UNIVERSIDAD POLITÉCNICA DE MADRID

Escuela Técnica Superior
de Ingeniería de Sistemas Informáticos

ARQUITECTURAS ORIENTADAS A SERVICIOS

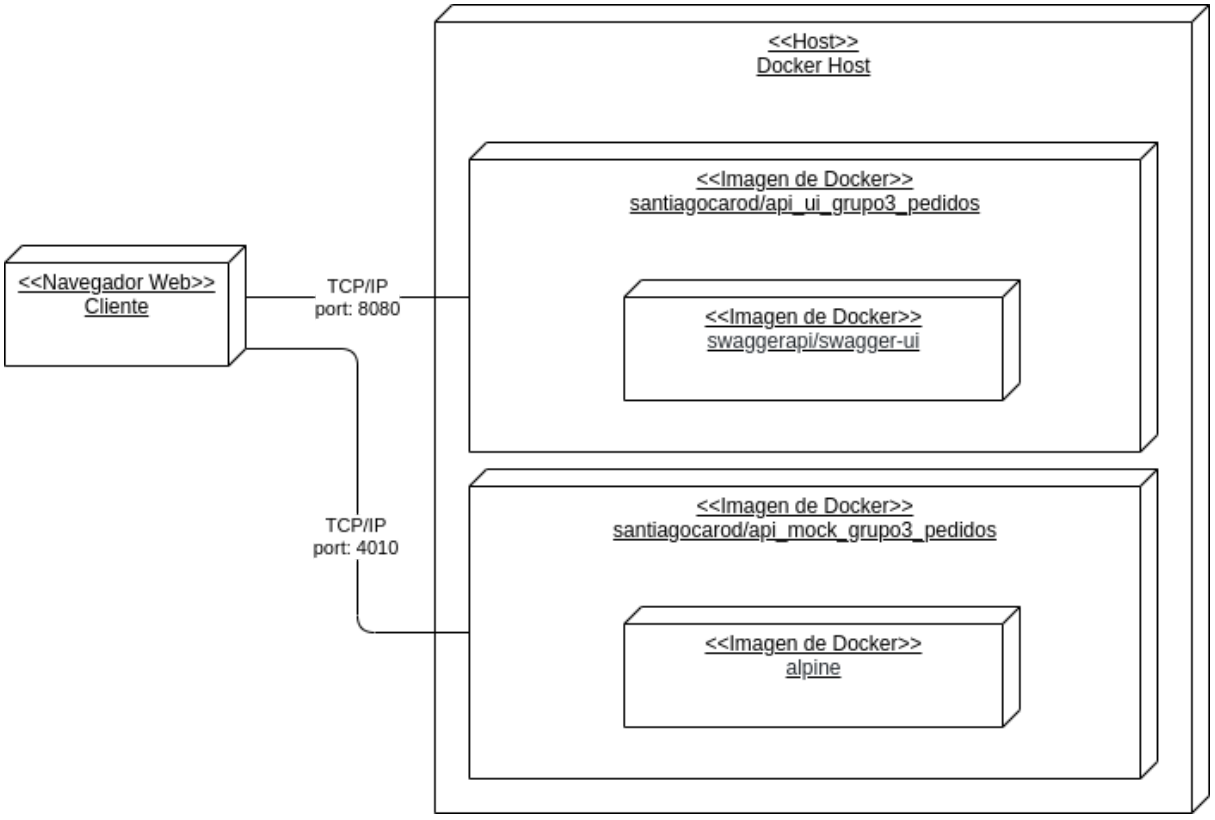
Memoria - Práctica Final

Alumnos:
Santiago Caro
Miguel Angel Perez
Ivan Pulido

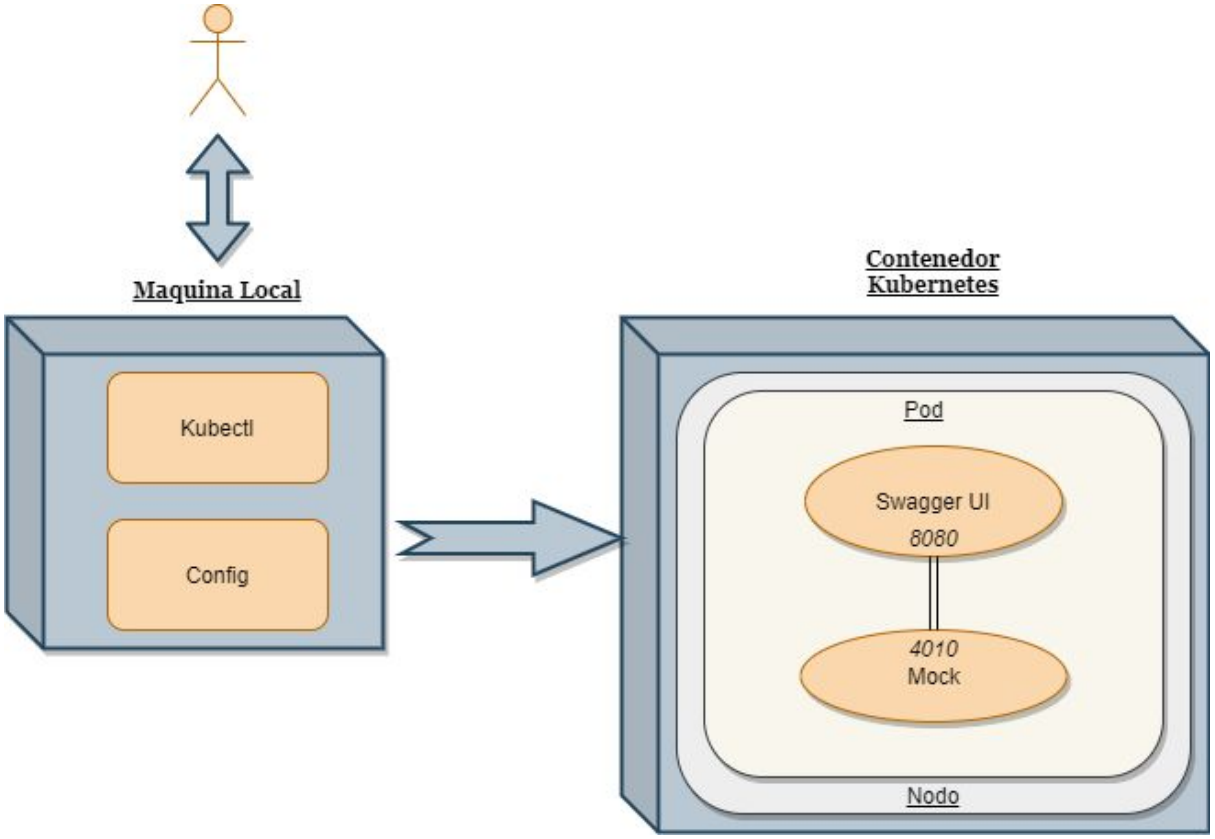
Universidad Politécnica de Madrid
ETSI SISTEMAS INFORMÁTICOS

Diagramas de Despliegue

Docker Compose



Kubernetes



Descripción de los contenedores y los Despliegues

Durante la realización de la práctica para el correcto funcionamiento de ambas funcionalidades, tanto para la interfaz de usuario que demuestra la API al usuario de una forma amigable como para la simulación del comportamiento de la API con Stoplight prism fue necesaria la creación de dos imágenes de docker. Cada una de estas imágenes funciona por separado y contienen dentro de si la especificación de la API en formato JSON para su correcto funcionamiento. Entre ellas se pueden comunicar ya que la imagen de la interfaz de usuario le puede hacer peticiones al servidor de simulación y así obtener una respuesta dentro del mismo contenedor.

Para él la imagen del primer contenedor la cual contiene la interfaz de usuario, se tomó como imagen base la que ya fue creada por la misma comunidad de Swagger y copiar dentro de su directorio base la especificación del API en formato JSON. Después es necesario darle un valor a la variable de entorno “*SWAGGER_JSON”*

para indicar en donde se encuentra dentro del sistema el archivo principal de la especificación de la API. Tomamos la decisión de escoger esta imagen como base ya que trae todo listo para mostrar la interfaz gráfica lista y no hay que realizar muchos cambios para permitir al usuario interactuar con la API. Por último se expone hacia el exterior del contenedor el puerto 8080 por el cual va a ser posible conectarse al Swagger UI

Por el otro lado el contenedor que contiene el servidor de simulación de la API tiene como base el sistema operativo liviano Alpine. A esta imagen se le agrega el manejador de paquetes de Javascript conocimo como npm y gracias a este se puede instalar Stoplight Prism dentro del contenedor para poder simular las respuestas de una API rest dentro del sistema. Se copia, asi mismo, la especificación de la API realizada para la entrega pasada, en formato json. Por último se indica que cuando el contenedor sea iniciado se ejecute el comando que le dice a Stoplight prism que escuche en el puerto 4010 en todas las direcciones dentro del contenedor por alguna petición que se encuentre dentro de las direcciones de la API.

Una vez tenemos estas dos imágenes construidas las podemos subir a Github y por medio de la integración continua de Docker Hub podemos crear las imágenes dentro de la plataforma de Docker Hub y así que sean públicas y se puedan acceder sin problema. Ahora para crear el archivo docker-compose lo que se hace principalmente es bajar estas dos imágenes desde docker-hub que ya han sido creadas con las configuraciones necesarias y exponer al exterior los puertos de cada contenedor para poder acceder a estos, así como indicar qué acciones debe tomar el controlador en caso de que alguna de los contenedores falle, en este caso se indica que se reinicie.

Para desplegar estos servicios en Kubernetes toma un poco más de trabajo al ser mucho más flexible toca especificar varias variables más. Primero que todo la estructura de este archivo llamado “Grupo3.yml” tiene dos partes principales las cuales interactúan entre sí, la primera es el Despliegue (“Deployment”) que contiene toda la información necesaria para la configuración de los contenedores y la segunda parte de este documento es la del servicio (“Service”) la cual describe la interfaz entre la máquina anfitriona y los contenedores. Para la primera parte, el despliegue, se le da un nombre a la aplicación, se le asigna una pequeña descripción y se menciona cuántas réplicas de los contenedores van a existir una vez se ejecute. Así mismo se menciona cuales son las imágenes que se van a utilizar para la ejecución y de cada una de ellas que puertos utilizan.

Teniendo toda la información del despliegue, se describe entonces la parte del servicio en la cual se menciona que es el servicio de la aplicación anteriormente,

mencionando el nombre asignado anteriormente. Después se dice que es un servicio de tipo NodePort para decir que cada nodo tiene unos puertos estáticos por los cuales se puede llegar a sus servicios. Por último se hace el mapeo de los puertos, en donde se menciona cual es el puerto que está dentro del contenedor y se mapea al puerto fuera del contenedor que a su vez se mapea a un puerto fuera de Kubernetes que tiene que ser en un rango de 30.000 a 32.000. Esto se hace para ambos contenedores. Una vez se tiene esta especificación completa para que funcionen ambos contenedores con sus servicios se despliega. Todas las comunicaciones se hacen por medio de paquetes transmitidos por medio de paquetes TCP/IP y con puertos correspondientes especificados dentro de los documentos.