**Desarrollo de un modelo para el análisis de pruebas de rendimiento de software en ambientes de big data, para identificar patrones mediante reglas de asociación**

**Nombre completo del investigador:** Jorge Mauricio Acuna Gómez.

**Breve descripción de su formación académica:** Ingeniero de Sistemas, MSc. (Maestría en Ciencias de la Computación).

Investigadores asociados.

**Introducción de la Investigación**

El ciclo de vida de desarrollo de una aplicación, contempla como una de sus fases o etapas, las pruebas del sistema desarrollado. Estas pruebas pueden ser, pruebas funcionales o de desempeño. Contar con un buen plan de pruebas ayuda a los equipos de trabajo a detectar cualquier problema o anomalía en la aplicación antes de hacer el release del sistema a producción. Las pruebas de rendimiento o desempeño tienen el objetivo de garantizar que la aplicación soporta en términos de rendimiento las cargas a las que se verá expuesta una vez que se haya puesto en marcha.

Dichas pruebas deben contemplar distintos aspectos de los elementos involucrados en cualquier sistema informático; de los cuales podemos nombrar la respuesta del sistema ante la ejecución de ciertas funcionalidades o procesos implementados y el estado de los componentes de hardware donde correrá la aplicación. Estos aspectos mencionados, anteriormente, son los que se utilizaron a la hora de medir o evaluar cada componente del sistema, ya sea de hardware o de software.

La investigación tuvo como objetivo principal proponer un modelo que permita mejorar la evaluación y el análisis de los resultados de estas pruebas.

La función de identificar problemas potenciales en los sistemas informáticos, luego de correr las pruebas puede ser muy compleja, dependiendo de la aplicación y su arquitectura; esto conlleva a la inversión de recursos y tiempo para evaluar dichas pruebas, por lo que la aplicación del modelo sería de gran ayuda para el desarrollo y puesta en marcha de un sistema.

Por otro lado, al final de estas pruebas se cuenta con una gran cantidad de archivos o información proveniente de diferentes fuentes, las cuales pueden ser computadoras clientes, servidores o las diferentes aplicaciones con las que podría interactuar la aplicación, y que son importantes para ser consideradas en las pruebas de desempeño.

Finalmente, lo que se persigue con el modelo es la identificación de patrones en los datos generados o procesados por medio de la obtención de reglas de asociación, pues por medio de ellas, se identificará un conjunto de relaciones presentado con cierta frecuencia en los datos producidos durante la aplicación de las pruebas. Este conjunto de reglas, indicará el comportamiento de la aplicación, el cual, dependiendo de los parámetros establecidos y resultados esperados, determinará si el sistema informático es adecuado o no para ser puesto en marcha.

**Métricas utilizadas**

Es de gran importancia la definición de las métricas que se van a utilizar para el análisis. Dentro de las métricas se definieron:

1. Los escenarios que se quieren probar en la aplicación, esto va a cambiar dependiendo del tipo de aplicación en cuestión. Aquí se definen los *logs* del lado de la aplicación, estos deben contener el tipo de prueba o proceso ejecutado, el momento en que se corrió y tiempo de respuesta.
2. Los *performance counters*, que consideran todas las mediciones asociadas al hardware de los servidores o computadoras donde está corriendo la aplicación, como por ejemplo la memoria RAM o utilización del procesador.
3. Y por último una vez definidos los escenarios de prueba es de suma importancia que el equipo técnico del proyecto defina de acuerdo a los *performance counters* y el tipo de prueba que es un resultado aceptable y cual no. Por ejemplo, al correr la prueba X la utilización del procesador debe ser menor a un X%.

Es muy importante definir tanto los escenarios de prueba, como los parámetros capturados en los *logs*. Por otro lado, las unidades de medición de las métricas de hardware y software evaluados son de suma importancia, ya que el sistema depende del análisis o la aprobación para mover a ambientes de producción. La evaluación de las reglas de asociación será un proceso manual, que los integrantes del equipo de desarrollo deberán discutir y evaluar para tomar la decisión final.

**Modelo Propuesto**

El modelo hará que el proceso de análisis de pruebas sea más eficiente y eficaz, evitando, además, el error humano al realizar el análisis.

El mismo también busca que los equipos de desarrollo puedan dedicar su tiempo a otras tareas y de esta forma ayudar al desarrollo de los proyectos en las organizaciones. Es importante mencionar que las herramientas utilizadas en la investigación son *Open Source*, donde se seleccionaron Hadoop y Linux para el procesamiento de los datos. Ya que las pruebas pueden ser ejecutadas por periodos largos de tiempo y la complejidad de las aplicaciones hoy en día puede llegar a ser grande, la cantidad de datos y archivos a procesar pueden ser muchos, por lo que en este aspecto Hadoop ayuda al procesamiento masivo de datos. En Hadoop se desarrollaron unos scripts (hql) para poder limpiar, dosificar y preparar los datos de tal forma que nos permita analizar los 2 tipos de fuentes de datos que se están utilizando (los *performance counters* y los datos arrojados por la aplicación). El modelo propuesto cuenta con una serie de pasos que se deben respetar para la aplicación del mismo. A dicho modelo se le llamó APRA “Análisis de pruebas de rendimiento automático”. Dichos pasos se detallan a continuación:

| **MODELO APRA** |
| --- |
| 1. Contar con los pre-requisitos antes de iniciar con el modelo.
	1. Contar con el ambiente para realizar las pruebas.
	2. Contar con *cluster* con Hadoop configurado y listo para llevar a cabo el pre procesamiento de los datos.
	3. Contar con una herramienta de minería de datos para aplicar el algoritmo Apriori compatible con el modelo propuesto.
 |
| 1. Definir los escenarios y la duración de las pruebas de la aplicación.
 |
| 1. Definir las unidades de medidas y valores aceptables para cada uno de los elementos a medir en el modelo.
 |
| 1. Incluir en los logs del cliente la información requerida por el modelo.
* Momento en que ocurrió la prueba.
* Tipo de prueba o escenario a probar.
* Tiempo de respuesta.
 |
| 1. Incluir en los logs del servidor la información requerida por el modelo.
* Momento cuando se recopila el estado de los componentes de hardware del servidor.
* Porcentaje de procesador utilizado.
* Cantidad de memoria RAM disponible.
* Tamaño de las solicitudes de I/O a disco.
* Transferencia en bytes de la interface de red.
 |
| 1. Generar los logs del cliente y servidor en un formato compatible con el sistema que se utiliza.
 |
| 1. Copiar los archivos de logs generados por las pruebas al sistema local de archivos de la computadora con Linux donde está configurado Hadoop.
 |
| 1. Cargar los archivos al HDFS (*Hadoop Distributed File System*).
 |
| 1. Crear la base de datos correspondiente en Hadoop.
 |
| 1. Ejecutar el script .hql desarrollado en esta investigación para limpiar, dosificar y preparar los datos.
 |
| 1. Generar el archivo plano desde la línea de comandos de Hadoop.
 |
| 1. Crear el archivo compatible con el sistema elegido (basado en el archivo final generado en el paso anterior) para la ejecución del algoritmo Apriori.
 |
| 1. Generar las reglas de asociación por medio de un sistema que implemente el algoritmo Apriori que sea compatible con el modelo propuesto.
 |
| 1. Analizar las reglas de asociación obtenidas y tomar las decisiones correspondientes de los resultados de las pruebas.
 |

Se considera una estandarización en la manera como se captura la información de las pruebas; en el caso de esta investigación se utilizaron archivos .csv, en donde la delimitación de los datos en los logs de las pruebas debe ser por medio de comas, la herramienta utilizada para el procesamiento debe ser del suite de herramientas disponibles en las distribuciones de Hadoop y, finalmente, la implementación del algoritmo Apriori para la generación de las reglas de asociación. Las reglas de asociación indican la ocurrencia de los escenarios de prueba y las condiciones o estado del hardware en el momento en que se corrieron las pruebas.

**Análisis de los resultados**

El resultado final a la hora de aplicar son las reglas de asociación, las cuales ayudan a relacionar los datos e identificar patrones en los mismos. Lo que se buscaba con la generación de las reglas de las reglas es que indicaran la mayor ocurrencia de escenarios ante la ejecución de las pruebas.

**Hallazgos de las reglas**

* Se puede deducir que durante el período cuando las pruebas fueron realizadas, el servidor siempre tuvo una memoria disponible entre 7GB y 8GB, con una utilización del procesador menor al 30%.
* También se puede mencionar que el tamaño promedio de las solitudes I/O al disco fueron entre 10 KB y 20 KB para una utilización de procesador menor al 30%.
* Para una velocidad de la interface de Red mayor a 50Kbs la utilización de procesador fue siempre menor al 30%.
* Para la mayoría de las pruebas el tiempo de respuesta fue de menos de 1 segundo, obteniendo una utilización de CPU de menos del 30% y con un porcentaje mayor al 50% de memoria disponible en el servidor (disponibilidad de ente 7GB y 8GB de memoria).
* El siguiente es un ejemplo de las reglas importantes detectadas después del análisis; donde se puede concluir que en los datos se encontró una relación entre el porcentaje de procesador utilizado, la memoria disponible y el tamaño promedio de las solicitudes de I/O hechas al disco.
	+ - *server\_percentage\_processor\_time = CPU\_LessThan30*

*memory\_available\_gb = MemoryAvailableGB\_Between7and8GB*

*==>*

*hd\_avg\_disk\_KB\_transfer =HDAvrgKBTransfer\_Between10and20KB*

En la siguiente regla se puede observar cómo se relacionó el tipo de prueba con el hardware de los servidores, donde al ejecutar la prueba 1 el porcentaje de procesador utilizado fue menor al 30% y la memoria RAM disponible fue de entre &GB y 8GB.

* + - *operation=TestScenario1 ==> server\_percentage\_processor\_time=CPU\_LessThan30 memory\_available\_gb=MemoryAvailableGB\_Between7and8GB*

**Conclusiones**

Luego de aplicar el modelo, podemos validar y comprobar que el sistema al que se le aplicaron las pruebas se comportó estable y soporta de manera exitosa y sin problemas los escenarios ejecutados sobre los servidores con las características específicas de hardware de producción. La característica de los datos recolectados de la aplicación probada no varía mucho durante el período de tiempo cuando se llevaron a cabo las pruebas, por lo que también se puede deducir que la confiabilidad de las reglas y resultados obtenidos son de alta confianza. También se notó que si una determinada prueba durante el período de la aplicación de las misma se ejecuta pocas veces en comparación con los otros escenarios, el resultado de ella pasa a ser despreciable para la generación de las reglas; por lo que puede no ser tomado en cuenta por el algoritmo. Debido a esto, es muy importante definir de manera precisa la frecuencia de las pruebas aplicadas. Otro punto importante a mencionar es que cuanto más cantidad de variables contenga la regla en “X” o en “Y”, hay más aporte al análisis de las pruebas. De lo contrario habrá que analizar mayor cantidad de reglas para tener una comprensión más completa y precisa de cómo sucedieron y se comportaron las pruebas.