



ISTIO –

SERVICE MESH

Computación Tolerante a Fallas

Docente: Dr. Michel Emmanuel
López Franco

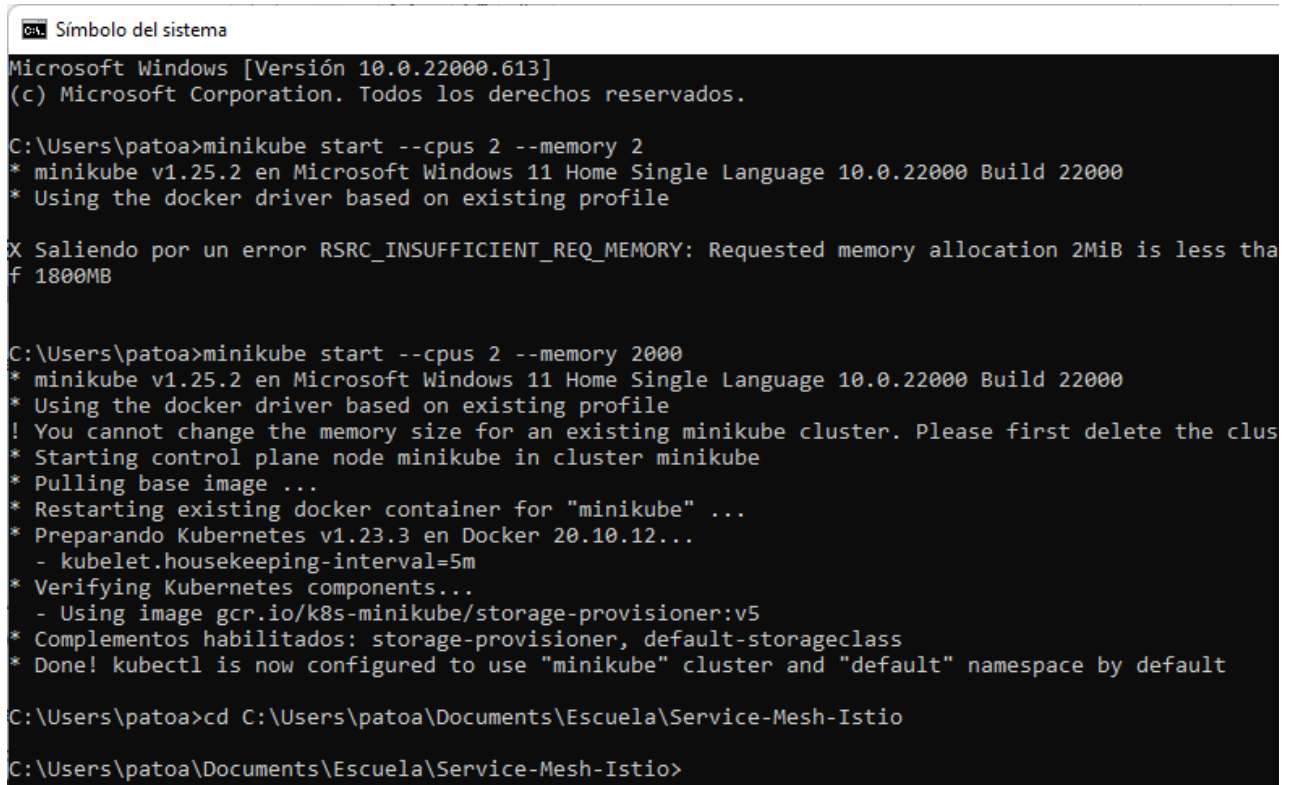
Sección D06

Murillo Cortes, Jeanette

DESARROLLO DE LA PRÁCTICA

Primeramente, se creó el clúster de minikube con sus características default, por lo que se utilizó el comando `minikube start --cpus 2 --memory 2000`

para definir la CPU y los recursos de memoria. En mi caso, para que pudiera funcionar tanto minikube como mi máquina, le puse que ocupara 2 de RAM de CPU y 2000 de memoria. De esta manera, se inició el clúster, como se muestra en la siguiente imagen.



```
Símbolo del sistema
Microsoft Windows [Versión 10.0.22000.613]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\patoa>minikube start --cpus 2 --memory 2
* minikube v1.25.2 en Microsoft Windows 11 Home Single Language 10.0.22000 Build 22000
* Using the docker driver based on existing profile

X Saliendo por un error RSRC_INSUFFICIENT_REQ_MEMORY: Requested memory allocation 2MiB is less than 1800MB

C:\Users\patoa>minikube start --cpus 2 --memory 2000
* minikube v1.25.2 en Microsoft Windows 11 Home Single Language 10.0.22000 Build 22000
* Using the docker driver based on existing profile
! You cannot change the memory size for an existing minikube cluster. Please first delete the cluster.
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
* Restarting existing docker container for "minikube" ...
* Preparando Kubernetes v1.23.3 en Docker 20.10.12...
  - kubelet.housekeeping-interval=5m
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Complementos habilitados: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default

C:\Users\patoa>cd C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>
```

Para que continuar, se debe de ir a la página de Istio <https://istio.io/> en el apartado de Documentación y Download. De esta manera, se puede descargar desde el comando que aparece en la página o llendo a al apartado de 'Istio releases' que te envia a una página de GitHub donde se descarga un un .zip de acuerdo a nuestro sistema operativo. En mi caso, como mi máquina es Windows 10, utilicé el de 'istio-1.9.0-win.zip'.

En mi caso, como usé el .zip, lo descomprimí en la carpeta del proyecto. De esta manera, lo siguiente que se hizo fue ingresar a la carpeta mediante el command line y declarar la ruta de donde esta Istio con los siguientes comandos: `cd C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3, echo $PATH, SET PATH=$PATH:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin, cd bin.`

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>cd C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3>echo $PATH
$PATH

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3>export PATH=$PATH:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin
"export" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3>SET PATH=$PATH:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3>istioctl
"istioctl" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3>cd bin
```

De esta manera que si ahora ingresamos el comando `istioctl` podemos observar que ahí se encuentra Istio. Se debe de aclarar que este comando solo funciona desde donde se haya agregado el comando Istio, en otras carpetas, no va a funcionar.

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin>istioctl
Istio configuration command line utility for service operators to
debug and diagnose their Istio mesh.

Usage:
  istioctl [command]

Available Commands:
  admin          Manage control plane (istiod) configuration
  analyze        Analyze Istio configuration and print validation messages
  authz          (authz is experimental. Use `istioctl experimental authz`)
  bug-report     Cluster information and log capture support tool.
  completion     Generate the autocompletion script for the specified shell
  create-remote-secret Create a secret with credentials to allow Istio to access remote Kubernetes apiserver
  dashboard      Access to Istio web UIs
  experimental   Experimental commands that may be modified or deprecated
  help           Help about any command
  install        Applies an Istio manifest, installing or reconfiguring Istio on a cluster.
  kube-inject    Inject Istio sidecar into Kubernetes pod resources
  manifest       Commands related to Istio manifests
  operator       Commands related to Istio operator controller.
  profile        Commands related to Istio configuration profiles
  proxy-config   Retrieve information about proxy configuration from Envoy [kube only]
  proxy-status   Retrieves the synchronization status of each Envoy in the mesh [kube only]
  remote-clusters Lists the remote clusters each istiod instance is connected to.
  tag            Command group used to interact with revision tags
  upgrade        Upgrade Istio control plane in-place
  validate       Validate Istio policy and rules files
  verify-install Verifies Istio Installation Status
  version        Prints out build version information

Flags:
  --context string  The name of the kubeconfig context to use
  -h, --help        help for istioctl
  -i, --istioNamespace string Istio system namespace (default "istio-system")
  -c, --kubeconfig string Kubernetes configuration file
  -n, --namespace string Config namespace
  --vklog Level     number for the log level verbosity. Like -v flag. ex: --vklog=9

Additional help topics:
  istioctl options  Displays istioctl global options
```

Para continuar, podemos observar que tenemos default namespaces, pero faltaria agregar Istio al clúster.

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin>kubectl get ns
NAME                STATUS   AGE
default             Active   21d
kube-node-lease     Active   21d
kube-public         Active   21d
kube-system         Active   21d

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin>kubectl get pod
No resources found in default namespace.
```

Este proceso de instalar Istio en el minikube cluster se hace con el comando `istioctl install` como se muestra en la siguiente imagen.

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin>istioctl install
This will install the Istio 1.13.3 default profile with ["Istio core" "Istiod" "Ingress gateways"] components into the cluster. Proceed? (y/N) y
✔ Istio core installed
✔ Istiod installed
✔ Ingress gateways installed
✔ Installation complete
Making this installation the default for injection and validation.

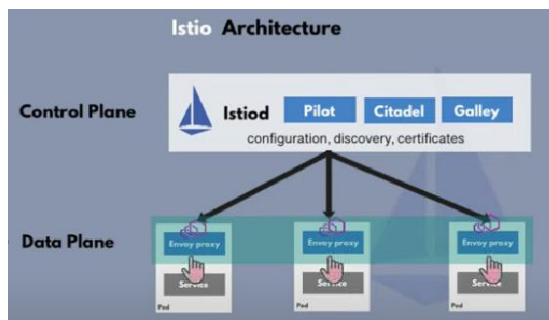
Thank you for installing Istio 1.13. Please take a few minutes to tell us about your install/upgrade experience! https://forms.gle/pzWZpAvMVBecaQ9h9
```

De manera que si usamos nuevamente el comando `kubectl get ns` se puede observar que ya nos aparece entre los namespaces.

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin>kubectl get ns
NAME                STATUS   AGE
default              Active   21d
istio-system         Active   3m37s
kube-node-lease      Active   21d
kube-public          Active   21d
kube-system          Active   21d
```

Donde dentro se encuentran Istio Ingress Gateway y el componente Istiod.

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin>kubectl get pod -n istio-system
NAME                                READY   STATUS    RESTARTS   AGE
istio-ingressgateway-6dc56fc9f9-r66nf  1/1     Running   0           2m55s
istiod-8488b9bdc7-45kmz               1/1     Running   0           4m18s
```



Istio Service Mesh Architecture

Istiod es el plano de control, donde que administra los procesos del cliente y servidor con servidores proxy, de manera que con una aplicación de microservicios, se le dará un pod por cada microservicio y luego Istio

inyectará el proxy enviado en cada uno de esos pods.

Al continuar con Istio, se utilizó una aplicación demo de Google Cloud la cuál tiene implementada bastante microservicios para administrar con el Service Mesh, de manera que se descargó desde el repositorio de GitHub <https://github.com/GoogleCloudPlatform/microservices-demo> para de esta manera poder utilizar el deployment con el comando `kubectl apply -f kubernetes-manifests.yaml` que va a crear e iniciar los microservicios que tiene

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3\bin>cd..
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win\istio-1.13.3>cd..
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio\istio-1.13.3-win>cd..
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>kubectl apply -f kubernetes-manifests.yaml
deployment.apps/emailservice created
service/emailservice created
deployment.apps/checkoutservice created
service/checkoutservice created
deployment.apps/recommendationservice created
service/recommendationservice created
deployment.apps/frontend created
service/frontend created
service/frontend-external created
deployment.apps/paymentservice created
service/paymentservice created
deployment.apps/productcatalogservice created
service/productcatalogservice created
deployment.apps/cartservice created
service/cartservice created
deployment.apps/loadgenerator created
deployment.apps/currencyservice created
service/currencyservice created
deployment.apps/shippingservice created
service/shippingservice created
deployment.apps/redis-cart created
service/redis-cart created
deployment.apps/adservice created
service/adservice created
```

declarado el archivo .yaml.

Una manera de observar esto es mediante el comando `kubectl get pod` que tiene cómo todos los microservicios que se están ejecutando como pods, como se

muestra en la lista de la siguiente imagen:

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>kubectl get pod
NAME                                READY   STATUS              RESTARTS   AGE
adservice-75656d5f44-7dx26         0/1     Pending             0           60s
cartservice-8c64564d4-t8x2z        0/1     Pending             0           62s
checkoutservice-5d45565464-c6nb5   1/1     Running             0           65s
currencyservice-7dc56c8-xh5nc       0/1     Pending             0           61s
emailservice-67b75bf988-z694f      0/1     ContainerCreating   0           65s
frontend-5db5d7b788-8tghz          0/1     ContainerCreating   0           64s
loadgenerator-77bc9cbc96-qj25q      0/1     Pending             0           62s
paymentservice-6f69f8b58d-dzcrs     0/1     ContainerCreating   0           63s
productcatalogservice-67f5c88476-kp7qf 0/1     Pending             0           63s
recommendationservice-7ddd87dccc-mbts7 0/1     ContainerCreating   0           65s
redis-cart-78746d49dc-djhhd         0/1     Pending             0           60s
shippingservice-55bd6c45bb-z4659    0/1     Pending             0           61s
```

Cada microservicio tiene un contenedor dentro del pod. Se puede mostrar cada etiqueta de nuestros pods, pero en este caso se definió la etiqueta istio-injection-enabled para poder borrar los pods y que estos puedan reconfigurar la configuración existente pero ahora con los proxys injectados como se muestran en las siguientes imagenes.

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>kubectl get ns default --show-labels
NAME      STATUS   AGE   LABELS
default   Active   21d   kubernetes.io/metadata.name=default

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>kubectl label namespace default istio-injection=enabled
namespace/default labeled

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>kubectl get ns default --show-labels
NAME      STATUS   AGE   LABELS
default   Active   21d   istio-injection=enabled,kubernetes.io/metadata.name=default
```

```
C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>kubectl delete -f kubernetes-manifests.yaml
deployment.apps "emailservice" deleted
service "emailservice" deleted
deployment.apps "checkoutservice" deleted
service "checkoutservice" deleted
deployment.apps "recommendationservice" deleted
service "recommendationservice" deleted
deployment.apps "frontend" deleted
service "frontend" deleted
service "frontend-external" deleted
deployment.apps "paymentservice" deleted
service "paymentservice" deleted
deployment.apps "productcatalogservice" deleted
service "productcatalogservice" deleted
deployment.apps "cartservice" deleted
service "cartservice" deleted
deployment.apps "loadgenerator" deleted
deployment.apps "currencyservice" deleted
service "currencyservice" deleted
deployment.apps "shippingservice" deleted
service "shippingservice" deleted
deployment.apps "redis-cart" deleted
service "redis-cart" deleted
deployment.apps "adservice" deleted
service "adservice" deleted

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>kubectl get pod
No resources found in default namespace.

C:\Users\patoa\Documents\Escuela\Service-Mesh-Istio>kubectl apply -f kubernetes-manifests.yaml
deployment.apps/emailservice created
service/emailservice created
deployment.apps/checkoutservice created
service/checkoutservice created
deployment.apps/recommendationservice created
service/recommendationservice created
deployment.apps/frontend created
service/frontend created
service/frontend-external created
deployment.apps/paymentservice created
service/paymentservice created
deployment.apps/productcatalogservice created
service/productcatalogservice created
deployment.apps/cartservice created
service/cartservice created
deployment.apps/loadgenerator created
deployment.apps/currencyservice created
service/currencyservice created
deployment.apps/shippingservice created
service/shippingservice created
deployment.apps/redis-cart created
service/redis-cart created
deployment.apps/adservice created
service/adservice created
```

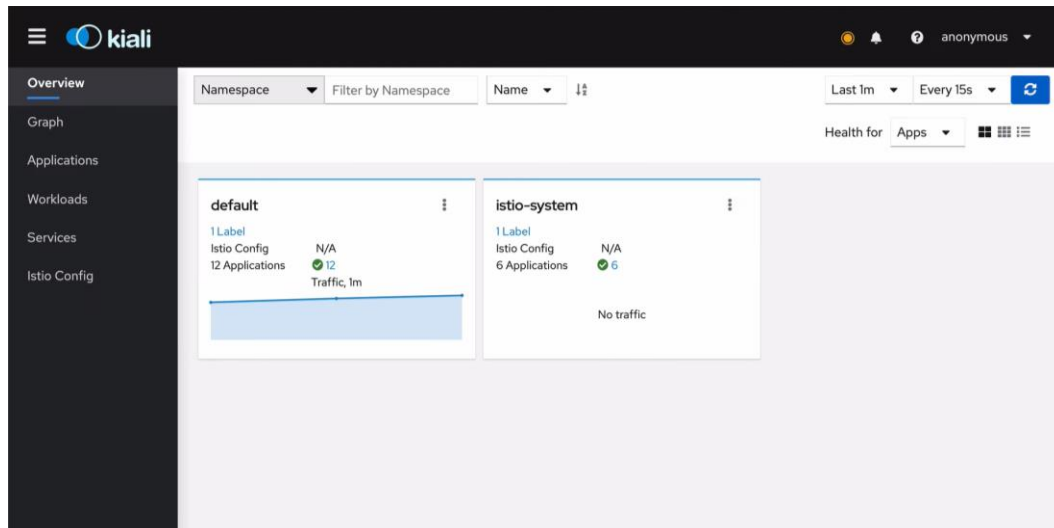
Ahora, para observar los microservicios, mediante el comando `kubectl get pod -n istio-system`, de manera que el servicio de Kiali nos muestra en qué

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
grafana	ClusterIP	10.96.128.204	<none>	3000/TCP
istio-ingressgateway	LoadBalancer	10.110.39.171	<pending>	15021:30557/TCP,80:30857/TCP,443:30291/TCP,15012:30861/TCP,15443:30369/TCP
istiod	ClusterIP	10.98.79.185	<none>	15010/TCP,15012/TCP,443/TCP,15014/TCP
jaeger-collector	ClusterIP	10.108.27.60	<none>	14268/TCP,14250/TCP
kiali	ClusterIP	10.98.210.25	<none>	20001/TCP,9090/TCP
prometheus	ClusterIP	10.105.197.150	<none>	9090/TCP
tracing	ClusterIP	10.99.52.153	<none>	80/TCP
zipkin	ClusterIP	10.104.122.240	<none>	9411/TCP

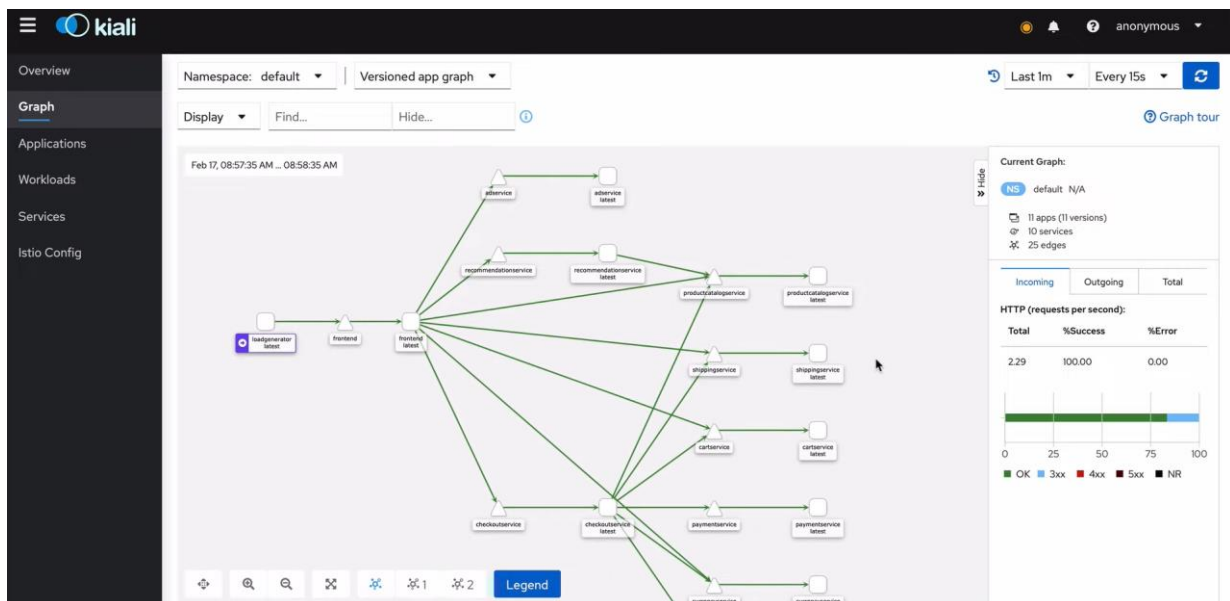
puerto se puede observar la monitorización de nuestros servicios, con el comando `kubectl port-forward srv/kiali -n istio-system 20001`

```
Forwarding from 127.0.0.1:20001 -> 20001
Forwarding from [::1]:20001 -> 20001
```

Así, al ir al URL que nos aparece en el la línea de comandos, el 127.0.0.1:20001, nos va a llevar a Kiali:




Si nos vamos al apartado del Grafo de Service Mesh, podemos observar cómo se están ejecutando todos los microservicios de la aplicación .



CONCLUSIONES

El uso de Service Mesh con el proyecto Istio nos permite como programadores conocer más y mejores herramientas que puedan hacer una buena administración a la arquitectura de los microservicios de nuestras aplicaciones. En el desarrollo de la práctica aprendí bastante acerca de esto mediante la investigación de Istio ya que era una herramienta bastante nueva para mí, pero que ahora me permite observar que su uso proporciona un buen



balance de carga y mantenimiento a los microservicios, de manera que si una aplicación tiene muchos, puede ser bastante escalable.

BIBLIOGRAFÍA

- TechWorld (Febrero, 2021). *"Istio Setup in Kubernetes | Step by Step Guide to install Istio Service Mesh"*. Recuperado del URL: <https://www.youtube.com/watch?v=voAyroDb6xk&t=202s>
- Craig Box -Google (Marzo, 2020). *"Istio / Introducción a istiod: simplificación del plano de control"*. Recuperado del URL: <https://istio.io/latest/blog/2020/istiod/>
- *"GoogleCloudPlatform/microservices-demo: Sample cloud-native application with 10 microservices showcasing Kubernetes, Istio, gRPC and OpenCensus"*. Recuperado del URL: <https://github.com/GoogleCloudPlatform/microservices-demo>
- "Istio". Recuperado del URL: <https://istio.io/>
- RedHat (Junio, 2018). *"¿Qué es la malla de servicios o service mesh?"* Recuperado del URL: <https://www.redhat.com/es/topics/microservices/what-is-a-service-mesh>