Rome | March 22 - 23, 2019

{codemotion}

Microservices with Istio, JHipster and Kubernetes

Deepu K Sasidharan

@deepu105 | deepu.js.org

About me

Deepu K Sasidharan

JHipster co-lead developer

Principal developer @ XebiaLabs

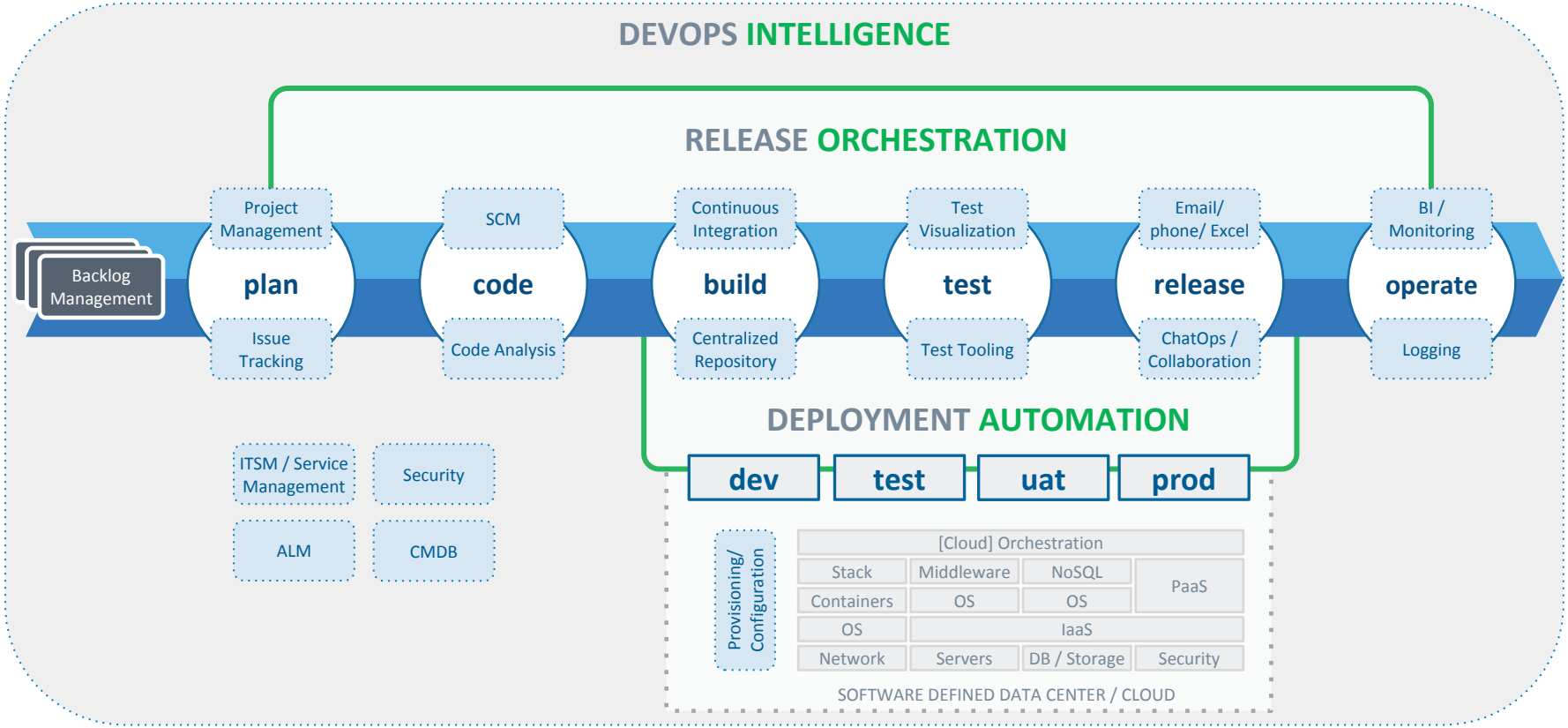
OSS aficionado, author, speaker

@deepu105

deepu.js.org

XebiaLabs

XebiaLabs


DEVOPS INTELLIGENCE
RELEASE ORCHESTRATION
Backlog Management
Project Management
Issue Tracking
plan
SCM
Code Analysis
code
Continuous Integration
Centralized Repository
build
Test Visualization
Test Tooling
test
Email/ phone/ Excel
ChatOps / Collaboration
release
BI / Monitoring
Logging
operate
DEPLOYMENT AUTOMATION
dev
test
uat
prod
ITSM / Service Management
Security
ALM
CMDB
Provisioning/ Configuration
[Cloud] Orchestration
Stack
Middleware
NoSQL
PaaS
Containers
OS
OS
OS
IaaS
Network
Servers
DB / Storage
Security
SOFTWARE DEFINED DATA CENTER / CLOUD

What about you?

How many of you are Java developers?

How many are web developers?

How many of you are doing microservices?

Are you a fan of Kubernetes?

Have you tried Istio?

Have you tried Linkerd?

About JHipster

Most popular Rapid Application Development platform for Java web applications and microservices

- 13k+ stars on GitHub
- 1.8M+ installations & 20k+ app generations per month
- 250k+ overall users
- 490+ contributors & 28 core team members
- 260+ companies using JHipster
- 70+ plugins

What can you do with JHipster?

- World peace!
- Fix your plumbing
- Order Pizza for you

What can you do with JHipster?

- Generate simple monolith web applications
- Generate complete microservice architectures
- Generate domain model (entities)
- Generate CI/CD pipelines
- Deploy to AWS, GCP, Azure, Heroku, Cloud Foundry
- Deploy to Docker, Kubernetes, Openshift, Rancher

Scaffolding! Isn’t that some junk boilerplate code?

Scaffolding! Isn’t that some junk boilerplate code?

sonarcloud

Pricing | Log in

Transparency makes sense and that’s why the trend is growing.

Check out these open-source projects showing users their commitment to quality.

Open Source	Matt Raible	SonarSource
Jhipster Sample A...	**21-points**	**SonarLint for Visu...**
Bugs 0 A	Bugs 0 A	Bugs 1 D
Vulnerabilities 0 A	Vulnerabilities 0 A	Vulnerabilities 0 A
Code Smells 60 A	Code Smells 51 A	Code Smells 35 A
Coverage 80.2%	Coverage 74.2%	Coverage 86.1%
Duplications 1.2%	Duplications 2.1%	Duplications 0.0%
43k lines of code / CSS, Java, ...	*14k lines of code / CSS, Java, ...*	*18k lines of code / C#*

Come join the fun, it’s entirely free for open-source projects!

GitHub | Bitbucket | Azure DevOps

Microservices with Netflix OSS Stack

Netflix OSS

Eureka

Service registry for service discovery and mid-tier load balancing.

Ribbon

Load balancing, fault tolerance, caching & batching.

Hystrix

Circuit breaking and fault tolerance

Zuul

Edge service(Routing, Load balancing, monitoring, security and more)

JHipster Registry

Eureka server

Provides service discovery

Spring cloud config server

Provides runtime configurations

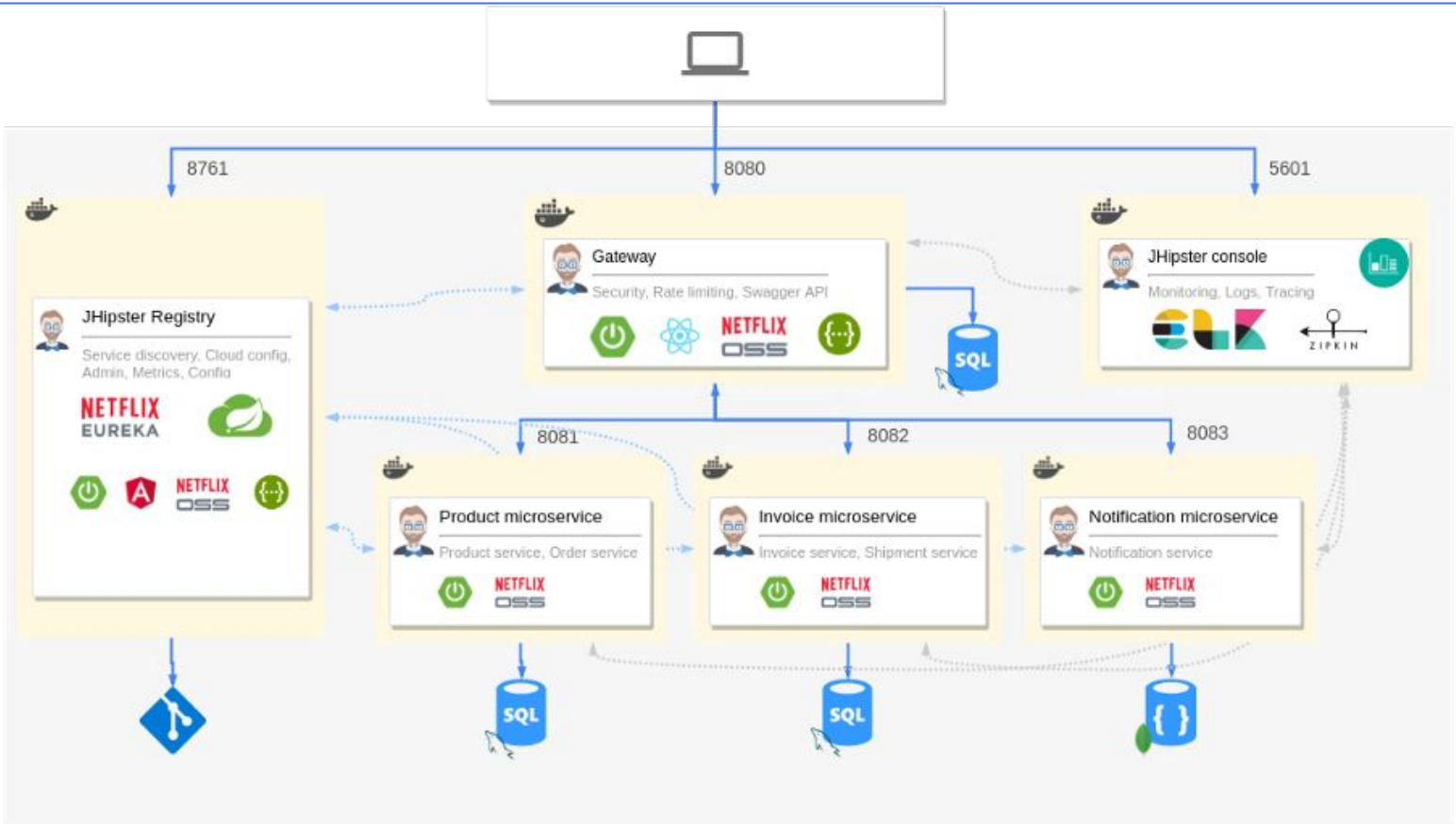
Administration server

Monitoring, logs, configurations and dashboards

Spring Boot

Provides the application runtime

Microservice with JHipster Registry

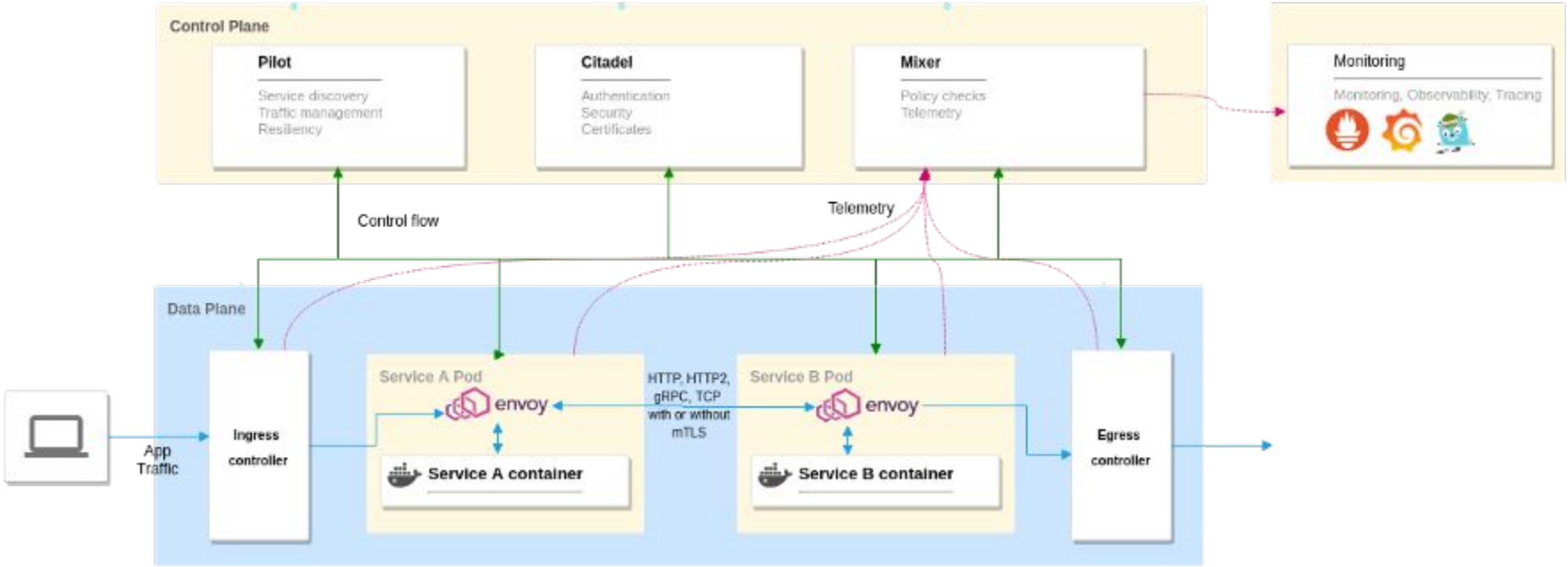


Microservices with Istio service mesh

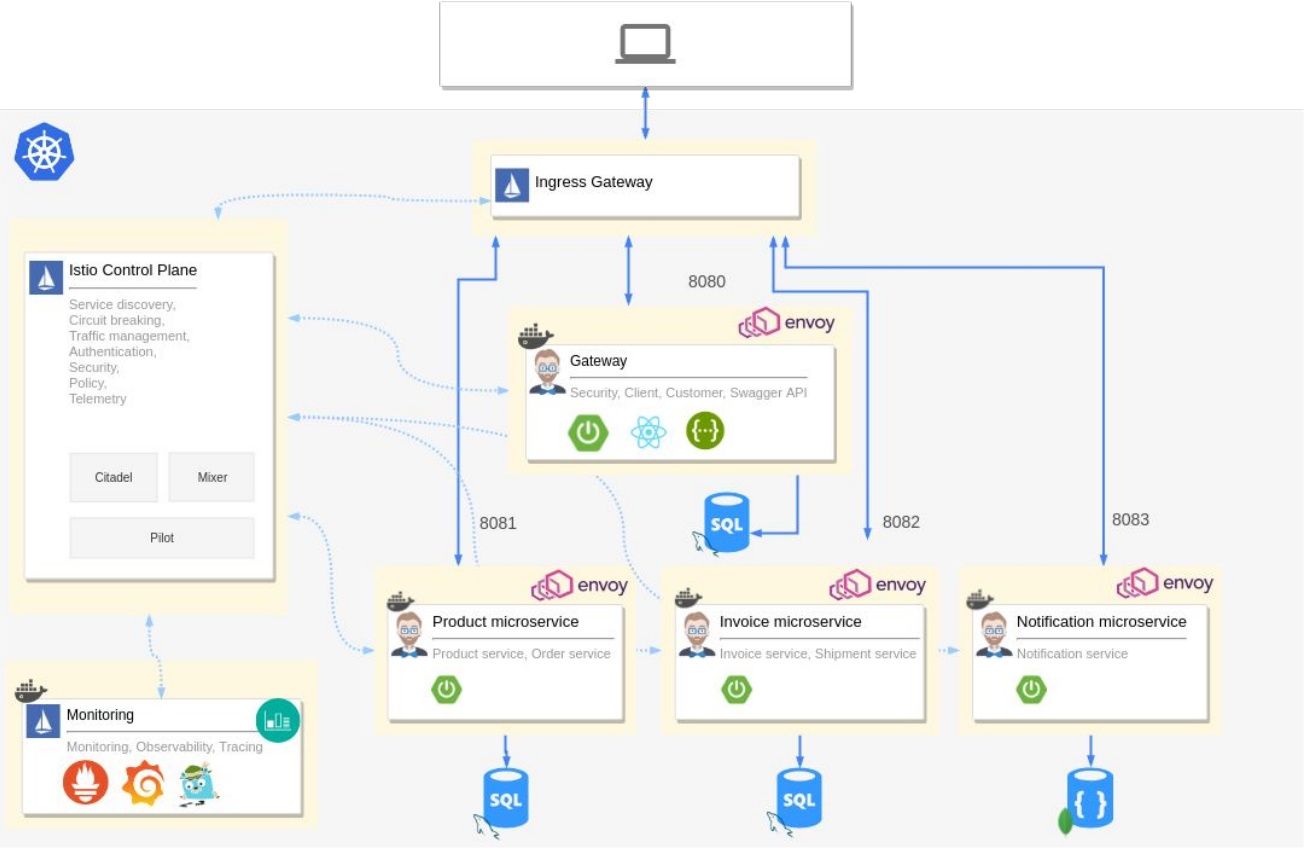
What does Istio do?

- Service discovery
- Automatic load balancing
- Routing, circuit breaking, retries, fail-overs, fault injection
- Policy enforcement for access control, rate limiting, A/B testing, traffic splits, and quotas
- Metrics, logs, and traces
- Secure service-to-service communication

Istio architecture



Microservice with Istio on Kubernetes



Prerequisite - Cluster (GKE)

Create GCP Project :

```
$ gcloud projects create jhipster-demo -  Enable billing and Kubernetes Engine API
$ gcloud config set project jhipster-demo
```

Create GKE Cluster :

```
$ gcloud container clusters create hello-hipster \
     --cluster-version 1.11 \
     --num-nodes 4 \
     --machine-type n1-standard-2 --zone europe-west1-b
```

Set Credentials:

```
$ gcloud container clusters get-credentials hello-hipster
```

GKE Cluster

Kubernetes clusters

CREATE CLUSTER | DEPLOY | REFRESH | DELETE

A Kubernetes cluster is a managed group of uniform VM instances for running Kubernetes. Learn more

Filter by label or name

Name	Location	Cluster size	Total cores	Total memory	Notifications	Labels	
hello-hipster	europe-west1-b	4	8 vCPUs	30.00 GB			Connect

Details | Storage | Nodes

Nodes

Filter nodes

Columns

Name	Status	CPU requested	CPU allocatable	Memory requested	Memory allocatable	Storage requested	Storage allocatable
gke-hello-hipster-default-pool-efe8b9c9-3f6r	Ready	850 mCPU	1.93 CPU	499.12 MB	5.92 GB	0 B	0 B
gke-hello-hipster-default-pool-efe8b9c9-61k4	Ready	908 mCPU	1.93 CPU	787.46 MB	5.92 GB	0 B	0 B
gke-hello-hipster-default-pool-efe8b9c9-t6b0	Ready	1 CPU	1.93 CPU	2.47 GB	5.92 GB	0 B	0 B
gke-hello-hipster-default-pool-efe8b9c9-v4xl	Ready	1.61 CPU	1.93 CPU	1.03 GB	5.92 GB	0 B	0 B

Prerequisite - Istio

Install Istio :

```
$ cd ~/
$ export ISTIO_VERSION=1.1.0
$ curl -L https://git.io/getLatestIstio | sh -
$ ln -sf istio-$ISTIO_VERSION istio
$ export PATH=~/istio/bin:$PATH
```

Install Istio in GKE cluster :

```
$ kubectl apply -f ~/istio/install/kubernetes/helm/istio/templates/crds.yaml
$ kubectl apply -f ~/istio/install/kubernetes/istio-demo.yaml \
    --as=admin --as-group=system:masters
```

Get Ingress Gateway IP:

```
$ kubectl get svc istio-ingressgateway -n istio-system
```

Istio pods

Workloads REFRESH DEPLOY

Workloads are deployable units of computing that can be created and managed in a cluster.

Is system object : False | Namespace : istio-system | Filter workloads | Columns

Name	Status	Type	Pods	Namespace	Cluster
grafana	OK	Deployment	1/1	istio-system	hello-hipster
istio-citadel	OK	Deployment	1/1	istio-system	hello-hipster
istio-cleanup-secrets	OK	Job	0/0	istio-system	hello-hipster
istio-egressgateway	OK	Deployment	1/1	istio-system	hello-hipster
istio-galley	OK	Deployment	1/1	istio-system	hello-hipster
istio-grafana-post-install	OK	Job	0/0	istio-system	hello-hipster
istio-ingressgateway	OK	Deployment	1/1	istio-system	hello-hipster
istio-pilot	OK	Deployment	1/1	istio-system	hello-hipster
istio-policy	OK	Deployment	2/2	istio-system	hello-hipster
istio-sidecar-injector	OK	Deployment	1/1	istio-system	hello-hipster
istio-statsd-prom-bridge	OK	Deployment	1/1	istio-system	hello-hipster
istio-telemetry	OK	Deployment	5/5	istio-system	hello-hipster
istio-tracing	OK	Deployment	1/1	istio-system	hello-hipster
prometheus	OK	Deployment	1/1	istio-system	hello-hipster
servicegraph	OK	Deployment	1/1	istio-system	hello-hipster

Rows per page: 50 1 - 15 of 15

JDL Application

```
application {
  config {
    baseName store
    applicationType gateway
    packageName com.jhipster.demo.store
    serviceDiscoveryType no
    authenticationType jwt
    prodDatabaseType mysql
    cacheProvider hazelcast
    buildTool gradle
    clientFramework react
    useSass true
    testFrameworks [protractor]
  }
  entities *
}
```

```
application {
  config {
    baseName notification
    applicationType microservice
    packageName com.jhipster.demo.notification
    serviceDiscoveryType no
    authenticationType jwt
    databaseType mongodb
    cacheProvider no
    enableHibernateCache false
    buildTool gradle
    serverPort 8083
  }
  entities Notification
}
```

JDL Entity & Relationship

```
entity ProductCategory {
    name String required
    description String
}

entity ProductOrder {
    placedDate Instant required
    status OrderStatus required
    code String required
    invoiceId Long
    customer String required
}

enum OrderStatus {
    COMPLETED, PENDING, CANCELLED
}

relationship OneToMany {
   ProductOrder{orderItem} to OrderItem{order(code) required} ,
   ProductCategory{product} to Product{productCategory(name)}
}

service Product, ProductCategory, ProductOrder, OrderItem with serviceClass
paginate Product, ProductOrder, OrderItem with pagination
microservice Product, ProductOrder, ProductCategory, OrderItem with product
```

JDL Deployment for Kubernetes with Istio

```
225 deployment {
226   deploymentType kubernetes
227   appsFolders [store, invoice, notification, product]
228   serviceDiscoveryType no
229   dockerRepositoryName "deepu105"
230   kubernetesNamespace jhipster
231   kubernetesServiceType Ingress
232   ingressDomain "35.241.165.213.nip.io"
233   istio autoInjection
234   istioRoute true
235 }
```

JDL

JHipster Domain Language

JDL : http://bit.ly/codemotion-rome-jdl

Reference : https://www.jhipster.tech/jdl/
Studio : https://start.jhipster.tech/jdl-studio/

JHipster magic

```
$ jhipster import-jdl app.jdl
```

```
WARNING! Kubernetes configuration generated, but no Jib cache found
If you forgot to generate the Docker image for this application, please run:
To generate the missing Docker image(s), please run:
  ./gradlew bootWar -Pprod jibDockerBuild in /home/deepu/workspace/temp/test2/store
  ./gradlew bootWar -Pprod jibDockerBuild in /home/deepu/workspace/temp/test2/invoice
  ./gradlew bootWar -Pprod jibDockerBuild in /home/deepu/workspace/temp/test2/notification
  ./gradlew bootWar -Pprod jibDockerBuild in /home/deepu/workspace/temp/test2/product

WARNING! You will need to push your image to a registry. If you have not done so, use the following commands to tag and push the images:
  docker image tag store deepu105/store
  docker push deepu105/store
  docker image tag invoice deepu105/invoice
  docker push deepu105/invoice
  docker image tag notification deepu105/notification
  docker push deepu105/notification
  docker image tag product deepu105/product
  docker push deepu105/product

You can deploy all your apps by running the following script:
  ./kubectl-apply.sh

Use these commands to find your application's IP addresses:
  kubectl get svc store -n jhipster

INFO! Congratulations, JHipster execution is complete!
INFO! Deployment: child process exited with code 0
```

Deploy to GKE

```
$ cd kubernetes
```

```
$ ./kubectl-apply.sh
```

```
$ watch kubectl get pods -n jhipster
```

Exploring the app

Application Gateway:

```
$ export INGRESS_IP=$(kubectl -n istio-system get svc istio-ingressgateway \
      -o jsonpath='{.status.loadBalancer.ingress[0].ip}')
```

```
$ google-chrome store.$INGRESS_IP.nip.io
```

Gateway

Store 0.0.1-SNAPSHOT

Home | Entities | Administration | English | Account

Welcome, Java Hipster!

This is your homepage

You are logged in as user "admin".

If you have any question on JHipster:

- **JHipster homepage**
- **JHipster on Stack Overflow**
- **JHipster bug tracker**
- **JHipster public chat room**
- **follow @java_hipster on Twitter**

If you like JHipster, don't forget to give us a star on **Github**!

Monitoring & logs

Grafana - Monitoring dashboard:

```
$ kubectl -n istio-system port-forward $(kubectl -n istio-system get pod \
    -l app=grafana -o jsonpath='{.items[0].metadata.name}') 3000:3000
```

```
$ google-chrome http://localhost:3000
```

Prometheus - Log metrics:

```
$ kubectl -n istio-system port-forward $(kubectl -n istio-system get pod -l
\
    app=prometheus -o jsonpath='{.items[0].metadata.name}') 9090:9090
```

```
$ google-chrome http://localhost:9090
```

Grafana dashboard

Grafana - Istio Service Dashboard - Google Chrome

Grafana - Istio Service Das

localhost:3000/d/LJ_uJAvmk/istio-service-dashboard?refresh=10s&orgId=1&var-service...

Istio Service Dashboard

Last 5 minutes Refresh every 10s

Service store.jhipster.svc.cluster.local | Client Workload Namespace All | Client Workload All | Service Workload Namespace All | Service Workload All

SERVICE: store.jhipster.svc.cluster.local

Client Request Volume: 0 ops

Client Success Rate (non-5xx responses): 100%

Client Request Duration: 100 ms, 90 ms, 80 ms, 70 ms; 13:24, 13:26, 13:28; P50, P90, P99

Client TCP Bandwidth: N/A

Server Request Volume: 0.13 ops

Server Success Rate (non-5xx response...: 100%

Server Request Duration: 150 ms, 100 ms, 50 ms, 0 ns; 13:24, 13:26, 13:28; P50, P90, P99

Server TCP Bandwidth: N/A

CLIENT WORKLOADS

Prometheus dashboard

Prometheus Time Series Collection and Processing Server - Google Chrome

Prometheus Time Series

localhost:9090/graph?g0.range_input=1h&g0.expr=envoy_cluster_inbound_8080__stor...

Prometheus | Alerts | Graph | Status | Help

Enable query history

envoy_cluster_inbound_8080__store_jhipster_svc_cluster_local_internal_upstream_rq_200

Load time: 48ms
Resolution: 14s
Total time series: 1

Execute | envoy_cluster_outbound_{

Graph | Console

1h | Until | Res. (s) | stacked

Wed, 14 Nov 2018 12:13:50 GMT

Wed, 14 Nov 2018 12:13:50 GMT
envoy_cluster_inbound_8080__store_jhipster_svc_cluster_local_internal_upstream_rq_200: 27
instance: 10.24.1.9:9102
job: envoy

28
26
24
22
20

Observability

Jaeger - Distributed tracing:

```
$  kubectl -n istio-system port-forward $(kubectl -n istio-system get pod -l \
    app=jaeger -o jsonpath='{.items[0].metadata.name}') 16686:16686
```

```
$ google-chrome http://localhost:16686
```

Service graph:

```
$ kubectl -n istio-system port-forward $(kubectl -n istio-system get pod -l \
    app=servicegraph -o jsonpath='{.items[0].metadata.name}') 8088:8088
```

```
$ google-chrome http://localhost:8088/force/forcegraph.html
```

Jaeger tracing

Jaeger UI - Google Chrome

Jaeger UI

localhost:16686/search?end=1542198799607000&limit=20&lookback=1h&maxDuration...

Jaeger UI | Lookup by Trace ID... | Search | Dependencies | About Jaeger

Find Traces

Service (9)

istio-ingressgateway

Operation (6)

all

Tags

http.status_code=200 error=true

Lookback

Last Hour

Min Duration

e.g. 1.2s, 100ms, 500us

Max Duration

e.g. 1.1s

Duration

60ms

40ms

20ms

Time

12:46:40 pm | 12:55:00 pm | 01:03:20 pm | 01:11:40 pm | 01:20:00 pm

20 Traces

Sort: Most Recent

istio-ingressgateway: store.jhipster.svc.cluster.local:8080/* 63.43ms

10 Spans | istio-ingressgateway (2) | istio-mixer (4) | istio-policy (2) | store (2)

Today 1:25:41 pm

8 minutes ago

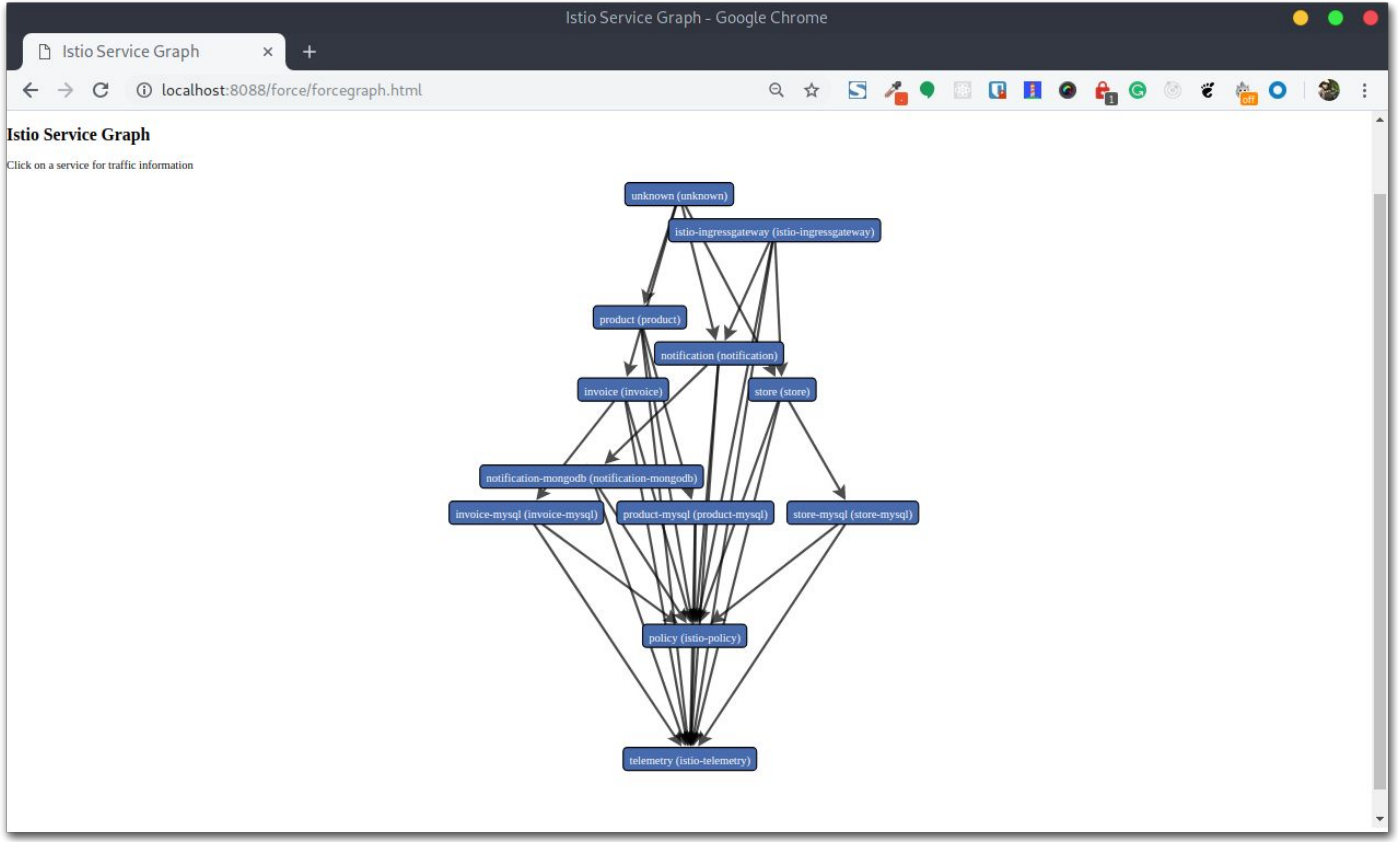
istio-ingressgateway: 35.205.83.85:80 4.24ms

5 Spans | istio-ingressgateway (2) | istio-mixer (2) | istio-policy (1)

Today 1:08:20 pm

25 minutes ago

Service graph



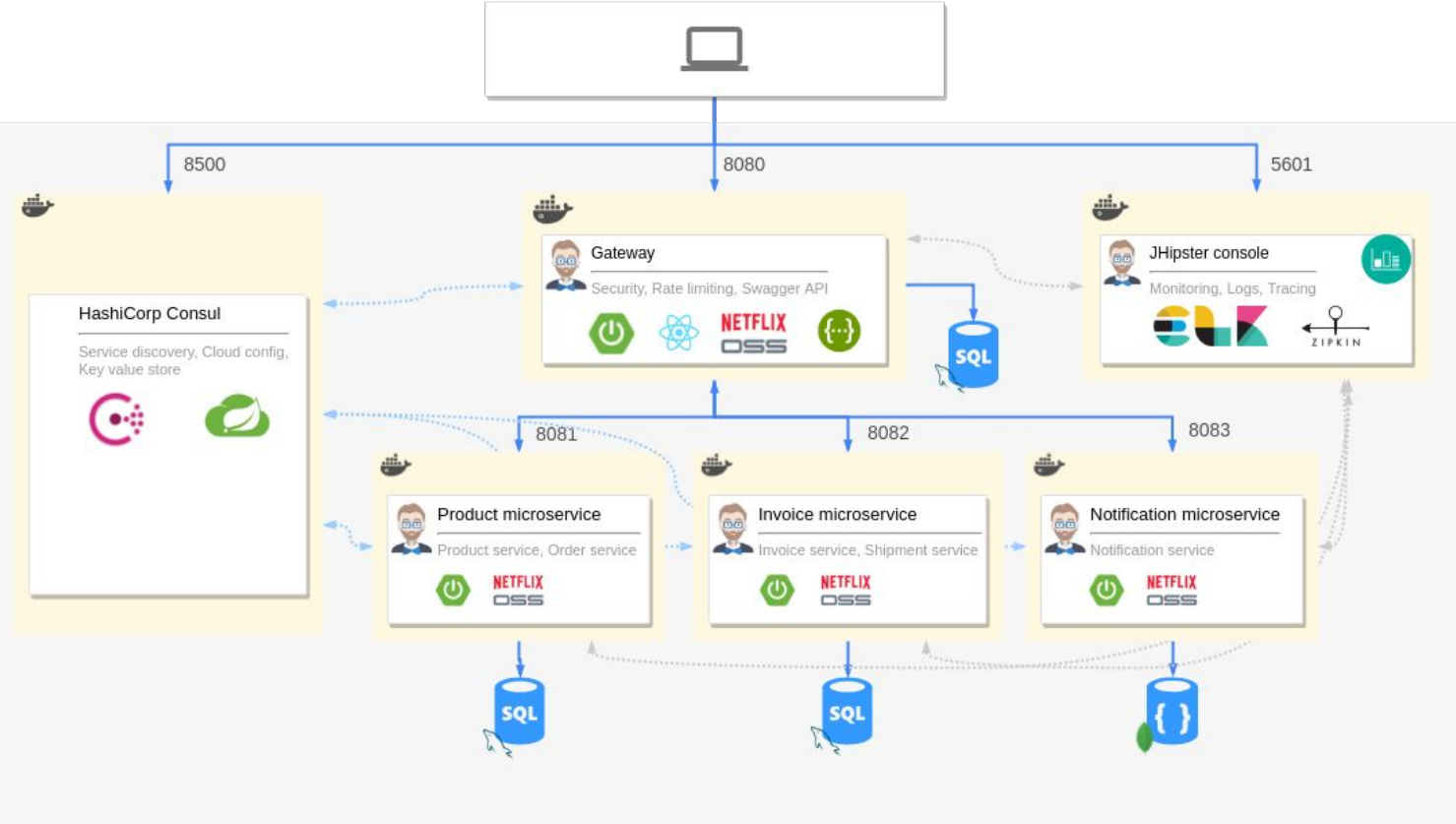
Is it worth the hype?

+ Kubernetes Native microservices
+ Reduced responsibilities for dev(Service discovery, security, tracing, etc)
+ No need to write/maintain any code for some of the complex parts of a microservice architecture.
+ A/B testing, canary releases, and lot more

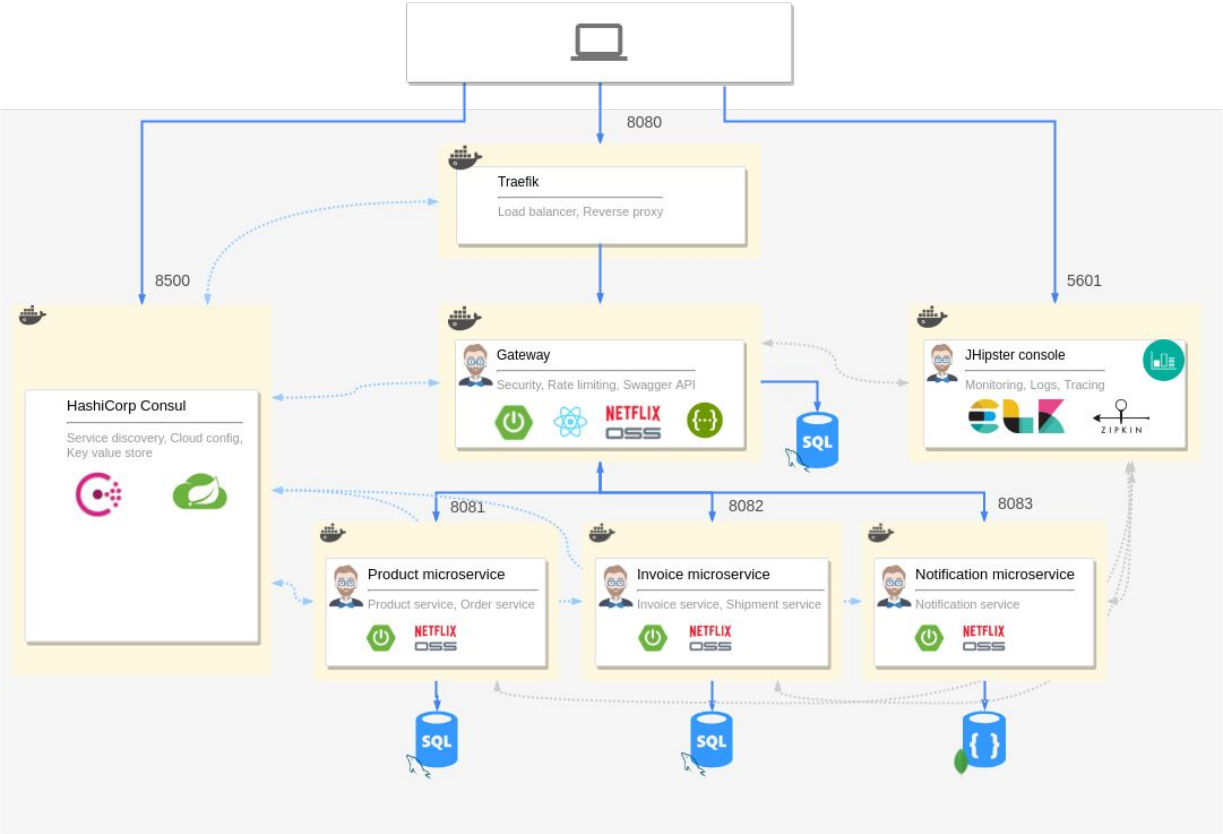
- New compared to other stable options
- Higher resource usage (CPU, Memory)
- Higher running costs
- Business logic related policies might be trickier

Other Microservices architecture options

Microservice with Consul



Microservice with Consul & Traefik



What’s coming?

JHipster 6 and beyond!

JHipster 6 and beyond

- Angular reactive forms
- VueJS blueprint
- HTML5 mode for routing
- Spring Boot 2.1
- Java 11
- Kubernetes support improvements
- Istio support improvements
- Kiali support for Istio
- Gradle 5
- JDL improvements

Spring Boot 2.x

- Migration to Spring Boot 2.1 complete
  - Another good reason to use JHipster!
  - JDK 11 support
- Work is under way for reactive programming support
  - Test it by running “jhipster --experimental”
  - Still work to do on entities and on the client-side

More information on JHipster

Main website	https://jhipster.tech
JHipster online	https://start.jhipster.tech
GitHub	https://github.com/jhipster/generator-jhipster
Twitter	https://twitter.com/java_hipster
Stack Overflow	https://stackoverflow.com/questions/tagged/jhipster?sort=newest

Thank you

Do rate the talk if you found it useful!