Herramienta informática para la selección de los servidores web Apache 2 y Nginx

Resumen– La presente investigación se centró en el objetivo de desarrollar una herramienta informática para aumentar la eficiencia en la selección de los servidores web Apache 2 y Nginx durante el proceso de migración a código abierto. Se aplicó el método Analítico-Sintético para el estudio de Apache 2 y Nginx y la determinación de las particularidades de cada uno. La realización de un estudio de caso aplicando la Norma Cubana ISO/IEC 25023:2017 y el método estadístico, permitió el procesamiento de la información recopilada acerca del comportamiento de Apache 2 y Nginx, esto sirvió de base para formular la teoría relacionada con la eficiencia de ambos servidores. A partir del tipo de contenido y del total de peticiones concurrentes se definieron 60 escenarios para la selección del servidor más eficiente, teniendo en cuenta todos los indicadores o cada medida específica: Rendimiento, Utilización de recursos y Capacidad. Las tres variantes a seleccionar son: Apache 2, Nginx o Nginx como proxy inverso de Apache 2. El resultado obtenido se materializa en la Herramienta para la Migración y Administración de Servicios Telemáticos, la cual posee un componente Web que permite la selección del servidor web que más se ajusta a la institución y la activación del módulo correspondiente para su administración. La herramienta se validó a través de un estudio de caso, del criterio de expertos en su variante Delphi y la técnica de Iadov. Finalmente la triangulación metodológica permitió confirmar el resultado satisfactorio de todos los métodos aplicados y el cumplimiento del objetivo planteado.

Palabras clave: Apache 2, eficiencia, Nginx, herramienta informática.

I. INTRODUCCIÓN

Con el transcurso de los años el número total de usuarios en Internet a nivel mundial ha aumentado significativamente. Las estadísticas ofrecidas por la compañía Statista muestran el número de usuarios de Internet en todo el mundo de 2009 a 2018, en América Latina y el Caribe la cifra ascendió de 186,9 millones en 2009 a 438,25 millones en 2018 [1]. Internet también ha penetrado en los diez países más poblados de Latinoamérica, siendo Cuba uno de ellos con 4,4 millones de usuarios [2]. Internet ha creado un nuevo escenario en el que las relaciones personales cobran protagonismo. Las nuevas plataformas y herramientas colaborativas han producido un cambio desde una web 1.0 basada en páginas estáticas, meramente informativas, sin capacidad de generar una participación del usuario, hacia una web dinámica donde se produce una interrelación que genera una suma de conocimientos y/o experiencias. Los servidores web desempeñan un papel dominante en la infraestructura de estos servicios.

Su tarea principal es recibir y procesar solicitudes de clientes que exigen objetos específicos de los servidores y devolverlos mediante respuestas relacionadas. Los mensajes de solicitud y respuesta asociados se transportan mediante el Protocolo de Transferencia de Hipertexto (HTTP, del inglés *Hypertext Transfer Protocol*) o el Protocolo de Transferencia de Hipertexto Seguro (HTTPS, del inglés *Hypertext Transfer Protocol Secure*) mediante conexiones TCP (del inglés *Transmission Control Protocol*) entre los clientes y el servidor. El rendimiento resultante de la prestación de servicios depende crucialmente del procesamiento adecuado y eficiente de estas tareas [3]. Existen varios tipos de servidores web en cuanto a la forma como procesan las peticiones que reciben de los usuarios: basados en procesos, basados en hilos y basados en eventos.

Actualmente son numerosos los servidores web, dentro de los que se encuentran Microsoft-IIS, Apache 2, Nginx, LiteSpeed, Google, Sun Java System, Lighttpd, IBM Servers y Yahoo Traffic Server. Algunos son más usados que otros debido a que poseen características distintivas que marcan la diferencia en cuanto a su selección. La información sobre el uso de servidores web es proporcionada por algunos sitios especializados en ofrecer estadísticas de diferentes tecnologías en la web. Ejemplo de ello es el sitio de la compañía inglesa Netcraft que realiza mensualmente mediciones de uso [4]. Según el reporte del mes de noviembre de 2018, los cuatro principales proveedores más importantes son Microsoft (39,47%), Nginx (21,71%), Apache (20,68%) y Google (1,46%) [5].

Por otra parte, Web Technology Surveys (W3Techs) también proporciona estadísticas del porcentaje de sitios web que hacen uso de determinado servidor web. En el reporte de noviembre de 2018 las estadísticas fueron: Apache (47,1%), Nginx (37,5%), Microsoft-IIS (10,1%), LiteSpeed (3,2%) y Google (1.0%) [6]. A partir de las estadísticas en ambos reportes, se calculó la media del porcentaje de uso de cada uno de los cuatro servidores web que coinciden, determinándose Apache con 33,89%, Nginx con 29,61%, Microsoft-IIS con 24,79% y Google Servers con 1,23%, donde Apache y Nginx son los dos servidores web de código abierto más usados.

En Cuba también se evidencia el gran uso de estos dos servidores web de código abierto. Se realizó un análisis de los 40 principales sitios en Cuba de interés cultural y de entretenimiento, informativo, educativo e investigativo [7] y las tecnologías que estos emplean. Para analizar los sitios web se utilizó la herramienta multiplataforma Wappalizer, que permite identificar la tecnología utilizada en sitios web, incluidos servidores, sistemas de gestión de contenido, plataformas de comercio electrónico, herramientas de análisis,

marcos de publicidad, etc. [8]. Del análisis realizado se determinó lo siguiente:

- 17 mostraron el sistema operativo, donde 88,2% se basan en código abierto (Ubuntu, Debian, RedHat, SUSE, CentOS) y 11,8% emplean Windows Server.
- 33 mostraron el servidor web, donde las estadísticas son Apache 2 (66,6%), Nginx (27,3%) y Microsoft-IIS (6,1%), evidenciándose que Apache 2 es notablemente el más usado.
- 28 mostraron el lenguaje de programación, PHP (89,3%), ASP.NET (7,1%) y Python (3,6%), destacándose PHP en la mayoría de los sitios.

Se puede apreciar que se hace extensivo el uso de tecnologías de código abierto y precisamente en Cuba el proceso de informatización tiene como pilar el empleo de software libre. El 28 de febrero de 2017 quedó aprobada por el Consejo de Ministros la política integral para el perfeccionamiento de la informatización de la sociedad, donde se debe asegurar la sostenibilidad y soberanía tecnológica. Como precisó Miguel Díaz-Canel Bermúdez, al intervenir en la jornada previa al Noveno Período Ordinario de sesiones de la VIII legislatura de la Asamblea Nacional del Poder Popular, existe la necesidad de que las instituciones avancen en la implementación de la política de informatización e incorporen masiva y ordenadamente el uso de aplicaciones y sistemas informáticos desarrollados por las entidades cubanas [9].

Para lograr la soberanía tecnológica y desempeñar el proceso de migración, el país creó una estructura compuesta por cuatro grupos de trabajo (Legal, Capacitación, Divulgación, Técnico) y el Grupo Ejecutivo, encargado de la dirección. Como parte del Grupo Técnico, la Universidad de las Ciencias Informáticas (UCI) se destaca por desarrollar la distribución cubana de GNU/Linux Nova y llevar a cabo el proceso de migración a tecnologías de software libre y plataformas de código abierto en diferentes instituciones cubanas [10]. La UCI ha realizado varios procesos de migración en diferentes entidades.

La ejecución de los procesos de migración a código abierto inicialmente tenía como base la "Guía Cubana de Migración a Software Libre" confeccionada en la propia universidad en el año 2009. A partir del año 2015 se aplica la "Estrategia para la migración a aplicaciones de código abierto", la cual corrige un conjunto de dificultades detectadas en la guía. La estrategia plantea que la migración de servicios telemáticos es el primer paso en el proceso de ejecución de la migración. Dentro de estos servicios se encuentran los ofrecidos por los servidores web. Se plantea que de cada servidor presente en la entidad debe obtenerse toda la información relacionada con el hardware, la información de los servicios brindados y la relación existente entre estos y los demás sistemas de la institución [10].

Además como apoyo al proceso de migración se encuentra el libro "Buenas Prácticas para la Migración a Código Abierto", el cual establece que de manera análoga debe realizarse la selección de las aplicaciones informáticas que servirán de soporte para los servicios telemáticos de la institución. Plantea los elementos de gran importancia a tener en cuenta en la selección de las alternativas para la migración de los servicios, como son los conocimientos y experiencia de los administradores de red sobre el uso de las tecnologías libres. Por otra parte, la selección de la alternativa libre debe adecuarse a las necesidades de la institución y no a la alternativa al servicio privativo existente. El libro define que las alternativas libres a servidores web privativos para PYMES (Pequeñas y Medianas Empresas) y grandes empresas son Apache 2 y Nginx [11].

Para los servidores web se definen las dos alternativas pero no cómo se debe realizar el proceso de selección de estos. A partir de una entrevista realizada a los especialistas en servicios telemáticos pertenecientes a la UCI, se identificó que para la selección no existe una guía que oriente qué servidor se ajusta más a la institución. No se tiene un registro del estudio de servidores web migrados en procesos anteriores. La situación existente ocasiona dificultades tales como:

- Se evidencia la pérdida de tiempo y se duplican esfuerzos estudiando la alternativa a seleccionar.
- No se reutiliza el conocimiento para la selección del servidor web.
- Muchas veces se selecciona el servidor web más conocido aunque sea el menos eficiente para la institución.

Dada la problemática existente, el objetivo general está centrado en desarrollar una herramienta informática para aumentar la eficiencia en la selección de los servidores web Apache 2 y Nginx durante el proceso de migración a código abierto.

II. MATERIALES Y MÉTODOS

Para el desarrollo de la investigación el objeto de estudio está orientado a los servidores web Apache 2 y Nginx. Se utilizaron los métodos científicos y técnicas que a continuación se mencionan: Analítico-Sintético, Hipotético deductivo, Modelación, Método Estadístico, Entrevista y Encuesta.

A. Arquitectura para el manejo de peticiones

La arquitectura de software de un servidor web proporciona un servicio que establece cómo un usuario accede a través del protocolo HTTP a los archivos de datos almacenados en el servidor web. Existen tres arquitecturas: basadas en procesos, basadas en hilos y basadas en eventos. En las dos primeras la E/S se bloquea durante el manejo de la solicitud, por tanto el consumo de recursos se incrementa linealmente de acuerdo con el número creciente de hilos o procesos. La tercera es mucho más flexible que las anteriores, con E/S sin bloqueo y manejo asíncrono de solicitudes [12].

Servidores basados en procesos: es el predecesor de los demás diseños. Cuando se crea un nuevo proceso, el sistema duplica la imagen del ejecutable en un nuevo espacio de direccionamiento y, por lo tanto, a partir de ese punto los

procesos son independientes [13]. Un proceso es una instancia específica de un programa de computadora en ejecución. Los servidores basados en procesos son excelentes porque son estables y el bloqueo de un proceso no afecta a otros procesos. Sin embargo, no pueden manejar tantos clientes porque la creación y destrucción de todos los procesos crea una gran cantidad de sobrecarga del procesador y requiere una gran cantidad de memoria [14]. El uso de la memoria, en este caso, es linealmente proporcional al número de procesos secundarios. Además, el sistema operativo necesita gastar más tiempo de CPU para cambiar entre los procesos secundarios (cambio de contexto) [12].

Servidores basados en hilos: al crear un nuevo hilo de ejecución, el sistema no crea un nuevo proceso ni duplica la imagen del ejecutable, sino que, sobre el mismo espacio de direcciones y compartiendo la misma área de memoria, pone en marcha un nuevo puntero de instrucción y comienza a ejecutar sentencias [13]. Un proceso puede controlar muchos hilos, esto es bueno para la administración de recursos. Mediante el uso de hilos en lugar de procesos, se puede atender un mayor número de solicitudes de clientes utilizando menos recursos del sistema que un servidor basado en procesos. En este tipo de servidores, las solicitudes pueden compartir memoria, haciéndolas más eficientes y por lo tanto capaces de atender más solicitudes de forma más rápida, sin embargo, un bloqueo en un hilo podría hacer que todo el servidor caiga [14].

Servidores basados en eventos: en la arquitectura controlada por eventos, un bucle de eventos de un solo proceso de un solo hilo ejecuta múltiples conexiones mediante mecanismos de E/S no bloqueantes. Puede descargar el cuello de botella en el rendimiento en caso de una alta carga de conexión gracias a su comunicación asíncrona. La arquitectura basada en eventos puede atender más solicitudes concurrentes que la basada en procesos o en hilos [12]. Esta arquitectura basa su funcionamiento en la utilización de lecturas y escrituras asíncronas sobre sockets. En lugar de iniciar un nuevo proceso o un nuevo hilo para cada solicitud, en una arquitectura impulsada por eventos, el flujo del programa está controlado por eventos, alguna forma de mensaje enviado desde otro proceso o hilo. Aquí, tiene un proceso que se ejecuta como un "detector" o "detector de eventos" que espera que una solicitud ingrese al servidor. Cuando llega la solicitud, en lugar de comenzar un nuevo proceso, se envía un mensaje a una parte diferente del servidor llamada "controlador de eventos", que luego realiza una tarea. El resultado simplificado de servir páginas web de esta manera es que se necesita menos tiempo de procesador y menos memoria [14].

Actualmente la mayoría de los diseños de servidores web de alto rendimiento implementan arquitecturas híbridas. Apache proporciona una arquitectura flexible basada en procesos y basada en hilos, mientras que Nginx presenta una potente arquitectura multiproceso e impulsada por eventos [12]. A continuación se describen ambos servidores.

B. Arquitectura de Apache 2

En Apache 2 cada solicitud es atendida por un hilo separado o proceso y utiliza sockets sincrónicos [15]. Con la creación de la versión 2.0 se introducen los Módulos de Multiprocesamiento (MPM), como una solución propuesta por el Grupo Apache debido a que la anterior arquitectura no trabajaba bien bajo plataformas que no estaban centradas en procesos como en el caso de Windows [16]. Los MPM son responsables de conectar con los puertos de red de la PC servidora, aceptar las peticiones y generar los procesos hijos que se encargan de servirlas. Existen diferentes MPM, encontrándose entre ellos prefork, worker, event, threaded y perchild, cada uno basado en un funcionamiento diferente. Los sitios web que necesitan más que nada escalabilidad pueden usar un MPM hebrado como worker, mientras que los sitios web que requieran por encima de otras cosas estabilidad o compatibilidad con software antiguo pueden usar prefork [17]. A continuación se describen los tres MPM que trae Apache 2 por defecto.

MPM prefork: este MPM crea un grupo de procesos hijos para servir peticiones. Cada proceso hijo tiene un solo hilo, por lo que si Apache inicia 20 procesos hijos, puede servir 20 peticiones simultáneamente, como se aprecia en la Fig. 1. En caso que ocurra un error grave y un proceso hijo muera, solo se pierde una petición: la que está atendiendo el hijo que acaba de cesar la actividad. El máximo y el mínimo de procesos hijos son parámetros configurables, permitiendo adaptar Apache a las capacidades de la PC servidora y también fijarle un máximo de carga. En el MPM prefork el rendimiento se degrada rápidamente después que las solicitudes superen el número de procesos, por lo que esta no es una buena opción en muchos escenarios. Cada proceso tiene un impacto significativo en el consumo de RAM, por lo que este MPM es difícil de escalar de manera efectiva. Puede ser una buena opción si se usa junto con otros componentes que no están construidos con hilos como PHP (el cual no es seguro para hilos), por lo que este MPM se recomienda como la única forma segura de trabajar con mod_php, el módulo de Apache para procesar estos archivos [18].

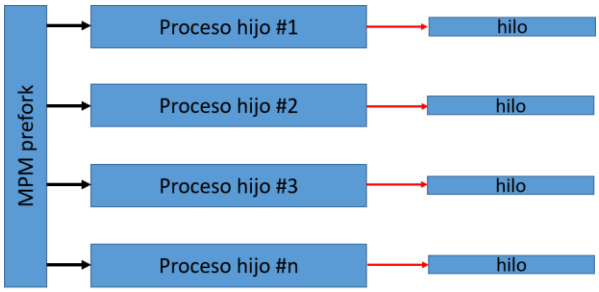

Fig. 1 Funcionamiento del MPM prefork (Fuente: elaboración propia).

MPM worker y MPM event: el MPM event es una mejora de worker pero se basa en el mismo funcionamiento multiproceso-multihilo, como se aprecia en la Fig. 2. Usan

hilos para atender peticiones, el servidor puede servir un mayor número de peticiones con menos recursos del sistema que un servidor basado únicamente en procesos. Si Apache inicia 20 procesos hijos y cada uno 30 hilos, se pueden servir 600 peticiones simultáneamente. El MPM event es similar a worker en la mayoría de las situaciones, pero está optimizado para manejar conexiones keep-alive [18]. Tanto en worker como event, hay dos opciones de rendimiento principales que se pueden ajustar. El primero es ThreadsPerChild, y el segundo es MaxRequestWorkers. ThreadsPerChild describe cuántos hilos se crean desde cada proceso hijo, mientras que MaxRequestWorkers determina los hilos máximos lanzados en total. Una sola conexión web de un usuario puede lanzar más de un hilo. Cada conexión de usuario aumenta la cantidad de recursos del sistema, como la RAM y el tiempo de CPU [19].

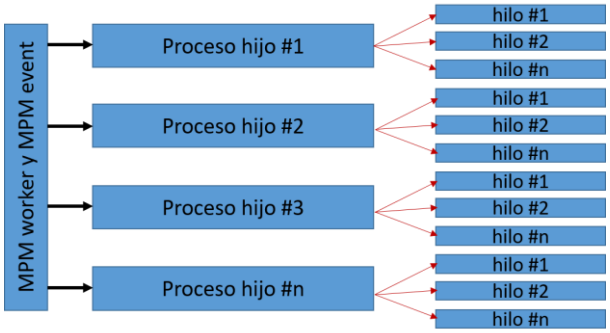

Fig. 2 Funcionamiento de los MPM worker y event (Fuente: elaboración propia).

B. *Arquitectura de Nginx*

Nginx utiliza una arquitectura basada en eventos, una única instancia de Nginx consta de un proceso maestro y procesos de trabajo. Cada proceso de trabajo maneja conexiones múltiples, esto se logra ejecutando un bucle de eventos que extrae eventos que ocurrieron en sockets abiertos desde el sistema operativo. El sistema operativo es el encargado de distribuir las conexiones entrantes entre los procesos de trabajo de forma rotatoria, ver Fig. 3 [20]. Desde el archivo de configuración se puede definir la cantidad de procesos, las conexiones máximas por proceso y otros parámetros [4].

Cada proceso de trabajo en Nginx tiene un solo hilo y se ejecuta de forma independiente. Su trabajo principal es obtener nuevas conexiones y procesarlas lo más rápido posible. Cuando se inician los procesos de trabajo, se inicializan con la configuración y el proceso maestro les dice que escuchen los sockets configurados. Una vez activos, leen y escriben contenido en el disco y se comunican con los servidores ascendentes. Como están todos bifurcados del proceso maestro, pueden usar la memoria compartida para datos en caché, datos de persistencia de sesión y otros recursos compartidos. En Windows, un hilo es comparativamente mucho más liviano que un proceso, a diferencia de Linux, donde sincronizar la memoria compartida es costoso, por eso en Nginx decidieron no crear múltiples hilos por proceso. El proceso maestro lee y evalúa los archivos de configuración, y también mantiene los procesos de trabajo. Es el proceso de trabajo el que hace el procesamiento real de las solicitudes [21].

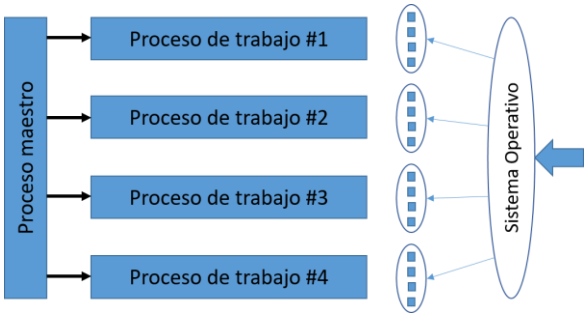

Fig. 3 Funcionamiento de la arquitectura de Nginx (Fuente: [20]).

Nginx es más eficiente que sus competidores, primero y ante todo, es más rápido al hacer uso de sockets asincrónicos. Nginx no genera procesos tantas veces como recibe solicitudes. Un proceso por núcleo es suficiente para manejar miles de conexiones, lo que genera una carga de CPU y un consumo de memoria mucho más ligeros. La razón por la cual Nginx está muy por delante de Apache en términos de rendimiento es porque es precisamente la razón por la que fue escrito [15].

C. *Correcto funcionamiento del servidor web*

Se realizó un estudio sobre Apache 2 y Nginx, identificándose que Nginx se usa con frecuencia como un servidor proxy [21]. El objetivo principal de configurar Nginx como un servidor de interfaz y darle a Apache una función de back-end, es mejorar la velocidad de publicación. Nginx tendría que diferenciar entre contenido estático y dinámico, en consecuencia, atender las solicitudes de archivos estáticos y reenviar las solicitudes de archivos dinámicos a un servidor back-end. Nginx actúa como un servidor proxy simple: recibe solicitudes HTTP del cliente (actuando como servidor HTTP) y las reenvía al servidor back-end (actuando como cliente HTTP) [15].

Se determina que la configuración de Nginx como proxy inverso de Apache 2 es una buena solución para entornos donde se desee instalar Apache 2 y a la misma vez obtener las ventajas de Nginx para el manejo de peticiones. Además de analizar cada servidor web de forma independiente, la autora decide incluir la combinación de Nginx como proxy inverso de Apache 2 como alternativa para las instituciones cubanas.

A partir de una comparación realizada entre ambos servidores, se puede llegar a la conclusión que Apache 2 y Nginx tienen muchas características en común, sin embargo la diferencia de mayor impacto entre ambos servidores es la forma en que atienden las peticiones realizadas por el usuario. Esto proporciona quizás la diferencia más significativa en la

forma en que responden a las diferentes condiciones del tráfico [18]. En cuanto al funcionamiento del servidor web, el otro elemento que distingue a ambos servidores es el soporte para protocolos basados en CGI, la forma en que procesan el contenido dinámico. A partir del estudio realizado, la autora elaboró el esquema que se aprecia en la Fig. 4.

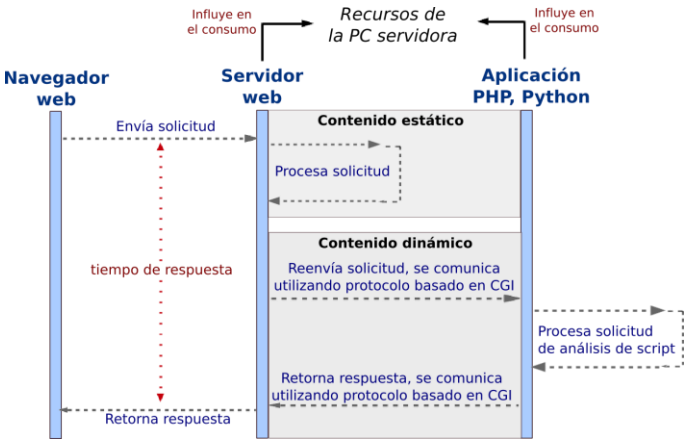

Fig. 4 Esquema de funcionamiento del servidor web (Fuente: elaboración propia).

Como se puede observar, el tiempo de respuesta y el consumo de recursos dependen de cómo el servidor maneje las peticiones que recibe y la forma en que procese el contenido a través de los protocolos basados en CGI. Por tanto la autora determina que el correcto funcionamiento del servidor web depende de la arquitectura, la cantidad de peticiones concurrentes y el tipo de contenido publicado.

C. Estudio de caso

La eficiencia en la selección de los servidores web Apache 2 y Nginx implica el desempeño adecuado del servidor web que se seleccione para un entorno determinado. Por tanto, se hace necesario el estudio y análisis de la eficiencia de ambos servidores mediante un estudio de caso. El objetivo es definir la diferencia entre los servidores web Apache 2 y Nginx en cuanto a la eficiencia de desempeño, teniendo en cuenta los indicadores de la Norma Cubana ISO/IEC 25023:2017 [22]: tiempo medio de conclusión de un trabajo, adecuación del tiempo de conclusión de un trabajo, rendimiento medio, la media de utilización del procesador, la media de utilización de la memoria, la media del uso de los dispositivos de entrada salida (E/S), utilización del ancho de banda, capacidad de procesamiento de transacciones y capacidad de acceso de usuario.

El estudio se basa en los elementos de los que depende el funcionamiento del servidor web, definidos en la sección anterior: arquitectura del servidor (se evidencia en el servidor web en cuestión), la cantidad de peticiones concurrentes y el tipo de contenido publicado. La PC servidora se estudiará en tres escenarios de prueba diferentes en cuanto al tipo de contenido que puede ser: estático, dinámico con PHP o dinámico con Python. A su vez para el tipo de contenido estático existen dos escenarios en cuanto al servidor web instalado que puede ser Apache 2 o Nginx, por otra parte para tipo de contenido dinámico con ambos lenguajes de programación se encuentran tres escenarios: Apache 2, Nginx o Nginx funcionando como proxy inverso de Apache 2 (la autora de la investigación lo denominó Proxy). De lo anteriormente explicado se concluye que para cada uno de los nueve indicadores de la variable eficiencia de desempeño, se estudiarán ocho escenarios.

El estudio consistió en realizar con la herramienta *ab* (*Apache Benchmark*) un número determinado de peticiones con cierta concurrencia, desde una PC cliente al servidor web instalado en la PC servidora. En cada uno de los ocho escenarios se realizaron 10 observaciones, para todas con un total de 50 000 peticiones y cada una con concurrencias de 10, 100, 250, 500, 750, 1 000, 5 000, 10 000, 15 000 y 20 000 peticiones respectivamente.

Además al mismo tiempo se realizó el monitoreo de los recursos de la PC servidora con la herramienta *dstat*. Para el cálculo de la eficiencia en cada indicador se definieron cinco escenarios en cuanto a la cantidad de peticiones concurrentes que pueden existir en las instituciones (denominada con la variable *p*), teniendo en cuenta las respuestas de los administradores de servidores web en una entrevista realizada. A continuación se describen cada uno de los escenarios.

1) $p \leq 500$: se refiere a una cantidad de peticiones concurrentes menor o igual que 500. Se incluyen cuatro observaciones con concurrencias 10, 100, 250 y 500 respectivamente.

2) $500 < p \leq 1000$: se refiere a una cantidad de peticiones concurrentes mayor que 500 y menor o igual que 1 000. Se incluyen dos observaciones con concurrencias 750 y 1 000 respectivamente.

3) $1000 < p \leq 10000$: se refiere a una cantidad de peticiones concurrentes mayor que 1 000 y menor o igual que 10 000. Se incluyen dos observaciones con concurrencias 5 000 y 10 000 respectivamente.

4) $p > 10000$: se refiere a una cantidad de peticiones concurrentes mayor que 10 000. Se incluyen dos observaciones con concurrencias 15 000 y 20 000 respectivamente.

5) Todas: se refiere a todas las concurrencias de peticiones incluyendo las 10 observaciones.

Para cada indicador se realizó el análisis correspondiente. En cuanto a la utilización de la memoria, durante la ejecución de la prueba no se usó la swap, la capacidad de consumo reportada fue de 0 MB tanto para contenido estático como dinámico con PHP y Python. En la Fig. 5 se aprecian cada una de las 10 observaciones, el tipo de contenido estático se expresa con columnas agrupadas, el dinámico con PHP mediante líneas y el dinámico con Python a través de líneas con marcadores. Para contenido estático, los dos servidores web consumen menos Memoria de Acceso Aleatorio (RAM,

del inglés *Random Access Memory*) que para contenido dinámico. Apache 2 es el servidor que más memoria utiliza, además el consumo aumenta más a mayor concurrencia de peticiones. Para contenido dinámico los tres servidores web utilizan más memoria con Python que con PHP. En ambos casos Apache utiliza notablemente más memoria que los demás servidores.

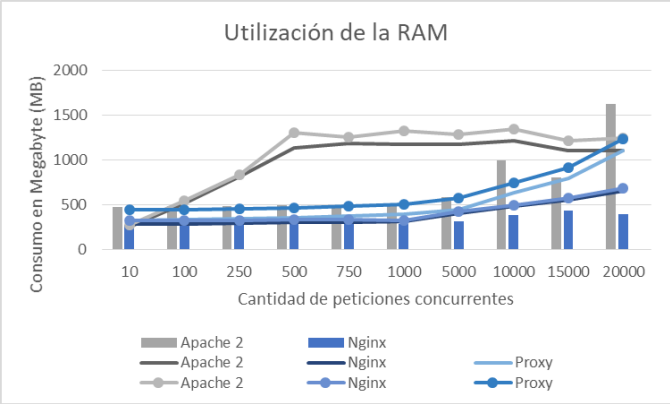

Fig. 5 Utilización de la RAM (Fuente: elaboración propia).

A partir del análisis de la eficiencia calculada en cada indicador, con el objetivo de medir todos los indicadores en una misma escala, se asignó en cada escenario en cuanto a la concurrencia de peticiones y atendiendo al tipo de contenido, los valores 3, 2 y 1 a los servidores web en correspondencia con el orden de su eficiencia, de mayor a menor. Para casos de igual eficiencia se asignó el mismo valor. Para cada uno de los cinco escenarios en cuanto a la concurrencia de peticiones, se calculó la puntuación final para cada servidor web teniendo en cuenta todos los indicadores y además para cada conjunto de indicadores correspondientes a las tres medidas de la eficiencia que son Rendimiento, Utilización de los recursos y Capacidad. Partiendo de la teoría definida, se desarrolló la solución como parte de la Herramienta para la Migración y Administración de Servicios Telemáticos.

III. RESULTADOS

La Herramienta para la Migración y Administración de Servicios Telemáticos (HMAST) en su versión 2.0 contiene varios módulos, dos de ellos corresponden a los servidores web Apache 2 y Nginx respectivamente. La herramienta permite la administración remota a través del protocolo SSH (*Secure SHell*) de diferentes PC servidoras. La solución informática de la presente investigación se sustenta en añadir a HMAST un componente Web que permita seleccionar el servidor web que se ajuste a la institución, a partir del resultado obtenido con la aplicación del estudio de caso del epígrafe anterior.

La herramienta presenta una arquitectura N-Capas orientada al dominio, distribuida en cinco componentes o paquetes: Presentación, Aplicación, Dominio, Persistencia e Infraestructura Transversal. La interacción entre los mismos se realiza a través de interfaces y utilizando inyección de dependencias [4]. En las capas de Presentación, Aplicación, Dominio y Persistencia se inserta un paquete llamado Web que a su vez contiene dos paquetes, Apache2 y Nginx, con todo lo referente a cada módulo.

Como solución de la presente investigación, en lugar de tener los módulos Apache 2 y Nginx aisladamente, se inserta en HMAST un componente Web que contiene dentro ambos módulos, que inicialmente están desactivados. Cuando se accede por primera vez al componente, se deben completar los campos en la opción Seleccionar servidor web: Cantidad total de usuarios que acceden al contenido (número entero positivo); Tipo de contenido (Estático/Dinámico); En caso de ser dinámico especificar lenguaje de programación (PHP/Python); Medida de la eficiencia que desea priorizar (Todos/Rendimiento/Utilización de los recursos/Capacidad).

Posteriormente se presiona el botón Mostrar resultado, la herramienta tomando como referencia los 60 escenarios obtenidos en la Tabla 1, visualiza el nombre del servidor web más indicado según los elementos especificados anteriormente. Se muestra la opción Instalar servidor web, que permite instalar el servidor seleccionado o ambos servidores en caso de ser Proxy la opción resultante. En este último caso, el sistema instala primero Apache 2 y se habilita solo el puerto 8080 escuchando por 127.0.0.1 para posteriormente instalar Nginx y que no existan conflictos. El usuario puede desinstalar el servidor web en el momento deseado.

Tabla 1 Escenarios definidos para la selección del servidor web (Fuente: elaboración propia).

Cantidad de peticiones concurrentes (p)	**Indicadores**	**Tipo de contenido**		
		Estático	**Dinámico**	
			PHP	**Python**
p≤500	**Rendimiento**	Nginx	Nginx	Nginx
	Recursos	Nginx	Nginx	Nginx
	Capacidad	Apache	Nginx	Proxy
	Todos	**Nginx**	**Nginx**	**Nginx**
500<p≤1000	**Rendimiento**	Nginx	Nginx	Proxy
	Recursos	Nginx	Nginx	Nginx
	Capacidad	Apache	Nginx	Nginx Proxy
	Todos	**Nginx**	**Nginx**	**Nginx**
1000<p≤10000	**Rendimiento**	Apache	Proxy	Proxy
	Recursos	Nginx	Nginx	Nginx
	Capacidad	Nginx	Proxy	Nginx Proxy
	Todos	**Nginx**	**Proxy**	**Nginx**
p>10000	**Rendimiento**	Nginx	Proxy	Proxy
	Recursos	Nginx	Nginx	Nginx
	Capacidad	Apache Nginx	Nginx	Nginx Proxy
	Todos	**Nginx**	**Nginx**	**Nginx**
Todas	**Rendimiento**	Nginx	Proxy	Proxy
	Recursos	Nginx	Nginx	Nginx
	Capacidad	Apache Nginx	Nginx	Nginx Proxy
	Todos	**Nginx**	**Nginx**	**Nginx**

Como se puede apreciar, teniendo en cuenta todos los indicadores, el servidor web seleccionado en todos los escenarios es Nginx excepto para contenido dinámico con PHP y concurrencia de peticiones entre 1 000 y 10 000, que el seleccionado es Proxy. En cuanto al consumo de recursos el más eficiente en todos los casos es Nginx. Teniendo en cuenta el rendimiento y la capacidad, los tres servidores son elegidos respectivamente en algunos escenarios. A continuación se describen los módulos Apache 2 y Nginx.

A. Módulo Apache 2

En el módulo Apache 2 se pueden administrar las tres secciones de configuración: entorno global, servidor principal y hosts virtuales. A continuación se describen cada una de las funcionalidades.

Especificar estado del servidor: las acciones de iniciar, reiniciar y detener el servidor web se realizan en el momento que el usuario lo desee. En el caso de recargar se realiza de forma automática después de aplicar los cambios a la PC servidora.

Instalar MPM a usar: uno de los aspectos relevantes que le permite a Apache 2 adaptarse a los diferentes entornos existentes es la instalación del MPM a usar. La interfaz muestra los tres MPM que trae Apache 2 por defecto (prefork, worker y event), lo que se debe seleccionar cuál es el que estará cargado, que debe ser solo uno.

Especificar parámetros en los MPM prefork, worker y event: se pueden gestionar los parámetros de cada uno de los MPM. En el caso de prefok se especifica el número de procesos hijos que se crean al iniciar Apache; los números mínimo y máximo de procesos hijos inactivos; el número máximo de procesos hijos que pueden crearse; el número máximo de peticiones que un proceso hijo atenderá durante su existencia donde el valor 0 es para ilimitadas peticiones.

Especificar parámetros en el servidor principal: se especifican parámetros como la dirección de correo del administrador; la información sobre versión del servidor, que es la información que se devuelve dentro de la cabecera HTTP que envía el servidor, donde los posibles valores de menor a mayor información son Prod, Min, Os y Full; y la ruta del fichero de registro de errores.

Especificar parámetros en el manejo de conexiones: se pueden modificar parámetros como el tiempo que Apache esperará antes de cerrar la conexión; el estado de la utilización de conexiones HTTP persistentes; el tiempo que el servidor esperará peticiones subsiguientes en conexiones persistentes y el número de solicitudes permitidas por conexión donde la cantidad es ilimitada para valor 0.

Especificar módulos a usar en Apache 2: se muestra una lista con todos los módulos existentes en Apache y de ellos están marcados los habilitados. Para habilitar o deshabilitar uno o más módulos, deben seleccionarse y presionar el botón Habilitar o Deshabilitar respectivamente.

Modificar puertos para las conexiones: la modificación de los puertos utilizados para las conexiones en Apache 2 se realizará de forma implícita cuando se guarden todos los cambios realizados, una vez que haya finalizado la gestión de todos los hosts virtuales. Se adicionarán todos los puertos que hayan sido asignados a algún host virtual, en caso que no se encuentren en la lista de puertos de Apache 2.

Especificar permisos de acceso a directorios y ficheros: se listan los directorios y ficheros a los que se les han establecido permisos de control de acceso. En cada directorio o fichero se especifica si se permite o deniega el acceso desde diferentes orígenes, donde cada uno puede ser un IP completo, un IP parcial, un nombre de dominio completo, un nombre de dominio parcial o una combinación IP/máscara.

Gestionar hosts virtuales: la funcionalidad de mayor impacto es gestionar hosts virtuales, la cual facilita un mejor entendimiento por parte de los administradores para la gestión de estos debido a que se muestra de forma detallada la estructura de almacenamiento lógico de los mismos. Apache soporta hosts virtuales basados en nombre y/o hosts virtuales basados en IP. Una PC servidora puede tener asignadas una o varias direcciones IP y cada una de ellas puede escuchar por una lista de puertos específicos y además por todos los puertos, que en la configuración se representa con el símbolo de asterisco (*). A cada combinación direcciónIP:puerto se puede asignar un host virtual basado en IP o una lista de hosts virtuales basados en nombre. Para gestionar los hosts virtuales primeramente se listan todos los asignados a la PC servidora, con un filtro para la búsqueda en dependencia de la dirección a la que están asignados (todos los IP, un IP específico o un nombre de dominio). Los hosts se agrupan por los puertos de escucha. También se muestra si están habilitados o deshabilitados. Con los botones que aparecen en la parte superior derecha del listado se podrán adicionar nuevos hosts, editar alguno de los existentes, eliminar, activar o desactivar uno o varios.

Adicionar host virtual: para la adición de un host virtual existen tres grupos de parámetros: Generales, Permisos de acceso y Otros parámetros. En Generales como campos obligatorios están el nombre del fichero que contendrá la configuración del host virtual que es el identificador del host pues no puede repetirse y el directorio raíz que es donde se almacena el contenido a publicar por ese host virtual. Además se requiere especificar una lista con las combinaciones dirección:puerto, donde dirección puede ser un IP específico, un nombre de dominio, todos los IP (se representa con el símbolo de asterisco) o por defecto (se representa como _default_); el puerto puede ser un número entre 1 y 65535 o todos (se representa con el símbolo de asterisco). En caso que Nginx funcione como proxy inverso de Apache 2, la combinación especificada es 127.0.0.1:8080. Se detallan también parámetros asociados con el soporte SSL y los errores. En Permisos de acceso se listan los directorios y ficheros a los que se les han establecido permisos de control de acceso. En Otros parámetros se gestionan alias del nombre, archivos índices y alias de directorios o ficheros.

B. Módulo Nginx

El módulo cuenta con las funcionalidades necesarias para configurar las conexiones en el servidor, controlar el tráfico HTTP, gestionar los hosts virtuales, configurar los permisos de acceso y alias para cada host virtual. A continuación se describen cada una de las funcionalidades [23]:

Especificar estado del servidor: el sistema permite al usuario iniciar, detener, reiniciar o recargar el servidor web Nginx. Para ejecutar iniciar, el servicio debe estar detenido, por otra parte para detener o reiniciar, el servicio debe estar ejecutándose. La acción recargar se ejecuta automáticamente después de aplicar los cambios a la PC servidora.

Gestionar hosts virtuales: el sistema permite al usuario mostrar el listado de todos los hosts virtuales existentes en el servidor web Nginx, en caso de estar vacío, se muestra un mensaje notificándolo. Para adicionar un host virtual los parámetros obligatorios son el nombre del fichero que contendrá la configuración del host virtual; la dirección IP que puede ser un IP específico de los asignados a la PC servidora o un asterisco (*) en caso de que se defina acceder a un host virtual desde cualquier dirección IP; el puerto de escucha asignado al host virtual que debe ser un número entero positivo entre 1 y 65535; el directorio raíz el cual almacena el contenido a publicar y el nombre del servidor que se refiere al nombre de dominio asignado al host virtual. Además se pueden especificar los archivos índices del directorio raíz del host virtual. Otro elemento es si Nginx funcionará como proxy inverso de Apache 2, con argumento on/off respectivamente, por defecto es (off), en caso que el usuario especifique (on), el sistema realiza la configuración en el host virtual para enviar el contenido dinámico a Apache 2, tomando el lenguaje de programación utilizado de los elementos especificados en la selección.

Una vez introducidos dichos parámetros se crea en la PC donde está instalada HMAST, el fichero de configuración del host. Para realizar la edición debe existir al menos un host virtual en el listado, posteriormente se selecciona uno y se presiona el botón Editar para que muestre la interfaz de edición. Para eliminar uno o varios hosts virtuales el usuario selecciona el/los que desea eliminar, una vez seleccionados se presiona el botón Eliminar, se muestra un mensaje de confirmación. El sistema permite al usuario habilitar o deshabilitar un host virtual, el usuario accede a la lista de los hosts virtuales y selecciona el/los que desea habilitar/deshabilitar. Posteriormente presiona el botón correspondiente para realizar la acción deseada, donde se muestra un mensaje de confirmación.

Especificar parámetros para las conexiones: el sistema permite al usuario especificar los parámetros referentes a las conexiones: el número máximo de conexiones simultáneas por procesos; si el servidor acepta o no varias peticiones por conexión, con argumento on/off que significa que el servidor acepta conexiones persistentes (on) o no acepta conexiones persistentes (off); la cantidad de segundos entre peticiones y el número máximo de procesos, si el valor de dicha directiva es auto, significa que el servidor intenta detectar de forma automática el número de núcleos disponibles de la CPU.

Otras funcionalidades son las referentes a la gestión de los directorios donde se almacena la información que va a publicar un host virtual, el sistema permite listar, adicionar, editar, mostrar y eliminar un directorio. El sistema permite además mostrar y editar las configuraciones del tráfico HTTP.

IV. VALIDACIÓN

La validación de la herramienta informática tiene la finalidad de verificar el cumplimiento del objetivo propuesto, que es aumentar la eficiencia en la selección de los servidores web Apache 2 y Nginx durante el proceso de migración a código abierto. Primeramente se evalúa la herramienta informática con la realización de un estudio de caso midiendo cada uno de los indicadores de la eficiencia de desempeño. Además se aplica la técnica de Iadov para determinar el nivel de satisfacción de los especialistas en servicios telemáticos y se evalúa la aplicabilidad de la solución con el criterio de expertos a través del método Delphi.

A. Estudio de caso en ECOAIND3

Se realizó un estudio de caso en la Empresa Constructora de Obras de Arquitectura e Industriales No. 3, que tuvo como objetivo comparar la eficiencia del servidor web seleccionado en dos proyectos de migración a código abierto: el proyecto de migración a aplicaciones de código abierto y el proyecto de actualización a la migración de los servidores. En el primero sin aplicar la solución se seleccionó Apache2 y en el segundo aplicándola, el servidor web seleccionado fue Nginx debido a que la empresa posee un total de 200 usuarios y el contenido publicado es estático (repositorio de Nova y la actualización del antivirus). Para la medición de la eficiencia se empleó la Norma Cubana ISO/IEC 25023:2017. Se realizaron cuatro observaciones, utilizando la herramienta ab mediante la cual se crearon un total de 200 peticiones con concurrencias 10, 100, 150 y 200 respectivamente, desde una PC cliente al servidor web instalado en la PC servidora. Al mismo tiempo se realizó el monitoreo de los recursos de la PC servidora con la herramienta dstat. Finalmente se realizó la comparación de la eficiencia de ambos servidores web, ver Tabla 2.

Tabla 2 Cálculo de la eficiencia en ECOAIND3 (Fuente: elaboración propia).

Indicadores	Apache	Nginx
Tiempo medio de conclusión de un trabajo (ms)	562,250	190,250
Adecuación del tiempo de conclusión de un trabajo (ms)	2,811	0,951
Rendimiento medio	1,783	5,264
La media de utilización del procesador	0,012	0,005
La media de utilización de la memoria	0,275	0,126
La media del uso de los dispositivos de E/S	0,090	0,090
Utilización del ancho de banda	0,647	0,543
Capacidad de procesamiento de transacciones	2,864	2,921
Capacidad de acceso de usuario	200	200

En la tabla se muestra en negrita el valor resultante del servidor más eficiente, donde en siete indicadores Nginx supera a Apache 2, se concluye que con la aplicación de la herramienta en ECOAIND3 se aumentó la eficiencia en la selección del servidor web.

B. Criterio de expertos

Existen varios métodos para la obtención de juicios de expertos, que pueden clasificarse según si la evaluación se realiza de manera individual o grupal. En el primer grupo se encuentra el método Delphi [24], que es un método de consenso que tiene como objetivo encontrar un acuerdo general entre un panel de expertos sobre un tema de investigación específico [25]. Cada juez realiza la evaluación individualmente, pero luego de analizar las respuestas se le envía a cada juez la mediana obtenida y se le pide que reconsidere su juicio hasta que se logre un consenso. Esta técnica ofrece un alto nivel de interacción entre los expertos [24].

Para la selección de los expertos se aplicó una encuesta a 10 especialistas que han desempeñado el rol de especialistas en servicios telemáticos en el Centro de Software Libre de la UCI, a partir de la información expresada en la misma se obtuvo el nivel de competencia de cada experto, donde el 90% posee un nivel de competencia Alto y un 10% un nivel Medio. Para la evaluación de la solución, se realizó el envío a los expertos de un documento que describe la solución desarrollada y una encuesta, donde se solicitó que evaluaran cinco aspectos relacionados con la misma. La evaluación de los aspectos se basó en cinco categorías: Muy Adecuado (MA), Bastante Adecuado (BA), Adecuado (A), Poco Adecuado (PA) y Nada Adecuado (NA).

Posteriormente se emplea el Modelo de Torgerson, que permite la conversión de la escala original cualitativa en una escala de intervalo (cuantitativa) y la valoración de cada uno de los aspectos de forma individual [26]. A partir de las valoraciones de los expertos se calculó la frecuencia absoluta por aspecto, la frecuencia acumulada por aspecto, la frecuencia acumulada relativa y los Puntos de Corte, que su ubicaron en una recta numérica, clasificando de esta forma cada aspecto sometido a evaluación. Se concluye que de los cinco aspectos, tres se evalúan de Muy Adecuado (60%) y dos de Bastante Adecuado (40%), sin mostrarse ninguno de los casos como Adecuado, Poco adecuado o Nada adecuado. Se demostró la existencia de un 100% de concordancia con la solución.

Los resultados de la validación de la herramienta informática a través del criterio de expertos permite confirmar la aplicabilidad de esta y concluir que:

- El apoyo de la herramienta informática al proceso de migración de servicios telemáticos; la posibilidad que la herramienta informática seleccione el servidor web teniendo en cuenta el tipo de contenido a publicar y el total de peticiones concurrentes; y los 60 escenarios definidos para la selección del servidor, son tres aspectos evaluados de Muy Adecuado.
- La herramienta informática como vía para una correcta selección del servidor web a instalar; y como medio para aumentar la eficiencia en la selección de los servidores web Apache 2 y Nginx, son dos aspectos evaluados de Bastante Adecuado.

C. Técnica de Iadov

Se realizó una encuesta a 10 especialistas en servicios telemáticos pertenecientes al Centro de Software Libre de la UCI utilizando la técnica de Iadov, para determinar su grado de satisfacción con la herramienta propuesta a partir de su experiencia en la migración y administración de servicios telemáticos. La técnica de Iadov se basa en el análisis de un cuestionario que tiene una estructura interna determinada, la cual sigue las relaciones que se establecen entre tres preguntas cerradas (cuya relación el sujeto desconoce) y el análisis posterior de cinco preguntas abiertas. La relación entre las preguntas cerradas se establece a través del denominado "Cuadro lógico de Iadov", indicando la posición de cada persona en la escala de satisfacción.

El número resultante de la interrelación de las tres preguntas indica la posición de cada sujeto en la escala de satisfacción. La escala de satisfacción es la siguiente: (1 = Clara satisfacción; 2 = Más satisfecho que insatisfecho; 3 = No definida; 4 = Más insatisfecho que satisfecho; 5 = Clara insatisfacción; 6 = Contradictoria). De los 10 especialistas encuestados, 7 respondieron clara satisfacción, 2 más satisfechos que insatisfechos y 1 indefinido. A continuación se calcula el índice de satisfacción grupal (ISG) a partir de la fórmula (1), donde A, B, C, D, E, representan el número de sujetos con índice individual 1; 2; 3 ó 6; 4; 5, respectivamente y donde N representa el número total de sujetos del grupo. 5

$$ISG = \frac{A(1) + B(0{,}5) + C(0) + D(-0{,}5) + E(-1)}{N} \quad (1)$$

$$ISG = \frac{7(1) + 2(0{,}5) + 1(0) + 0(-0{,}5) + 0(-1)}{10} = 0{,}8$$

Posteriormente se trabaja con los diferentes niveles de satisfacción que se expresan en la escala numérica que oscila entre +1 y - 1, debido a que el valor 0,8 se encuentra entre 0,5 y 1, indica que los especialistas en servicios telemáticos presentan satisfacción con la herramienta informática.

D. Triangulación metodológica

La triangulación es un procedimiento de validez en el que los investigadores buscan la convergencia entre múltiples y diferentes fuentes de información para formar temas o categorías en un estudio. La triangulación metodológica se define como el uso de más de dos métodos para estudiar el mismo fenómeno bajo investigación. Se basa en el uso de métodos y análisis de recolección de datos tanto cualitativos como cuantitativos en el estudio del mismo fenómeno [27]. Se concluye que los resultados obtenidos en los métodos cuantitativos y cualitativos coinciden, por lo que se puede

afirmar que la herramienta informática aumenta la eficiencia en la selección de los servidores web Apache 2 y Nginx durante el proceso de migración a código abierto

IV. CONCLUSIONES

Al finalizar la presente investigación se concluye lo siguiente:

- El análisis de las fuentes bibliográficas relacionadas con los servidores web Apache 2 y Nginx, permitió incluir como alternativa para las instituciones cubanas la configuración de Nginx como proxy inverso de Apache 2 y determinar que el correcto funcionamiento del servidor web depende de la arquitectura del servidor, la cantidad de peticiones concurrentes y el tipo de contenido publicado.
- El empleo de un estudio de caso aplicando la Norma Cubana ISO/IEC 25023:2017 y calculando la eficiencia de cada servidor web a partir del tipo de contenido y la cantidad de peticiones concurrentes, permitió el desarrollo de un componente Web en la Herramienta para la Migración y Administración de Servicios Telemáticos, que selecciona el servidor web más eficiente en cuanto a 60 escenarios definidos.
- La validación de la herramienta informática mediante un estudio de caso en ECOAIND3 analizando dos proyectos de migración; el empleo del criterio de expertos a través del método Delphi y el modelo de Torgerson; la aplicación de la técnica de Iadov para medir el grado de satisfacción; y la triangulación metodológica; permitió evidenciar el cumplimiento del objetivo de la presente investigación.

AGRADECIMIENTOS

Al Centro de Software Libre (CESOL) de la Universidad de las Ciencias Informáticas y especialmente al equipo de desarrollo de la Herramienta para la Migración y Administración de Servicios Telemáticos (HMAST).

REFERENCIAS

[1] STATISTA. Number of internet users worldwide from 2009 to 2017, by region (in millions). [En línea]. 2018b. [Consultado el: 13 de diciembre de 2018]. Disponible en: [https://www.statista.com/statistics/265147/number-of-worldwide-internet-users-by-region/].

[2] STATISTA. ¿Cuántos usuarios de Internet hay en América Latina? [En línea]. 2018c. [Consultado el: 13 de diciembre de 2018]. Disponible en: [https://es.statista.com/grafico/13903/cuantos-usuarios-de-internet-hay-en-america-latina/].

[3] VAN, Tien; KRIEGER, Udo R.; CHAKKA, Ram. Performance modeling of an apache web server with a dynamic pool of service processes. Telecommunication Systems, 2008, 39 (2): p. 117-129.

[4] PALMA, Nurisel. Módulo para la administración de los servidores web en HMAST. Tesis para optar por el título de Ingeniero en Ciencias Informáticas, Universidad de las Ciencias Informáticas, La Habana, 2013.

[5] NETCRAFT. Most Reliable Hosting Company Sites in November 2018. [En línea]. 2018. [Consultado el: 6 de diciembre de 2018]. Disponible en: [https://news.netcraft.com/archives/2018/].

[6] W3TECHS. Usage of web servers for websites. [En línea]. 2018. [Consultado el: 6 de diciembre de 2018]. Disponible en: [https://w3techs.com/technologies/overview/web_server/all].

[7] ETECSA. Sitios web y portales nacionales de interés informativo, cultural e investigativo. [En línea]. 2018. [Consultado el: 5 de diciembre de 2018]. Disponible en: [http://www.etecsa.cu/inicio/sitios_web_nacionales/].

[8] RAKHMAWATI, Nur Aini, et al. A survey of web Technologies in Indonesia Local Governments. Jurnal Sisfo, 2018, 7 (3): p. 213-222.

[9] RODRÍGUEZ, Lissy. Informatización de la sociedad: principios y resultados de una política. [En línea]. Periódico Granma, 13 de julio de 2017. [Consultado el: 15 de octubre de 2017]. Disponible en: [http://www.granma.cu/cuba/2017-07-13/informatizacion-de-la-sociedad-principios-y-resultados-de-una-politica-13-07-2017-14-07-49].

[10] PÉREZ, Yoandy. Estrategia para la migración a aplicaciones de código abierto. Tesis para optar por el título de Máster en Informática Aplicada, Universidad de las Ciencias Informáticas, La Habana, 2015.

[11] PÉREZ, Yoandy; GARCÍA, Abel; GOÑI, Angel. Buenas Prácticas para la Migración a Código Abierto. La Habana, Ediciones Fututo, 2015. 106 p.

[12] NAM, Van. Comparative Performance Evaluation of Web Servers. VNU Journal of Science: Computer Science and Communication Engineering, 2017, 31 (3): p. 28–34.

[13] CHARTE, Francisco. Programación GNU/Linux. España, Anaya Multimedia, 2003. 720 p.

[14] HELMKE, Matthew. Ubuntu Unleashed 2015 Edition. Londres, Pearson Education, 2015. 912 p.

[15] NEDELCU, Clément. Nginx HTTP Server. Third Edition. Birmingham, Packt Publishing, 2015. 318 p.

[16] KABIR, Mohammed. La Biblia del Servidor Apache 2. España, Anaya Multimedia, 2003. 845 p.

[17] APACHE SOFTWARE FOUNDATION. Multi-Processing Modules (MPMs). [En línea]. 2017. [Consultado el: 20 de octubre de 2017]. Disponible en: [http://httpd.apache.org/docs/2.4/mpm.html].

[18] CAMPOVERDE, Ariel M.; HERNÁNDEZ, Dixys L.; MAZÓN, Bertha E. Cloud computing con herramientas open-source para Internet de las cosas. Maskana, 2015, 6 (Número Especial): p. 173-182.

[19] MATOTEK, Dennis; TURNBULL, James; LIEVERDINK, Peter. Pro Linux System Administration: Learn to Build Systems for Your Business Using Free and Open Source Software. Second Edition. New York, Apress, 2017. 1008 p.

[20] KHOLODKOV, Valery. Nginx Essentials. Birmingham, Packt Publishing Ltd, 2015. 151 p.

[21] SONI, Rahul. Nginx: From Beginner to Pro. New York, Apress, 2016. 255 p.

[22] OFICINA NACIONAL DE NORMALIZACIÓN. NORMA CUBANA NC ISO/IEC 25023:2017. INGENIERÍA DE SOFTWARE Y SISTEMAS – REQUISITOS DE LA CALIDAD Y EVALUACIÓN DE SOFTWARE Y SISTEMAS (SQuaRE) – MEDICIÓN DE LA CALIDAD DEL PRODUCTO DE SOFTWARE Y SISTEMA. 2017.

[23] MOLINA, Rachel. Módulo para administrar el servidor web Nginx desde la Herramienta para la Migración y Administración de Servicios Telemáticos. Tesis para optar por el título de Ingeniero en Ciencias Informáticas, Universidad de las Ciencias Informáticas, La Habana, 2017.

[24] ESCOBAR, Jazmine; CUERVO, Ángela. Validez de contenido y juicio de expertos: una aproximación a su utilización. Avances en medición, 2008, 6 (1): p. 27-36.

[25] GALANIS, P. The Delphi method. Archives of Hellenic Medicine, 2018, 35 (4): p. 564-570.

[26] GARCÍA, Mª Elena; LENA, Francisco Javier. Aplicación del método delphi en el diseño de una investigación cuantitativa sobre el fenómeno FABLAB. Empiria - Revista de metodología de ciencias sociales, 2018, 2 (40): p. 129-166.

[27] HUSSEIN, Ashatu. The use of triangulation in social sciences research: Can qualitative and quantitative methods be combined? Journal of comparative social work, 2009, 4 (1): p. 1-8.