Tarea 3: Deployment y modificación HelloWorld

Informe sobre Mediciones de Atributos de Calidad de Rendimiento en Nuestro Sistema Cliente-Servidor

Presentado por:
- **Adri Martinez-A00400842**
- **Johan Guzmán-A00401480**

Introducción

En este informe, presentamos las mediciones de los atributos de calidad relacionados con el rendimiento de nuestro sistema de software cliente-servidor. Hemos automatizado las pruebas utilizando dos clientes y un servidor, ejecutando 5 iteraciones por comando para un total de 5 comandos: "hello", "23" (que calcula la secuencia de Fibonacci hasta 23), "listifs" (lista de interfaces de red), "listports 127.0.0.1" (lista de puertos abiertos en localhost) y "!ls" (lista de archivos en el directorio). Utilizamos un script en PowerShell (run_tests.ps1) para coordinar la ejecución, registrar logs y calcular métricas básicas, que hemos refinado y verificado con análisis adicionales.

Como equipo, hemos diseñado estas pruebas para evaluar cómo el sistema maneja solicitudes concurrentes y medir los subatributos de rendimiento definidos a continuación. Los resultados se basan en los logs generados (client1log.csv, client2log.csv, etc.), donde capturamos timestamps, comandos, respuestas, tiempos de procesamiento en el servidor (ServerTime) y tiempos de ida y vuelta (RTT). El tiempo total de las pruebas fue de aproximadamente 47.07 segundos, con 50 solicitudes procesadas exitosamente.

A continuación, recordamos la definición de los requisitos no funcionales y atributos de calidad, tal como los hemos considerado en nuestro análisis.

Requisitos No Funcionales: Atributos de Calidad

Cada sistema de software implementa un conjunto de funcionalidades, que idealmente corresponden al conjunto de requisitos funcionales solicitados por las partes interesadas. Sin embargo, este conjunto no es la única preocupación del software; las restricciones con las que estas funcionalidades deben ser ejecutadas y entregadas representan la lista de requisitos no funcionales que el sistema de software debe cumplir. Entre estos requisitos no funcionales se encuentran los atributos de calidad del software, como, por ejemplo, el plazo máximo de respuesta a cualquier solicitud. Los atributos de calidad son una preocupación importante en la ingeniería de software porque, a menudo, los sistemas deben ser rediseñados debido a deficiencias en estos atributos.

Los atributos de calidad relevantes para un sistema dado dependen de los requisitos, características y restricciones de cada sistema. Así, lograr un nivel de calidad esperado en cualquier atributo de calidad depende de cómo se implementan y diseñan las funciones

del sistema. Lograr los atributos de calidad también depende de la implementación y despliegue del software. Por ejemplo, el rendimiento depende de (i) la comunicación entre componentes, (ii) las funcionalidades de cada componente, (iii) la ubicación de los recursos compartidos, (iv) los algoritmos que implementan las funcionalidades, y (v) la codificación de estos algoritmos. A la luz de esto, existen muchos aspectos a considerar cuando se busca alcanzar un nivel específico de un atributo de calidad.

Rendimiento

El atributo de calidad de rendimiento se define como "El grado en que un sistema o componente cumple con sus funciones designadas dentro de las restricciones dadas, tales como velocidad, precisión o uso de memoria." Este atributo de calidad está relacionado con el tiempo que le toma a un sistema responder a los eventos. Existen muchos tipos de eventos, tales como interrupciones, mensajes, solicitudes de usuarios, entre otros. Los dos aspectos principales que hacen que este atributo de calidad sea complejo son: (i) la cantidad de fuentes de eventos y (ii) los patrones de llegada de los eventos. Ejemplos de fuentes de eventos son el sistema operativo y las aplicaciones de usuario. Los patrones de llegada de eventos pueden caracterizarse como periódicos, esporádicos o estocásticos. Los eventos periódicos tienen un patrón de repetición y posiblemente una fecha límite, mientras que los eventos estocásticos ocurren según una distribución probabilística; los eventos esporádicos son aquellos que no pueden clasificarse como periódicos ni estocásticos.

Debido a sus características, el rendimiento tiene aspectos del sistema que contribuyen a él. Estos aspectos son conocidos como subatributos o factores de rendimiento. Los subatributos de rendimiento a través de los cuales se puede medir concretamente son los siguientes:

1. **Rendimiento (Throughput)**: La cantidad de trabajo por unidad de tiempo.
2. **Tiempo de Respuesta (Response Time)**: El tiempo transcurrido entre la llegada del evento y la respuesta del sistema (también conocido como latencia), por ejemplo, el tiempo de carga de una aplicación, apertura de pantalla y tiempos de actualización, etc.
3. **Fecha Límite (Deadline)**: Restricción de tiempo para completar una tarea.
4. **Jitter de Respuesta**: Variación de la latencia.
5. **Tasa de Pérdida (Missing rate)**: Cantidad de eventos perdidos por unidad de tiempo.
6. **Tasa de No Procesados (Unprocess rate)**: Cantidad de eventos no procesados por unidad de tiempo.
7. **Tiempos de Procesamiento**: Funciones, cálculos, importaciones, exportaciones.
8. **Tiempos de Consultas y Reportes**: Cargas iniciales y cargas posteriores.

Metodología de Pruebas

Hemos automatizado las pruebas utilizando el script run_tests.ps1, que compila el proyecto, inicia el servidor, ejecuta dos clientes en paralelo en modo automático (5 iteraciones por comando) y procesa los logs para calcular métricas. El script calcula

throughput, tiempos de respuesta promedio, tasa de misses en deadlines, jitter, tasa de pérdidas y tiempos de procesamiento y consultas. Hemos utilizado deadlines predefinidos: 1000 ms para "hello", "23" y "listifs"; 10000 ms para "listports 127.0.0.1" y "!ls". No observamos pérdidas ni eventos no procesados en los logs, ya que todas las solicitudes recibieron respuestas.

- **Configuración del Entorno**

  - **Sistema bajo prueba**: Aplicación cliente-servidor Java
  - **Infraestructura**: 1 servidor + 2 clientes en la misma red
  - **Herramienta de automatización**: Script PowerShell personalizado (run_tests.ps1)
  - **Iteraciones por comando**: 5 ejecuciones por cada tipo de comando
  - **Comandos evaluados**: hello, fibonacci(23), listifs, listports, !ls

- **Métricas Capturadas**

  Hemos recopilado datos de:

  - Timestamp de cada solicitud
  - Tiempo de procesamiento en servidor (ServerTime)
  - Tiempo round-trip (RTT)
  - Tipo de comando ejecutado

Resultados y Fórmulas Utilizadas

A continuación, documentamos cada subatributo de rendimiento, para cada atributo, incluimos:

1. **Definición**: Basada en la descripción proporcionada en tu consulta.
2. **Fórmula**: La ecuación utilizada para calcularlo, adaptada al contexto de tus pruebas (usando ServerTime como tiempo de procesamiento en el servidor, ya que representa el tiempo entre la llegada y la respuesta del sistema).
3. **Medición**: Los valores calculados a partir de los datos, presentados en tablas donde sea efectivo para comparaciones o enumeraciones.
4. **Análisis**: Observaciones breves sobre los resultados en este setup.

Análisis por cada atributo de calidad:

1. **Rendimiento (Throughput)**

   1.1. Definición: Este atributo se refiere a la cantidad de trabajo realizado por el sistema por unidad de tiempo, como el número de transacciones procesadas. Se espera que el sistema pueda manejar un alto volumen de eventos o solicitudes dentro de un periodo determinado, asegurando su eficiencia.

1.2. Formula: Para calcularlo se hace uso de la formula:

$$\text{Throughput} = \frac{\text{Número total de solicitudes procesadas}}{\text{Tiempo total de ejecución (en segundos)}}$$

1.3.Medición:

- Número total de solicitudes procesadas: 50
- Tiempo total (desde la primera hasta la última solicitud): 47.07 segundos
- Throughput: 1.062 solicitudes/segundo

1.4.Análisis: El throughput es bajo, lo que indica que el sistema procesa aproximadamente 1 solicitud por segundo en promedio. Esto podría deberse a comandos intensivos como listports o listifs, que toman más tiempo.

2. Tiempo de Respuesta (Response Time)

2.1. Definición: Este atributo se refiere al tiempo que transcurre entre la solicitud de un evento y la respuesta del sistema. Se espera que el sistema responda de manera rápida, minimizando la latencia y asegurando una experiencia de usuario fluida.

2.2. Fórmula:

$$\text{Tiempo de Respuesta Promedio} = \frac{\sum \text{ServerTime de cada solicitud}}{\text{Número de solicitudes}}$$

(Se calcula por comando para mayor detalle).

2.3. Medición:

Command	Count	ST min	ST max	ST mean (ms)	ST std	RTT min	RTT max	RTT mean	RTT std
!ls	10	140	590	237.300	132.553	141	4224	1599.000	1739.645
23	10	0	34	8.900	12.381	2	403	69.100	123.157
hello	10	0	19	3.400	5.333	1	24	6.200	6.194
listifs	10	390	792	508.100	144.884	418	4284	1663.200	1632.653
listports 127.0.0.1	10	2953	4336	3678.800	385.301	3432	7616	5030.300	1644.428

2.4. Análisis: Los comandos simples como "hello" y "23" (cálculo de Fibonacci) tienen tiempos bajos (<10 ms), mientras que "listports" es el más lento debido a la exploración de puertos. Esto resalta dependencias en la implementación (e.g., algoritmos de escaneo).

3. Fecha Límite (Deadline)

3.1. Definición: Este atributo se refiere al tiempo máximo permitido para completar una tarea o evento. Se espera que el sistema cumpla con las fechas límite para garantizar que las tareas se completen dentro del tiempo estipulado sin afectar el rendimiento global. Aquí, se mide a través de la tasa de incumplimientos (miss rate).

3.2. Fórmula:

$$DeadlineMissRate = \frac{\text{Número de solicitudes donde ServerTime > Deadline}}{\text{Tiempo total de ejecución (en segundos)}}$$

$$DeadlineMissRate = \frac{NumeroIncumplimientos}{\text{Tiempo total de ejecución (en segundos)}}$$

(Deadlines usados: hello=1000 ms, 23=1000 ms, listifs=1000 ms, listports=10000 ms,!ls=10000 ms, esto se puede ver en nuestro script run_tests.ps1).

3.3. Medición:

- Número de incumplimientos: 0 (ninguna solicitud excedió su deadline).
- Tasa de incumplimientos: 0.000000 incumplimientos/segundo.

3.4. Análisis: El sistema cumple todas las deadlines definidas, lo que es positivo. Si se ajustan deadlines más estrictos (e.g., 500 ms para listifs), podrían aparecer incumplimientos en comandos intensivos.

4. Jitter de Respuesta Variación de la latencia.

4.1. Definición: Este atributo se refiere a la variabilidad en el tiempo de respuesta del sistema (latencia). Se espera que el sistema mantenga una respuesta consistente y predecible, evitando fluctuaciones significativas que puedan afectar la estabilidad y el desempeño.

4.2. Fórmula:

$$\text{Jitter} = \sqrt{\frac{\sum(\text{ServerTime}_i - \text{Promedio ServerTime})^2}{\text{Número de solicitudes - 1}}}$$

(Desviación estándar de ServerTime, por comando para más detalle).

4.3. Medición:

Comando	Jitter (ms)
!ls	139.72
23	13.05
hello	5.62
listifs	152.72
listports 127.0.0.1	406.14

4.4. Análisis:

Los comandos simples presentan bajo jitter, mientras que los comandos de sistema muestran alta variabilidad, posiblemente debido a la carga del sistema operativo y condiciones de red variables.

5. Tasa de Pérdida (Missing rate)

5.1. Definición: Este atributo se refiere a la cantidad de eventos que el sistema no puede procesar o pierde por unidad de tiempo. Se espera que el sistema minimice la pérdida de eventos para garantizar la precisión y la integridad de la información procesada.

5.2. Fórmula:

$$Missing\ Rate\ = \frac{(Eventos\_enviados\ -\ Eventos\_recibidos)}{T\_total}$$

5.3. Medición:

- Eventos enviados: 50 (5 iteraciones × 5 comandos × 2 clientes).
- Eventos respondidos: 50.
- Tasa de pérdida: 0.000000 eventos/segundo.

5.4. Análisis:

No hay pérdidas en estas pruebas, lo que sugiere una conexión confiable entre clientes y servidor. En escenarios con alta carga o red inestable, esto podría aumentar.

6. Tasa de No Procesados (Unprocess rate)

6.1. Definición: Este atributo se refiere a los eventos que llegan al sistema, pero no se procesan adecuadamente. Se espera que el sistema procese todos los eventos sin fallos, manteniendo un alto nivel de eficiencia en el manejo de los datos.

6.2. Fórmula:

$$Unprocess\ Rate\ = \frac{N\_no\_procesados}{T\_total\ (s)}$$

6.3. Medición

- Eventos no procesados: 0.
- Tasa de no procesados: 0.000000 eventos/segundo

6.4. Análisis

El servidor procesó todos los eventos recibidos. Si el server_log.csv mostrara conteos incompletos (solo tiene una entrada con MessageCount=0), podría indicar un problema de logging, pero los client logs confirman procesamiento completo.

7. Tiempos de Procesamiento Funciones, cálculos, importaciones, exportaciones.

7.1. Definición: Este atributo se refiere al tiempo que el sistema tarda en ejecutar funciones internas como cálculos, importaciones o exportaciones de datos. Se espera que estos procesos se realicen de manera eficiente y en tiempos adecuados para evitar cuellos de botella y retrasos.

7.2. Fórmula:

$$Tiempo\ de\ Procesamiento\ Promedio = \frac{\sum ServerTime\ de\ solicitudes}{Número\ Solicitudes}$$

7.3. Medición

Command	First (ms)	Later avg (ms)
!ls	372	222.333
23	23	7.333
hello	19	1.667
listifs	470	512.333
listports 127.0.0.1	3595	3688.111

7.4. Análisis

Los tiempos de procesamiento varían según la complejidad de los comandos. *hello* y 23 son rápidos, ya que involucran operaciones en memoria (saludos y cálculos de Fibonacci). Por otro lado, *listifs* requiere acceso a información de red del sistema, lo que introduce latencia debido a consultas al sistema operativo. Del mismo modo, listports es el más costoso, probablemente por iteraciones repetitivas sobre puertos o llamadas a APIs del sistema (e.g., netstat o equivalentes).

8. Tiempos de Consultas y Reportes Cargas iniciales y cargas posteriores.

8.1. Definición: Este atributo se refiere al tiempo que tarda el sistema en cargar datos o generar reportes. Se espera que el sistema realice consultas y genere reportes de manera eficiente, incluso en cargas iniciales y subsecuentes, sin demoras notables para el usuario.

8.2. Fórmula:

RTT = tiempo de ida y vuelta (incluye red + procesamiento)

8.3. Medición

Comando	Carga Inicial (ms)	Cargas Posteriores Promedio (ms)
!ls	**372.00**	**222.33**
23	**23.00**	**7.33**
hello	**19.00**	**1.67**
listifs	**470.00**	**512.33**
listports 127.0.0.1	**3595.00**	**3688.11**

8.4. Análisis

En la mayoría de los comandos, las cargas posteriores son más rápidas (e.g., "hello" baja de 19 ms a 1.67 ms), sugiriendo optimizaciones como cachés. Sin embargo, "listifs" y "listports" no muestran mejora, posiblemente por variabilidad en recursos del sistema.

- **Conclusiones y Recomendaciones**

Como equipo, consideramos que nuestro sistema muestra un rendimiento aceptable para comandos simples, con throughput moderado y sin misses en deadlines. Sin embargo, comandos intensivos como "listports" y "listifs" tienen tiempos altos y jitter significativo, lo que sugiere oportunidades de optimización, como mejorar algoritmos o distribuir carga. En futuras iteraciones, planeamos aumentar el número de clientes para simular más concurrencia y monitorear uso de memoria/CPU. Este análisis nos ayuda a validar los atributos no funcionales y guiar rediseños si es necesario.

Referencias

- ✓ **ISO/IEC 25010:2011.** *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) System and software quality models.* International Organization for Standardization, 2011.