning and resource allocation
- **Telemetry and Cluster Intelligence**
  - Predictive maintenance / fault-tolerance over large-scale deployments

Who’s using K8s?

OVERVIEW

- An overview of Kubernetes and the value it is bringing for automating deployment, scaling, and management of containerized applications
- How organizations can simplify management and container deployment on Cloud, Hybrid or On-premises environments with GigaSpaces InsightEdge
- 3 top open-source tools for production: HELM, Istio, and Prometheus
- A Kubernetes services comparison between cloud providers: AWS vs. Azure vs. GCP

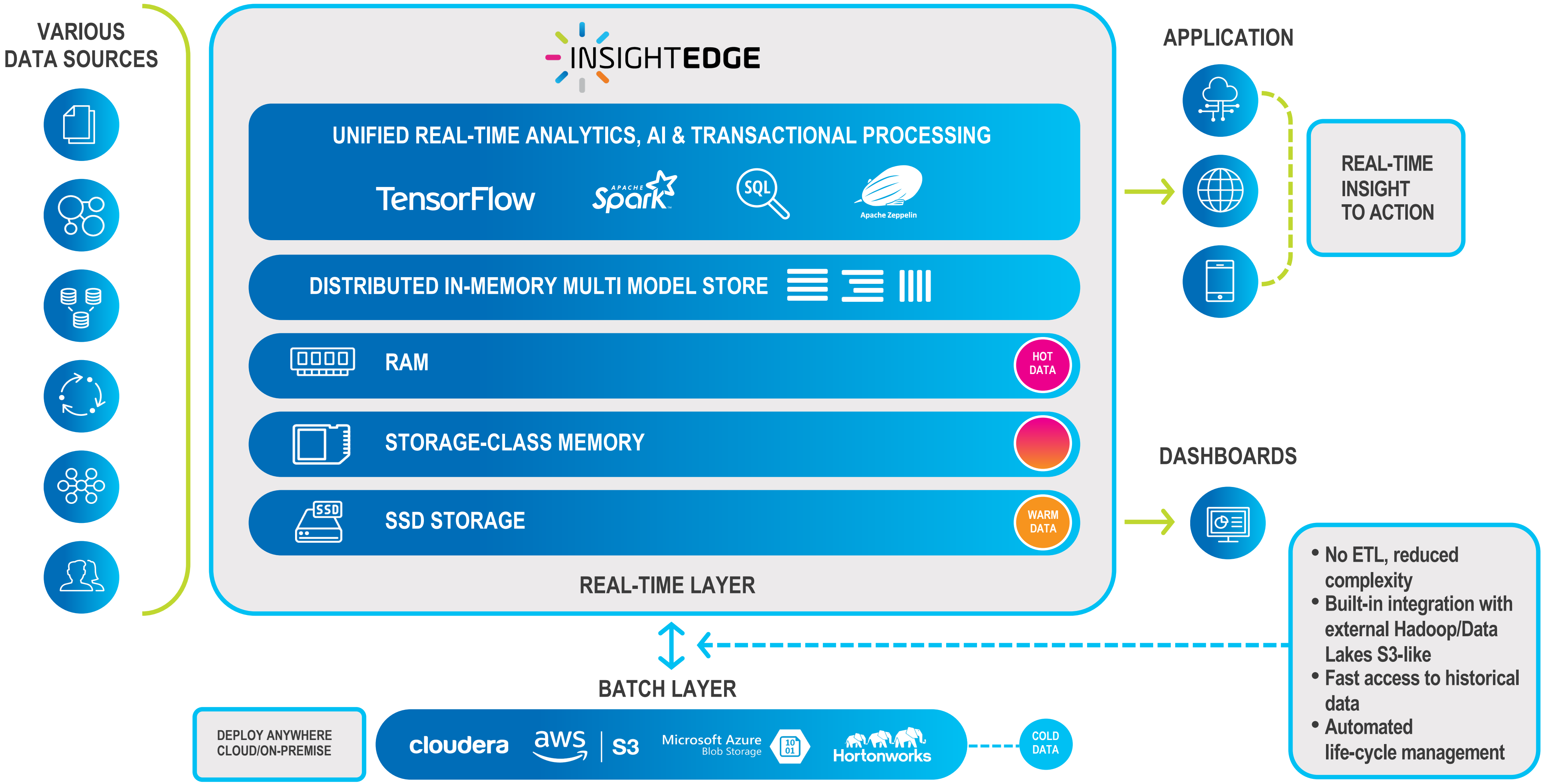
How Can You Gain the Most Value from Your Data?



Near real-time data is highly valuable if you act on it on time

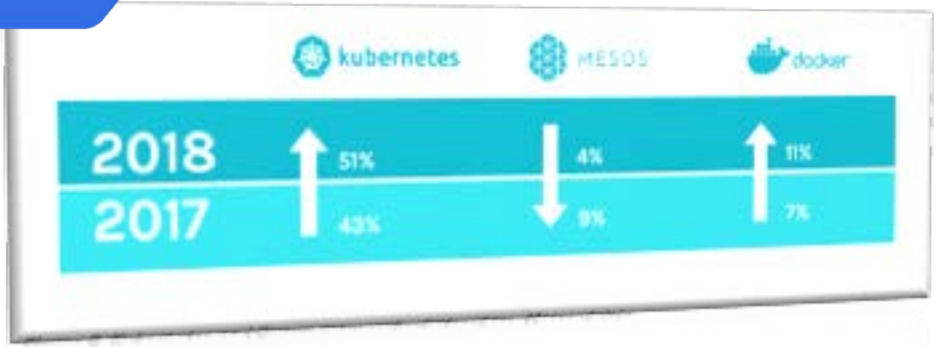
Historical + near real-time data is **more** valuable if you have the means to combine them

InsightEdge: Real-time Analytics for Instant Insights To Action



- No ETL, reduced complexity
- Built-in integration with external Hadoop/Data Lakes S3-like
- Fast access to historical data
- Automated life-cycle management

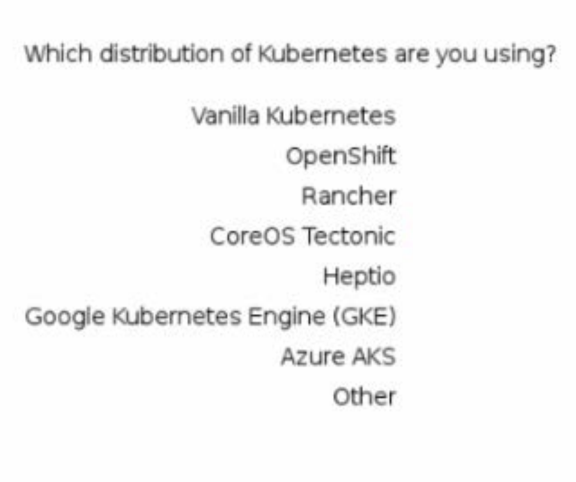
Kubernetes

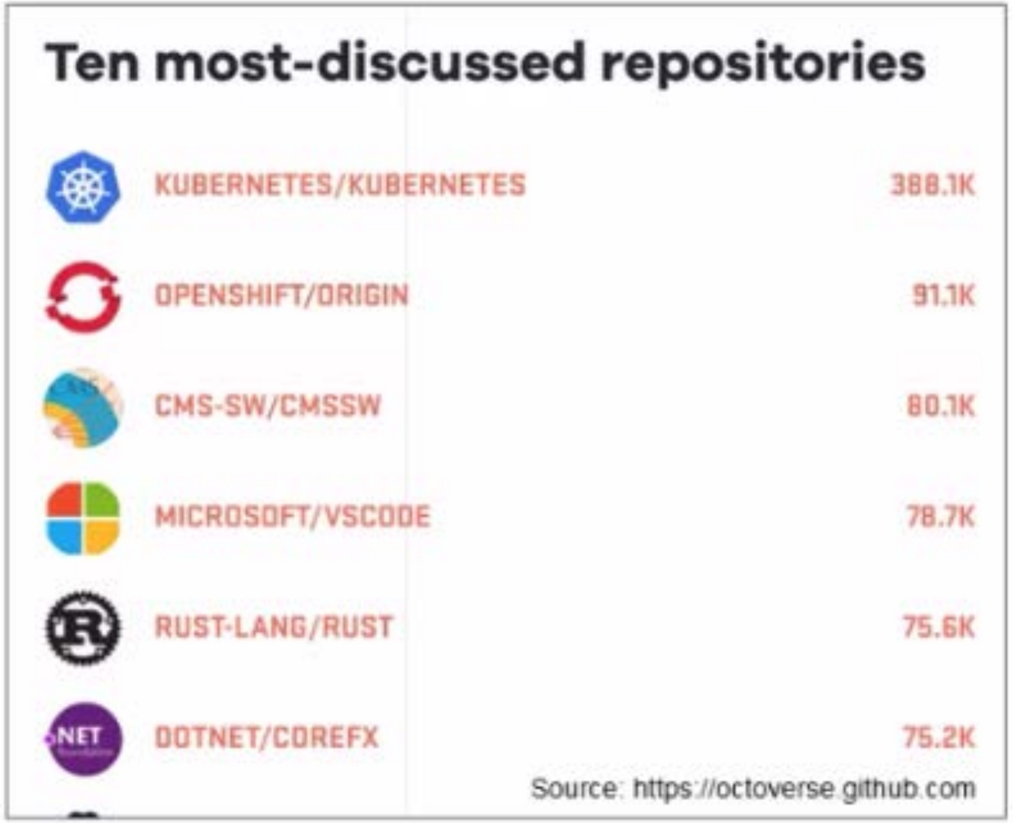


- At least 54% of the Fortune 500 were hiring for Kubernetes skills in 2017
- Around 51% growth for Kubernetes share in the market in 2018

Kubernetes is the Winner

- #1 discussed project on GitHub
- Top 2 in number of contributors
- ~400K users on Slack





Ten most-discussed repositories

KUBERNETES/KUBERNETES	388.1K
OPENSHIFT/ORIGIN	91.1K
CMS-SW/CMSSW	80.1K
MICROSOFT/VSCODE	78.7K
RUST-LANG/RUST	75.6K
DOTNET/COREFX	75.2K

Source: https://octoverse.github.com

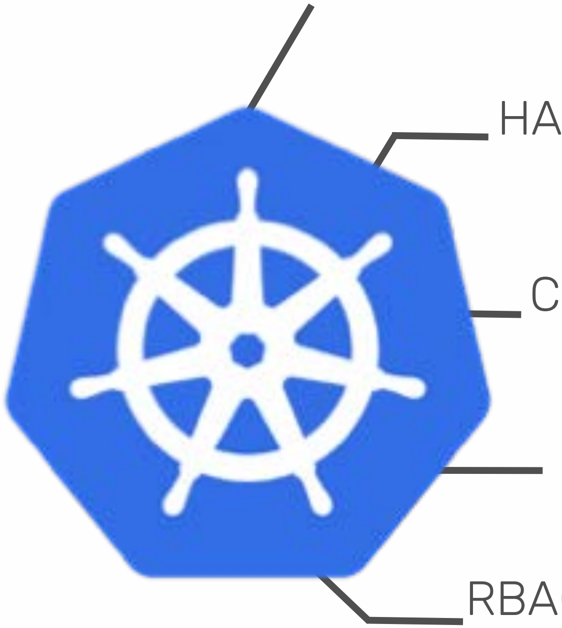
Business Landscape

- The leading orchestration tool vs. Docker Swarm, Mesos, OpenShift and Cloud Foundry and most used CNCF project
- All cloud vendors have a managed Kubernetes service (EKS, AKS and GKE)
- Apache Spark 2.3 has native Kubernetes support

Why Kubernetes?

Key building blocks for a “cloud like” platform as a service

- Auto deployment of data services, functions and frameworks (Spark ML, SQL, Zeppelin, etc.)
- Orchestration automation with cloud native solutions (auto scale, self healing)



Desired State Scheduler

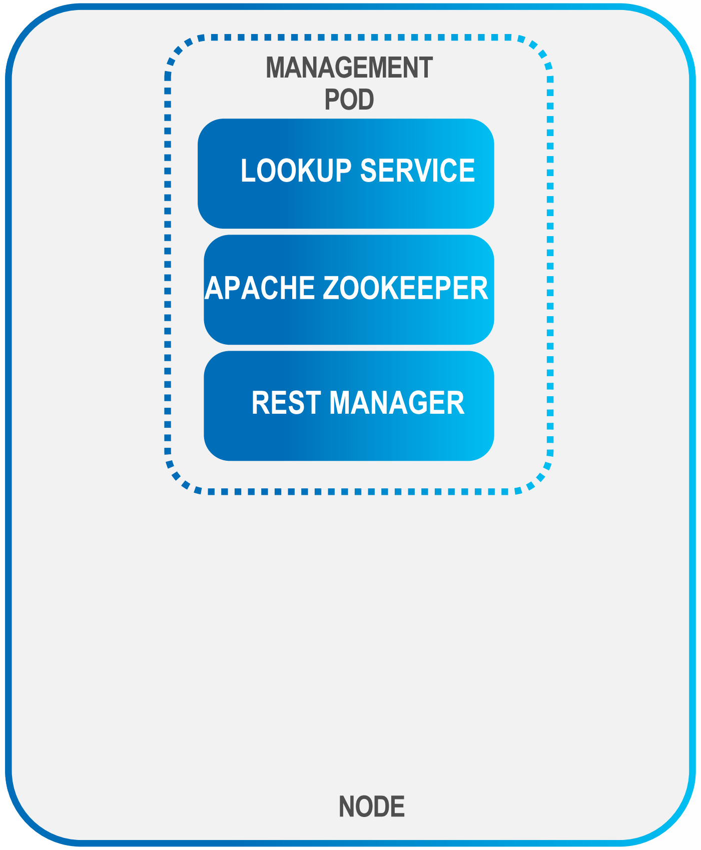
HA Architecture

Cooperative Multi-Tenancy

Service Account Authentication

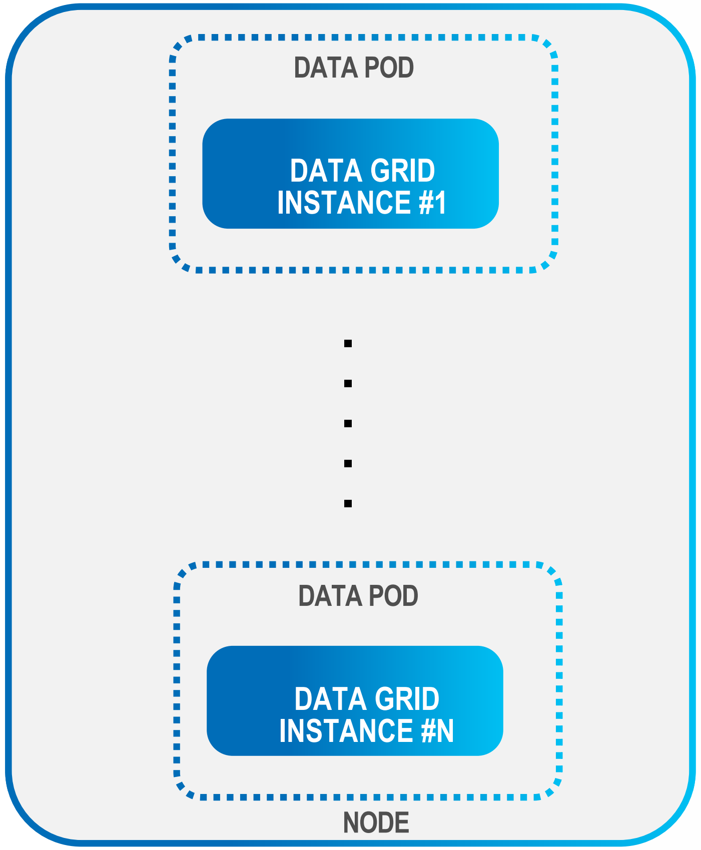
RBAC Authorization

Kubernetes – Management POD



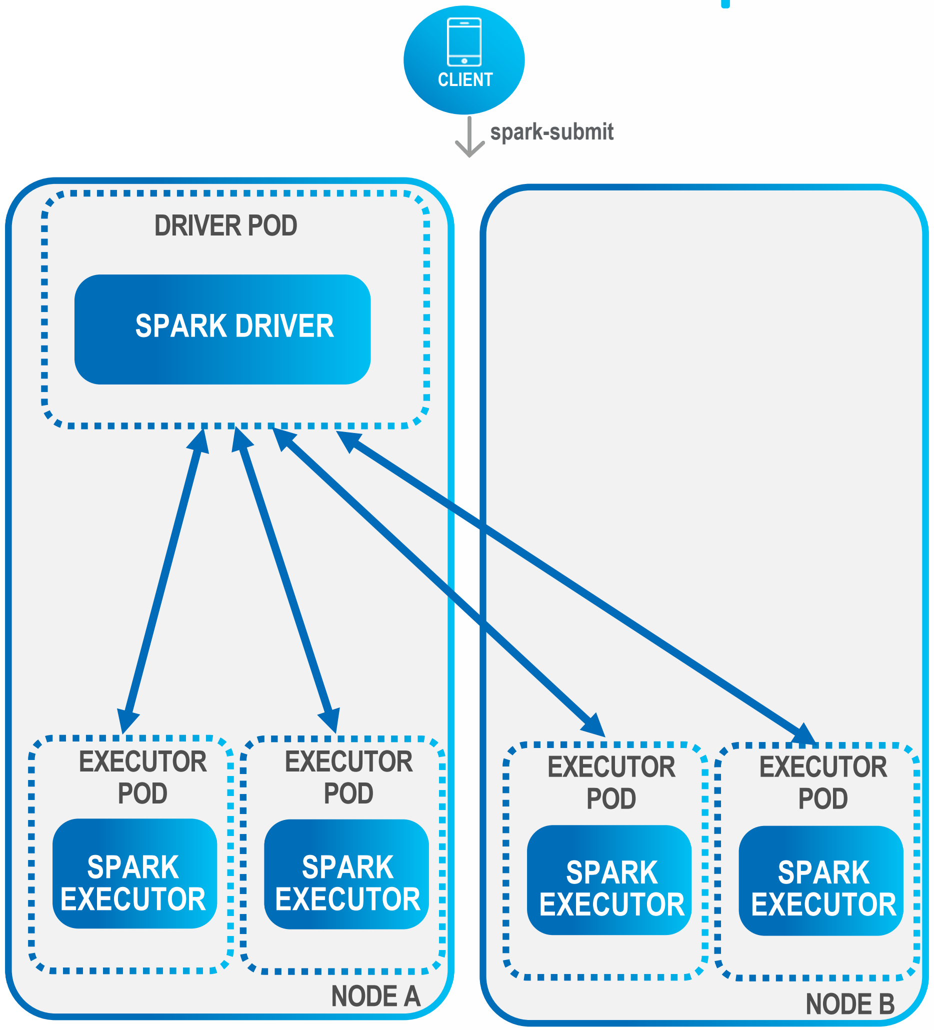
- Lookup Service (LUS) - The Lookup Service provides a mechanism for services to discover each other. For example, querying the LUS to find active GSCs.
- Apache ZooKeeper - Zookeeper is a centralized service used for space leader election
- REST Manager - RESTful API for managing the environment remotely from any platform

Kubernetes – Data POD



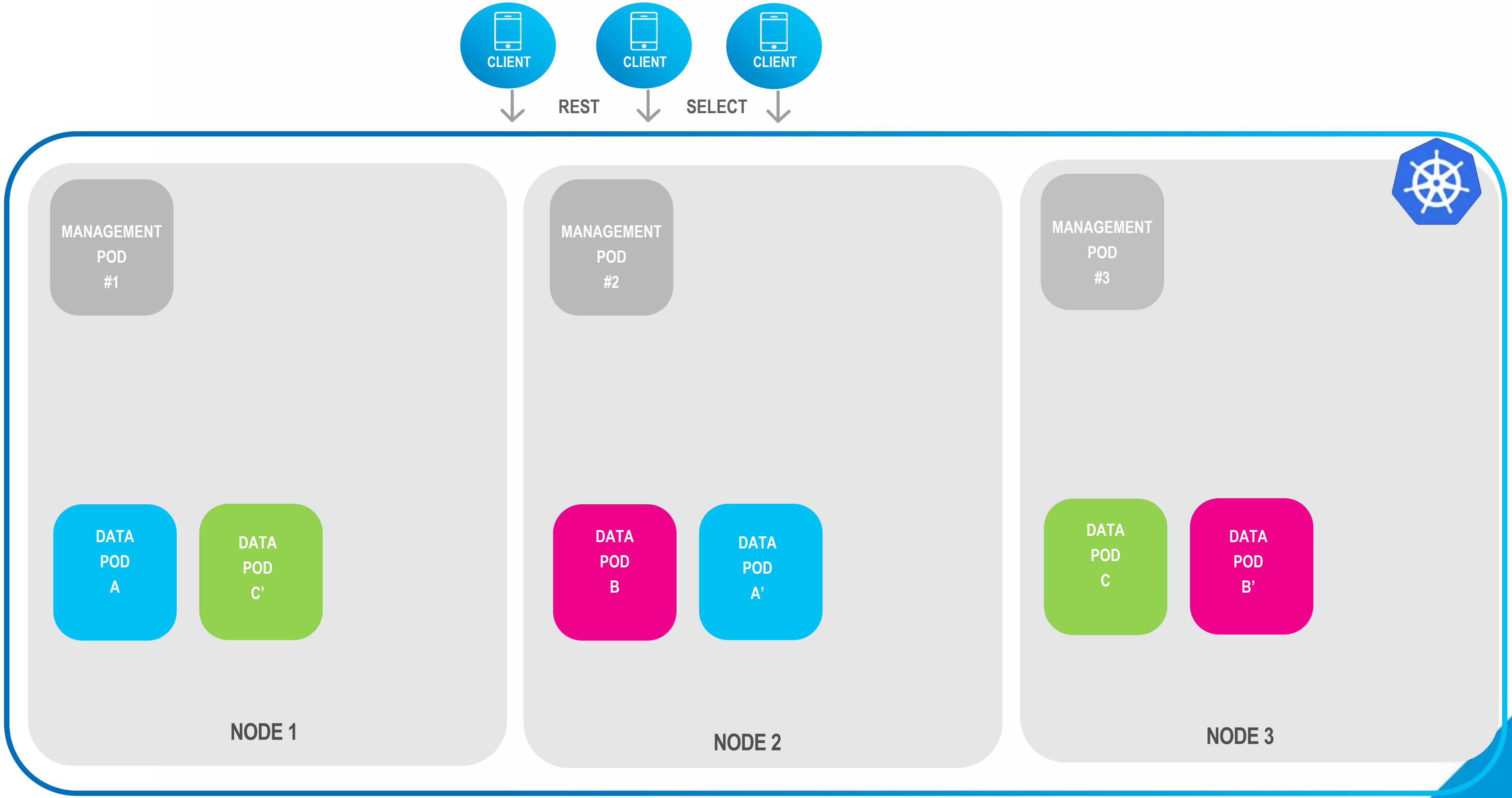
- Data Grid Instance - This is the fundamental unit of deployment in the data grid. A Processing Unit instance is the actual runtime entity.
- Each Data POD contains a single instance to provide cloud native support using Kubernetes built-in controllers (auto scale, self healing)

Kubernetes – Spark POD

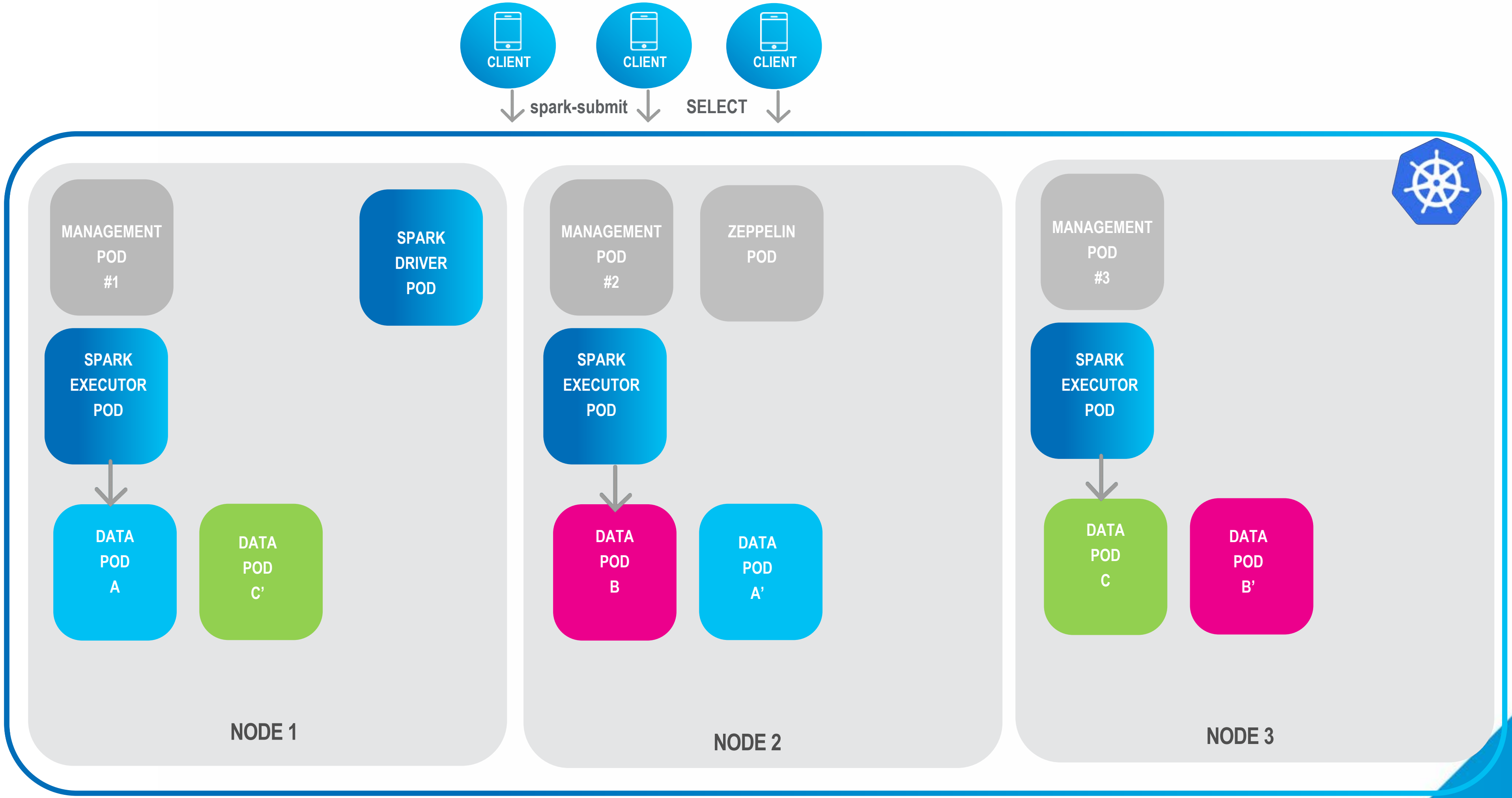


- Driver Pod – The Spark driver is running within a POD. The driver creates executors, connects to them, and executes the applicative code.
- Executor Pod – When the application completes, the executors' pods terminate and are cleaned up, but the master pod persists logs and remains in "completed" state

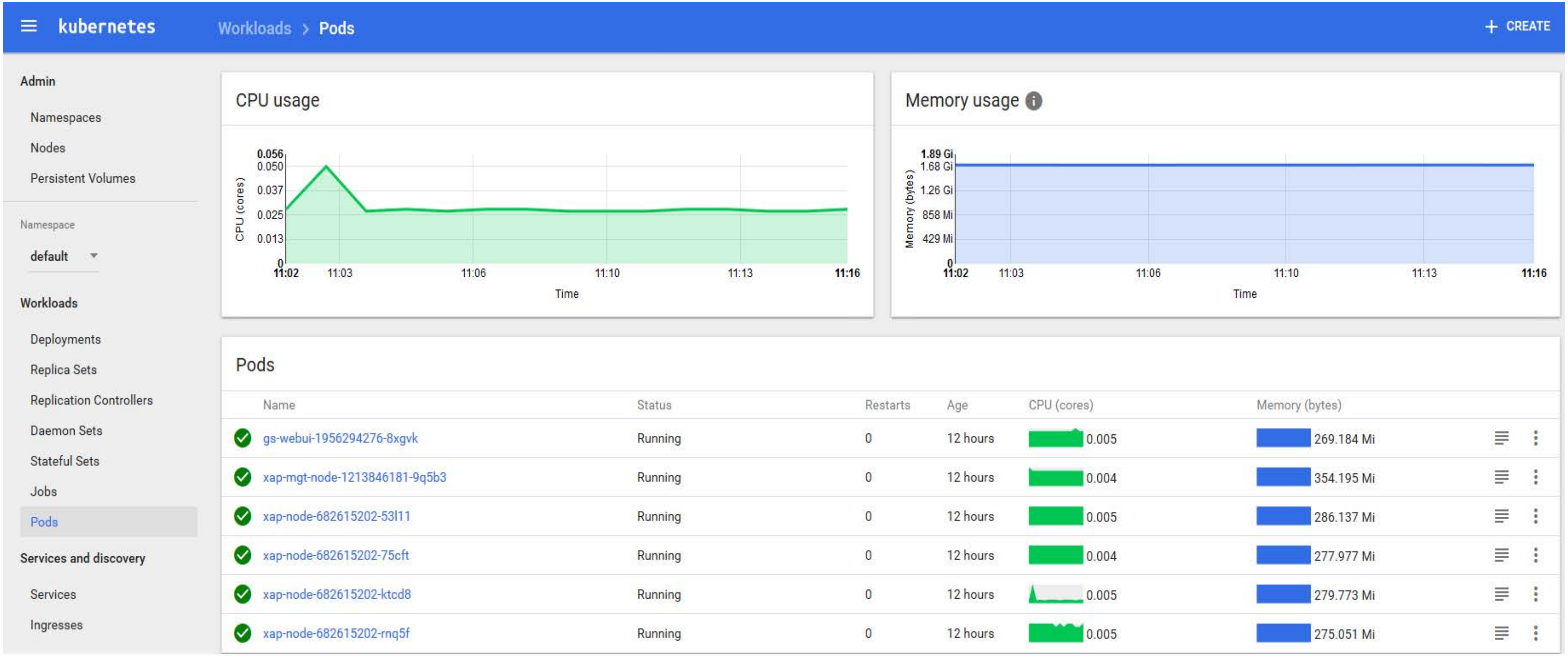
XAP High Level Overview 3,1



InsightEdge High Level Overview 3,1



Kubernetes Dashboard View



Pods

	Name	Status	Restarts	Age	CPU (cores)	Memory (bytes)
✓	gs-webui-1956294276-8xgvk	Running	0	12 hours	0.005	269.184 Mi
✓	xap-mgt-node-1213846181-9q5b3	Running	0	12 hours	0.004	354.195 Mi
✓	xap-node-682615202-53l11	Running	0	12 hours	0.005	286.137 Mi
✓	xap-node-682615202-75cft	Running	0	12 hours	0.004	277.977 Mi
✓	xap-node-682615202-ktcd8	Running	0	12 hours	0.005	279.773 Mi
✓	xap-node-682615202-rnq5f	Running	0	12 hours	0.005	275.051 Mi

“Under the Hood” Guidelines

- Apply a POD Anti-Affinity using label selectors for both Data and Management PODs
  - For example: spread the primary and backup data pods from this service across zones
- Each POD has a persistent identifier that is maintained across any rescheduling using StatefulSets
  - For example: automated rolling updates/scale up data pod one-by-one

Installation

- HELM – The package manager for Kubernetes
- Helm Charts helps you define, install and upgrade both XAP and InsightEdge

```
# helm install gigaspaces/insightedge --version=14.0 --name demo
```

Installation – Define Capacity

- The following Helm deploys a cluster with 3 partitions with 512MiB allocated for each partition:

```
# helm install gigaspaces/insightedge --version=14.0 --name demo
--set pu.partitions=3 ,pu.resources.limits.memory=512Mi
```

Installation – Define High Availability

- The following Helm command deploys a cluster in a high availability topology, with anti-affinity enabled:

```
# helm install gigaspaces/insightedge --version=14.0 --name demo
--set pu.ha=true,pu.antiAffinity.enabled=true
```

Testing for Liveness

- Use liveness probes to notify Kubernetes that your application’s processes are unhealthy and it should restart them
- The probe calls a bash script

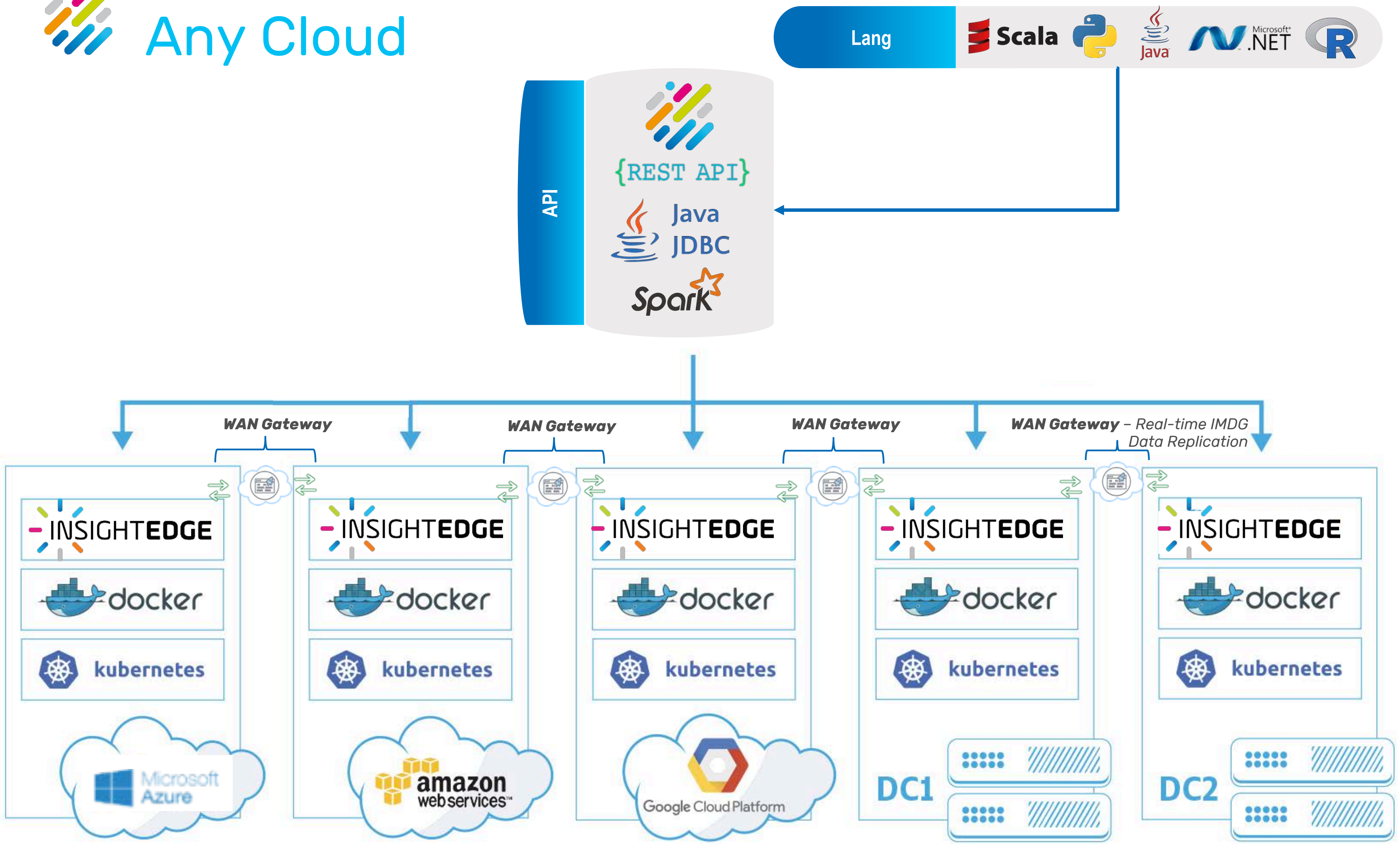
```
livenessProbe:
           exec:
                 command:
                 - sh
                 - -c - “data-pod-liveness 3181"
           initialDelaySeconds: 15
           timeoutSeconds: 5
```

Testing for Readiness

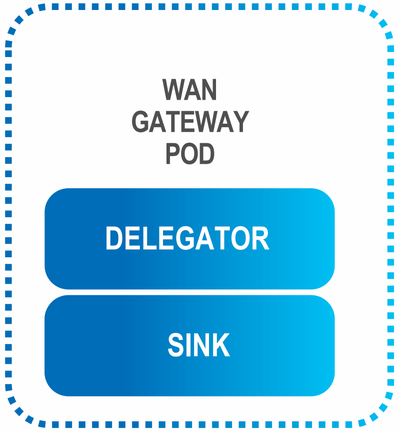
- Use readiness probes to notify Kubernetes that your application’s processes are able to process input, for example: when data is loading the pod not yet ready.
- The probe calls a bash script

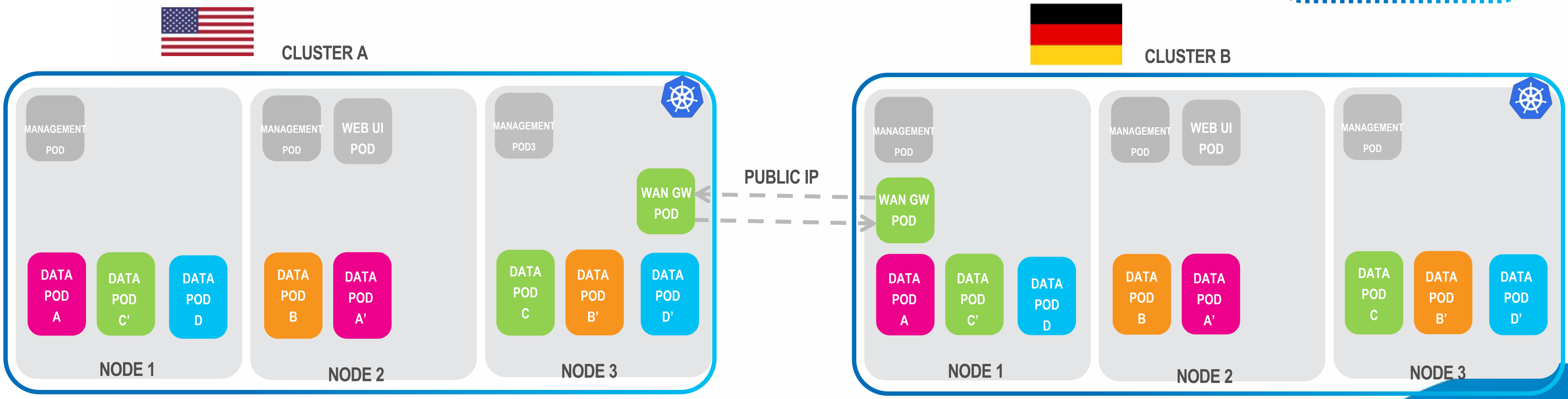
```
readienssProbe:
           exec:
                 command:
                 - sh
                 - -c - “data-pod-ready 2251"
           initialDelaySeconds: 15
           timeoutSeconds: 5
```

Any Cloud

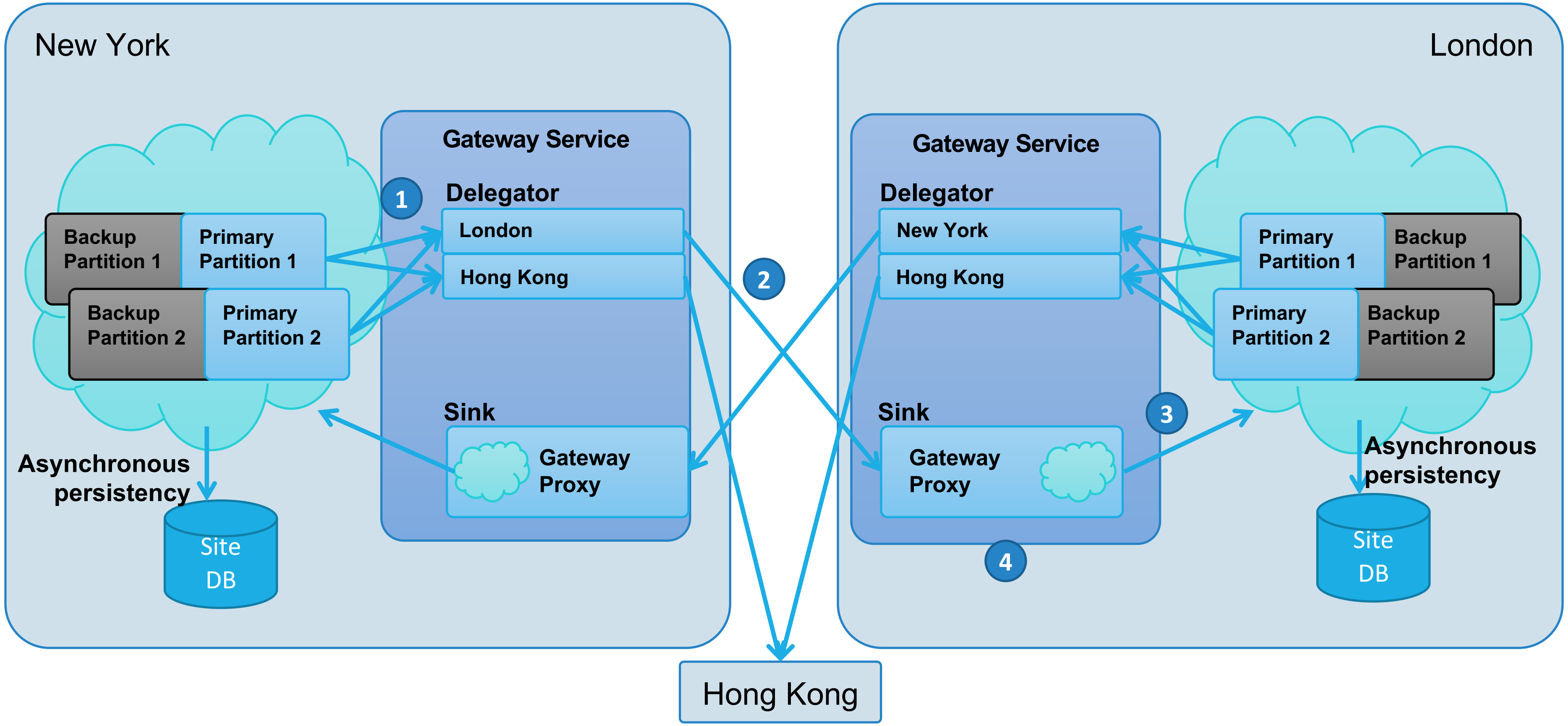

Lang
Scala
Java
Microsoft .NET
R
API
{REST API}
Java
JDBC
Spark
WAN Gateway
WAN Gateway
WAN Gateway
WAN Gateway – Real-time IMDG Data Replication
INSIGHTEDGE
docker
kubernetes
Microsoft Azure
amazon web services
Google Cloud Platform
DC1
DC2

WAN Gateway


WAN GATEWAY POD
DELEGATOR
SINK

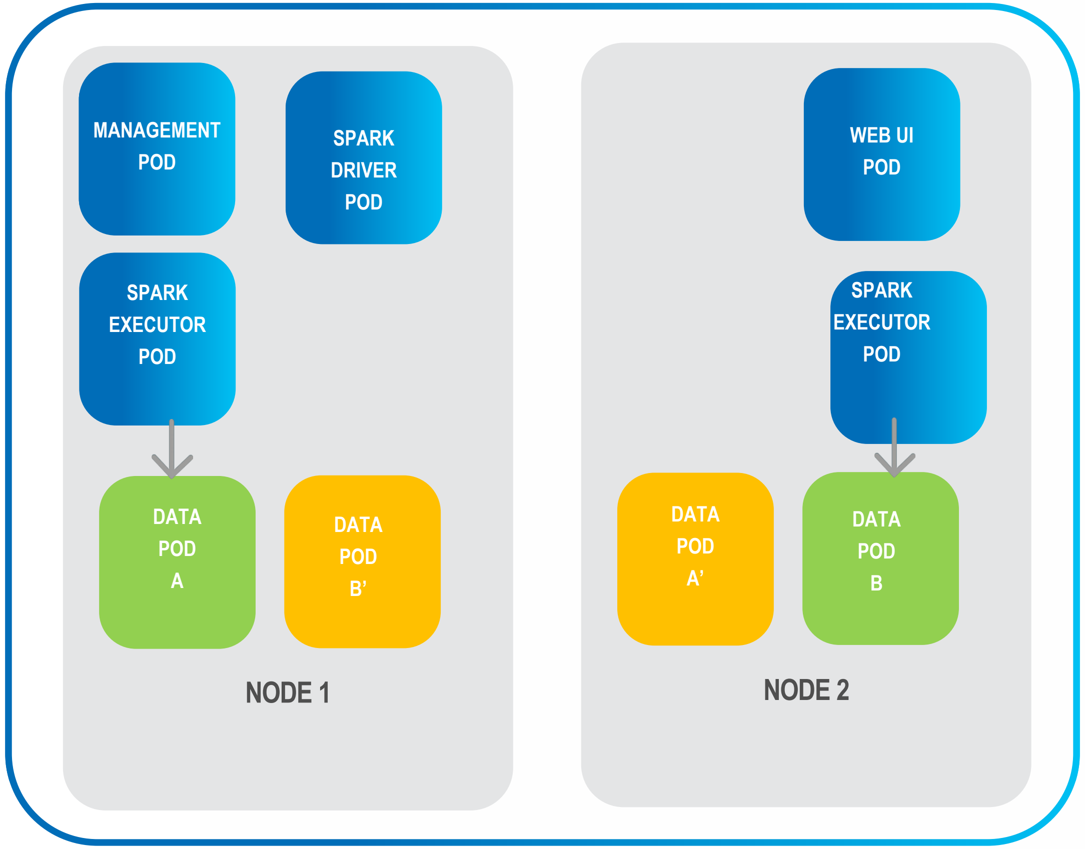

CLUSTER A
MANAGEMENT POD
MANAGEMENT POD
WEB UI POD
MANAGEMENT POD3
WAN GW POD
DATA POD A
DATA POD C'
DATA POD D
DATA POD B
DATA POD A'
DATA POD C
DATA POD B'
DATA POD D'
NODE 1
NODE 2
NODE 3
PUBLIC IP
CLUSTER B
MANAGEMENT POD
WAN GW POD
MANAGEMENT POD
WEB UI POD
MANAGEMENT POD
DATA POD A
DATA POD C'
DATA POD D
DATA POD B
DATA POD A'
DATA POD C
DATA POD B'
DATA POD D'
NODE 1
NODE 2
NODE 3

Replication Flow

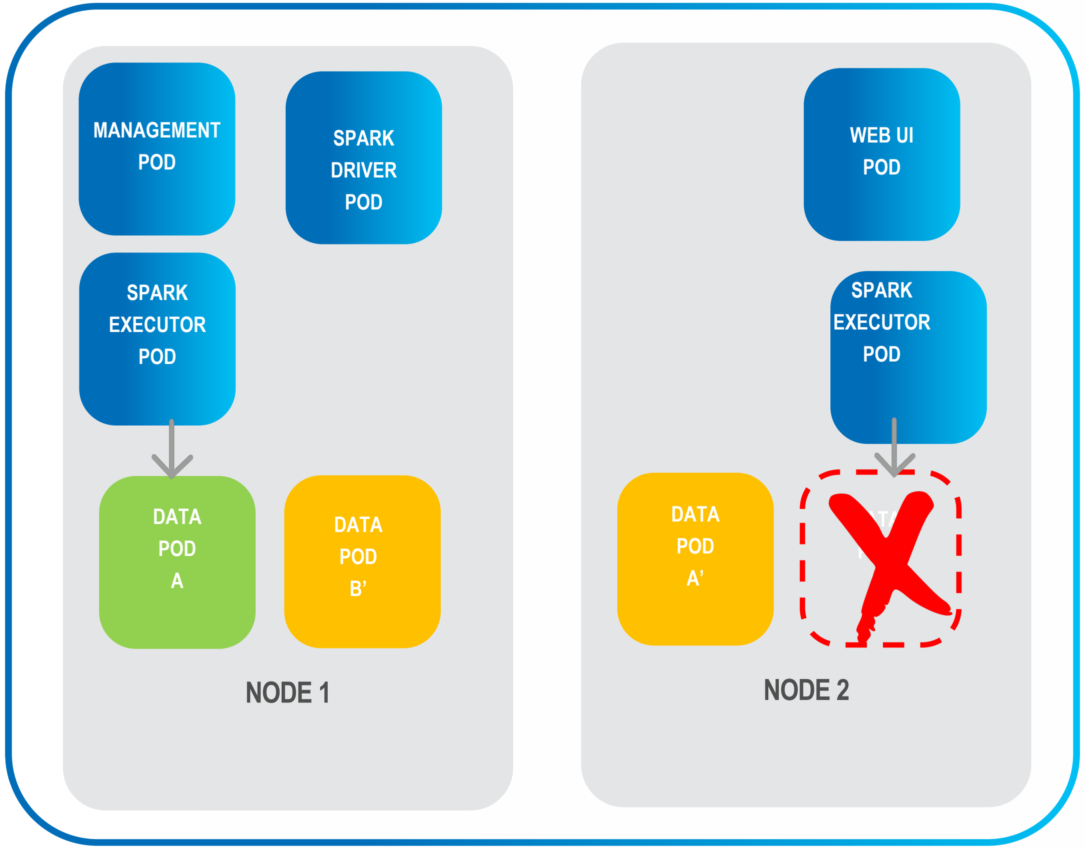


1. Updates in New York cluster are pushed to local Delegator
2. Delegator sends the updates to the list of target sites configured in New York Gateway
3. London Sink will write the data to London Cluster
4. Any conflicts that occur are resolved using the custom Conflict Resolution algorithm

Auto Pod Failover

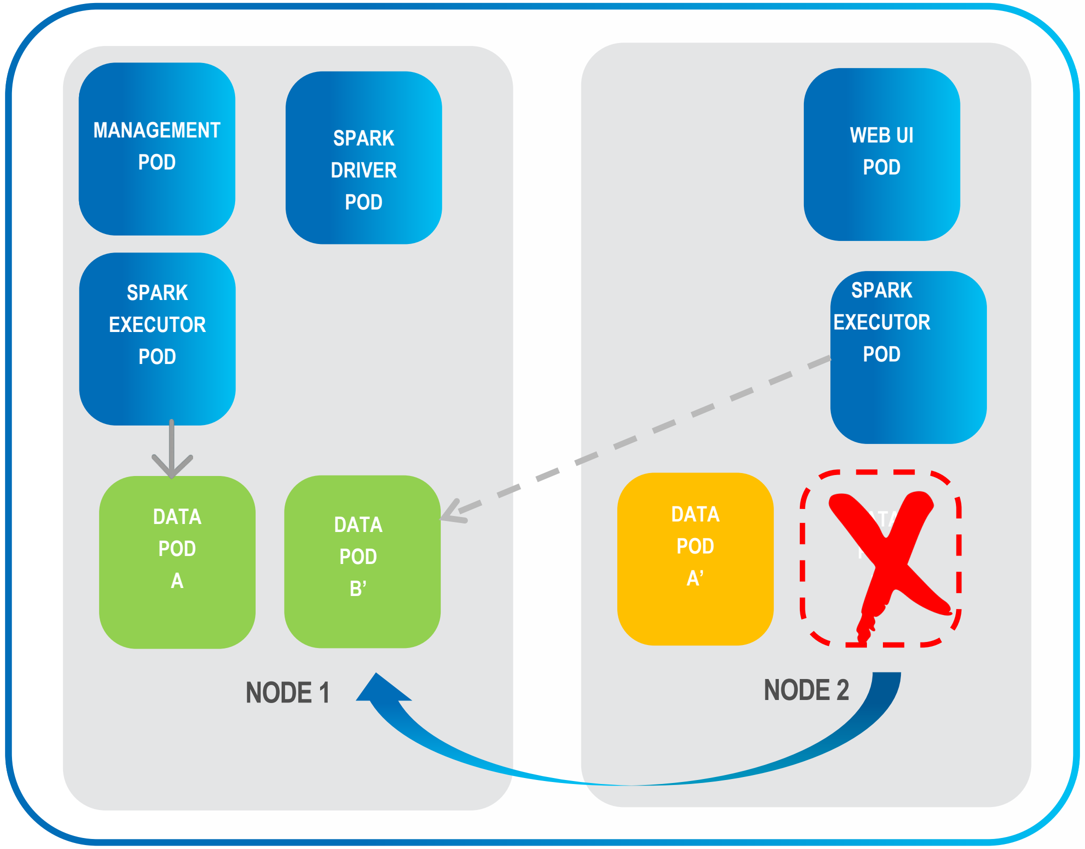


Auto Pod Failover



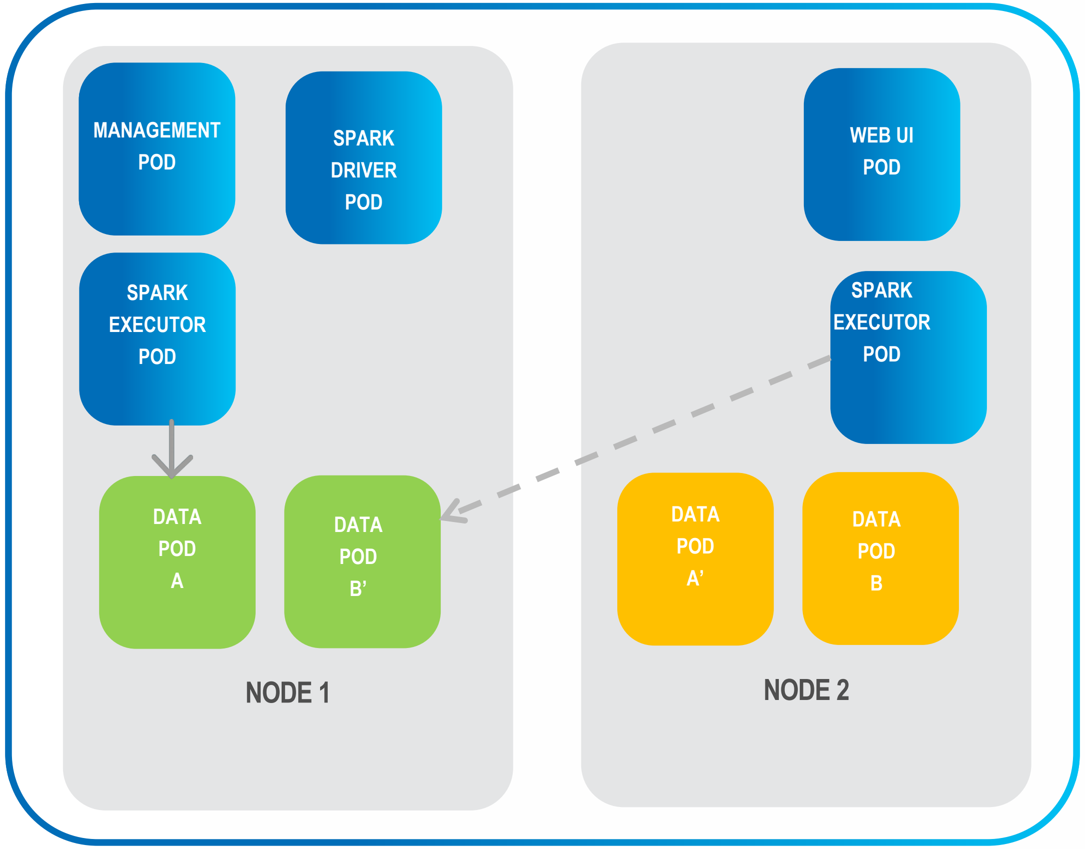
1 Data Pod B Fails

Auto Pod Failover



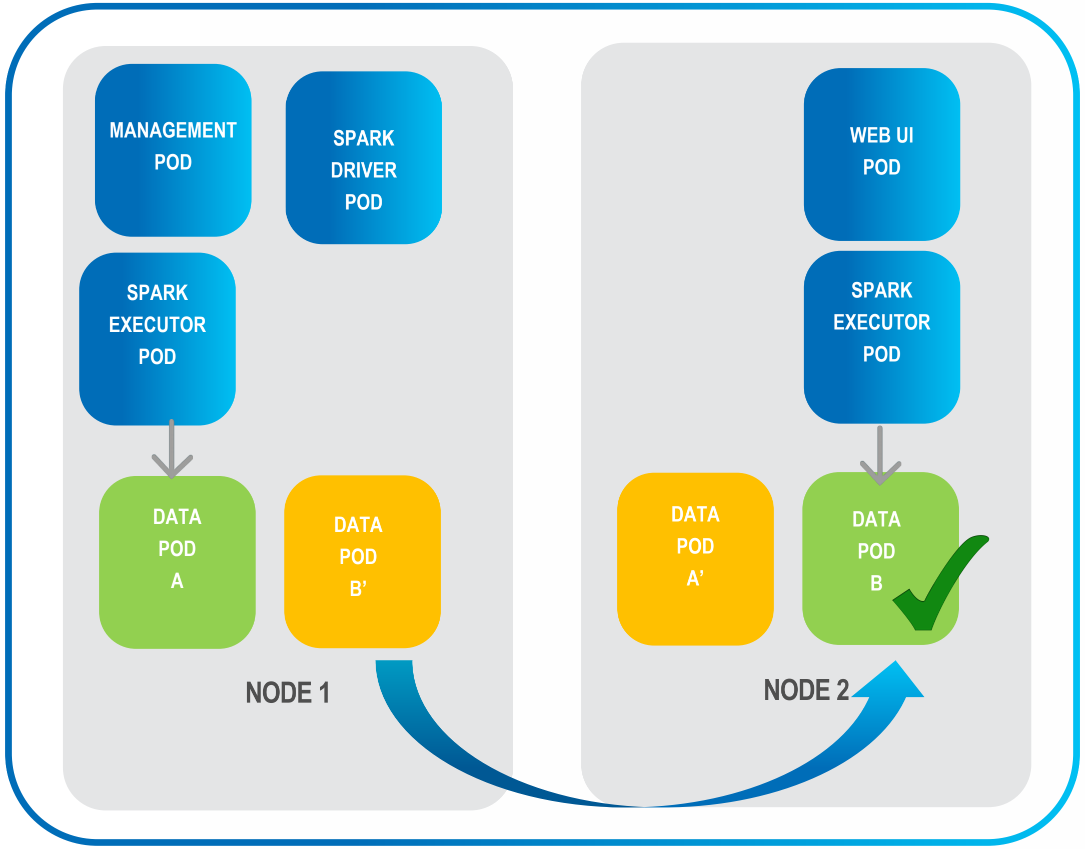
1. Data Pod B Fails
2. Failover to Data Pod B'

Auto Pod Failover



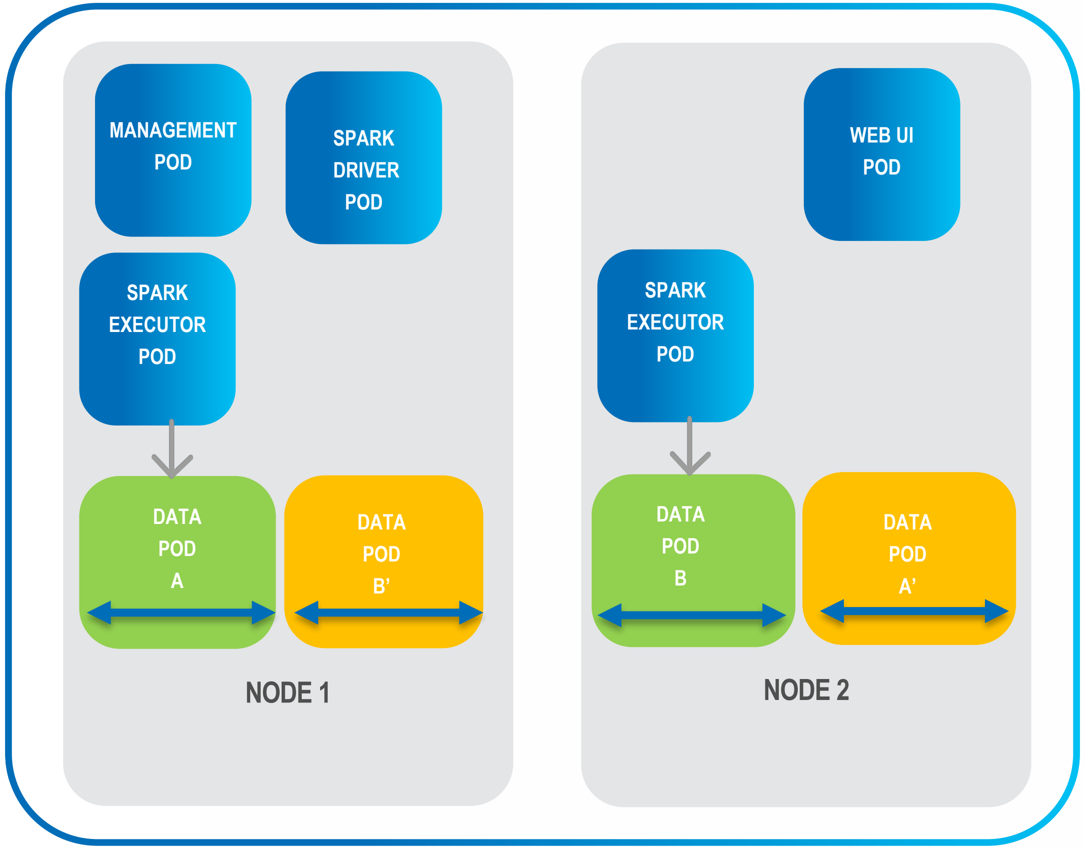
1. Data Pod B Fails
2. Failover to Data Pod B'
3. Data B is back up

Auto Pod Failback



1. Data Pod B Fails
2. Failover to Data Pod B'
3. Detect failure and restart Pod B
4. Once ready failback to Pod B as "proffered primary"

Automated Rolling Scale Up



1 Take Down Pod A'

2 Restart Pod A' with X2 RAM

3 Fail over to Pod A' and restart Pod A with X2 RAM

4 Fail back to Pod A

Repeat for each Pod

Kubernetes Comparison

Feature/ Service	**GCP**	**Azure**	**AWS**	**IBM**
Automatic Update	Auto or On-demand	On-demand	On-demand	On-Demand
Auto-scaling nodes	Yes	No, available thorough k8s autoscale	Yes	No
Node Pools	Yes	No	Yes	No
Multiple Zones	Yes	No	Yes	Yes
RBAC	Yes	Yes	Yes	Yes
Bare Metal Nodes	No	No	Yes	Yes

3 Key Technologies for Kubernetes

- Prometheus – Monitoring

Monitor applications and infrastructure running in Kubernetes, supports service discovery, built-in alerts, and more.

- Istio - Service Mesh

Istio manages and routes encrypted network traffic, balances loads across microservices, enforces access policies, verifies service identity and provides tracing, aggregates service to service telemetry.

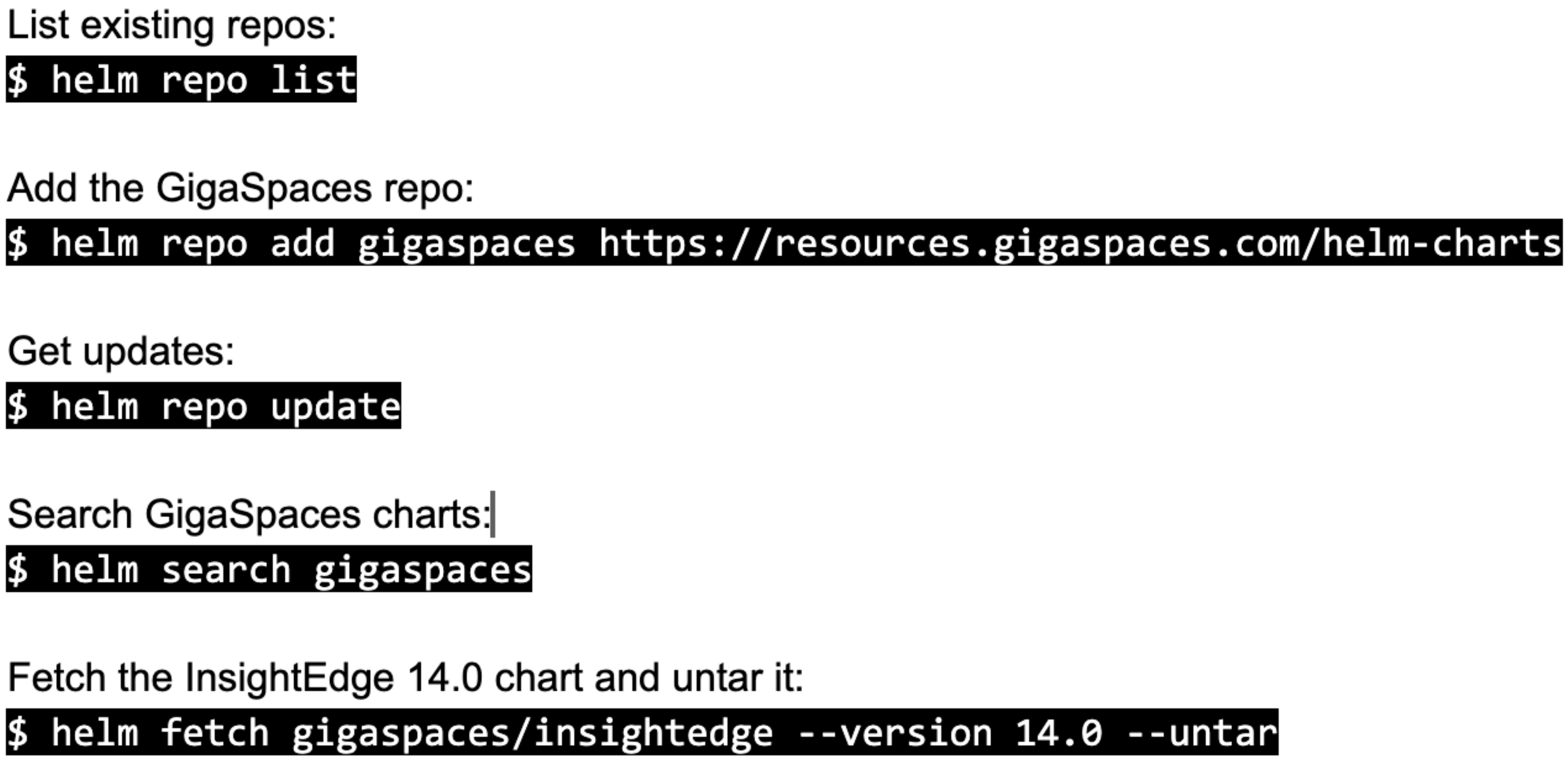
- Helm - Package Manager for Continuous Deployments

Repeatable deployments without all of the overhead and complication of keeping dependencies up to date and consistent

RECORDED DEMO

LINK: https://www.youtube.com/watch?v=i4Z4__I8N9Q

Fetch InsightEdge Helm Chart



List existing repos:

```
$ helm repo list
```

Add the GigaSpaces repo:

```
$ helm repo add gigaspaces https://resources.gigaspaces.com/helm-charts
```

Get updates:

```
$ helm repo update
```

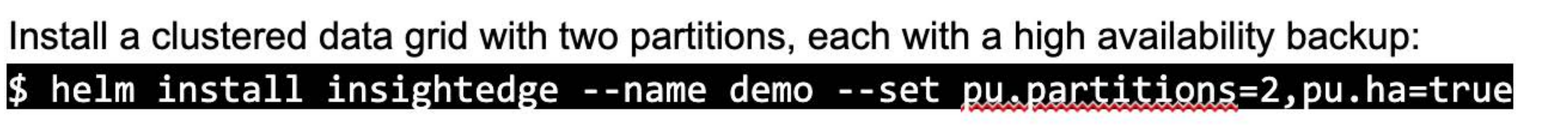
Search GigaSpaces charts:

```
$ helm search gigaspaces
```

Fetch the InsightEdge 14.0 chart and untar it:

```
$ helm fetch gigaspaces/insightedge --version 14.0 --untar
```

Installing a Data Grid



Install a clustered data grid with two partitions, each with a high availability backup:

```
$ helm install insightedge --name demo --set pu.partitions=2,pu.ha=true
```

Monitoring

Using Helm

```
$ helm status demo
```

Using Kubernetes Command Line

```
$ kubectl get pods
```

Using Kubernetes Dashboard

```
$ minikube dashboard
```

Using GigaSpaces REST Manager

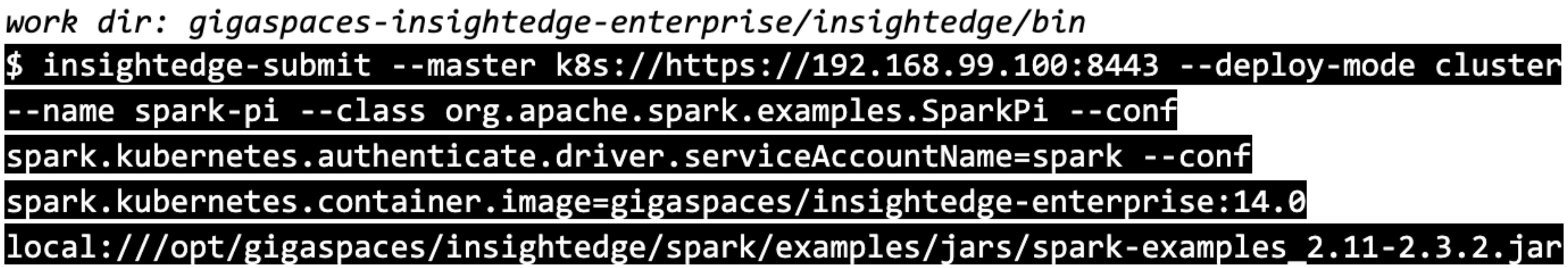
http://192.168.99.100:30890

Using GigaSpaces Command Line

```
$ insightedge --server=192.168.99.100:30890 space list
```

Running a Spark job

Run the following InsightEdge submit script for the SparkPi example. It calculates a Pi approximation. The result of the calculation is printed to the log.

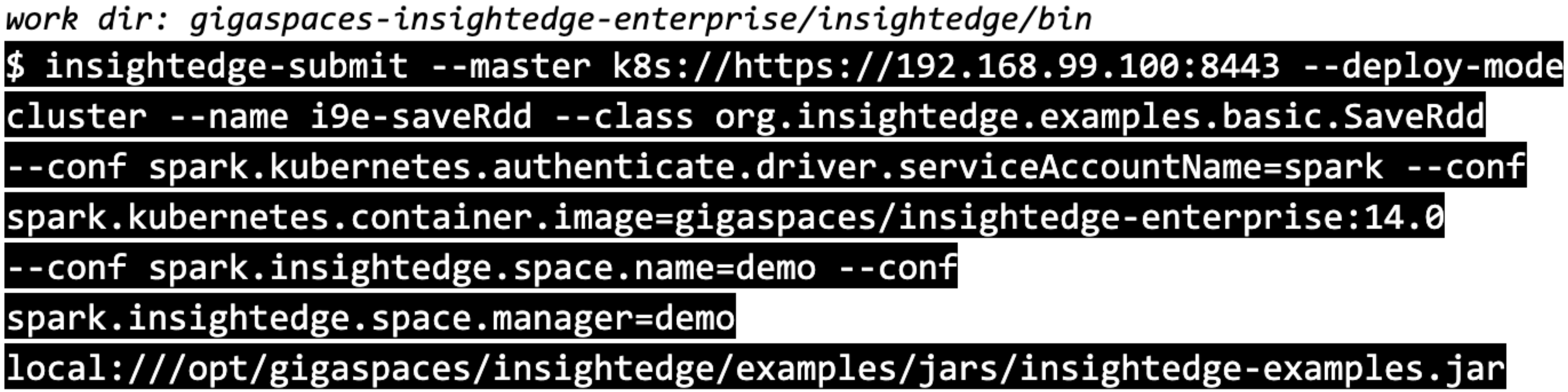


```
work dir: gigaspaces-insightedge-enterprise/insightedge/bin
$ insightedge-submit --master k8s://https://192.168.99.100:8443 --deploy-mode cluster
--name spark-pi --class org.apache.spark.examples.SparkPi --conf
spark.kubernetes.authenticate.driver.serviceAccountName=spark --conf
spark.kubernetes.container.image=gigaspaces/insightedge-enterprise:14.0
local:///opt/gigaspaces/insightedge/spark/examples/jars/spark-examples_2.11-2.3.2.jar
```

(Go to the driver pod and see the Pi value that was calculated, e.g. “`Pi is roughly 3.1391756458782296`”)

Running an InsightEdge Spark Job

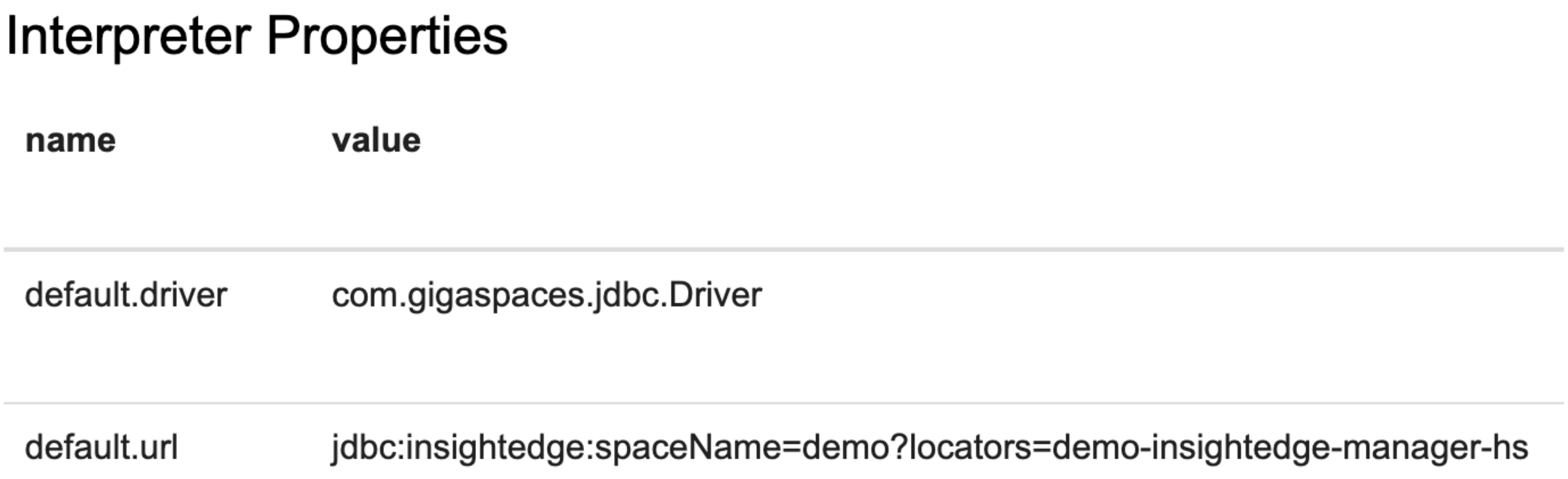
Run the following InsightEdge submit script for the SaveRDD example, which generates 100,000 Products, converts them to RDD, and saves them to the data grid.



```
work dir: gigaspaces-insightedge-enterprise/insightedge/bin
$ insightedge-submit --master k8s://https://192.168.99.100:8443 --deploy-mode
cluster --name i9e-saveRdd --class org.insightedge.examples.basic.SaveRdd
--conf spark.kubernetes.authenticate.driver.serviceAccountName=spark --conf
spark.kubernetes.container.image=gigaspaces/insightedge-enterprise:14.0
--conf spark.insightedge.space.name=demo --conf
spark.insightedge.space.manager=demo
local:///opt/gigaspaces/insightedge/examples/jars/insightedge-examples.jar
```

Apache Zeppelin

Zeppelin URL: http://192.168.99.100:30990



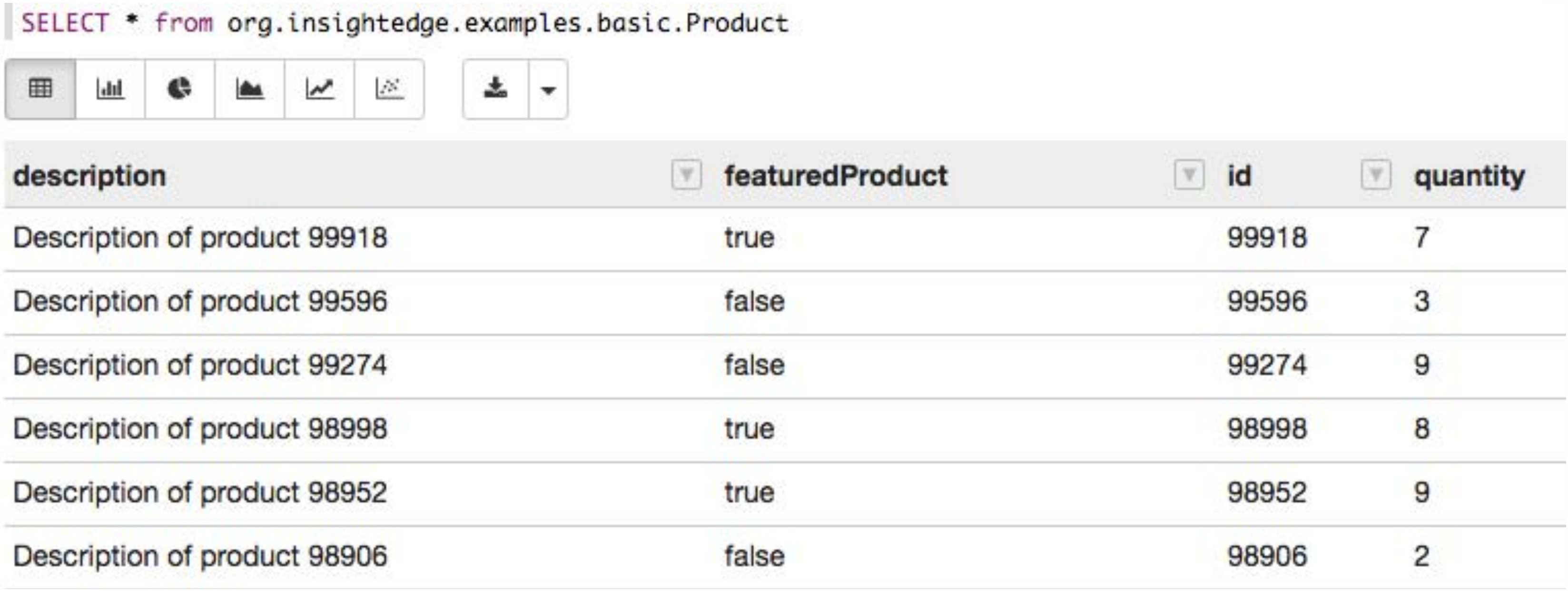
Interpreter Properties

name	value
default.driver	com.gigaspaces.jdbc.Driver
default.url	jdbc:insightedge:spaceName=demo?locators=demo-insightedge-manager-hs

SQL Queries

The following SQL Queries can be executed to analyze the data in the data grid.

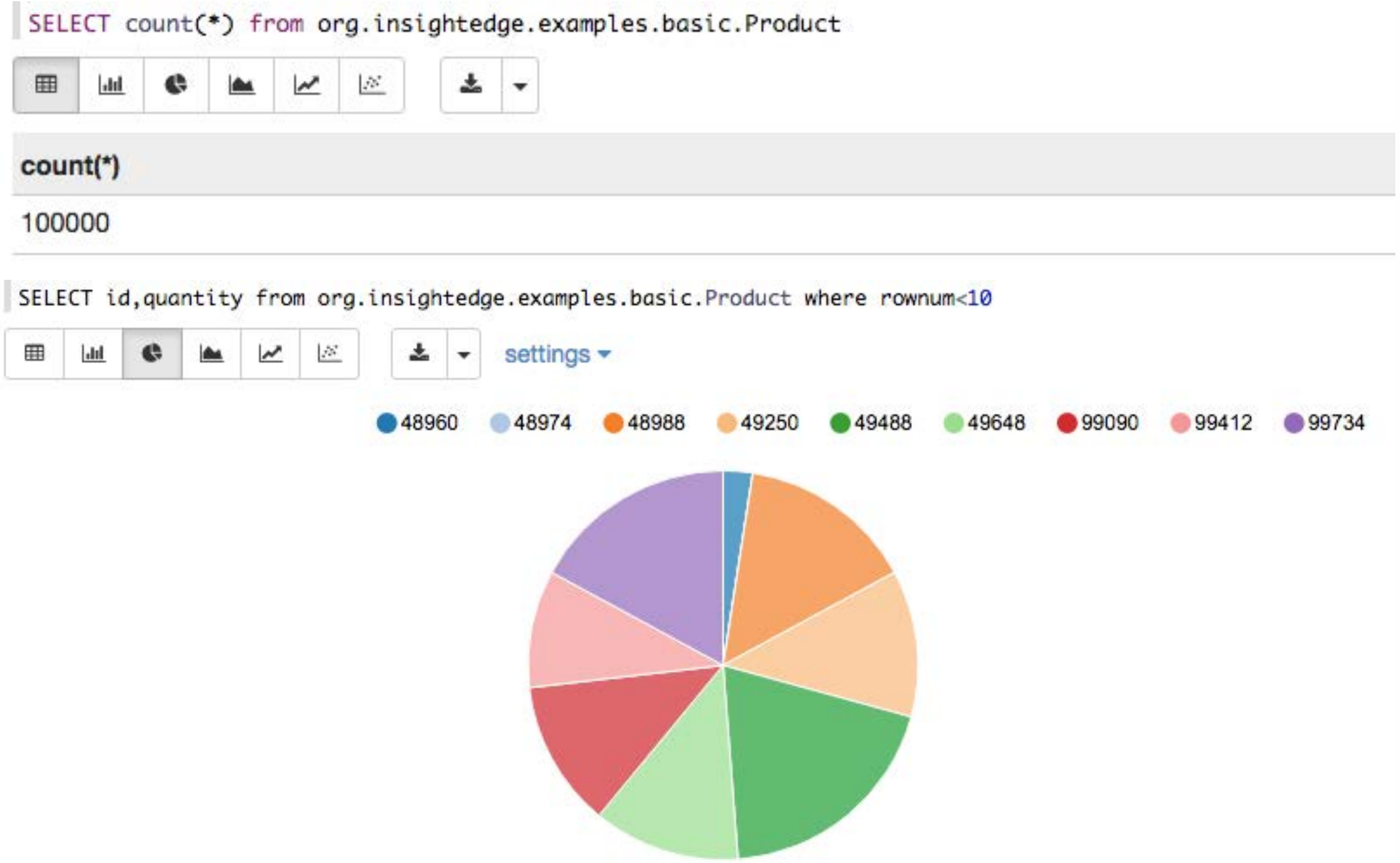
```
%insightedge_jdbc
SELECT * from Product
SELECT count(*) from Product
SELECT id,quantity from Product where id<15
```



```
SELECT * from org.insightedge.examples.basic.Product
```

description	featuredProduct	id	quantity
Description of product 99918	true	99918	7
Description of product 99596	false	99596	3
Description of product 99274	false	99274	9
Description of product 98998	true	98998	8
Description of product 98952	true	98952	9
Description of product 98906	false	98906	2

SQL Queries



```
SELECT count(*) from org.insightedge.examples.basic.Product
```

count(*)
100000

```
SELECT id,quantity from org.insightedge.examples.basic.Product where rownum<10
```

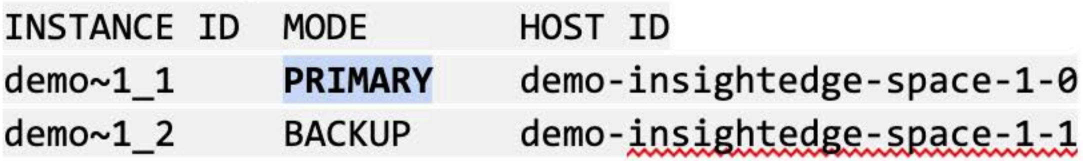
Failover

1. Using command line, list Space instances and check which Space is elected primary

work dir: gigaspaces-insightedge-enterprise/bin/

```
$ insightedge --server=192.168.99.100:30890 space list-instances demo
```

Example output:



```
INSTANCE ID   MODE        HOST ID
demo~1_1      PRIMARY     demo-insightedge-space-1-0
demo~1_2      BACKUP      demo-insightedge-space-1-1
```

2. Using Kubernetes Dashboard, ‘Exec’ into the Pod of the primary Space instance

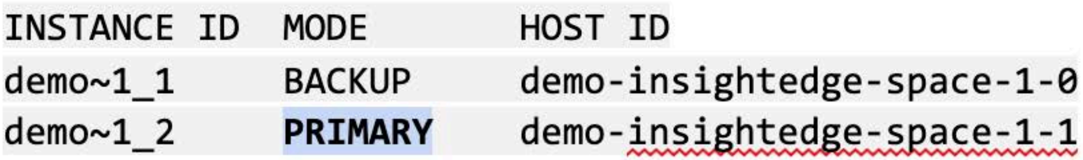
Failover

3. Execute the following command

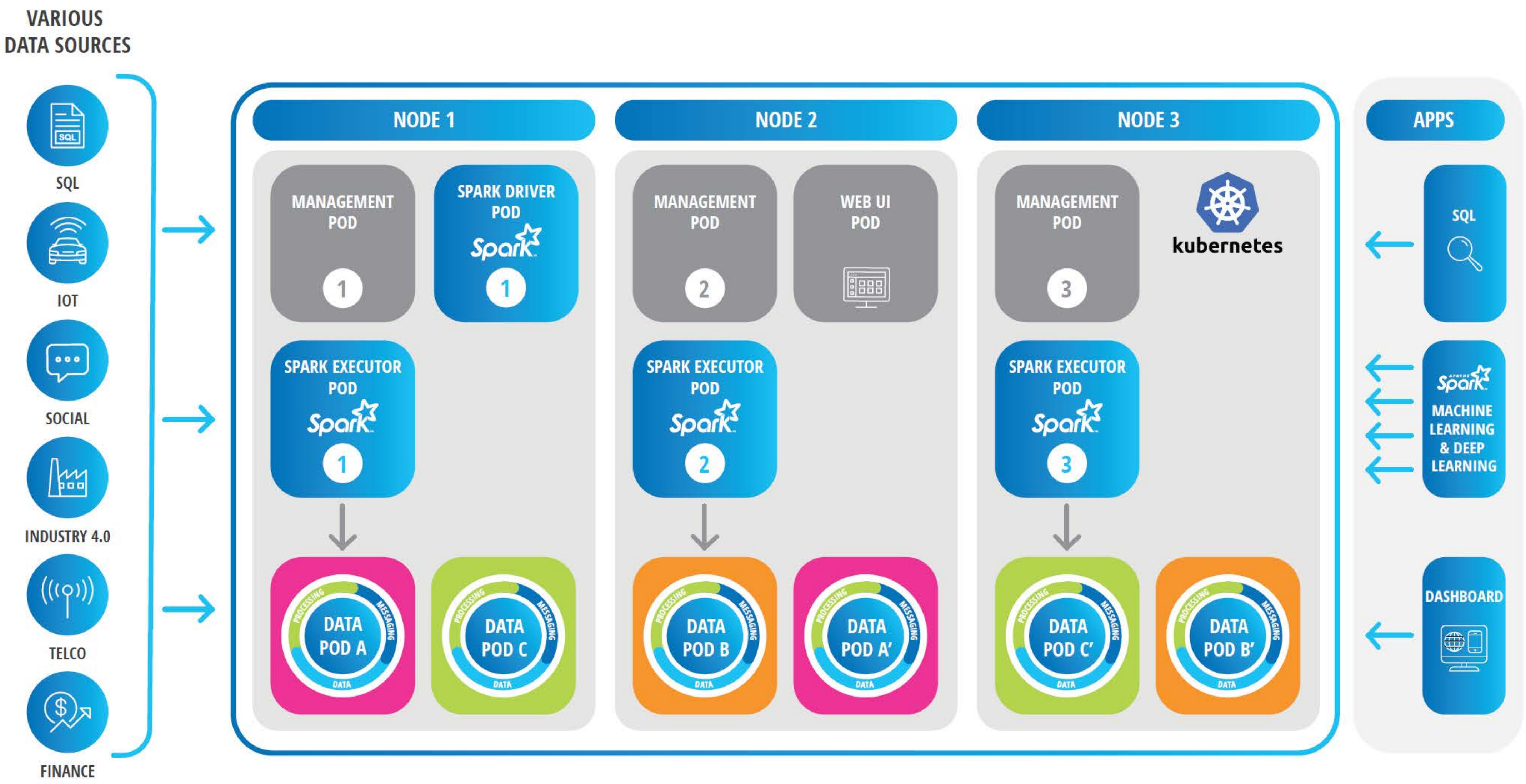
```
$ pkill -9 java
```

4. Execute list Space instances command line again and verify new elected primary

Example output:



```
INSTANCE ID   MODE         HOST ID
demo~1_1      BACKUP       demo-insightedge-space-1-0
demo~1_2      PRIMARY      demo-insightedge-space-1-1
```



VARIOUS DATA SOURCES

SQL

IOT

SOCIAL

INDUSTRY 4.0

TELCO

FINANCE

NODE 1

MANAGEMENT POD 1

SPARK DRIVER POD Spark 1

SPARK EXECUTOR POD Spark 1

DATA POD A

DATA POD C

NODE 2

MANAGEMENT POD 2

WEB UI POD

SPARK EXECUTOR POD Spark 2

DATA POD B

DATA POD A’

NODE 3

MANAGEMENT POD 3

kubernetes

SPARK EXECUTOR POD Spark 3

DATA POD C’

DATA POD B’

PROCESSING MESSAGING DATA

APPS

SQL

APACHE Spark

MACHINE LEARNING & DEEP LEARNING

DASHBOARD

To make a long story short, we've built space**ships**

THANK YOU

@taldoron

in taldoron84

✉ tald@gigaspaces.com

Tal Doron

Director, Technology Innovation