Instituto Politécnico Nacional

Escuela Superior de Cómputo

Sistemas Distribuidos

Pineda Guerrero Carlos

Tarea 1

4CV13

Humberto Alejandro Ortega Alcocer (2016630495)

14 de Marzo del 2023

Índice

Objetivo

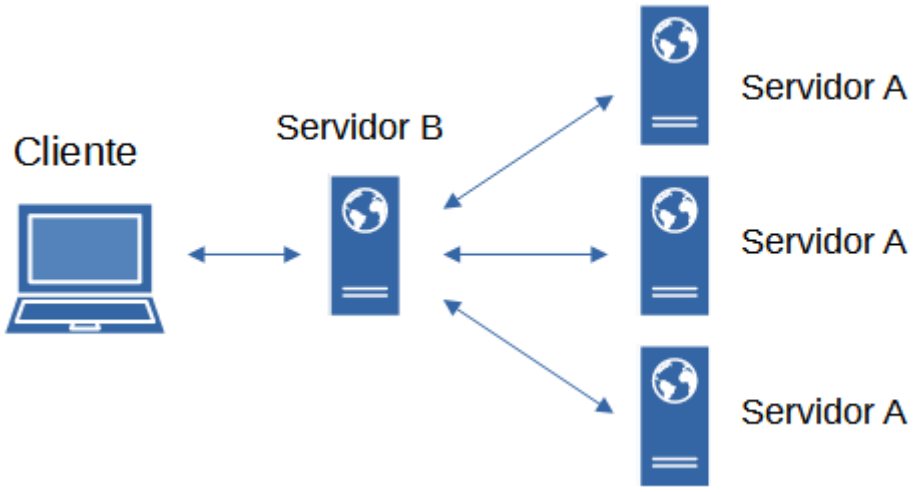
Desarrollar un sistema distribuido en Java de acuerdo a las siguientes especificaciones:

1. Desarrollar un servidor TCP (servidor A) multithread el cual recibirá tres números enteros de 64 bits (NÚMERO, NÚMERO_INICIAL y NÚMERO_FINAL).

El servidor A deberá dividir el NÚMERO entre cada número n desde NÚMERO_INICIAL hasta NÚMERO_FINAL, si hay un número n que divide a NÚMERO, entonces el servidor A regresará al cliente la palabra "DIVIDE", de otra manera regresará al cliente "NO DIVIDE".

Se sugiere utilizar el operador módulo % para determinar si un número n divide a NÚMERO, esto es:

Si NÚMERO%n==0 entonces n divide a NÚMERO



2. Desarrollar un servidor TCP (servidor B) el cual brindará el servicio de determinar si un número entero mayor que 1 es primo o no.

Utilizando el programa cliente que se describe en el inciso 3, se enviará al servidor B un número entero de 64 bits.

Para determinar si NÚMERO es primo, el servidor B dividirá el intervalo [2,NÚMERO-1] en tres intervalos, entonces el servidor B se conectará a tres instancias del servidor A. El servidor B enviará, a cada instancia del servidor A, NÚMERO y el intervalo [NÚMERO_INICIAL, NÚMERO_FINAL] correspondiente.

Cada instancia del servidor A deberá ejecutar en una ventana independiente de Windows o Linux.

Si las tres instancias del servidor A regresan "NO DIVIDE" al servidor B, entonces el servidor B regresará al cliente las palabras "ES PRIMO". Si alguna instancia del servidor A regresa "DIVIDE", entonces el servidor B regresará al cliente las palabras "NO ES PRIMO".

Por ejemplo, si el servidor B recibe el número 1234567811, entonces el servidor B enviará los siguientes números a las instancias:

Instancia 1:

```
1234567811,2,411522603
```

Instancia 2:

```
1234567811,411522604,823045206
```

Instancia 3:

```
1234567811,823045207,1234567810
```

Entonces cada servidor A regresará "NO DIVIDE", por tanto, el servidor B regresará al cliente "ES PRIMO".

Se sugiere utilizar las siguientes fórmulas para obtener los tres intervalos.

Sea K=NÚMERO/3 (la división es entera), entonces:

Intervalo	NÚMERO INICIAL	NÚMERO FINAL
1	2	K
2	K+1	2*K
3	2*K+1	NÚMERO-1

3. Desarrollar un cliente TCP que reciba como parámetro un número entero de 64 bits y lo envíe al servidor B, entonces el cliente deberá recibir del servidor B una string. El cliente desplegará la string que recibió del servidor B.

Desarrollo

Servidor A

Para comenzar con el desarrollo del sistema, lo primero será implementar el servidor A, el cual podemos caracterizar de la siguiente manera:

- En paralelo, un hilo por cada cliente.
- El algoritmo es secuencial.
- El socket debe cerrarse una vez finalizada la ejecución.
- Su único argumento será el puerto dónde estará escuchando.

Así, su ejecución es simple pues sólamente deberá iniciar y esperar que algún servidor B se conecte enviando los datos requeridos (número, límite inferior y límite superior).

La ejecución individualmente de un servidor A es:

```
❯ javac servidorA.java && java servidorA 2021
- Número: 10 (2-3), resultado: DIVIDE
- Número: 13 (2-4), resultado: NO DIVIDE
- Número: 188 (2-62), resultado: DIVIDE
- Número: 187 (2-62), resultado: DIVIDE
- Número: 137 (2-45), resultado: NO DIVIDE
- Número: -1 (2-0), resultado: NO DIVIDE
- Número: -10 (2--3), resultado: NO DIVIDE
```

Aquí estamos asignando el puerto 2021 a este servidor específicamente y los únicos *logs* que emite el programa son meramente informativos para tener un control visual del estado de la ejecución.

Servidor B

Ahora que tenemos el servidor A funcionando, para el servidor B lo primero que debemos hacer es caracterizarlo:

- Single-threaded (sin hilos por cliente)

- Hilos para las llamadas a los 3 servidores A.
- El algoritmo es secuencial hasta las 3 llamadas en paralelo a los servidores A.
- Sus argumentos son:
    - Puerto de ejecución (de sí mismo)
    - Puerto dónde se está ejecutando servidor A #1.
    - Puerto dónde se está ejecutando servidor A #2.
    - Puerto dónde se está ejecutando servidor A #3.

Para esto, lo primero será recibir los argumentos y almacenarlos, para hacerlo simple usaremos los datos de forma fija y asumiremos que los hilos siempre se crearán bajo la premisa de que los tres servidores de tipo A son indistintos en su precedencia.

Así, la ejecución del servidor B individualmente se verá de la siguiente forma:

```
❯ javac servidorB.java && java servidorB 2020 2021 2022 2023
- Número: 10
        - ServidorA 1: 2 - 3 , resultado: DIVIDE
        - ServidorA 2: 4 - 6 , resultado: DIVIDE
        - ServidorA 3: 7 - 10 , resultado: NO DIVIDE
- Resultado: NO ES PRIMO
- Número: 13
        - ServidorA 1: 2 - 4 , resultado: NO DIVIDE
        - ServidorA 2: 5 - 8 , resultado: NO DIVIDE
        - ServidorA 3: 9 - 13 , resultado: NO DIVIDE
- Resultado: ES PRIMO
```

Aquí se puede ver claramente que los argumentos son:
- Puerto del servidor B: 2020.
- Puerto del primer servidor A: 2021.
- Puerto del segundo servidor A: 2022.
- Puerto del tercer servidor A: 2023.

En el código se puede observar que las llamadas al servidor A se realizan en paralelo y posteriormente se espera su finalización.

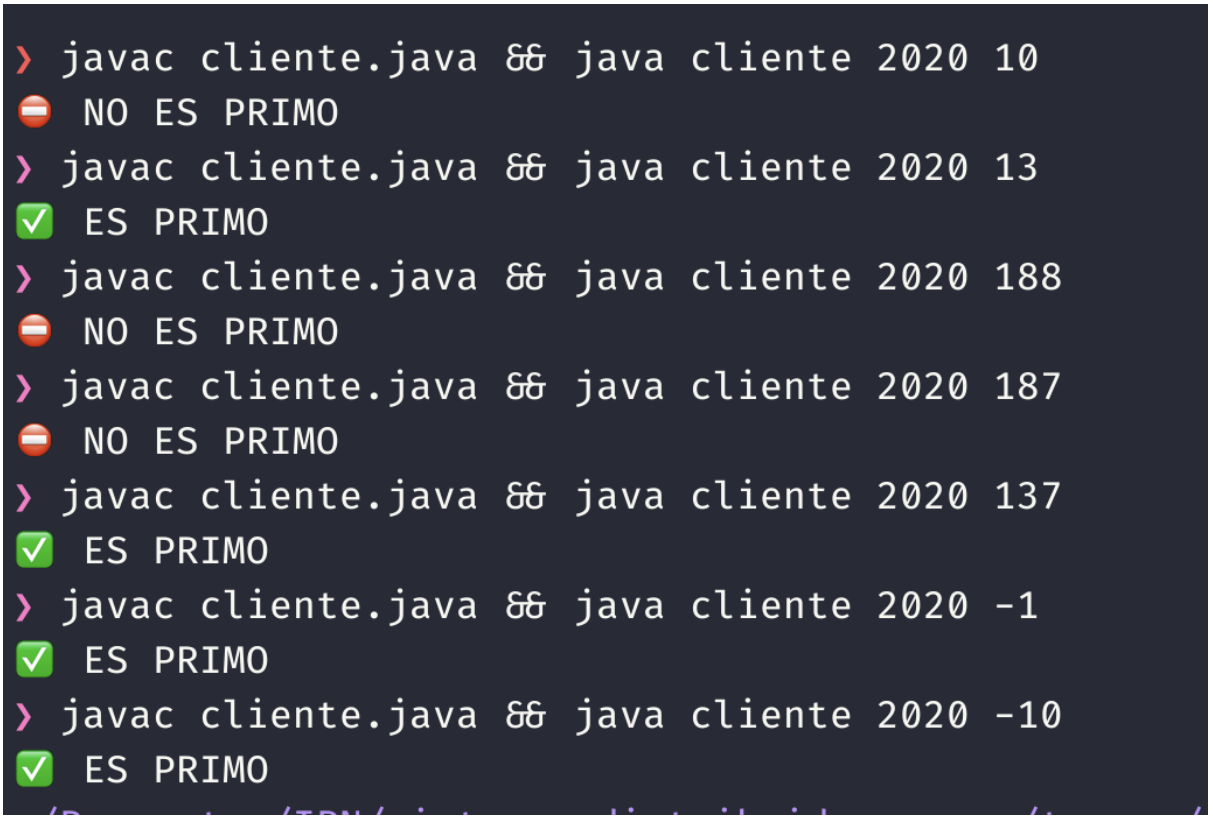
Cliente

En el caso del cliente, su caracterización es muy simple:

- Un solo hilo de ejecución.
- Dos argumentos:

- El puerto del servidor B.
- El número a verificar si es primo.

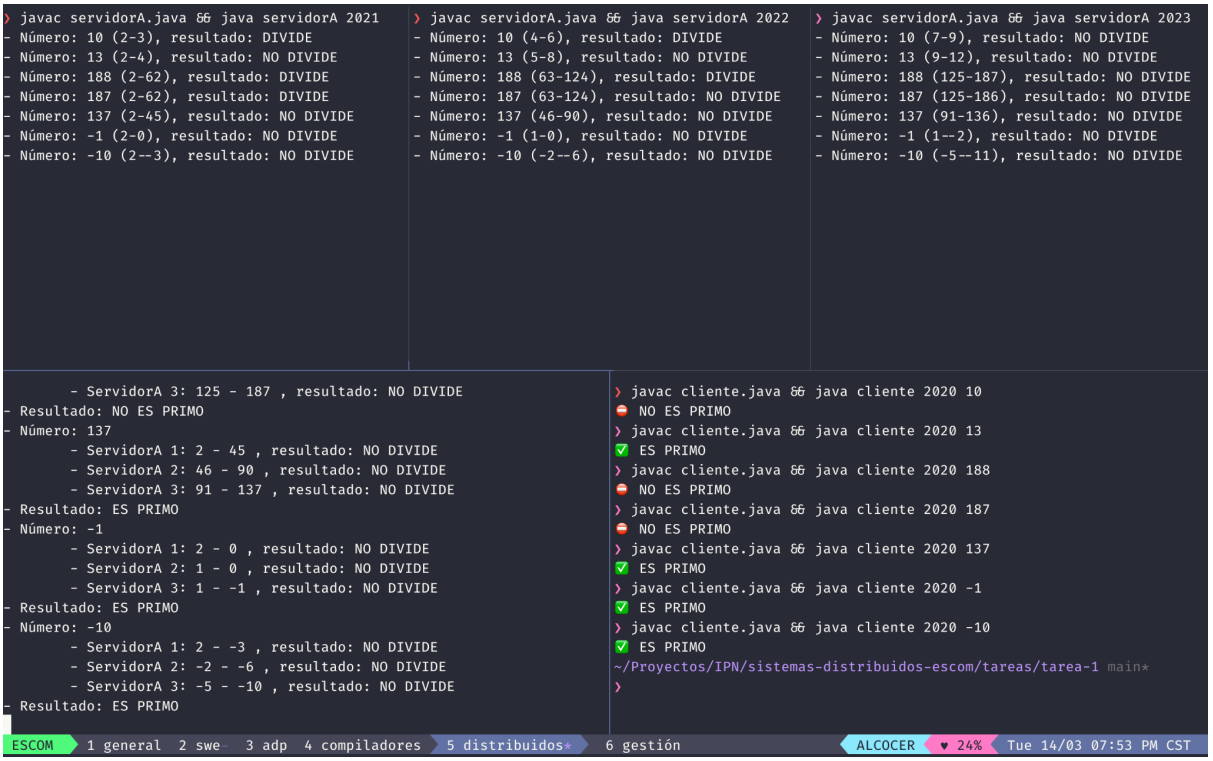
La ejecución del cliente individualmente se visualizará de la siguiente manera:



Se agregó un emoji al principio únicamente porque se veía más bonito, no afecta en nada la ejecución del programa.

Ejecución de Sistema

Finalmente, la ejecución en conjunto de los tres programas se visualiza de la siguiente manera:



Como se puede observar, el cliente (abajo derecha) envía el número al servidor B (abajo izquierda) quien determina los intervalos (basándose en la tabla proporcionada por el profesor) y hace las llamadas en paralelo a los tres servidores A (las tres ventanas superiores) los cuales verifican si el número es divisible entre el rango determinado por los límites proporcionados y responden al servidor B quién enviará la respuesta correspondiente al cliente.

Conclusión

Desarrollar sistemas distribuídos siempre es un reto, considerar todas las pequeñas partes y cómo interactúan entre sí no es tarea simple. En esta tarea pude desarrollar un sistema acorde a las especificaciones del profesor y logré comprender de mejor forma cómo funcionan varios métodos, librerías y herramientas para eficientar el desarrollo de la misma. Me gustó mucho que, para poder llevar a cabo el desarrollo, primero escribí un cliente que *específicamente* funcionaba para el servidor A, una vez que estaba seguro que el servidor A estaba bien, entonces hice el B y desarrollé el cliente final que se conectaba a este mismo, pero al mismo tiempo obtuve herramientas intermedias para poder diagnosticar y corregir cualquier error en el mismo. No soy una persona muy adepta a Java como lenguaje (prefiero C porque me permite tener más control y me parece didáctico), pero me gustó cómo pude implementar una solución que, de entrada pudiese parecer muy compleja, en pocas líneas de código y sin requerir de una complejidad muy elevada.

– Humberto Alejandro Ortega Alcocer.