BRUNO SANCHO DELTELL - 2O519135F

MEMORIA P2 - Art With Drones

INFORME DE DESARROLLO

PRODUCERS/CONSUMERS:

El programa maneja el flujo de información por KAFKA mediante 3 Topics:

-**DESTINO**(producer en engine y consumer en dron):
Envía los destinos de todos los drones a todos los drones. Al ser recibido por los drones, cada uno coge el destino que le corresponde.

-**MOVIMIENTOS**(producer en dron y consumer en engine):
Cada vez que un dron avanza (en su lista interna de movimientos), envía el movimiento realizado al engine para que este pueda actualizar el mapa.

-**MAPA**(producer en engine y consumer en dron):
Envía el mapa actualizado a los drones para que ambos puedan mostrarlo.

Todos los *producers* y *consumers* se encuentran definidos en archivos externos y se importan como librería cuando es necesario su uso.
Kafka se ha implementado por dockers.

AD_Engine:

Al iniciar Engine se inicializará el primer hilo del programa “hilosocketsdron”, que es el encargado de manejar las autenticaciones de los drones. Engine leerá del archivo compartido con Registry el token correspondiente al ID del dron que trata de conectarte, y en caso de que coincida con el que él le proporciona(que tiene almacenado en un fichero individual para cada dron) la autentificación será válida.

A continuación, se llama al módulo encargado de leer las figuras de JSON. Junto a él se inicializan los 2 producers y el consumers correspondientes a Engine.
Tras una breve carga (para dejar que kafka funcione correctamente), los drones se empezarán a conectar (sin superar el límite marcado por parámetros) y cuando el usuario pulse *Enter* (teniendo en cuenta los tiempos de carga de AD_Dron), se iniciaran los otros 4 hilos y el espectáculo comenzará.
Esos 4 hilos tienen las siguientes funciones:

-**hiloEspectaculo**: se encarga de dirigir el flujo principal de envio de figuras por el “Topic DESTINO”y de determinar cuando una figura se ha completado.
-**hiloMapa**: es el hilo más complejo, es el encargado de la comunicación con los drones para el Topic MOVIMIENTOS, de actualizar el mapa general, imprimirlo por pantalla junto a más información relevante(estado de los drones, estado de conexión de el weather, información del mismo….) y por último, envíar el mapa actualizado a los dron por el Topic MAPA.

-**hiloDrones:** es el encargado de comprobar el estado de los drones durante la ejecución de todo el programa. Utiliza un diccionario para llevar control de cuándo ha sido el último input de un dron por el Topic Movimientos, y si pasan más de 2 segundos (el flujo es casi constante hasta cuando un dron llega a su destino), se supondrá que está desconectado y se guardará su última posición por si se reconecta.
-**hiloWeather**: se encarga de la conexión por sockets con el servidor de weather, además de determinar su posible desconexión y por supuesto de enviar los drones a la posición 0,0 en caso de que la temperatura esté por debajo de -5º o por encima de 45º. En tal caso se envía un figura especial (con nombre, detener espectáculo y destino 0,0) a los drones. Los drones ya no realizarán más figuras y deberán ser desconectados por los propietarios de los mismos.

Si el engine se desconecta en cualquier momento, AD_Weather cerrará su conexión y los drones notará la desconexión. Sin embargo no se ha implementado un protocolo de reconexión del engine por falta de tiempo.

AD_Dron:

La aplicación AD_Dron tiene 2 formas de ejecución dependiendo de los parámetros de entrada.
Registrar un nuevo Dron(2 parámetros**):**
Se conectará con AD_Registry el cual le asignará el siguiente ID disponible y un token. El dron creará un fichero con el formato DRON_{ID} en el que almacenará esta información para una futura conexión con Engine. También se tendrán en cuenta situaciones de error como que la BBDD conjunta del Registry y Engine esté mal formateada y no se posible registrar un nuevo dron.

I**nscribir Dron en espectáculo(**5 parámetros**):**
Se tratará de autenticar en el engine, en caso de que lo consiga comienza el flujo principal de la aplicación tras cerrar la conexión por sockets. También sabrá si un dron se está tratando de reconectar a un espectáculo y actuará en consecuencia.

Se define los 2 consumers y el producers correspondientes, se esperan unos segundos y la aplicación está lista para comenzar una vez reciba el primer destino desde Engine.
El programa se divide en 2 hilos secundarios y el principal del propio programa:
-**hilo_actualiza_mapa**(): se encarga de recibir los mapas actualizados de Engine e imprimirlos.
-**hilo_comprueba_engine**(): lleva un conteo del último mensaje que se recibió por el Topic de DESTINO (es decir una figura) y si pasan más de 10 segundos (el flujo de figuras es constante, estas se envía continuamente hasta que se terminen de realizar por los drones) , se determinará que el engine ha perdido la conexión.

-**hiloPrincipal**(): se encarga de recibir las figuras del Topic MOVIMIENTOS, calcular los movimientos necesarios para llegar al destino de cada dron y enviarlos poco a poco al Engine. Se tiene en cuenta situaciones de error como que se deba parar el espectáculo por las condiciones climáticas.

AD_Registry

Su función principal es manejar la conexión por sockets con nuevos drones que quieran obtener un ID y un Token(generada con un Hash sha256). Se tienen en cuenta situaciones de error varias, como por ejemplo límite de drones simultáneos conectados, imposibilidad de leer el fichero de BBDD, inexistencia de este o registro de un dron ya registrado entre otras…

Si todo el proceso es correcto para un dron, se le devolverá su ID y su Token, se registrará en la BBDD compartida con engine y se cerrará la conexión por sockets.

AD_Weather

Su función es enviar la temperatura leída en un fichero cada 1.5 segundos a engine mediante sockets. Se contemplan situaciones de error como imposibilidad para enviar la temperatura, desconexión con Engine (en este caso se acabará el programa informando de lo sucedido) imposibilidad de leer el fichero entre otras…

DESPLIEGUE

Ejecutar Kafka:
docker-compose up(en termina en la carpeta *prod_cons*)

Ejecutar AD_ENGINE:
python AD_Engine.py <Puerto de escucha> <Máximo de drones> <IP del Broker> <Puerto del Broker> <IP del AD_Weather> <Puerto del AD_Weather>

Ejecutar AD_DRON:
Ejecución para conexión a engine:
python AD_Dron.py <Engine IP> <Engine Port> <Brocker IP> <Brocker Port> <Dron ID>
Ejecución para conexión a registry:
python AD_Dron.py <Registry IP> <Registry Port>

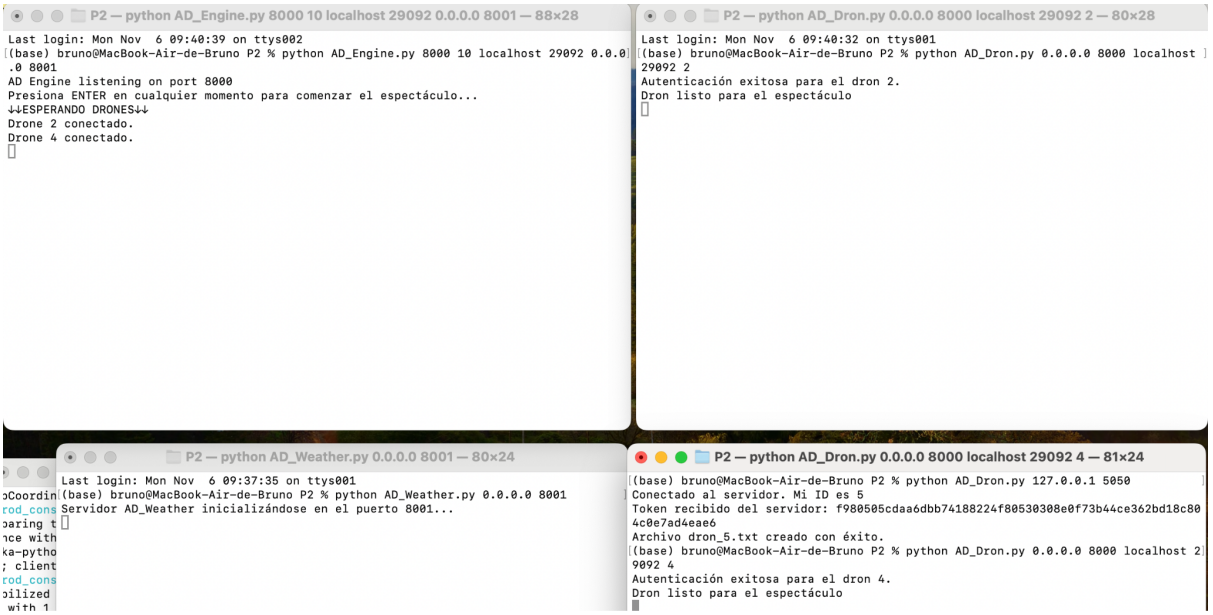
Ejecutar AD_REGISTRY:
python AD_Registry.py <Registry IP> <Registry Port>

Ejecutar AD_WEATHER:
python AD_Weather <I> <Puerto>

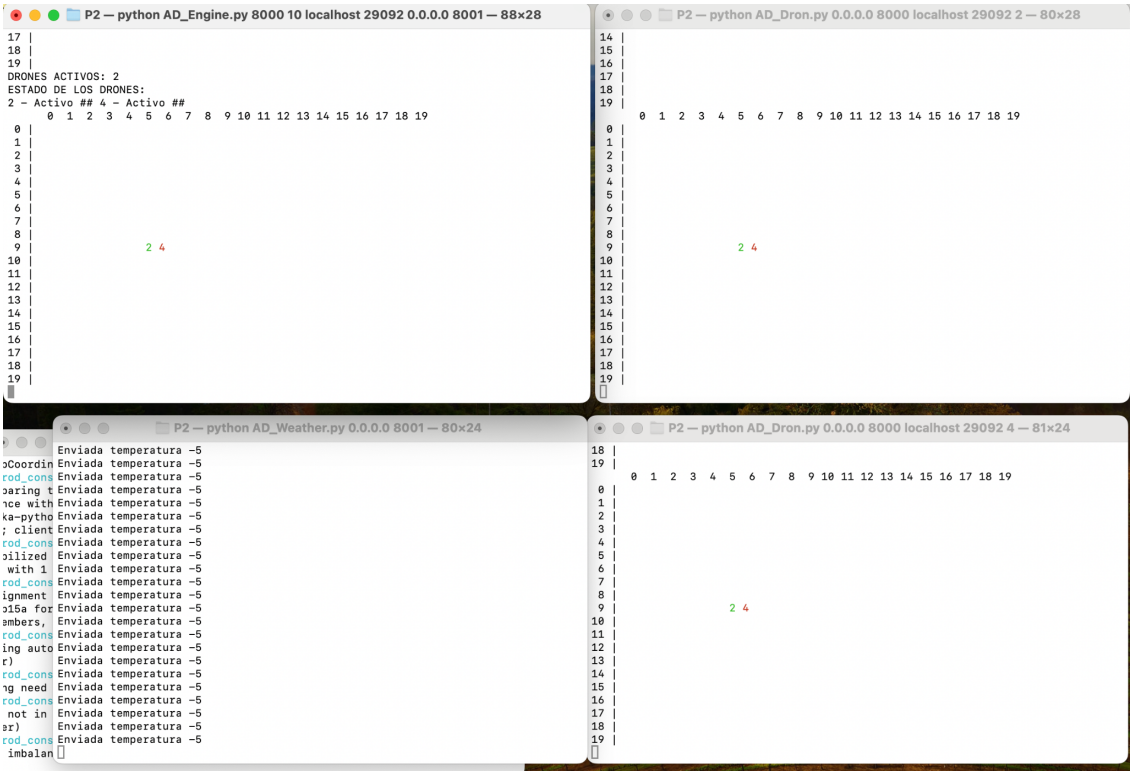
EJECUCIÓN NORMAL DEL PROGRAMA

Registro de un nuevo dron:

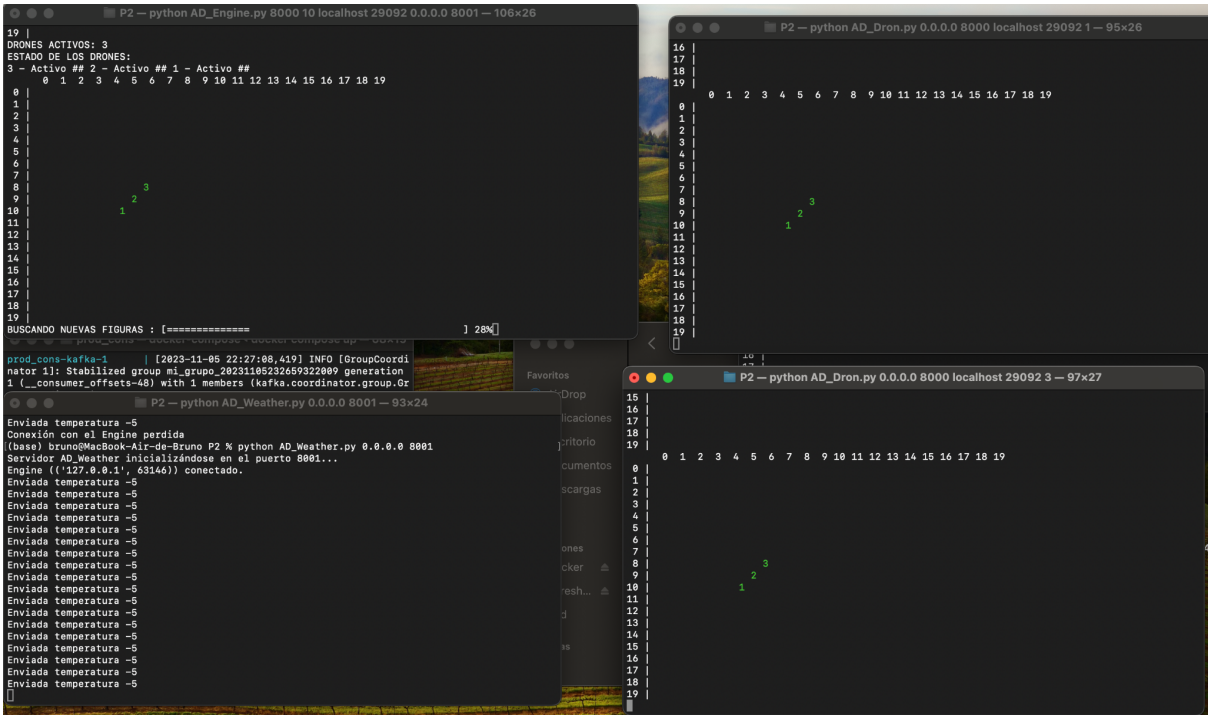
Todo listo para empezar el espectáculo:



Flujo Intermedio de un espectáculo:



Una Figura completada:



Todas las figuras completadas: