Práctica 1

SISTEMAS DISTRIBUIDOS | EINA - UNIZAR

Roberto Carlos (720100) y Samuel Ruiz de Gopegui Muñoz (685127)

Introducción

En esta práctica nos hemos enfrentado al reto de comenzar a desarrollar pequeños programas con el lenguaje Elixir, basado en Erlang. Además de ello, comenzamos el desarrollo en sistemas distribuidos con la elaboración de un programa de cálculo de números primos en varias estructuras: cliente-servidor y master-worker.

Cliente-Servidor es la más utilizada puesto que permite centralizar todas las herramientas en un lugar. Es una buena manera de abaratar costes en software y mantenimiento, pero por el contrario atenta contra la escalabilidad de un sistema distribuido al necesitar un nodo central.

Master-Worker también cuenta con las mismas desventajas pero el funcionamiento es distinto. Mientras que en Cliente-servidor el cliente pide al servidor que le realice tareas, en Maestro-esclavo el maestro pide a los esclavos/trabajadores que trabajen para él unificando luego los resultados.

Estos patrones junto con Peer–to-peer son las bases de cualquier sistema distribuido.

Arquitectura Cliente-Servidor

En esta arquitectura se implementa el protocolo “request-reply” ya que se intercambian mensajes de manera asíncrona, el envío de mensajes no es bloqueante, pero la recepción sí.

Se ha programado de manera secuencial y concurrente, en concurrente, hay un numero indeterminado de clientes preguntando al servidor el cual responde.

Es la implementación más sencilla puesto que se trata de una simple respuesta a un proceso que pregunta.

Arquitectura Master-Worker

En esta arquitectura, se sigue el protocolo “request-reply” pero el master solicita a los workers que hagan partes de la tarea dada, para ser él quien unifique los resultados.

Es la opción preferida en tareas de gran carga dado que puede segmentarse el trabajo en distintos procesos o máquinas que trabajen concurrentemente para ahorrar así bastante tiempo.

Problemas enfrentados

Durante la realización de la práctica nos hemos encontrado con varios problemas, la mayoría de ellos provocados por la poca experiencia que todavía disponemos con Elixir y agravados por la falta de tiempo.

El mayor reto, por encima de la realización de las diferentes arquitecturas, se encuentra en separar el fragmento de números primos requerido en otros más pequeños y distribuirlos entre los workers para aprovechar el tiempo de la mejor manera.

Ese problema y junto a que se devolvian listas vacias o numeros chinos, nos ha sido imposible poder realizar los test para las gráficas posteriores.