**DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN**

**ARQUITECTURA Y DISEÑO DE SOFTWARE (ISIS2503)**

**PROFESORA: PhD KELLY GARCÉS**

**PRESENTADO POR:**

**ANDRÉS CAMILO ZULETA – 201413405**

**CAMILO CARRILLO – 201215749**

**SERGIO MIGUEL MADERA – 201417957**

**FECHA DE PRESENTACIÓN:**

**15 DE MARZO DE 2016**

**UNIVERSIDAD DE LOS ANDES**

**RESUMEN EJECUTIVO**

Dentro del presente documento se encuentra el trabajo realizado por el grupo de trabajo número cinco para la entrega preliminar del primer experimento correspondiente al curso de Arquitectura y diseño de software. Este documento incluye un análisis de las decisiones tomadas en cuanto a la arquitectura y la justificación de estas, tomando en cuenta los requerimientos no funcionales planteados para esta entrega del experimento. En adición, se describe la planeación, el desarrollo y la ejecución de los servicios y pruebas de los atributos de calidad realizados sobre el sistema SATT construido.

**ARQUITECTURA SELECCIONADA Y JUSTIFICACIÓN**

La arquitectura seleccionada para la realización del proyecto es la arquitectura basada en servicios REST de JAX-RS. Dicha decisión fue tomada inicialmente debido a la facilidad de manejo que esta otorga y por el potencial que esta le otorga al equipo desarrollador para desarrollar, valga la redundancia, de manera eficaz y ágil. Adicionalmente esta arquitectura cuenta con los beneficios que REST tiene para ofrecer y su separación por capaz permite un fácil manejo de sus distintos componentes, facilitando así su mantenimiento en el caso de que se deseara una longeva esperanza de vida para la aplicación y además su fácil modificación en caso de ser requerida.

Ahora, esta arquitectura resulta también beneficiosa para los objetivos del experimento, es decir, el cumplimiento de los atributos de calidad de desempeño y escalabilidad. Esto podemos deducirlo a partir de los resultados que obtuvimos durante el laboratorio de desempeño y escalabilidad llevado a cabo durante el curso, donde se pudo ver que una aplicación REST soportaba de manera eficaz tanto el desempeño como la escalabilidad para un número significante de usuarios (más de diez mil). Por tanto a partir de esto se puede deducir que JAX-RS permite manejar de manera adecuada el número de solicitudes que se espera para la aplicación realizada durante el experimento (alrededor de cuatro mil).

Sin embargo también es menester mencionar que el grupo de trabajo planea desarrollar en paralelo la misma aplicación utilizando una arquitectura de tipo Play! Y comparar los resultados entre ambas arquitecturas para que al momento de la entrega final se escoja aquella arquitectura con la cual se obtengan los resultados más satisfactorios. Decisión que se fortaleció en el grupo al observar las dificultades que presentó la arquitectura JAX-RS para su despliegue en plataformas que ofrecen servicios cloud y adicionalmente a los resultados obtenidos durante esta fase del proyecto que no terminaron de convencer a todos los miembros del grupo.

**DISEÑO Y DESARROLLO DE LAS PRUEBAS**

**Pre-experimentación**

Se probará que los servicios que el sistema ofrece cumplen con los escenarios de calidad recibidos, incluso tras la adición de la capa de persistencia y capa de presentación web de la aplicación. Ha de ser posible para el sistema generar boletines de alerta con un latencia inferior a 500 milisegundos en todo momento; para probar esto, utilizaremos cuatro mil hilos con ejecución simultánea que accederán al servicio. En el mismo contexto de ejecución, el sistema ha de poder recibir información de todos los sensores con una duración total de menos de un minuto y ha de ser capaz de responder a cambios del nivel de altura con una latencia inferior a un segundo. Finalmente, debe estar dentro de las capacidades del sistema la recepción de información de 4000 sensores con el objetivo de actualizar el boletín de alerta con un tiempo de respuesta total de menos de un minuto. Esta prueba se realizará en el mismo contexto que las demás descritas anteriormente.

Las pruebas se realizarán utilizando JMeter. De esta forma, podemos modelar los escenarios en los que es necesario que la aplicación responda adecuadamente y analizar su desempeño con los datos reportados por JMeter. Todas las pruebas sobre los servicios consistirán de cuatro etapas, en cada una de las cuales el número de hilos que acceden de manera concurrente a cada servicio variará. En una etapa serán 500 hilos, en otra 1000, en otra 2000, y en otra 4000. Sin embargo, la prueba sobre la recepción constante de información de sensores y respuesta al cambio de altura tendrá tres etapas adicionales, diseñadas para realizar la implementación de un cluster de JMeter; en cada una hay 4000 threads recibiendo información de los sensores, y lo que variamos es el número de sensores enviando información al mismo tiempo. En cada etapa, son 100, 500 y 4000 respectivamente. Si los datos obtenidos durante cada prueba evidencian que el servicio al que corresponden cumplen con los requerimientos descritos en el párrafo anterior, será posible concluir que la aplicación como tal tiene las capacidades por el bien de las cuales fueron tomadas decisiones durante el proceso de desarrollo.

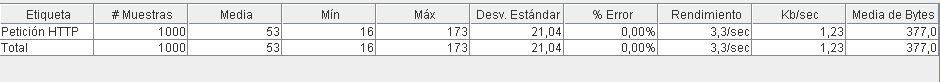
Es importante especificar además que para el despliegue de la aplicación se utilizará NetBeans IDE 8.1 y GlassFish Server 4.1.1, y que no se utilizó ningún recurso adicional además del programa desarrollado, SATT, y Apache JMeter. En la sección concerniente a la post-experimentación se mostrarán pantallazos que muestran los resultados más importantes de cada prueba, además de un breve análisis sobre los datos obtenidos y el desempeño de la aplicación durante la misma. Un análisis más detallado de lo observado sobre el experimento será anexado en un documento de Microsoft Excel al proyecto. No está de más mencionar que cada una de las pruebas, con sus etapas, se ejecutará por separado y en un ambiente aislado de los efectos que pudieran tener otras; una consecuencia importante de esto es que las tablas de la base de datos se encuentran en el mismo estado de control al inicio de cada prueba.

**Post-experimentación**

A continuación, se analizarán los resultados obtenidos al realizar las pruebas.

**Prueba 1 – Generación de boletín de alerta:**

C:\Users\Sergio\Downloads\testEntregaExp1 (1)\testEntregaExp1\generarReporte\para4000threads.PNG



La latencia máxima obtenida fue de 1454 milisegundos, y fue obtenida en la etapa con 4000 hilos concurrentes. Esta latencia es superior a la máxima tolerada de 500 milisegundos. No obstante, la latencia máxima obtenida para 1000 o menos hilos en ejecución concurrente generando el boletín de alerta fue de 173, menor a la máxima propuesta de 500 milisegundos, por lo que argüiremos a continuación que el desempeño observado es suficiente para los contextos reales a los que será sometido el sistema.

A pesar de los resultados obtenidos para la etapa de 4000 hilos concurrentes, por aquellos mencionados consideramos que el desempeño del que es capaz es suficiente en todos los casos prácticos, es decir, en todos los contextos a los que se someterá la aplicación en la realidad. Esto es porque no se generarán más de 1000 boletines de alerta simultáneamente en una situación real, dado que no se generará tal cantidad de tsunamis al mismo tiempo en la realidad. Una situación bastante posible en la realidad es, sin embargo, que deban haber más de 1000 actualizaciones sobre boletín de alerta simultáneamente, pero este servicio funciona de forma completamente satisfactoria en todas las etapas de su prueba, la cual se describirá a continuación. De esta forma, concluimos que el servicio cumple con el desempeño que necesitará durante las situaciones reales a las que será sometido.

Para un análisis más detallado de la prueba, por favor referirse al documento de Microsoft Excel anexo al proyecto.

**Prueba 2 – Recepción información y respuesta a cambios de nivel de altura**

C:\Users\Sergio\Downloads\testEntregaExp1 (1)\testEntregaExp1\enviarSeñalCambioOla\para4000Threads.PNG

La latencia máxima al recibir la información enviada desde los sensores con frecuencia de 1 minuto y responder a los cambios del nivel de altura en la etapa con mayor carga de la prueba fue de 70 milisegundos, considerablemente inferior a la latencia máxima impuesta de 1 segundo, que equivale a 1000 milisegundos. Por esto, concluimos que la aplicación cumple con las metas impuestas al principio de la planeación, por el bien de las cuales se tomaron muchas decisiones sobre el diseño y la implementación de la aplicación. Sin embargo, no ha de subestimarse el hecho de que responder a cambios de nivel de altura puede implicar la actualización de boletín de alerta, lo que a su vez implica el acceso a la base de datos. Dado lo relativamente lento que es el mismo, esto puede convertirse en un gran obstáculo conforme la exigencia sobre la aplicación crezca y ha de ser implementado lo más eficientemente posible.

Para un análisis más detallado de la prueba, por favor referirse al documento de Microsoft Excel anexo al proyecto.

**Prueba 3 – Escalabilidad:**

C:\Users\Sergio\Downloads\testEntregaExp1\testEntregaExp1\mostrarSeñales\Para4000ThreadsVsSensoresVariables\4000ThreadsVS4000Sensores.PNG

La latencia máxima al recibir la información de 4000 los sensores cuando se requiere actualizar el boletín de alerta fue de 686 milisegundos, inferior al tiempo de respuesta máximo de 1 minuto cuando se requiere actualizar el boletín de alerta. El error fue de 0.00% en todos los casos evaluados, por lo que concluimos que el sistema cumple con el requisito solicitado. Deducimos de esto la arquitectura elegida y las decisiones realizadas durante el desarrollo de cada una de sus partes fueron un factor determinante para el buen desempeño del servicio en cuestión. No obstante, de forma similar a la prueba 2, debemos tener en cuenta que la actualización de boletín de alerta requiere de acceso a la base de datos, aspecto que puede ser fatal conforme el estrés de la aplicación incrementa.

Para un análisis más detallado de la prueba, por favor referirse al documento de Microsoft Excel anexo al proyecto.

**CONCLUSIONES**

Para finalizar, tras la realización de las pruebas y el análisis de los datos obtenidos a partir de las mismas, observamos que la latencia máxima obtenida para la Prueba 1 en la etapa más exigente fue mayor a la máxima propuesta de 500 milisegundos, pero que en aquellas etapas de la prueba similares a situaciones reales el desempeño estuvo por debajo de este límite. Para la Prueba 2, observamos que el desempeño de la aplicación era satisfactorio en todos los contextos a los que fue sometida, siendo capaz de responder a los cambios del nivel de altura con una latencia máxima de 70 milisegundos. Para la Prueba 3, de manera análoga a la Prueba 2, observamos que incluso en la etapa más exigente, el sistema escaló de manera exitosa. Por lo anterior mencionado, concluimos que el sistema desarrollado es capaz de responder satisfactoriamente a todos los contextos a los que se verá sometido en la práctica.

Sin embargo, dada la naturaleza de los servicios que fueron testeados, concluimos también que debemos tener presente en todo momento que requieren de la realización de transacciones en la base de datos, de modo que mantengamos dichos accesos implementados de la forma más eficiente posible en todo momento para evitar un incremento en el tiempo de respuesta general de la aplicación. Adicionalmente se debe mencionar que el grupo de trabajo deberá observar cuidadosamente los cambios que realice en la arquitectura con el fin de obtener mejores resultados en los distintos escenarios de calidad pues se nota una diferencia considerable en las distintas pruebas luego de añadir los elementos de persistencia e interfaz gráfica que terminaron teniendo un impacto más grande (en algunos requerimientos) del esperado.