PROYECTO ALPHA

Elaborado por: Abraham Martínez, Patricio Pizaña

1. Introducción

En este documento se presenta la evaluación experimental de desempeño del juego distribuido **"Wack a Mole"**, el cual consta de un servidor (`StressSender`) y múltiples clientes (`StressReceiver`). El juego se basa en:

- **Registro de jugadores** mediante **Sockets TCP**.
- **Comunicación de eventos** de aparición de monstruos vía **ActiveMQ** (JMS).
- **Golpes** enviados por los clientes también a través de **TCP**.

El objetivo es medir el **tiempo de respuesta**, la **desviación estándar** de dicho tiempo, el **tiempo de registro**, la **desviación estándar** del registro y el **porcentaje de conexiones exitosas** bajo diferentes cargas de clientes simultáneos (50, 100, 150, 250 y 500). También se van a comparar entre un número distinto de puntos necesarios para ganar (5, 20).

2. Definición del Experimento

1. **Entorno de Pruebas**
    a. **Servidor**: `StressSender` corre en la máquina con [Intel Core i7-10870H @2.20 GHz, 16 GB RAM, Windows 11 Home].
    b. **Clientes**: `StressReceiver` simula entre 50 y 500 jugadores, lanzados casi simultáneamente.
    c. **Red**: Se utiliza la misma máquina o una red local, según sea el caso. En este caso se probo con una red WiFi de 150 Mbps de subida y bajada.
    d. **Configuración del Juego**:
        i. Se requiere **5/20 golpes** (WIN_CONDITION=5/20) para ganar.
        ii. El servidor se detiene al finalizar 1 partida por ejecución (MAX_GAMES=1).
2. **Métricas Recolectadas**
    a. **Tiempo de Registro (ms)**: Tiempo transcurrido desde que un cliente abre el socket hasta que envía su nombre y el servidor lo registra.
    b. **Tiempo de Reacción (ms)**: Tiempo entre que el cliente decide golpear (marca el `System.currentTimeMillis()`) y el servidor procesa ese golpe.

c. **Desviación Estándar** de ambos tiempos (registro y reacción).
d. **Porcentaje de Conexiones Exitosas**: (Conexiones Efectivas / Conexiones Esperadas) * 100.

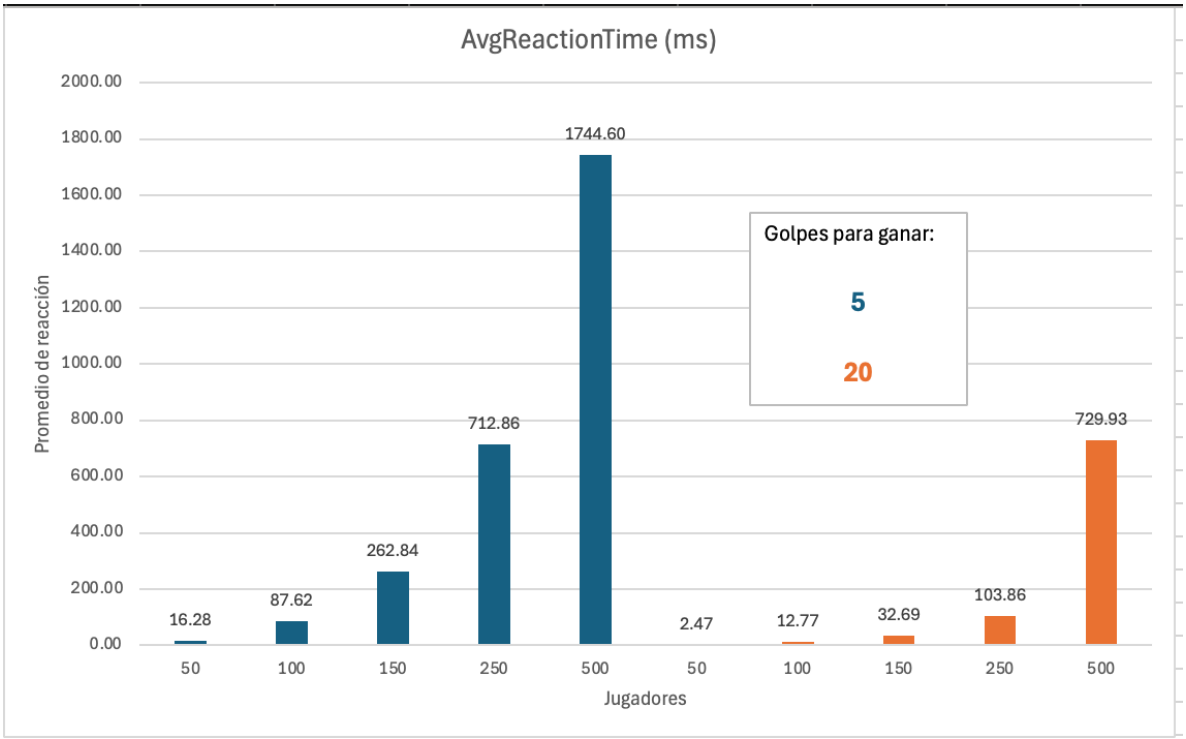
3. **Configuraciones de Carga**
   a. Se realizaron pruebas con **50, 100, 150, 250 y 500** clientes simultáneos.
   b. Cada configuración se ejecutó **10 veces** para obtener datos estadísticamente confiables.
4. **Procedimiento que se llevó a cabo**

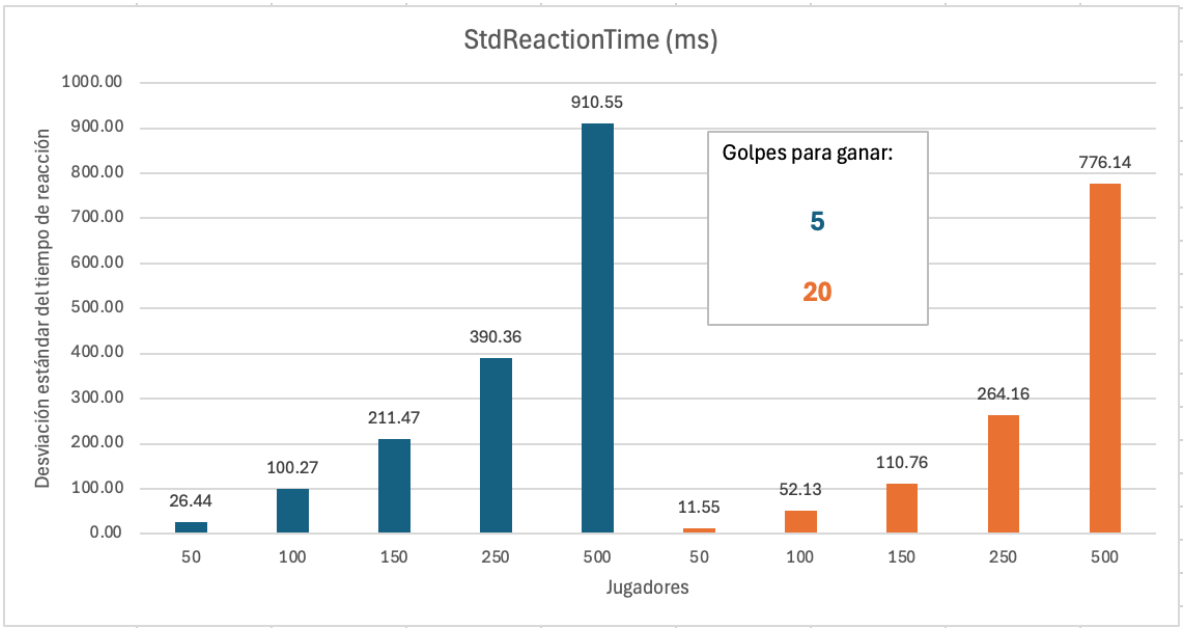
   a. **Ejecutar** el servidor con la instrucción:

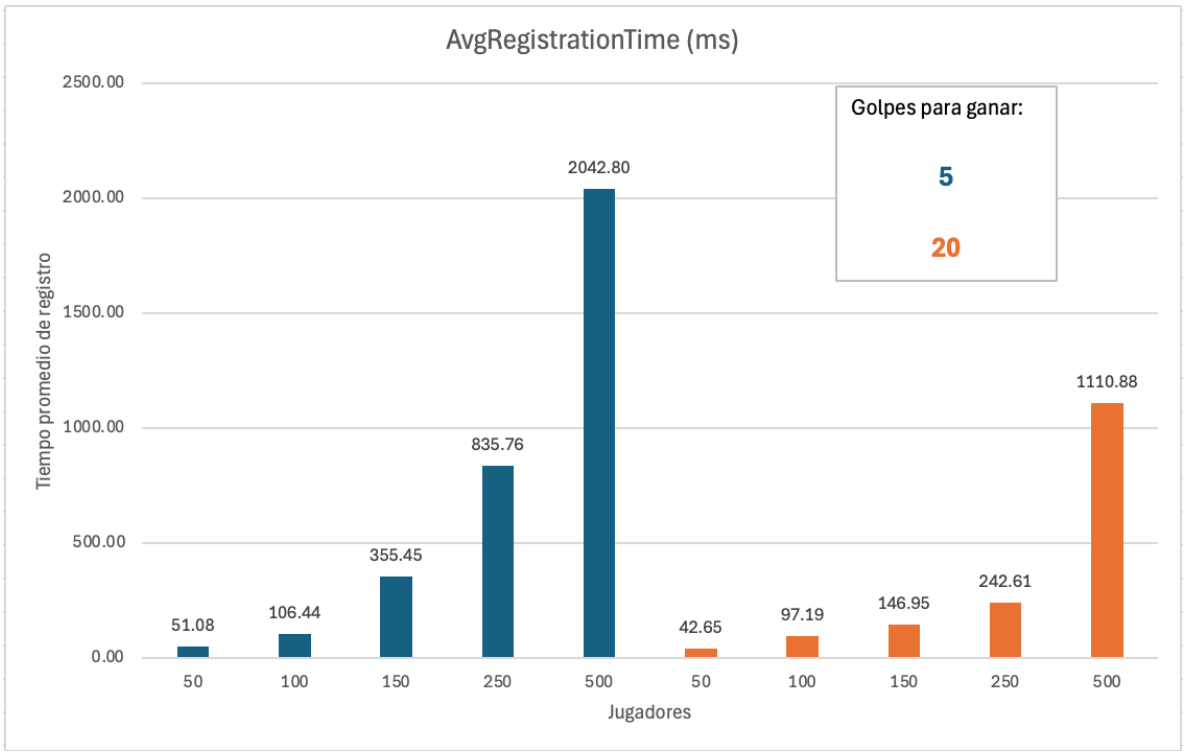
      ```
      java StressSender <numClientesEsperados>
      ```

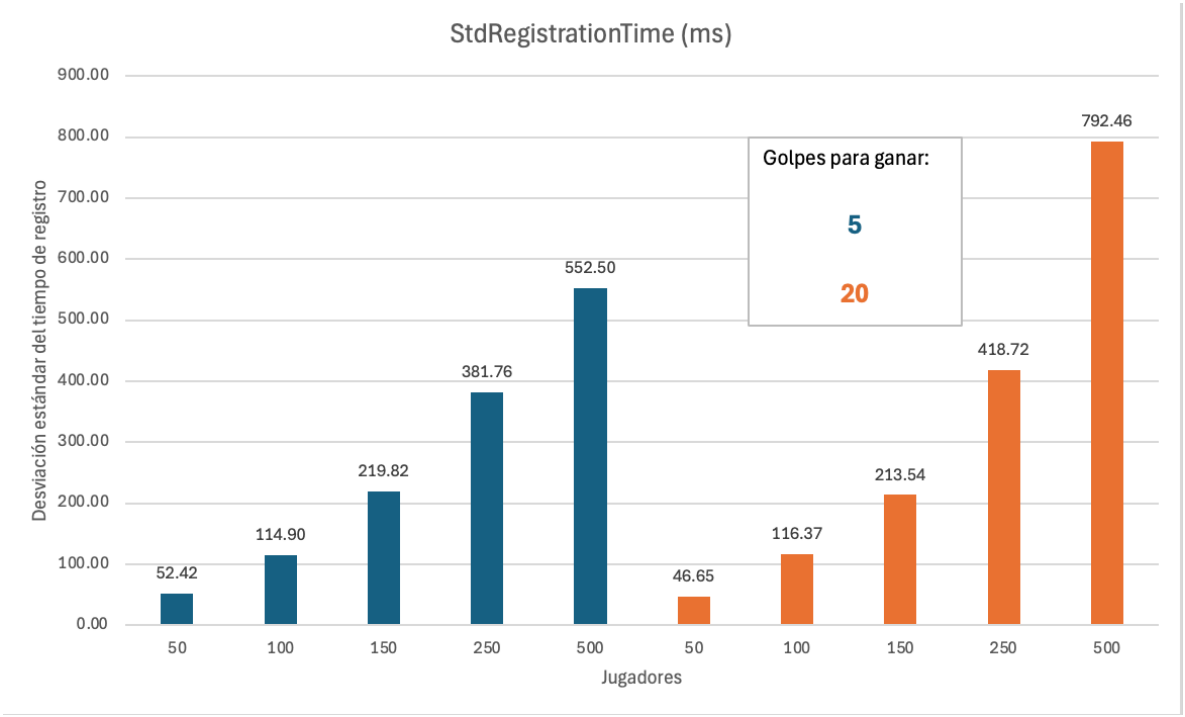
      donde `<numClientesEsperados>` es 50, 100, 150, 250 o 500.
   b. **Ejecutar** el programa `StressReceiver` con la cantidad de clientes (`NUM_CLIENTS`) correspondiente.
   c. Esperar a que **un jugador** llegue a 5 golpes (WIN_CONDITION).
   d. El servidor escribe los resultados de esa partida en un archivo CSV (`stress_results.csv`).
   e. Repetir el paso anterior **10 veces** por configuración.

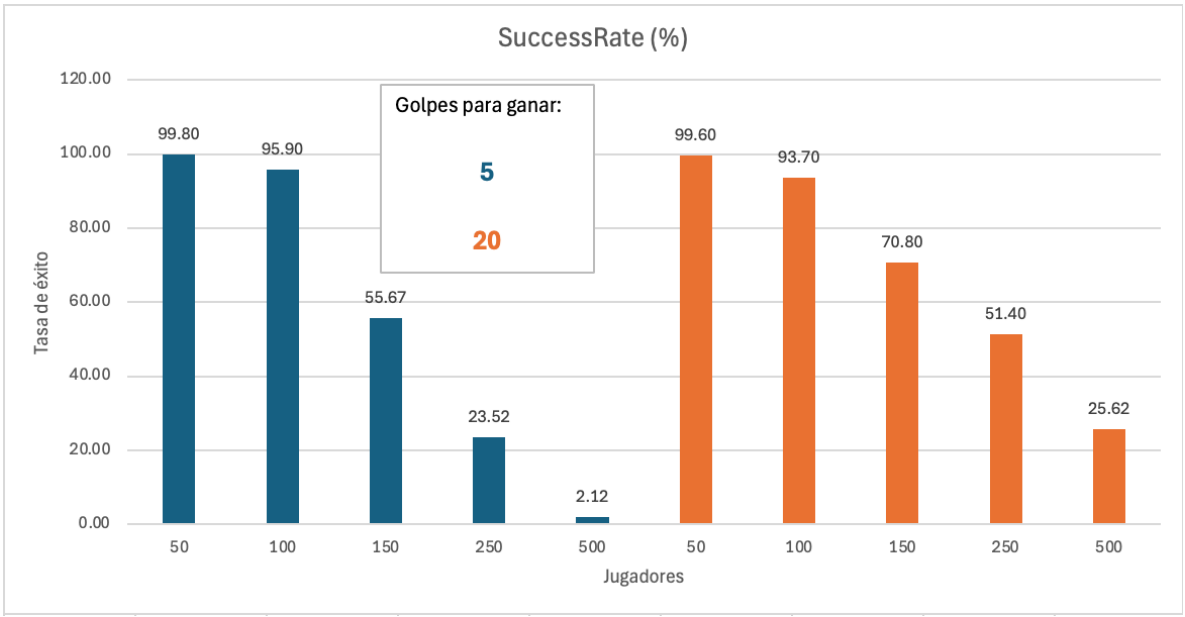
5. **Resultados**




StdReactionTime (ms)
Desviación estándar del tiempo de reacción
1000.00
900.00
800.00
700.00
600.00
500.00
400.00
300.00
200.00
100.00
0.00
26.44
100.27
211.47
390.36
910.55
11.55
52.13
110.76
264.16
776.14
50
100
150
250
500
50
100
150
250
500
Jugadores
Golpes para ganar:
5
20


AvgRegistrationTime (ms)
Tiempo promedio de registro
2500.00
2000.00
1500.00
1000.00
500.00
0.00
51.08
106.44
355.45
835.76
2042.80
42.65
97.19
146.95
242.61
1110.88
50
100
150
250
500
50
100
150
250
500
Jugadores
Golpes para ganar:
5
20


StdRegistrationTime (ms)
Desviación estándar del tiempo de registro
900.00
800.00
700.00
600.00
500.00
400.00
300.00
200.00
100.00
0.00
Golpes para ganar:
5
20
52.42
114.90
219.82
381.76
552.50
46.65
116.37
213.54
418.72
792.46
50
100
150
250
500
50
100
150
250
500
Jugadores


SuccessRate (%)
Tasa de éxito
120.00
100.00
80.00
60.00
40.00
20.00
0.00
Golpes para ganar:
5
20
99.80
95.90
55.67
23.52
2.12
99.60
93.70
70.80
51.40
25.62
50
100
150
250
500
50
100
150
250
500
Jugadores

6. Análisis e interpretación de resultados

Los resultados de las pruebas muestran cómo el desempeño del juego **"Whack a Mole"** se ve afectado por el número de jugadores y la cantidad de golpes requeridos para ganar. Comparando dos configuraciones, **5 golpes** y **20 golpes**, se observa que el servidor maneja bien hasta **100 jugadores**, pero más allá de **150 jugadores**, la carga comienza a degradar el rendimiento.

En la configuración de **5 golpes para ganar**, el **tiempo promedio de reacción** y su **desviación estándar** aumentan drásticamente en 250 y 500 jugadores, alcanzando **1744.6 ms** en 500 jugadores, indicando colapso del servidor. En cambio, con **20 golpes**, los tiempos son más estables y la degradación ocurre de manera más gradual. Esto sugiere que partidas más largas distribuyen mejor la carga de procesamiento.

El **tiempo de registro** sigue un patrón similar: con 5 golpes, el tiempo aumenta abruptamente en configuraciones altas, mientras que con 20 golpes el incremento es más progresivo. La **desviación estándar del registro** también es menor con 20 golpes, indicando tiempos más uniformes y predecibles.

El **porcentaje de conexiones exitosas** refleja el mayor problema de la configuración de 5 golpes. Aunque con **50 y 100 jugadores** se logran tasas de éxito cercanas al **100%**, en **500 jugadores la tasa cae a casi 0%**, ya que muchos clientes no alcanzan a registrarse antes de que alguien gane. En contraste, con **20 golpes**, la tasa de éxito es mayor, incluso en 250 y 500 jugadores, permitiendo que más jugadores participen antes de que termine la partida.

En conclusión, el servidor **escala bien hasta 100 jugadores**, pero con **más de 150**, la carga genera tiempos de respuesta altos y conexiones fallidas. Un **WIN_CONDITION de 20 golpes** mejora la estabilidad y permite mayor concurrencia sin saturación extrema. Si tenemos 5 golpes, hay muchos jugadores que no podrán ingresar al servidor ya que este termina (por cuestión de recolección de datos). Para soportar cargas más altas, sería recomendable optimizar la arquitectura, escalando horizontalmente o separando los servicios de registro y juego.