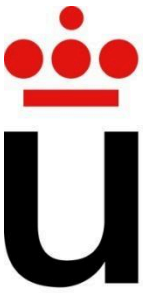


Universidad Rey Juan Carlos

ESCUELA TÉCNICA SUPERIOR
DE INGENIERÍA INFORMÁTICA

GRADO EN INTELIGENCIA ARTIFICIAL

Práctica 2:

Sistemas Operativos

Curso 2023 – 2024

Redactado por : **Jorge Camacho Mejías**

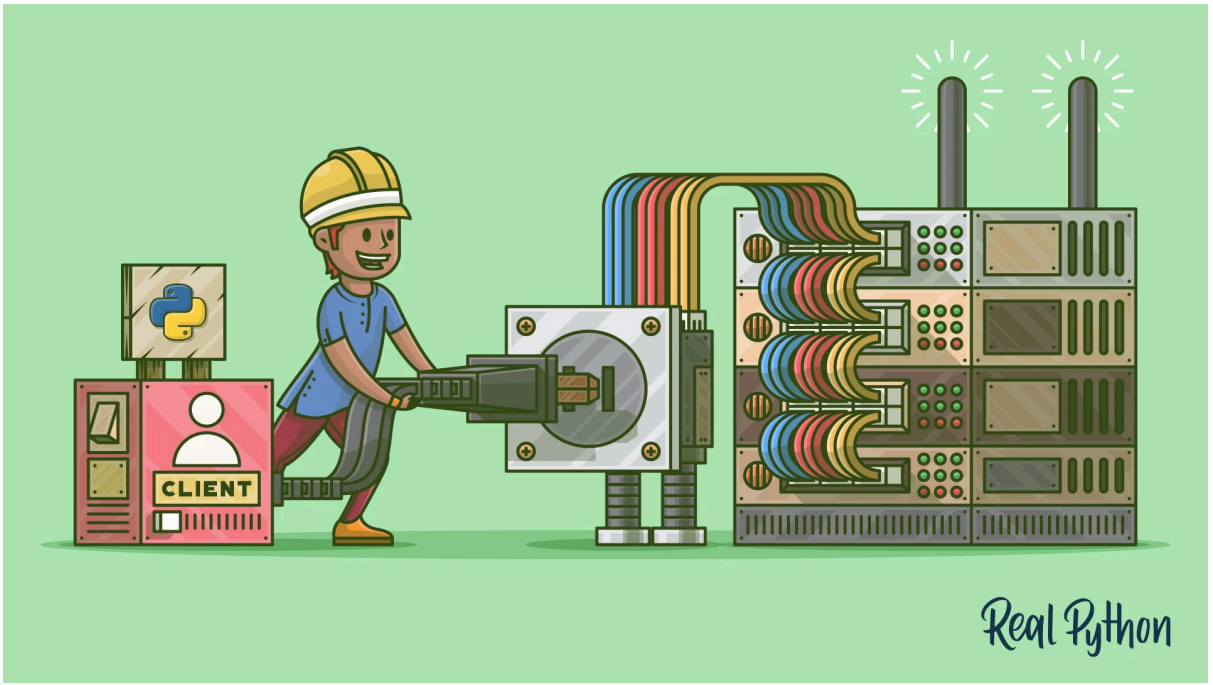
ÍNDICE

INTRODUCCIÓN

En esta práctica se nos encomendó la creación de un juego y la integración de este mismo utilizando sockets para poder alojar varias partidas con jugadores simultáneos en estas, y así aplicar conceptos que hemos visto en clase tales cómo los hilos, secciones críticas, mutex, etc.

Para el desarrollo de esta práctica utilizaremos python cómo lenguaje de programación utilizando librerías cómo socket y threading.

Dado el nivel de dificultad de la práctica no todos los puntos de la misma se han podido realizar exitosamente, sin embargo me parece importante resaltar que el mismo hecho de no conseguir el resultado esperado es igual o incluso más instructivo que el conseguirlo por lo que en esta memoria me gustaría reflejar los problemas que me han ido surgiendo en el desarrollo de la misma.



CREACIÓN SERVIDOR

Para comenzar el desarrollo de esta práctica comenzaremos explicando la creación del servidor, gracias a las guías proporcionadas para el trabajo, el primer paso fue sencillo de crear, importar librerías y crear un objeto tipo socket que sea el que administre la partida.

Hasta ahí no fue difícil avanzar sin embargo comenzaba a complicarse, lo siguiente que hice fue crear la lógica del juego, hice que se generara una letra aleatoria del alfabeto inglés y creé un diccionario con las categorías a rellenar, también implementé algunas funciones auxiliares cómo mostrar tablero o finalizar partida, las cuales fui modificando poco a poco respecto a lo que veía necesario.

Ahora empezaba el verdadero desafío, creada la conexión con los clientes comenzaba el hecho de gestionar a estos mismos, lo que hice es que cada vez que un cliente se unía al servidor, creaba un hilo de proceso que apuntaba a una función llamada manejar cliente, donde la función principal de esta era escribir en el tablero, al estar utilizando hilos y ser el tablero un recurso compartido utilice un mutex para que no hubiese conflictos.

Por último acabé de configurar la lógica e interfaz para el usuario para que se sintiera más cómodo de jugar, también añadí la función de ganar, el que se pudiera perder si pasan más de 2 minutos y que un usuario no pudiese tener una categoría bloqueada por más de 5 segundos.

CREACIÓN CLIENTE

Tras crear el archivo del servidor me puse con la parte del cliente, y que, aunque pueda parecer más simple, esta fue un verdadero quebradero de cabeza por los puntos que más tarde desarrollaré en profundidad.

Dicho esto, comencé con la creación del socket de cliente, lo cual no supuso nada de dificultad, el problema comenzó a llegar con el hecho de que los clientes deben tanto escuchar cómo mandar información al servidor, y aunque intenté hacer este proceso sin utilizar hilos y hacerlo así de una forma iterativa me di cuenta de que esto no era posible dado que el servidor manda información a los clientes aunque ellos no hagan nada, dado que si otro cliente modifica el tablero esto debe ser notificado a todos los jugadores en la partida.

Prosiguiendo con la creación de enviar información me encontré con el problema de que al haber creado un demonio que estuviera escuchando y monopolizando toda la información que enviaba el servidor, al enviar la categoría que el usuario desea rellenar, no podía recibir la información del servidor de sí esa categoría existía o no, por lo que si enviaba la palabra el programa dejaba de funcionar correctamente y lo hacía muy confuso ya que el cliente te pedía que introdujeras la palabra y el servidor estaba esperando la categoría (por ejemplo), así que mi solución fue intentar que el usuario interpretara que primero se metia la categoría y luego la palabra.

Por último al enviar la categoría erróneamente me aparecia el error de que el servidor lanzaba dos veces el mensaje de “Categoría no válida.” sinceramente sigo sin saber por que sucedia ya que esto se solucionó al meter una pequeña pausa entre mensaje y mensaje, mi teoría es que al estar usando hilos estos podían estar desincronizados y que al esperar usando time.sleep, esto le daba el suficiente tiempo para que no fallara.

Tras todos estos problemas el cliente estaba lo suficientemente correcto cómo para poder disfrutar de una partida en localhost con varios jugadores, sin embargo yo quería darle una vuelta de tuerca más, e intenté utilizar la API creada en la práctica 1 para que el juego pudiera crear varias partidas automáticamente y jugar desde la web.

INTEGRACIÓN API

Teniendo ambos programas separados únicamente debía fusionarlos con lo aprendido en la primera práctica, busqué cómo hacer que al ejecutar un archivo desde la terminal pudieras pasar varios parámetros a dicho programa, cosa que con la librería sys fue relativamente fácil de hacer utilizando el comando sys.argv.

Lo siguiente era hacer que cuando se mandase una consulta a la API para crear una partida, esta automáticamente ejecutara el archivo servidor.py pasandole el host y el puerto donde esta partida se pueda alojar, si bien con la librería os pude conseguir que ejecutara comandos sigo sin ser capaz de hacer que estas partidas sean accesibles por los usuarios. Mi teoría es que al estar utilizando supervisor para que la API siempre esté en funcionamiento estos dos programas puedan estar entrando en conflicto y por ello no funcione correctamente.

CONCLUSIÓN

Si bien este trabajo me ha parecido el más complejo de todos, creo que me ha enseñado librerías y conceptos que se pueden aplicar en cualquier lado, por ejemplo el uso de hilos agiliza mucho el rendimiento de cualquier programa si eres capaz de utilizarlos correctamente.

Sin embargo, me voy con una sensación agridulce dado que me hubiese gustado desarrollar el juego cómo yo lo tenía pensado desde un principio, y cuanto más avanzaba más me daba cuenta de la dificultad desmedida de este, sigo con dudas de si lo que en un principio pensé en hacer es siquiera posible sin entrar en temas de desarrollo web y otros lenguajes cómo javascript o HTML.