Servidor web concurrente y cliente en C2 AWS(septiembre 2019)

Italo Orlando Cufiño Novoa

# INTRODUCIÓN

E

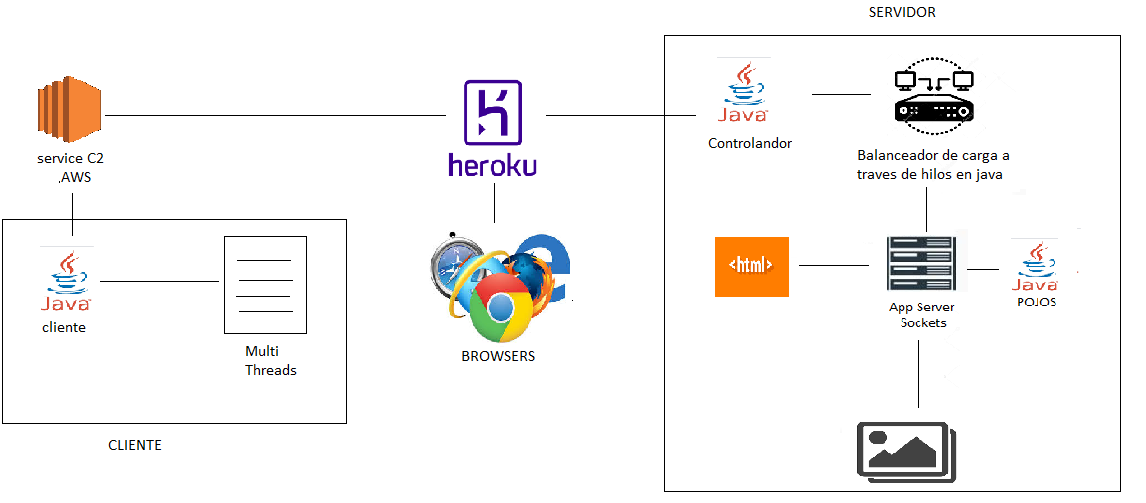
ste documento explica la implementación y manejo de disponibilidad de un servicio por medio de un servidor web concurrente a través de un balanceador de carga el cual funciona a través del uso de hilos manejados para el control de peticiones múltiples al mismo SocketService y a un cliente desplegado en C2 AWS que se encarga de hacer N peticiones a través de N hilos a nuestro servidor web concurrente.

El servidor web concurrente desplegado en Heroku permite consultas de html´s, imágenes y texto el cual pueden ser consultados a través de este link <https://boiling-scrubland-02497.herokuapp.com/> y podrás saber de acerca de su arquitectura y funciones a través de este link <https://github.com/ItaloNovoa/ProyectoAREM>

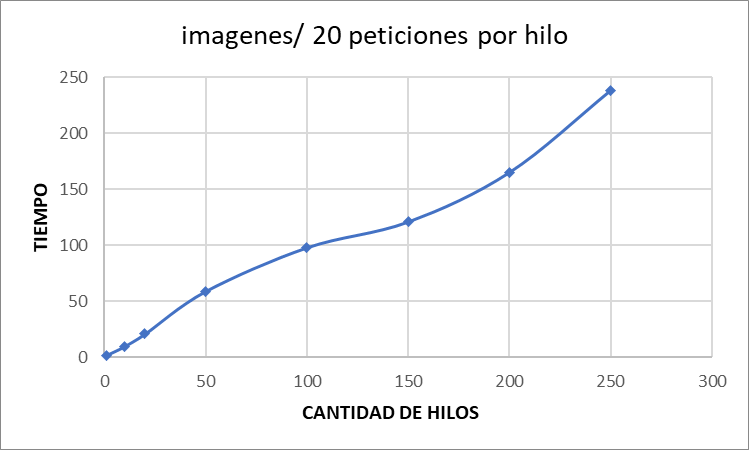
El cliente desplegado en una máquina de Linux a través de AWS en el servicio C2 es utilizado para consultar concurrentemente al servidor por medio de hilos que trabajan simultáneamente haciendo 20 consultas a distintos recursos del servidor en el tiempo de vida de cada hilo su implementación se podrá ver en el siguiente enlace

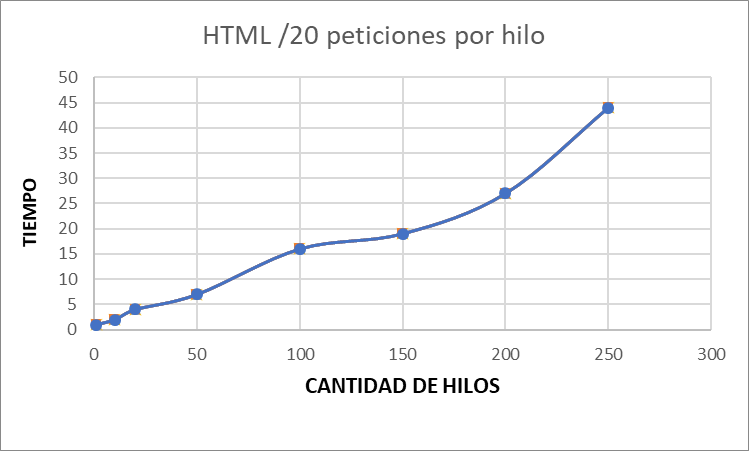
Para entender un poco de la distribución del servidor y el cliente mostraremos un diagrama de clases con los dos proyectos

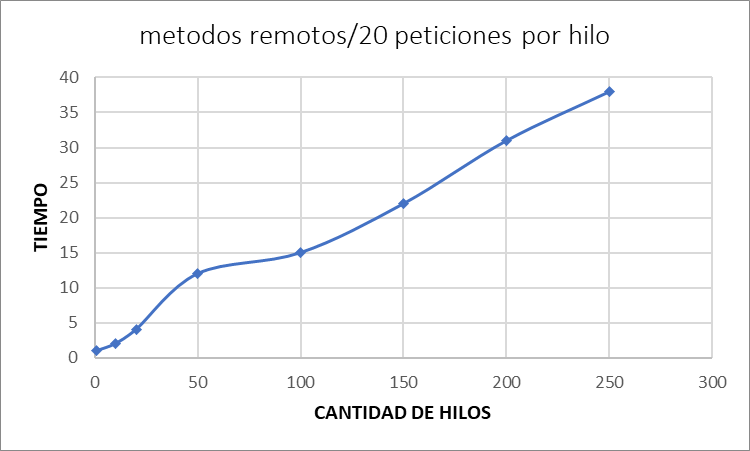
# Arquitectura De software



Con este diagrama podemos ver el ciclo o comunicación que se genera de la clase cliente con sus múltiples hilos manejados por medio de otra clase java, consultar desde una maquina Linux desplegada en Ubuntu a un link de heroku en el cual esta desplegada nuestra maquina servidor, con la intención de medir los tiempos que se demoran N hilos en consultar distintos tipos de recursos a continuación veremos unas graficas el cual muestran el tiempo que demora en hacer las peticiones en el servidor desplegado en heroku con N hilos teniendo en cuenta que cada hilo genera 20 peticiones en su tiempo de vida, a continuación veremos los tiempos generados por la solicitudes







Debido a las graficas anteriores debemos aclarar que la grafica de HTML no contiene una imagen ya que al momento de hacer las mediciones dividimos los componentes para poder ser más subjetivos y con tiempos más exactos para cada componente

# Conclusiones

* La consulta de los recursos de imágenes son los mas demorados debido al peso o cantidad de bytes que contiene una imagen
* Las consultas de los métodos remotos muestran que el uso del Servidor como framework no genera un gran tiempo de consulta con respecto a las imágenes
* El uso de un balanceador de carga nos permite un uso contante a un servicio por medio de muchos usuarios al mismo tiempo permitiendo ofrecer a mas personas el servicio