# PROYECTO: SISTEMA DISTRIBUIDO DE PRÉSTAMO DE LIBROS

ABRIL CANO CASTRO
JUAN MANUEL DURAN RUEDA
JUAN DAVID ROBLEDO GARZÓN
ANGEL DAVID TALERO PEÑUELA

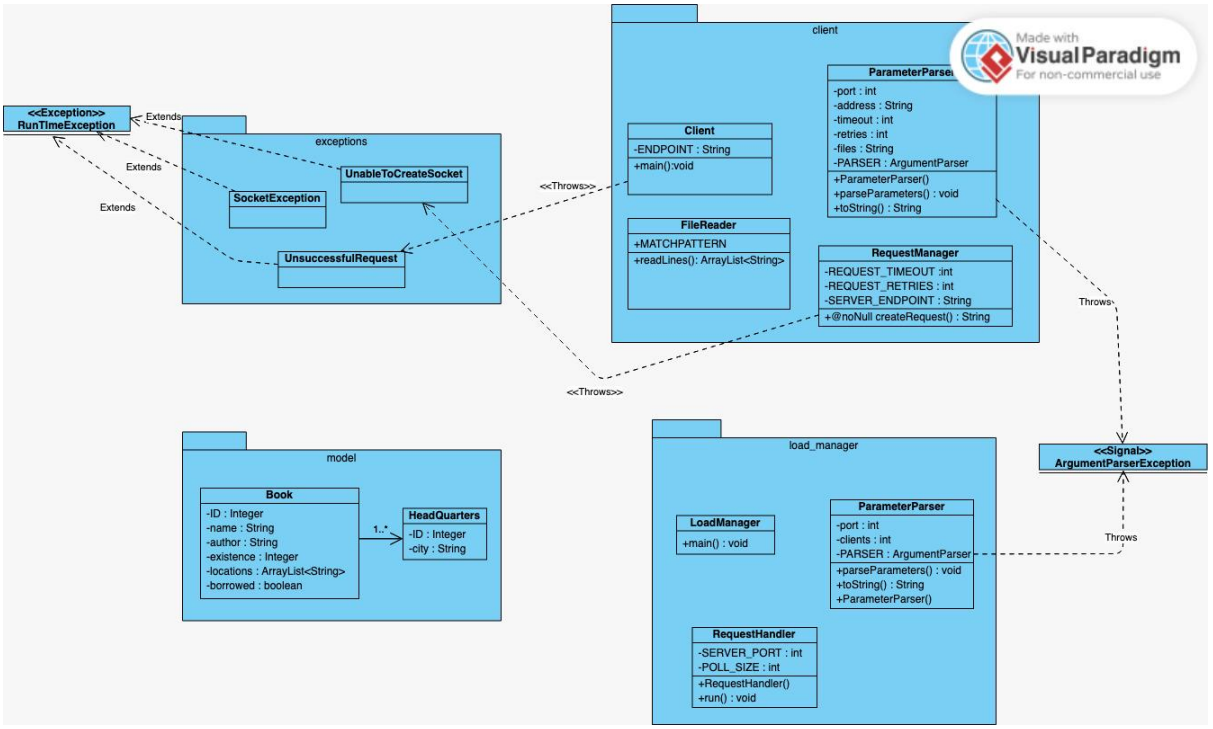
Entregado a
Mariela Curiel Huerfano


PONTIFICIA UNIVERSIDAD JAVERIANA
BOGOTÁ D.C
2023

# I. Diagramas de Diseño

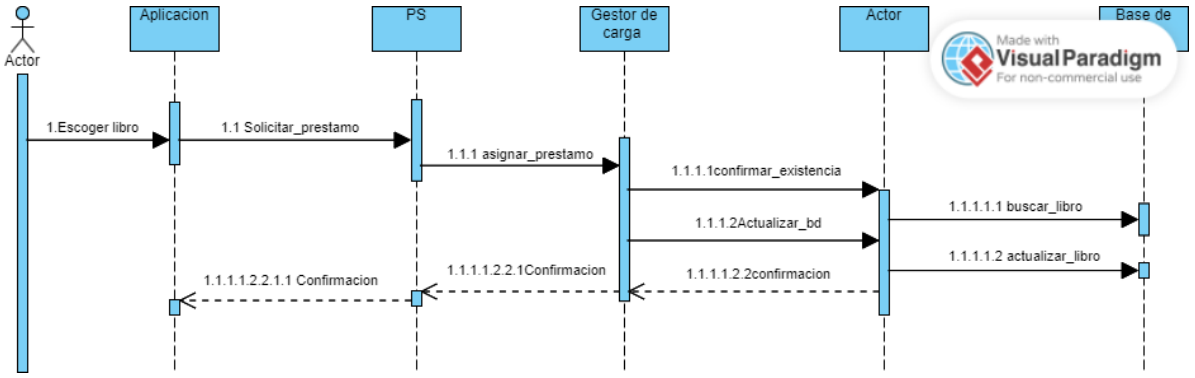
## 1.1 Diagrama de clases:

Se muestra en el diagrama de clases el empaquetamiento de las clases junto a los métodos utilizados por cada una. La carpeta de excepciones, como es natural, no cuenta con atributos ni métodos, ya que se extienden a las excepciones de tipo runtime.
Se muestran las demás clases y las excepciones que pueden arrojar.

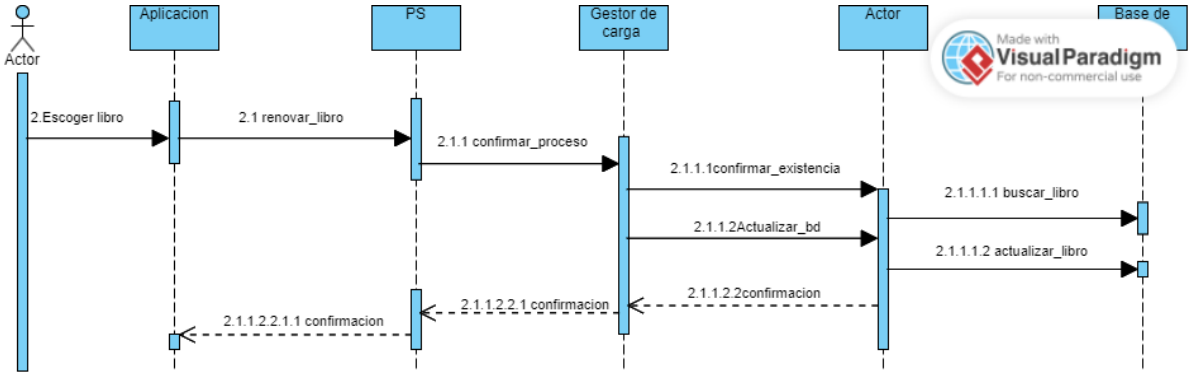


## 1.2 Diagramas de secuencia:

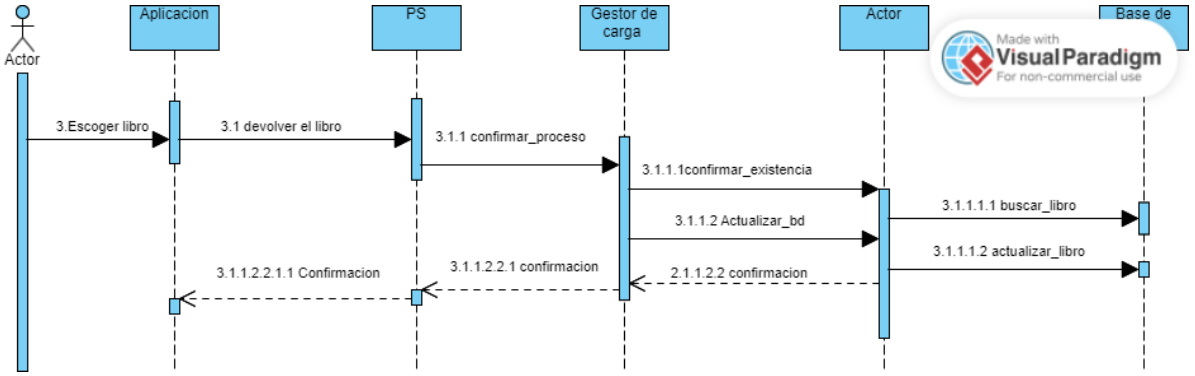
1. Solicitar libro:




2. Renovar libro




3. Devolver libro



# II. Pruebas

## 2.1 Protocolo de pruebas

Para la evaluación del funcionamiento y rendimiento del sistema se propone utilizar “Apache JMeter”, herramienta que permite someter el desarrollo a diferentes escenarios y cargas. De esta manera se puede observar cómo este reacciona bajo diferentes condiciones y comprobar sus propiedades de calidad.

1. Pruebas de Carga
    a. Las pruebas de carga permiten evaluar si el sistema puede manejar el volumen de tráfico esperado y cómo este se comporta ante el ingreso de múltiples niveles de procesos o clientes. Dentro del planteamiento de este proyecto se proponen hasta 15 procesos concurrentes en las dos sedes por lo que se deberán generar estás cargas pero a su vez se pueden realizar pruebas con un mayor número de procesos para revisar el comportamiento del sistema en estas condiciones. Realizar este tipo de pruebas permitirá identificar falencias en el funcionamiento del proyecto tal como cuellos de botellas, pero a su vez que tanto el gestor de cargas como las bases de datos y sus réplicas al igual que los actores estén cumpliendo con sus requerimientos adecuadamente.

2. Pruebas de Estrés
    a. Las pruebas de estrés pretenden llevar el sistema por encima de los límites de la carga que pueda soportar el sistema. De esta manera podemos identificar la carga máxima del servidor antes del fallo, el comportamiento de este ante esta eventualidad y así estimar cuando es necesario redirigir cargas para evitar el colapso del sistema.
3. Pruebas de Rendimiento
    a. Las pruebas de rendimiento tienen en cuenta las pruebas anteriores para evaluar de manera general el funcionamiento total del sistema y como este se comporta en diferentes escenarios. Esto permitirá concluir el funcionamiento esperado y correcto o no de los diferentes componentes del sistema.

| Prueba | Objetivo |
| --- | --- |
| Múltiples procesos envían solicitudes al sistema de manera simultánea | ● Verificar el correcto funcionamiento del sistema<br>● Identificar potenciales cuellos de botella<br>● Verificar las respuestas de los componentes del sistema |
| Falla de uno de los gestores de carga | ● Verificar la capacidad del sistema de redirigir cargas<br>● Comprobar la capacidad de recuperarse ante esta eventualidad<br>● Verificar la disponibilidad de este sistema ante una falla |
| Evaluar la consistencia del sistema ante los cambios | ● Verificar que la información en las bases de datos y sus réplicas sea consistente y verídica ante los cambios realizados por los procesos. |

## 2.2. Pruebas en JMeter:

Se realizaron pruebas dentro de una red local, con el servidor configurado para atender 15 solicitudes concurrentes, teniendo en un plazo de 15 segundos, con una cantidad máxima de reintentos de 3. Se estableció como un valor objetivo de 5000 solicitudes, sin embargo, no se pudo llegar a este valor, obteniendo un número máximo de hilos de 953.
Para el experimento en el que solo se envió un request por cada uno de los clientes, sin necesidad de hacer ningún re-intento.

*Se ejecutaron dos pruebas:*
1. El cliente envía una única petición a un servidor alojado en una red local
2. Los clientes se ejecutan en el mismo computador que el servidor para hacer control sobre las variables de red como: ancho de banda, latencia, etc.

## 2.3. Análisis de Resultados:

Para el análisis de los resultados se tuvieron en cuenta las siguientes variables:

## Hilos vs Tiempo de Vida:

Hace referencia al tiempo que transcurre desde que JMeter crea el hilo cliente hasta que el hilo finaliza, no está necesariamente relacionada al tiempo de respuesta pues depende de otras variables como los algoritmos de planificación del SO.

## Cantidad de Clientes en el Tiempo:

Número de hilos concurrentes realizando peticiones a través del tiempo

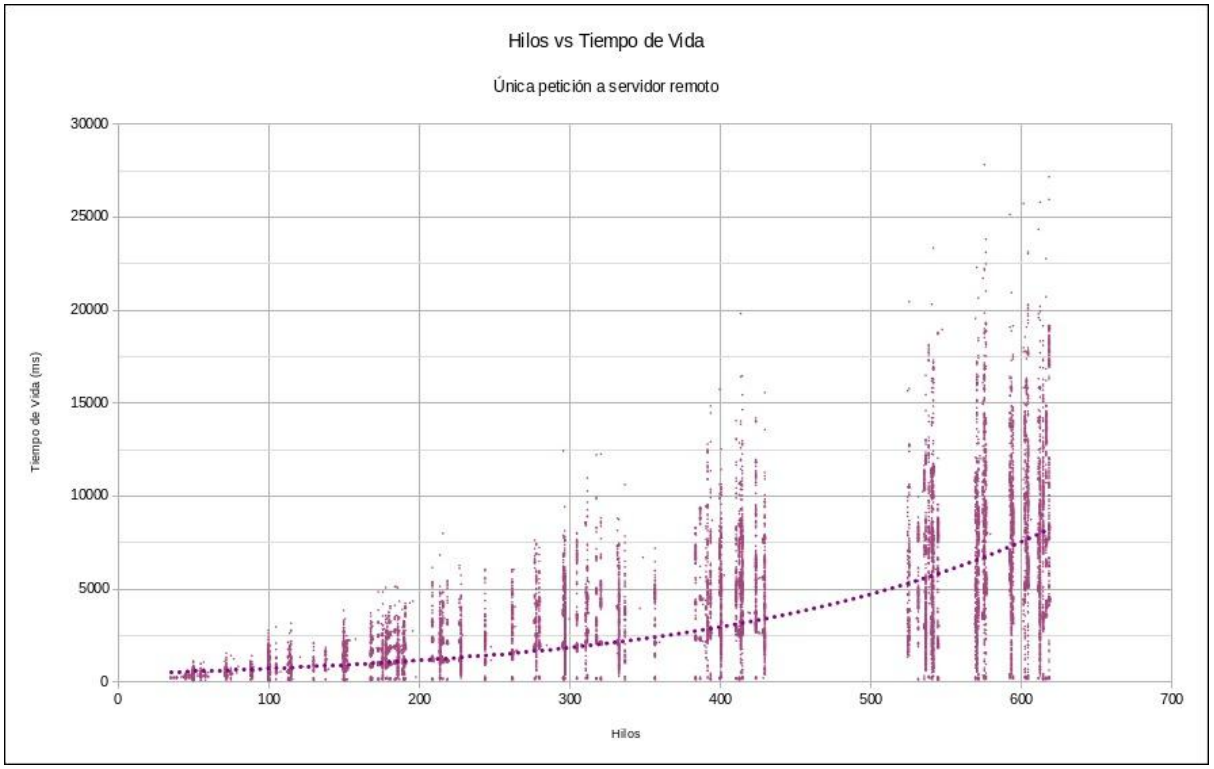
## Número de Hilos vs Tiempo de Respuesta:

Tiempo que tarda el servidor central en atender la petición de un cliente en función de la cantidad de clientes concurrentes

## Throughput vs Hilos:

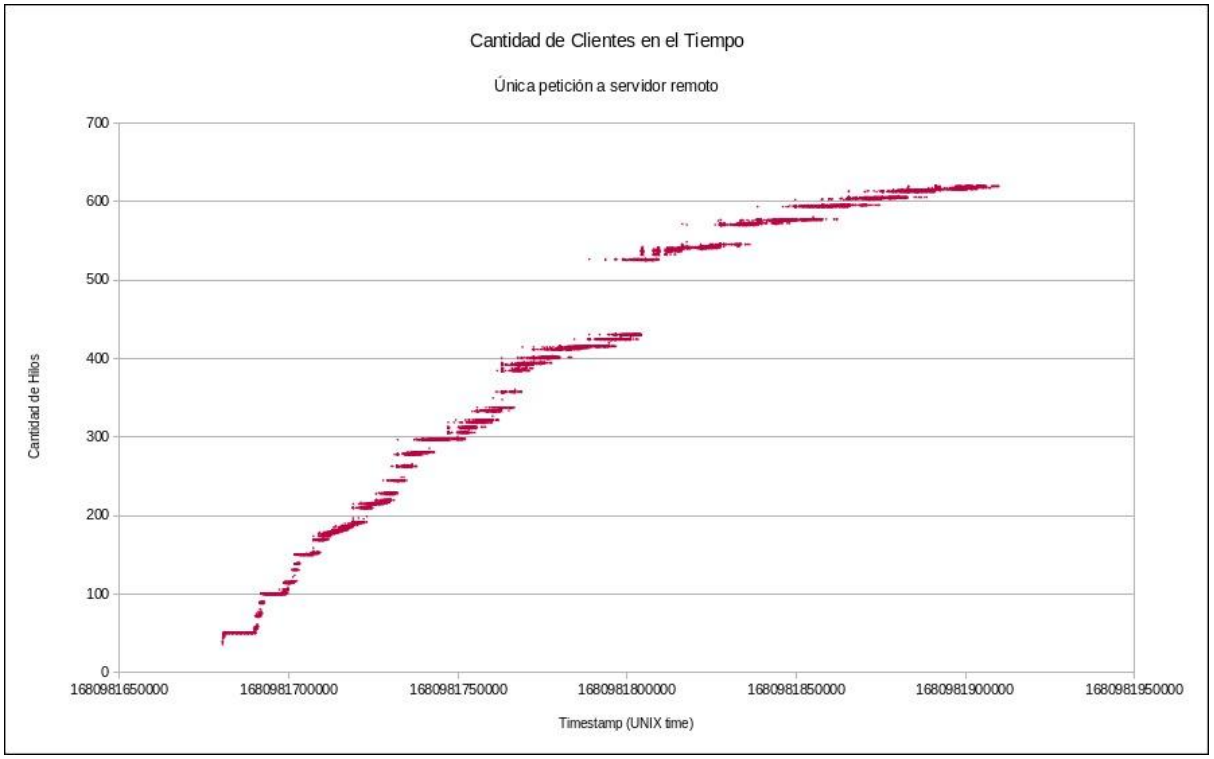
El *Throughput* hace referencia a las "Unidades de trabajo procesadas por unidad de tiempo", es decir, la cantidad de clientes en un segundo, que es capaz de atender el servidor

**Experimento en dos computadores a través de una red local:**

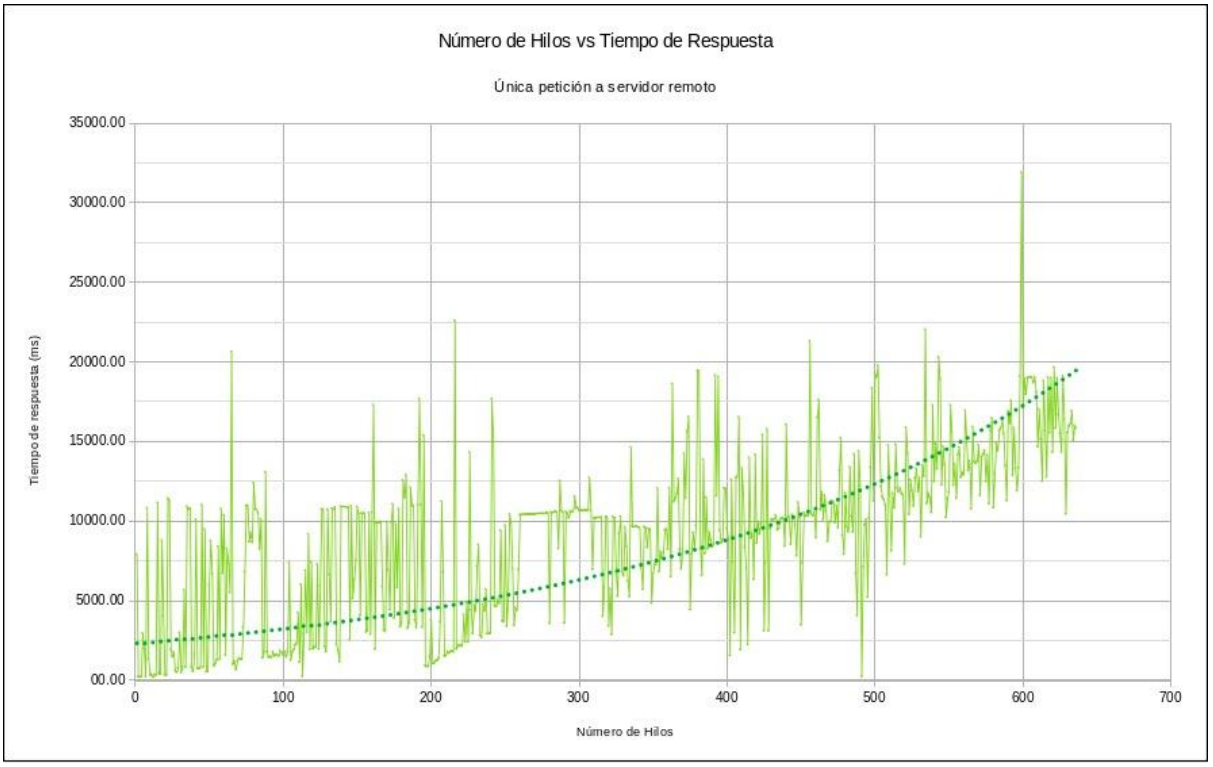



*Tiempo de vida de los hilos(clientes) vs cantidad de clientes*

- Tiempo de vida mínimo de 53 milisegundos y un máximo de 55 segundos.

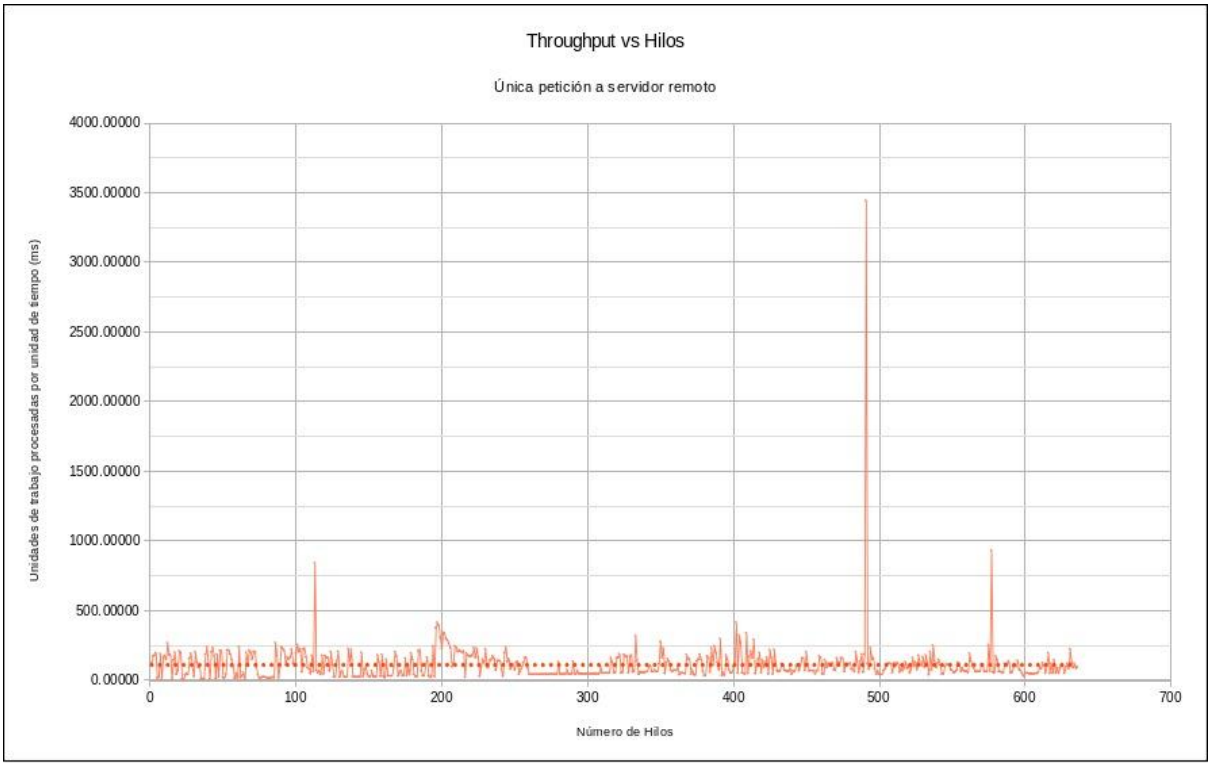



*Cantidad de hilos en el tiempo*




*Tiempo de respuesta del servidor en función de la cantidad de hilos*

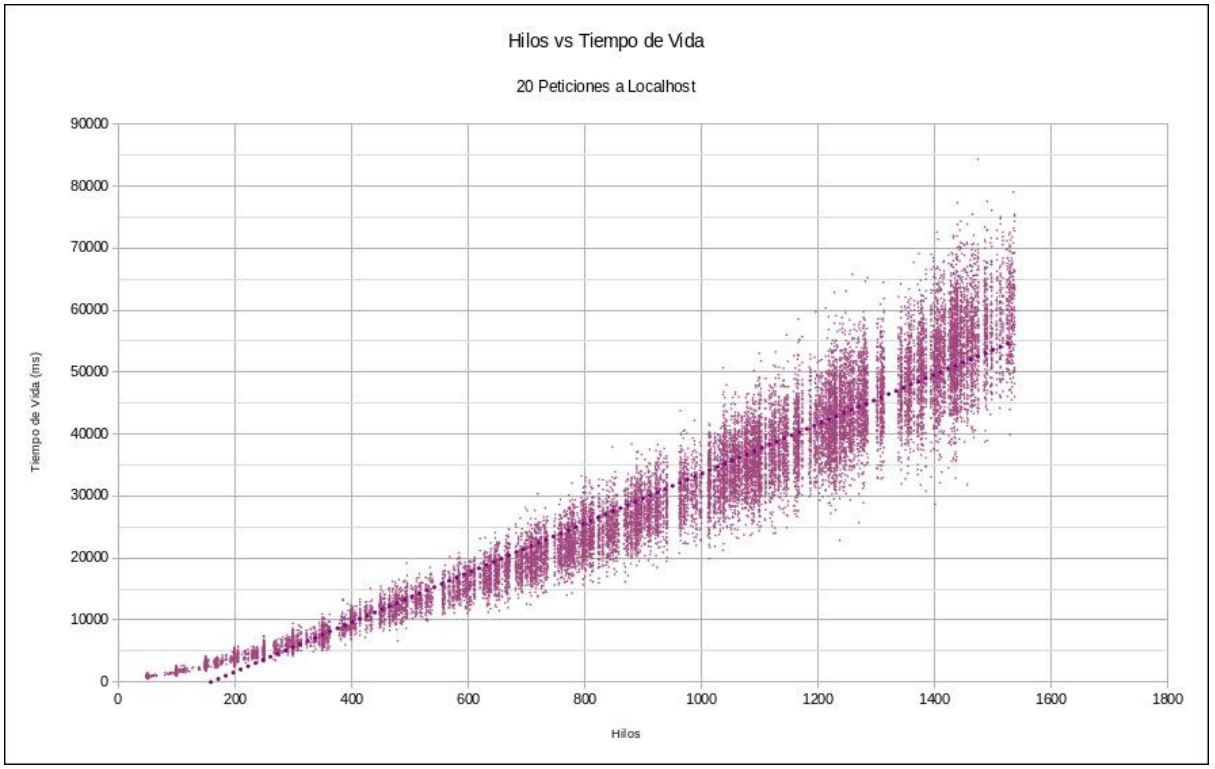
- Se obtuvo un tiempo máximo 32 segundos y mínimo 134 milisegundos




*Número de clientes que atiende el servidor vs cantidad de hilos generados*

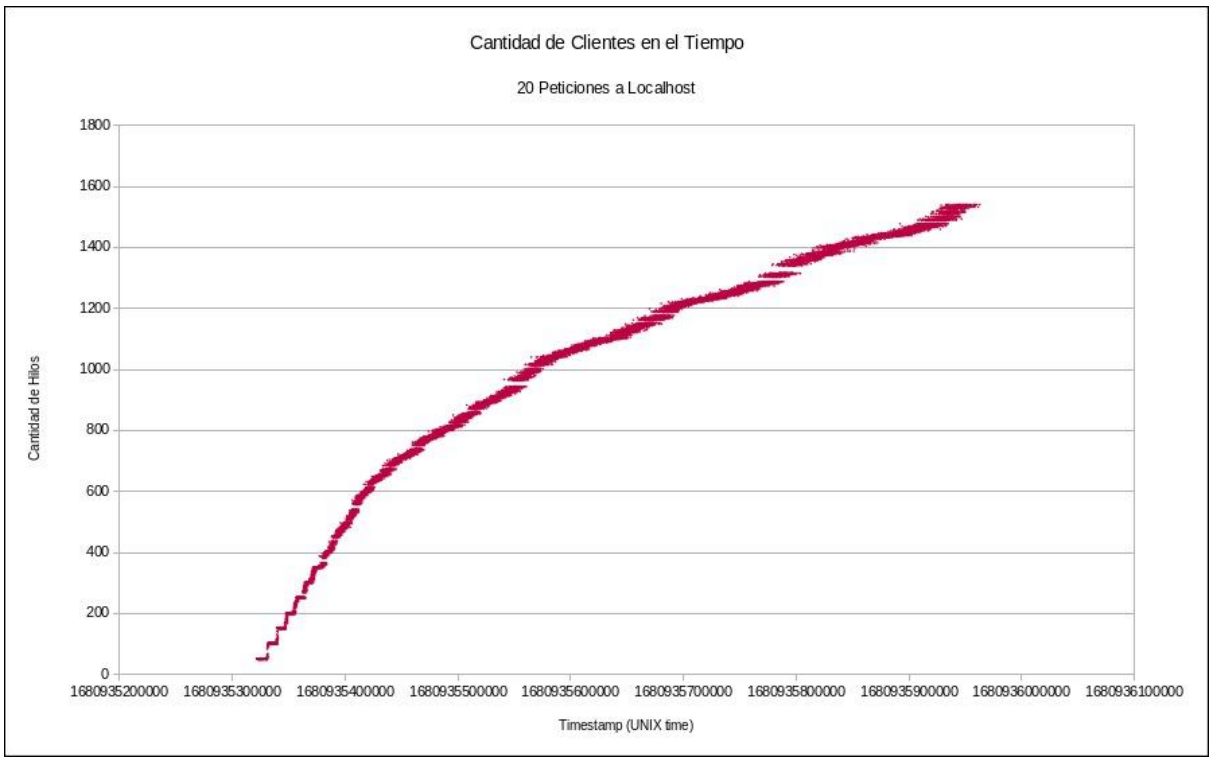
- Promedio de atención: 113 clientes por segundo.

**Experimentos en un solo computador (localhost):**

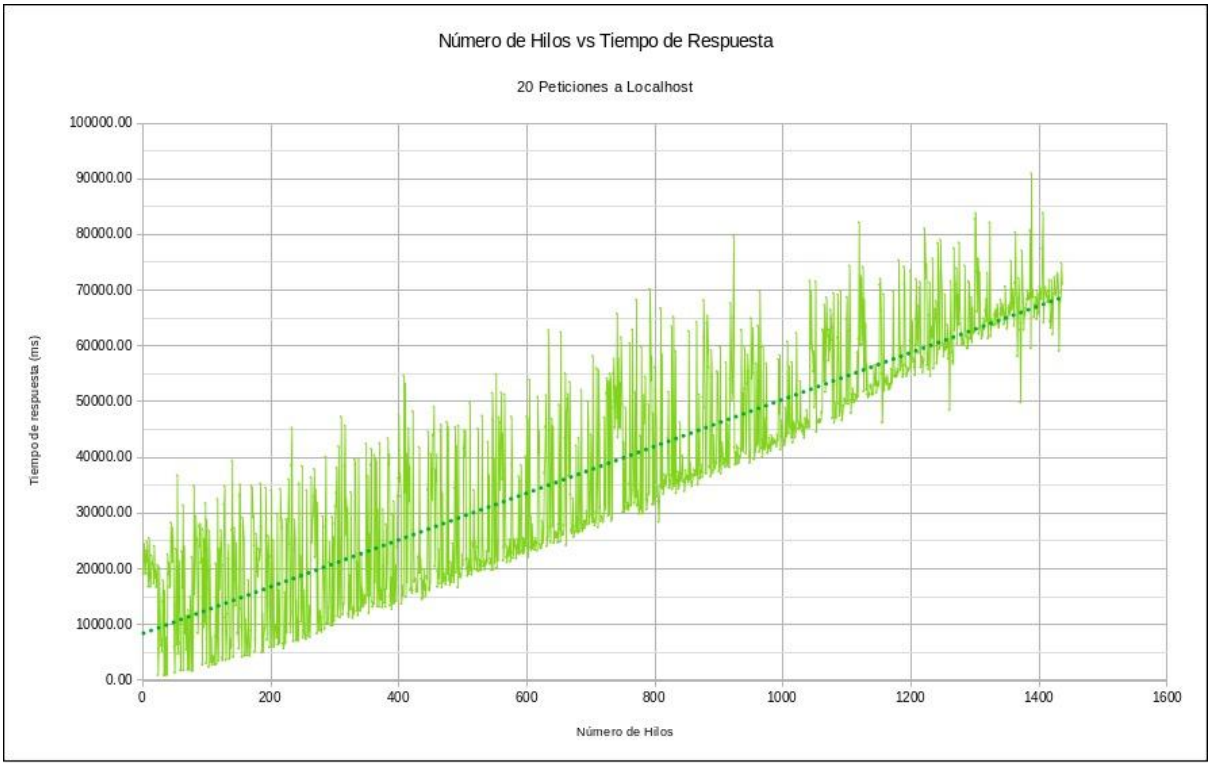



*Tiempo de vida de los hilos(clientes) vs cantidad de clientes*

- Se obtuvo un valor máximo de 1,7 segundos y un valor mínimo de 13 milisegundos.

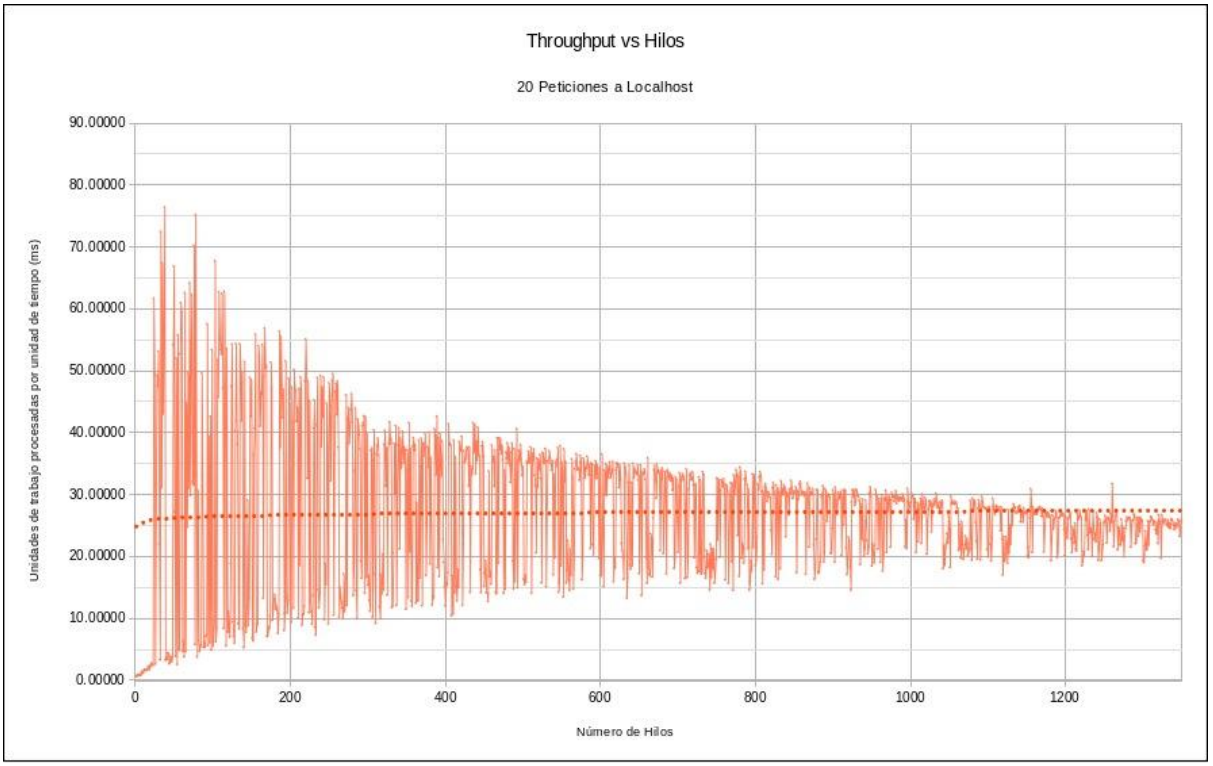



*Cantidad de hilos en el tiempo*




*Tiempo de respuesta del servidor en función de la cantidad de hilos*

- Se obtuvo un tiempo máximo 4,5 segundos y mínimo 40 milisegundos




*Número de clientes que atiende el servidor vs cantidad de hilos generados*

- Promedio de atención: 27 clientes por segundo.

En este caso, la cantidad de clientes atendidos por segundo es menor debido a que se deben crear los hilos al tiempo que se ejecuta el servidor por lo que hay menos recursos a su disposición.

# Conclusión:

Como puede ser evidenciado en las gráficas, aunque exista una gran dispersión en los datos, el tiempo que tarda el servidor en atender las solicitudes escala aproximadamente de manera lineal con la cantidad de clientes concurrentes, sin embargo; en un entorno de red real, esta escala puede llegar a ser exponencial pues existen otras variables que influyen en el resultado como lo son la saturación de la red, ancho de banda o latencia en la comunicación.