Sistema funcional

Daniel Hernández Ferrándiz - Diego Silva López

Abril 2020

1 Introducción

En este documento se describe el tipo de despliegue realizado para este proyecto y el funcionamiento del sistema final. El sistema de detección de árboles se ha implementado mediante Docker, que permite la ejecución de programas en contenedores. Este tipo de sistemas tiene una gran ventaja, y es que gracias al uso de Docker el sistema implementado funcionará en cualquier máquina que pueda ejecutar sus contenedores independientemente del hardware y su configuración.

1.1 Descripción del sistema

Para la aplicación de detección de árboles se ha decidido desarrollar una aplicación cliente/servidor. Esto permite dividir el flujo de trabajo en dos sistemas: el cliente, que tan solo enviará una petición al servidor (que podrá ser una imagen en la que analizar la masa arbórea o una dirección o coordenadas para analizar) y esperará la respuesta, y el servidor, que será el encargado de recibir peticiones, procesarlas y enviar las métricas deseadas al cliente.

Con esto se consigue que la aplicación cliente sea relativamente simple y rápida mientras que el servidor es el que contiene todo el procesado, resultando en una fácil actualización de la aplicación para todos los usuarios si es necesario y liberando de carga computacional al cliente. En este caso, el cliente y el servidor estarán montados en dos contenedores diferentes que se comunicarán mediante Docker compose, abriendo una conexión local en el puerto 8000.

1.2 Enlace al repositorio GitHub

https://github.com/dsilvalo28/AIVA-DAIA

2 Despliegue

En esta sección se detalla el diseño del despliegue propuesto para el sistema de detección de árboles mediante un diagrama UML de despliegue y la explicación de los elementos que lo forman.

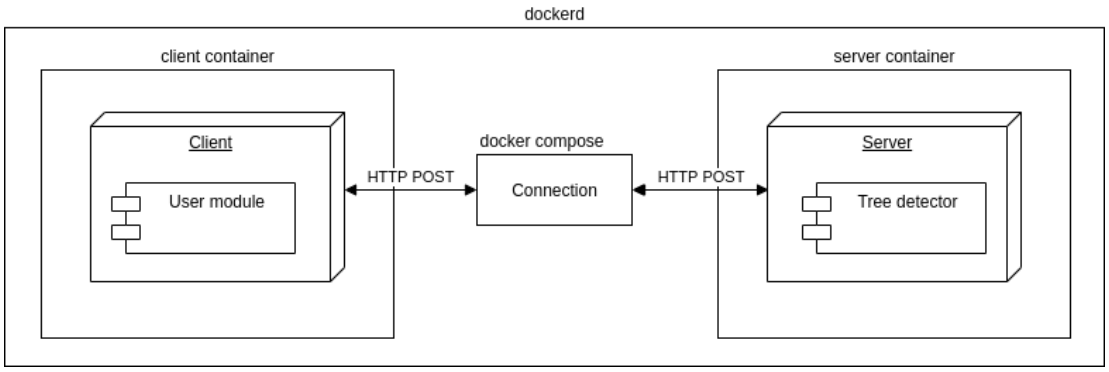

Figure 1: Diagrama de despliegue propuesto para la aplicación de detección.

El sistema consiste en dos contenedores de Docker, que serán gestionados por el gestor dockerd y se comunicarán a través de una conexión local mediante docker compose:

1. Cliente: El cliente será el encargado de recoger la información de entrada del usuario y codificarla y enviarla al servidor. El usuario pedirá el análisis de una zona mediante una imagen, unas coordenadas o una dirección. Después quedará a la espera de la respuesta del servidor y mostrará las métricas por pantalla.

2. Servidor: El servidor se dedicará a recibir POSTs, decodificarlos y procesarlos en función del tipo de entrada dada por el usuario. Si es un mapa se procesará directamente, y si es una dirección o coordenadas conseguirá el mapa de la zona y lo procesará posteriormente. Tras esto, enviará la respuesta con las métricas calculadas.

3 Funcionamiento del sistema

En este apartado se describe el proceso completo que se sigue al ejecutar la aplicación mediante un diagrama UML de secuencia que se detalla y explica a continuación.

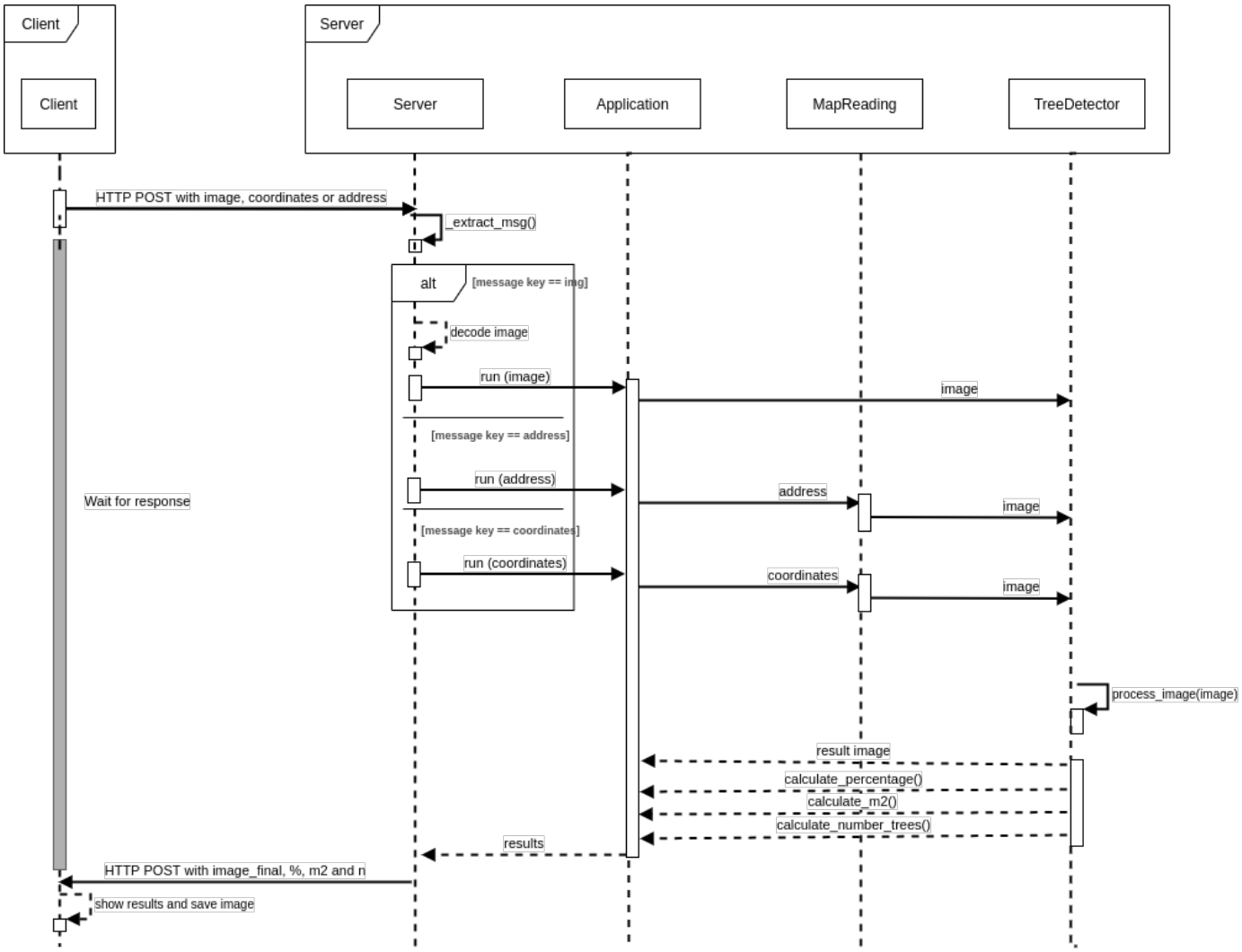

Figure 2: Diagrama de secuencia de la aplicación.

3.1 Petición del cliente

El sistema empieza con la introducción de una localización para la detección de árboles por parte del usuario. Esta localización puede ser una imagen, unas coordenadas o una dirección. Con esta información se rellena un POST en el cliente que será enviado al servidor, con tan solo uno de los tres campos posibles relleno. Tras esto, el cliente pasará a un estado de espera a la respuesta del servidor.

3.2 Procesamiento en el servidor

El servidor está constantemente activo, en espera de la llegada de un POST. Cuando llega un mensaje de este tipo extrae la información y se inicia la Application en función del tipo de datos de entrada. Si llega directamente una imagen pasa al TreeDetector para el procesado. Si llega una dirección o coordenadas, primero se obtiene el mapa en MapReading y se procesa con TreeDetector. Una vez procesada la imagen, se obtienen los resultados en la Application que inició el Server y están listos para ser enviados en un POST como respuesta al cliente.

3.3 Respuesta

Al llegar la respuesta, el cliente simplemente decodificará la imagen procesada del servidor y guardará una copia local de esta. Además, mostrará por pantalla las métricas del procesamiento para que el usuario pueda ver la información.