UNIVERSIDAD TÉCNICA DE BABAHOYO.

FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA.

PROCESO DE TITULACIÓN

NOVIIEMBRE 2022 – ABRIL 2023

EXAMEN COMPLEXIVO DE GRADO O DE FIN DE CARRERA

PRUEBA PRÁCTICA

PREVIO A LA OBTENCIÓN DEL TÍTULO DE:

INGENIERIA EN SISTEMAS.

TEMA:

ANALISIS DE LAS HERRAMIENTAS OPEN SOURCE “CLIF Y GATLING”, PARA PRUEBAS DE FIABILIDAD Y RENDIMIENTO EN APLICACIONES EN ENTORNO WEB

ESTUDIANTE:

WILSON DARIO VILLALVA RODRIGUEZ.

TUTOR:

MONTECE MORENO OMAR RODRIGO

AÑO 2023

ANALISIS DE LAS HERRAMIENTAS OPEN SOURCE “CLIF Y GATLING”,

PARA PRUEBAS DE FIABILIDAD Y RENDIMIENTO EN APLICACIONES EN ENTORNO WEB

RESUMEN.

El desarrollo de este caso de estudio se inicia con la búsqueda en el internet de las herramientas Open Source Clif y Gatling las mismas que permiten la evaluación de métricas de fiabilidad y/o rendimiento, se seleccionó estas herramientas ya que son las que mas uso tienen, la documentación existente, caracteristicas y por su fácil manejo al momento de ejecutar en una aplicación web.

El principal objetivo de este caso de estdio es evaluar herramientas Open Source para realizar pruebas de fiabilidad y rendimiento de aplicaciones web. Este caso está enfocado en cierta medida para , pequeños y medianos desarrolladores, quienes no pueden acceder a herramientas de evaluación de calidad con licencia propietaria, por lo tanto, este caso de estudio tiene como finalidad pretender demostrar que existen herramientas Open Source que funcionan adecuadamente y que está al alcance de todos.

Para llegar al cumplimiento del objetivo de este caso de estudio se realiza un proceso que está conformado de varias fases: revisión literaria, estrategia, mención de métodos para genera datos y el análisis de los datos.

Los resultados obtenidos en una primera búsqueda son diez herramientas básicas, posteriormente se filtro bajo ciertos parámetros que reduce la lista de herramientas a dos. Finalmente, luego de realizar el análisis literarios de las herramientas a pruebas se puede comprobar un funcionamiento satisfactorio.

Las herramientas Open Source que se mencionan en este caso de estudio evaluan rendimiento de aplicaciones en entorno web, son una buena alternativa frente al software propietario, las mismas cumplen un adecuado funcionamiento. Sin embargo, cabe aclarar que a un nivel de exigencia mucho mas elevado no funcionaría bien.

clave: Open Source, calidad de software, rendimiento, fiabilidad, evaluación.

ABSTRACT

The development of this case study begins with a search on the internet for the Open Source tools Clif and Gatling, the same ones that allow the evaluation of reliability and/or performance metrics, these tools were selected since they are the ones that have the most use. , the existing documentation, characteristics and for its easy handling when running in a web application.

The main objective of this case study is to evaluate Open Source tools to carry out reliability and performance tests of web applications. This case is focused to a certain extent for small and medium developers, who cannot access quality evaluation tools with a proprietary license, therefore, this case study aims to demonstrate that there are Open Source tools that work properly and that is within everyone's reach.

In order to achieve the objective of this case study, a process is carried out that is made up of several phases: literary review, strategy, mention of methods to generate data and data analysis.

The results obtained in a first search are ten basic tools, later it is filtered under certain parameters that reduces the list of tools to two. Finally, after carrying out the literary analysis of the tools under test, satisfactory operation can be verified.

The Open Source tools that are mentioned in this case study evaluate the performance of applications in a web environment, they are a good alternative to proprietary software, they perform adequately. However, it should be clarified that at a much higher level of demand it would not work well.

Keywords: Open Source, software quality, performance, reliability, evaluation.

INTRODUCCIÓN

Los micro, pequeñas y medianas empresas buscan innovar a través de la automatización de sus procesos con el fin de optimizar recursos y hacer funcional sus negocios, los aspectos de seguridad en el manejo de la información, ahorrar dinero y ganar tiempo.

En la búsqueda de cumplir con esos objetivos se apoyan en la compra o desarrollo de productos de software. Según la revista Vanguardia los negocios que son gestionados con software pueden incrementar sus ganancias hasta un 70%.

La importancia del software en el desarrollo de las empresas en la actualidad es indiscutible, existe un alto mercado amplio y variado en cuanto a productos de software. La calidad de estos producto de software determinará el éxito empresarial en cuanto a la aceptación de los usuarios.

Las grandes empresas de desarrollo de software cuentan por lo general con áreas destinadas a realizar pruebas a sus productos de software, tanto en su etapa de desarrollo como en producción, para ello usan herramientas con licencia propietaria. Los pequeños desarrolladores por situación económica posiblemente no pueden acceder a estas herramientas de pago.

Este caso de estudio busca demostrar las alternativas de funcionamiento de algunas herramientas Open Source que permiten evaluar la calidad de las aplicaciones en entorno web, presentando como una alternativa a las herramientas con licencia propietaria que son usadas por las grandes empresas de desarrollo de software .

DESARROLLO

Calidad en el producto de Software

La palabra calidad está definida por la Real Academia de la Lengua como "*Propiedad o conjunto de propiedades inherentes a algo, que permiten juzgar su valor*" [1]. En el ámbito del desarrollo de software, el modelo de calidad ISO/ICE 25000 define la calidad del producto de software como "Capacidad del producto de software para satisfacer las necesidades declaradas e implícitas cuando se utiliza bajo condiciones especificadas".

Al mencionar la definición a la calidad de un producto de software se puede mencionar que la esencia de calidad sigue siendo la misma, pero con un complemento muy importante que es la satisfacción de las necesidades para las que fue hecho el producto de software.

La calidad de un producto de software es una de las principales características y realidad al momento del desarrollo de aplicaciones las mismas que se las puede percibir en las diferentes actividades computacionales diarias y puede representar, sin recargar, el éxito o el fracaso de un negocio.

Un producto de software con fallas puede provocar: pérdidas de tiempo, dinero e incluso la pérdida de vidas humanas, según el ámbito para el cual fue desarrollado. El nivel de gravedad es más alto si los afectados son sistemas críticos.

Por otro lado, un producto de software que se tomo muy en serio considerar altos niveles de calidad puede ser un factor clave para el éxito de la empresa desarrolladora como para los clientes que adquieren el software.

Fiabilidad del producto de Software

Sánchez Michel, G., González Garay, S., & Ramirez Reyes, M. (2021). Los autores indican que la fiabilidad es la probabilidad de que el software se ejecute durante un periodo de tiempo determinado sin que se produzca un fallo.

Maila Maila, E. F. (2018), el autor menciona que la fiabilidad es el grado en que el producto de software puede mantener un nivel de rendimiento especifico en cuanto se usa bajo condiciones especificas. Las características de desgaste o envejecimiento no ocurre los software. Las limitaciones de fiabilidad en un producto de software se deben principalmente a fallos en los requisitos, diseño e implementación.

Las fallas dependen de la forma en que se utilice el producto de software y de las opciones de programas seleccionadas. La característica fiabilidad tiene las siguientes subcacterísticas; disponibilidad, madurez, tolerancia a fallos, y recuperabilidad. A continuación, se detalla las subcaracteristicas y descripción de cada una de ellas :

- **Madurez**: es el grado en que el sistema satisface las necesidades de fiabilidad en condiciones normales de funcionamiento.

- **Disponibilidad**: es el grado en que un componente de software es operativo y disponible cuando se requiera para su uso.

- **Tolerancia a fallos**: es el grado en que el producto de software puede mantener un nivel específico de trabajo en casos de fallos del software.

- **Recuperabilidad**: es el grado en que el producto de software puede volver a instaurar un nivel específico de rendimiento y que pueda recuperar los datos directamente afectados en el caso de una falla.

Rendimiento en los productos de Software

Muñoz, C. C., Velthuis, M. G. P., & de la Rubia, M. Á. M. (2010), los autores hacen referencia a que el rendimiento es el grado en que el producto de software proporciona su funcionabilidad adecuada, en relación con la cantidad de recursos utilizados, que se encuentra bajo condiciones establecidas. Estos recursos pueden estar relacionados o se deben incluir a otros productos de software, a los componentes hardware del sistema y materiales adicionales como pueden ser dispositivos de almacenamiento externos, impresoras, etc.

La evaluación del rendimiento de un producto de software que es operado por un usuario se lo puede medir externamente por calidad de uso. El rendimiento tiene las siguientes subcaracterísticas:

- **Comportamiento en el tiempo**: es cuando se indica el grado en que el producto de software proporciona los tiempos de respuesta y de procesamiento, y las tasas de rendimiento al realizar su función, bajo las condiciones establecidas en sus procesos.

- **Utilización de recursos**: indica el grado en que el producto de software utiliza cantidades y tipos de recursos apropiados de hardware, cuando el software lleva a cabo sus funciones bajo condiciones establecidas en sus parámetros.

Métricas de fiabilidad y rendimiento según la Norma ISO/ICE 25023

Las métricas de calidad usadas son las que presentan la ISO/ICE 25023 [4], este estándar define métricas que permiten la medición cuantitativa de los productos software.

Las métricas de calidad externa se usan para medir la calidad del producto/sistema de software evaluando el comportamiento del sistema del cual forma parte, esta evaluación se la puede realizar durante las pruebas del ciclo de vida y/o en su etapa operativa. Se debe tener en cuenta que los requisitos especificados por las medidas de calidad deben utilizarse como criterios cuando se evalúa un producto.

La clasificación de las métricas es en base a las características y subcaracterísticas de la norma ISO/ICE 25010.

En este trabajo el uso de las métricas establecidas por la norma ISO/ICE 25023 es de cumplir dos roles importantes; primero permite la selección de las herramientas que cubren la mayor cantidad de métricas de fiabilidad y rendimiento, segundo la evaluación en este caso de estudio.

Métricas de rendimiento según la Norma ISO/ICE 25023

Las métricas de rendimiento deben tener la capacidad de medir el rendimiento en la relación con la cantidad de recursos utilizados en condiciones establecidas. Es importante aclarar que estos recursos pueden ser otros productos de software, el hardware del sistema y materiales externos como dispositivos de almacenamiento externos.

En la medición de rendimiento se debe tomar en consideración las condiciones establecidas, es decir se debe de tener en cuenta la configuración del sistema y de hardware. La medición de rendimiento externo debe permitir medir atributos tales como el consumo de tiempo y el comportamiento de utilización de recursos del sistema incluyendo el software durante las operaciones o las pruebas.

Es importante tener en cuenta el papel que desempeña los factores como:

CPU y memoria utilizada por otro software, tráfico de red y procesos programados.

Las métricas de rendimiento en la evaluación de calidad externa tienen un 13% de importancia. A continuación, se puede observar un detalle de las subcaracterísticas de rendimiento con sus respectivas métricas.

SUBCARACTERÍSTICAS DE RENDIMIENTO CON SUS RESPECTIVAS MÉTRICAS

COMPORTAMIENTO EN EL TIEMPO

Vivanco Villamar, A. A. (2011), el autor hace referencia a las métricas de comportamiento en el tiempo de espera deben permitir medir los atributos como el comportamiento del tiempo de respuesta del sistema informático, incluyendo software durante la prueba o las operaciones. Las métricas de la sub característica comportamiento en el tiempo son: tiempo de respuesta, tiempo de espera y rendimiento.

Tiempo de espera

El tiempo de espera es el tiempo transcurrido en una computadora entre recibir un mensaje y enviar el resultado. A veces solo significa el tiempo usado para una aplicación.

Tiempo de respuesta

El tiempo de respuesta es la duración en el cual se emite un comando para iniciar un lote de tareas hasta recibir la primera respuesta. El tiempo de respuesta incluye tiempo de procesamiento y tiempo de transmisión. En el caso del sistema de Internet u otro sistema de tiempo real, a veces el tiempo de transmisión es mucho más largo.

Rendimiento

Es el tiempo que refleja en cuántas tareas se pueden procesar por la unidad de tiempo.

UTILIZACION DE RECURSOS

Las métricas de utilización de recursos externos permiten medir tales atributos como el comportamiento de los recursos utilizados del sistema informático, incluyendo el software durante las pruebas o la operación. Las métricas de la sub característica utilización de recursos son, utilización de CPU, utilización de memoria y utilización de dispositivos de E/S.

Utilización de CPU

La utilización de CPU permite medir cuánto tiempo de CPU se utiliza para realizar una tarea determinada.

Utilización de memoria

La utilización de memoria permite medir cuánto espacio de memoria se utiliza para realizar una tarea determinada.

Utilización de dispositivos de E/S

La utilización de dispositivo de E/S permite medir cuánto tiempo consumen los dispositivos de E/S para realizar una tarea determinada

CAPACIDAD

Las métricas de capacidad permiten medir el grado en que los límites máximos de un producto o parámetro del sistema cumplen los requisitos.

Numero de peticiones en línea

Esta métrica permite medir cuántas solicitudes en línea pueden procesarse por unidad de tiempo.

Numero de accesos simultáneos

Esta métrica permite medir cuántos usuarios pueden acceder al sistema simultáneamente en un momento determinado.

Sistemas de transmisión de ancho de banda

Esta métrica permite medir cuánto es el valor límite absoluto de transmisión requerido para cumplir con las funciones.

SUB CARACTERÍSTICAS DE FIABILIDAD CON SUS RESPECTIVAS MÉTRICAS MADUREZ

Las métricas de madurez externa permiten medir atributos tales como la libertad de software de los fallos causados por fallas existentes. Las métricas de esta sub característica son: eliminación de errores, cobertura de pruebas y el tiempo medio entre fallos.

Eliminación de fallos

La métrica de eliminación de fallos permite determinar qué proporción de fallos detectados se han corregido.

Cobertura de pruebas

La métrica cobertura de pruebas permite determinar cuánto de los casos de prueba requeridos se han ejecutado durante las pruebas.

Tiempo medio entre fallos

La métrica de tiempo de tiempo entre fallos (Mean Time Between Failures) permite medir con qué frecuencia falla el funcionamiento del sistema o software.

DISPONIBILIDAD

La disponibilidad en la evaluación de calidad externa puede ser evaluada por la proporción del tiempo total durante el cual una aplicación está en estado activo. Por lo tanto, la disponibilidad es una combinación de madurez, tolerancia a fallos y capacidad de recuperación la cual se encarga de regular la duración del tiempo de inactividad después de cada falla. Las métricas de esta sub característica son el tiempo de servicio y el tiempo de inactividad.

Tiempo de servicio

La métrica de tiempo de servicio permite medir la proporción del tiempo de servicio del sistema se proporciona realmente.

Tiempo de inactividad

La métrica de tiempo de inactividad permite medir cual es el tiempo promedio en que el sistema permanece indisponible cuando se produce un fallo.

TOLERANCIA A FALLOS

La sub característica tolerancia a fallos externa debe estar relacionada con la capacidad de la aplicación de mantener un nivel de rendimiento especificado en casos de fallos de operación o infracción de su interfaz especificada. Las métricas de esta sub característica son: prevención de fallas y redundancia.

Prevención a fallas

Esta métrica permite medir cuántos patrones de fallas fueron puestos bajo control para evitar fallas críticas y graves.

Redundancia

Esta métrica permite medir cuántos tipos de componentes del sistema se instalan de forma redundante para evitar fallos del sistema.

CAPACIDAD DE RECUPERACION

La sub característica capacidad de recuperación es una medida que permite medir atributos tales como el software con el sistema capaz de restablecer su nivel adecuado de rendimiento y recuperar los datos directamente afectados en el caso de un fallo. Esta sub característica tiene la métrica tiempo medio de recuperación.

Tiempo de recuperación

Esta métrica permite medir el tiempo promedio que tarda el sistema en completar la recuperación del fallo.

Procesos de pruebas de software (ISTQB)

La ISTQB (*International Software Testing Qualifications Board*) es la organización internacional que tiene como misión "definir y mantener un Cuerpo de Conocimientos que permita certificar a los probadores basándose en las mejores prácticas, conectando la comunidad internacional de pruebas de software y fomentando la investigación"

Un proceso de pruebas permite la medición de la calidad de un producto de software en términos de los defectos encontrados. SI el producto de software pasa una prueba de calidad diseñada correctamente reduce el nivel general del riesgo de un sistema.

Un proceso de pruebas puede tener los siguientes objetivos:

- Identificar fallas en el producto de software.
- Aumentar la confianza en el nivel de calidad.
- Facilitar información para la toma de decisiones.
- Evitar la aparición de fallas.

La ISTQB recomienda que el proceso de pruebas debería estar implicado desde el inicio del ciclo de vida del software para evitar la introducción de defectos en el código, para este caso de estudio se parte ya con productos Open Source terminados.

Los siete principios de las pruebas de software

ISTQB ha propuesto una serie de principios que establecen ciertas pautas generales para las pruebas de software.

1. Las pruebas de productos de software pueden demostrar la existencia de defectos, pero no al contrario, es decir no pueden demostrar que el software no tenga defectos. Al realizar las pruebas de software se reduce la probabilidad de que existas defectos ocultos.
2. Realizar las pruebas de software con todas las combinaciones de entrada y salida es imposible, por lo que es más importante realizar un análisis de pruebas y priorizarlas.
3. Las pruebas tempranas se usan para la identificación de defectos en una etapa anticipada y se la debe realizar lo más pronto posible en ciclo de vida del software.
4. La agrupación de defectos por lo general se concentra en un número reducido de módulos, estos defectos deben ser detectados en una fase de pruebas previa al lanzamiento del software.
5. La paradoja de pesticida hace referencia a que, si se realizan repetitivamente las mismas pruebas una y otra vez, en consecuencia, se dejaran de encontrar defectos nuevos.
6. Las pruebas dependen del contexto en que se las realice, es decir, no es lo mismo probar un sistema financiero a un sistema de publicidad.
7. A pesar que se realicen todas las pruebas y se hagan todas las correcciones de los defectos detectados, no servirá de nada si no cumple las expectativas y necesidades de los usuarios.
8.

A pesar que en muchos de los puntos se referencia al ciclo de vida de un producto de software, cabe recalcar que en este proyecto las pruebas de calidad se las realiza a aplicaciones web finalizadas.

Procesos básicos de pruebas de software

Florián, B. E., Solarte, O., & Reyes, J. M. (2010). Los autores indican que proceso de pruebas de software se basan en un grupo de actividades que no solo se enfoca en la ejecución de las pruebas, por lo tanto, hay que tomar en cuenta muchas actividades.

En el proceso de pruebas debe tener los tiempos necesarios que permitan planificar las pruebas, diseñar los casos de prueba, preparar la ejecución de las pruebas y finalmente evaluar los resultados.

PROCESOS BASICOS DE PRUEBAS

En la siguiente lista se detalla las actividades ordenadas secuencialmente, esto no significa que es obligatorio esperar a terminar una actividad para iniciar la siguiente. Todo depende de las necesidades se las puede solapar o realizar 2 o más a la vez, adaptando las actividades a contexto del proyecto.

a) Planificación y control

En la etapa de planificación y control se definen los objetivos de las pruebas de software y la especificaciones de las actividades de pruebas con enfoque a cumplir los objetivos establecidos.

En el control se realiza un seguimiento, se compara el progreso real con el plan establecido, además de informar el estado en el que se encuentran las pruebas. En esta actividad se deben incluir, en caso de haber existido, desviaciones con respecto a lo planificado.

b) Análisis y diseño de pruebas

En esta etapa los objetivos de las pruebas de software se transforman en condiciones de prueba y casos de prueba.

Esta actividad cuenta con una serie de tareas consideradas como principales, las cuales se señalan a continuación:

1. Revisar la base para las pruebas, esto se refiere a los requisitos, informes de análisis de riesgos, arquitectura, diseño y especificaciones de interfaz.
2. Evaluar la capacidad para ser probados los productos de software.
3. Identificar las condiciones de prueba y dar prioridades en base a un análisis.
4. Diseñar y priorizar los casos de prueba.
5. Identificar los datos de prueba necesarios que soporten las condicione de pruebas y los casos de pruebas.
6. Diseñar el entorno de pruebas e identificar herramientas e infraestructura en caso de ser necesario.

c) Implementación y ejecución de pruebas

En esta etapa de implementación y ejecución de pruebas de software se especifican los procedimientos para las pruebas. Se inician con los casos de prueba en un orden determinado y se puede incluir información adicional de ser necesaria para la ejecución de las pruebas. Además, se configura el ambiente en el cual se ejecutan las pruebas.

La actividad de implementación y ejecución de pruebas tiene las siguientes tareas principales.

- Finalizar, implementar y dar prioridades a los casos de prueba.
- Desarrollar y dar prioridades a los procedimientos de pruebas.
- Revisar y verificar que el entorno de pruebas ha sido correctamente configurado.
- Ejecutar los procedimientos de pruebas de manera manual o mediante el uso de herramientas de ejecución de pruebas.
- Registrar los resultados obtenidos en la ejecución de las pruebas acompañado del detalle de la versión de software y las herramientas usadas en las pruebas.
- Realizar un reporte con las discrepancias encontradas en forma de incidencias y analizar en vistas de establecer que las causaron.
-

d) Evaluación de los criterios de salida e informe

La actividad previa a la finalización de todo el proceso es la evaluación de criterios de salida e informes. En esta actividad se evalúa la ejecución de las pruebas frente a los objetivos establecidos inicialmente.

Esta actividad consta con las siguientes tareas principales:

- Comprobar los registros de pruebas con los criterios de salida previstos en la planificación de las pruebas.
- Evaluar si se requiere que se realicen más pruebas o si se deberían modificar los criterios de salida previamente especificados.
- Realizar un resumen de las pruebas para las partes interesadas.

e) Actividades de cierre de pruebas

Esta actividad es el cierre del proceso de pruebas, en la cual se recopilan los datos obtenidos con el objetivo de consolidar la experiencia. Esta actividad es importante en el desarrollo de un proyecto de software, puede ser la actividad previa a un lanzamiento de un producto de software, la finalización de un proyecto de pruebas o la finalización de una versión de mantenimiento.

En la actividad de cierre de pruebas se incluyen las siguientes tareas principales:

- Comprobar cuales son los productos de software entregables previstos y cuales han sido definitivamente entregables.
- Finalizar los informes de incidencias o aportar modificaciones a aquellos que siguen abiertos.
- Documentar la aceptación del producto de software.
- Archivar los productos de soporte de prueba, el entorno de pruebas y la infraestructura de pruebas.
- Analizar las lecciones aprendidas en el proceso de pruebas para determinar futuras versiones y proyectos.
- Utilizar la información recopilada para mejorar la madurez de las pruebas.

Tipo de pruebas.

En busca del cumplimiento de los objetivos es importante definir los tipos de pruebas que permiten medir métricas de rendimiento y fiabilidad, se ha considero las pruebas de carga y estrés.

Pruebas de Carga

Este tipo de pruebas permiten determinar y validar la respuesta de una aplicación cuando es sometida a una carga de usuarios, transacciones o procesos simultáneamente, tratando de emular un ambiente de producción real.

Pruebas de Estrés

En este tipo de pruebas se trata de cargar el sistema o los componentes del sistema hasta que llegan a los límites de funcionamiento. Esto permite encontrar el volumen de datos (o el tiempo) en que la aplicación deja de ser capaz de responder a las peticiones como se espera.

Dependiendo de las funcionalidades disponibles de cada herramienta se aplicará el tipo de prueba correspondiente. Cabe aclarar que las pruebas son de caja negra, las herramientas que realicen pruebas con el código de la aplicación serán descartadas.

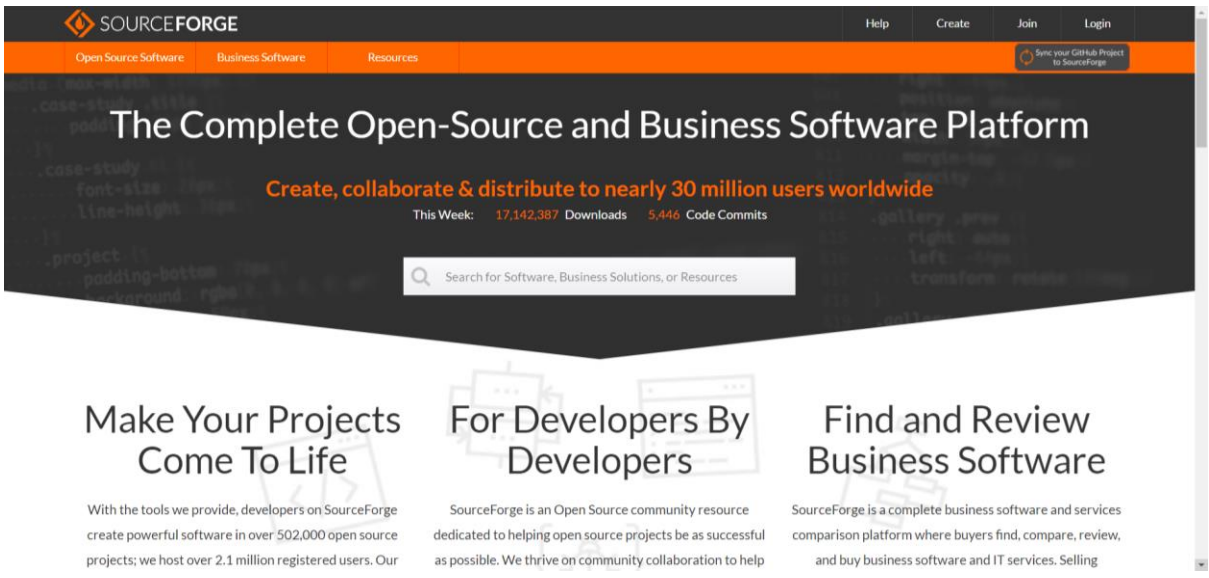
PROCESO DE SELECCIÓN DE HERRAMIENTAS

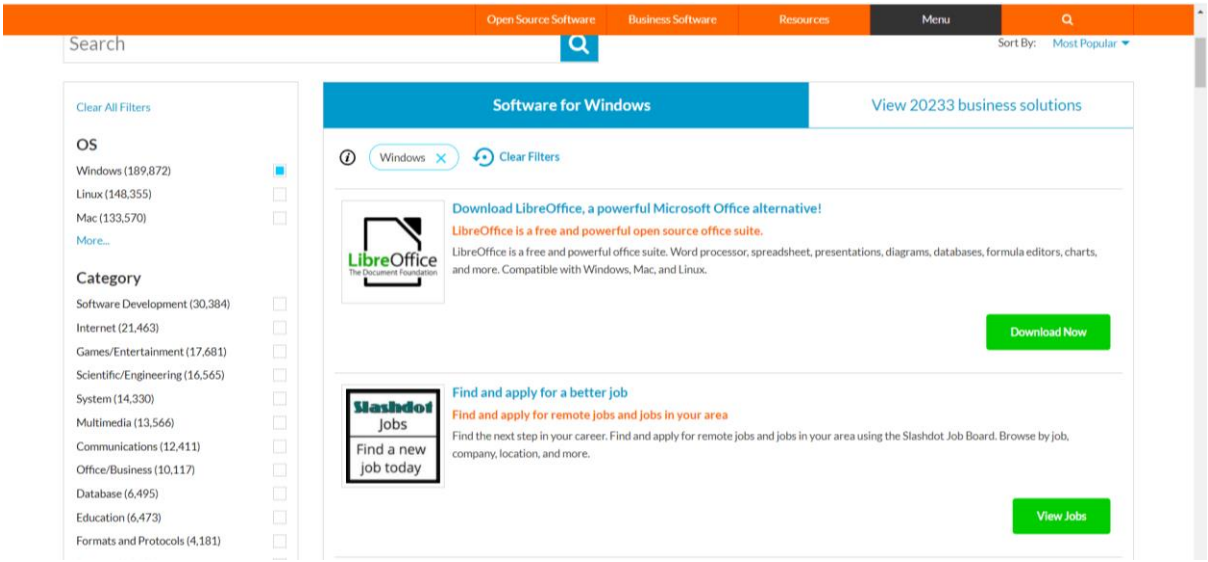
Búsqueda de herramientas Open Source para pruebas de fiabilidad y rendimiento.

Chiesa, F. (2004). El autor indica que la búsqueda de las herramientas en esta actividad se inicia con el proceso de selección de las herramientas, teniendo en cuenta que en esta actividad se agrupan todas las posibles herramientas a ser evaluadas. Al tener una lista de *n* herramientas se aplicarán ciertos criterios de selección con lo que se reducirá el universo, posteriormente se probarán las herramientas y finalmente se escogerá una que se aplicará a este caso de estudio.

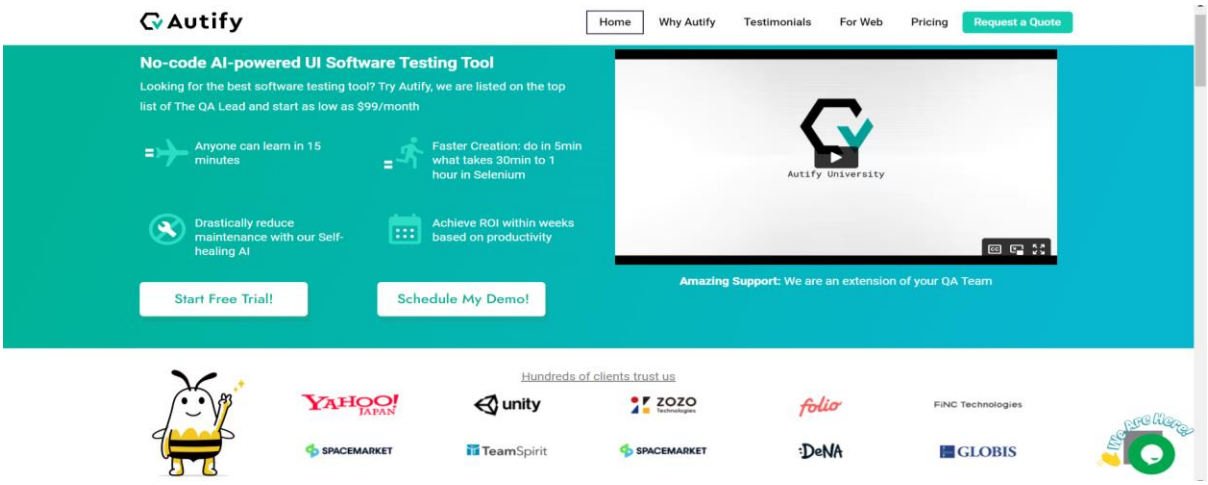
La búsqueda previa de las herramientas Open Source se la realiza en el buscador de Internet Google, posteriormente la búsqueda se centraliza en dos repositorios de productos de software Open Source que son *SourceForge* y *QA Testing Tools*. En el primer repositorio mencionado se encuentran aplicaciones para todas las necesidades de los usuarios, es decir, no es un repositorio específicamente de herramientas enfocadas a pruebas de calidad. Sin embargo, el segundo repositorio es una comunidad dedicado solo en productos para pruebas de software. Cabe recalcar que estos repositorios abarcan una gran cantidad de productos de software.

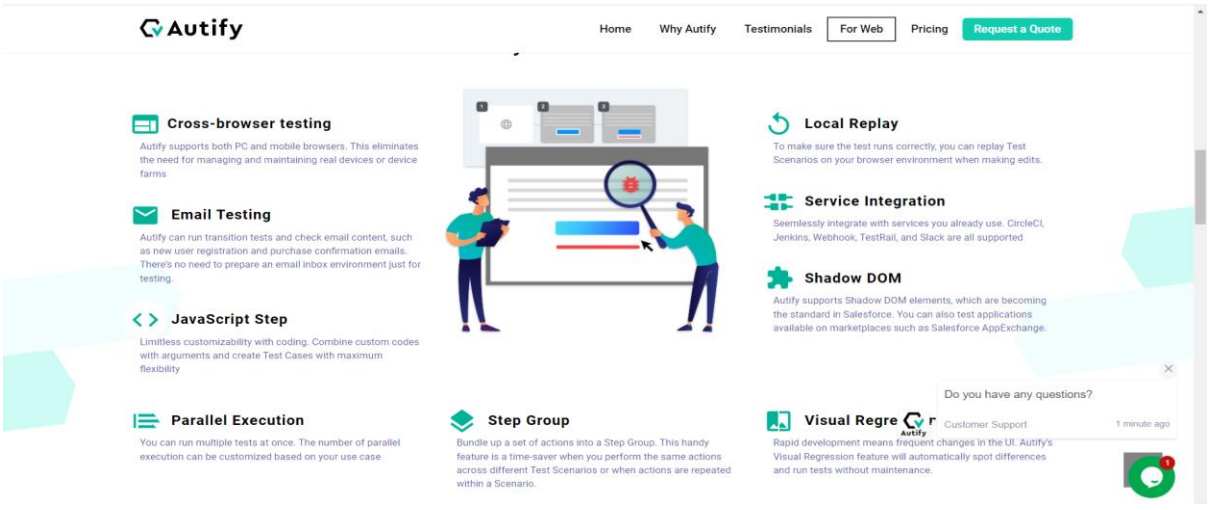
SourceForge





QA Testing Tools





LISTA PRELIMINAR DE HERRAMIENTAS OPEN SOURCE PARA PRUEBAS DE FIABILIDAD Y RENDIMIENTO DE SOFTWARE.

NO	HERRAMIENTAS
1	Canoo WebTest
2	CLIF
3	D-ITG
4	Fast Web Performance Test Tool
5	Funkload
6	Gatling
7	Grinder – Java Load Testing Framework
8	JMeter – Load and Performance tester
9	MStone
10	Multi-Mechanize – web performance and load

Fuente: Internet

Autor: Villalva Rodriguez Wilson Dario.

Para obtener esta lista de herramientas la búsqueda se la hizo en Internet en los buscadores como en los sitios web de los repositorios. Con este número de herramientas se procede a buscar información más detallada de cada una de ella, que permitirá reducir el número de herramientas bajo ciertos criterios que se detallan a continuación

Criterios de preselección de herramientas.

Con la finalidad de evaluar la herramienta Open Source con el mejor perfil, se plantean ciertos criterios de selección. Si uno de estos criterios no es cumplido por la herramienta será descartada.

Para la evaluación de cada criterio se revisa la página oficial o el repositorio de la herramienta donde se obtendrá la información. Esta información se verá reflejada en tabla que se detalla a continuación, en esta se marcará con un visto (□) si cumple con el criterio y con una (**x**) si no lo cumple.

Los criterios que se considerarán para la preselección de herramientas son los siguientes:

- Ser Open Source.
- Fecha de la versión de la herramienta.
- La documentación.

- Un instalador disponible.
- Evalúe al menos una de las métricas de las características de fiabilidad y/o rendimiento con orientación a las aplicaciones web.

Open Source

El propósito de este caso de estudio es evaluar herramientas Open Source, aunque resulte evidente que las herramientas que serán evaluadas tienen que ser Open Source no está por demás aclararlo.

Fecha de la última versión

La separación de aplicaciones por la fecha de última versión es debido a que las aplicaciones Open Source no garantizan su correcto funcionamiento, por lo que están en constante actualización de su software; Una herramienta que no tiene una versión reciente es porque ha sido abandonada por sus desarrolladores o la comunidad y en caso de presentarse algún inconveniente con la herramienta es posible que no se tenga una respuesta. Por lo antes mencionado se van a tomar en consideración herramientas que tengan como mínimo una última actualización del 1 de enero del año 2017 en adelante.

Documentación

La documentación es muy importante no solo en los productos de software Open Source, sino en todos los productos de software en general. Con el trascurrir de los años la práctica de documentación ha mejorado, esto ayuda a que los productos de software mejoren en sus fases de desarrollo y de mantenimiento.

En este caso de estudio es primordial la existencia de la documentación de cada una de las herramientas, tanto para tener en claro el funcionamiento de las mismas como sus limitaciones. La herramienta que no tenga documentación se la descartara.

Instalador disponible

Es necesario un instalador para probar la herramienta, pero existen casos en los repositorios en los que hay la descripción de las herramientas mas no existe el archivo de descarga, lo que la convierte en una herramienta inexistente para los usuarios. Por lo tanto, es importante verificar que están disponibles los instaladores de todas las aplicaciones de la lista preliminar.

Evalúa una métrica de fiabilidad y/o rendimiento

Las herramientas obtenidas en la búsqueda preliminar deben cumplir con el requisito de evaluar al menos una métrica ya sea de fiabilidad y/o rendimiento orientado a aplicaciones

web. En caso de no cubrir al menos una métrica de las antes mencionadas, la herramienta será descarta.

Herramientas y verificación de los criterios propuestos

No.	Herramienta	Open Source	Ultima Versión	Documentación	Instalador	Tipo de pruebas		Página oficial de la aplicación
						Fiabilidad	Rendimiento	
1	Canoo WebTest	□	19/07/2016	□	□	x	x	http://webtest.canoo.com/webtest/manual/WebTestHome.html
2	CLIF	□	6/4/2020	□	□	□	□	http://clif.ow2.org/index.html
3	D-ITG	□	03/06/2007	□	□	x	x	http://www.grid.unina.it/software/ITG/
4	Fast Web Performance Test Tool	□	22/11/2016	□	□	x	□	http://fwptt.sourceforge.net/
5	funkload	□	10/3/20011	□	□	□	x	https://funkload.nuxeo.org/intro.html
6	Gatling	□	20/04/2021	□	□	□	□	http://gatling.io
7	Grinder – Java Load Testing Framework	□	21/04/2015	□	□	x	□	http://grinder.sourceforge.net/
8	JMeter – Load and Performance tester	□	19/11/2016	□	□	x	□	http://jmeter.apache.org/
9	MStone	□	25/04/2013	x	□	□	x	https://www.qatestingtools.com/testing-tool/mstone
10	Multi-Mechanize – web performance and load testing framework	□	01/01/2013	□	□	x	□	https://multi-mechanize.readthedocs.io/en/latest/

Fuente: Sitio web de cada aplicación

Autor: Villalva Rodríguez Wilson Dario.

En la Tabla 1.1 se muestra una lista preliminar de herramientas Open Source para evaluar métricas de fiabilidad y rendimiento en los cuales se encuentran los criterios de preselección que fueron planteados anteriormente. Se obtiene como resultado dos herramientas que cumplen con los criterios de preselección, estas herramientas están resaltadas con fondo amarillo, las herramientas que no cumplen con los criterios de selección están marcadas de color rojo.

Selección de herramientas

La selección de herramientas que permitan evaluar las métricas de fiabilidad y rendimiento se realiza valorando con diferente ponderación cada métrica. Tal como se muestra en la figura 1.1 Tomando en cuenta que, hay herramientas que evalúan métricas de las dos características mencionadas, se plantea seleccionar 2 de ellas que son las que cumplen con los parámetros requeridos, obteniendo como seleccionadas las herramientas Clif y Gatling.

HERRAMIENTA OPEN SOURCE CLIF

CLIF es una herramienta para pruebas de carga, esto incluye inyecciones de carga y es compatible con varios protocolos como son: HTTP, FTP, SIP. Además, permite medir el uso de recursos por parte de la aplicación como es: procesador, memoria, y red. Las características que más resalta de CLIF son las siguientes.

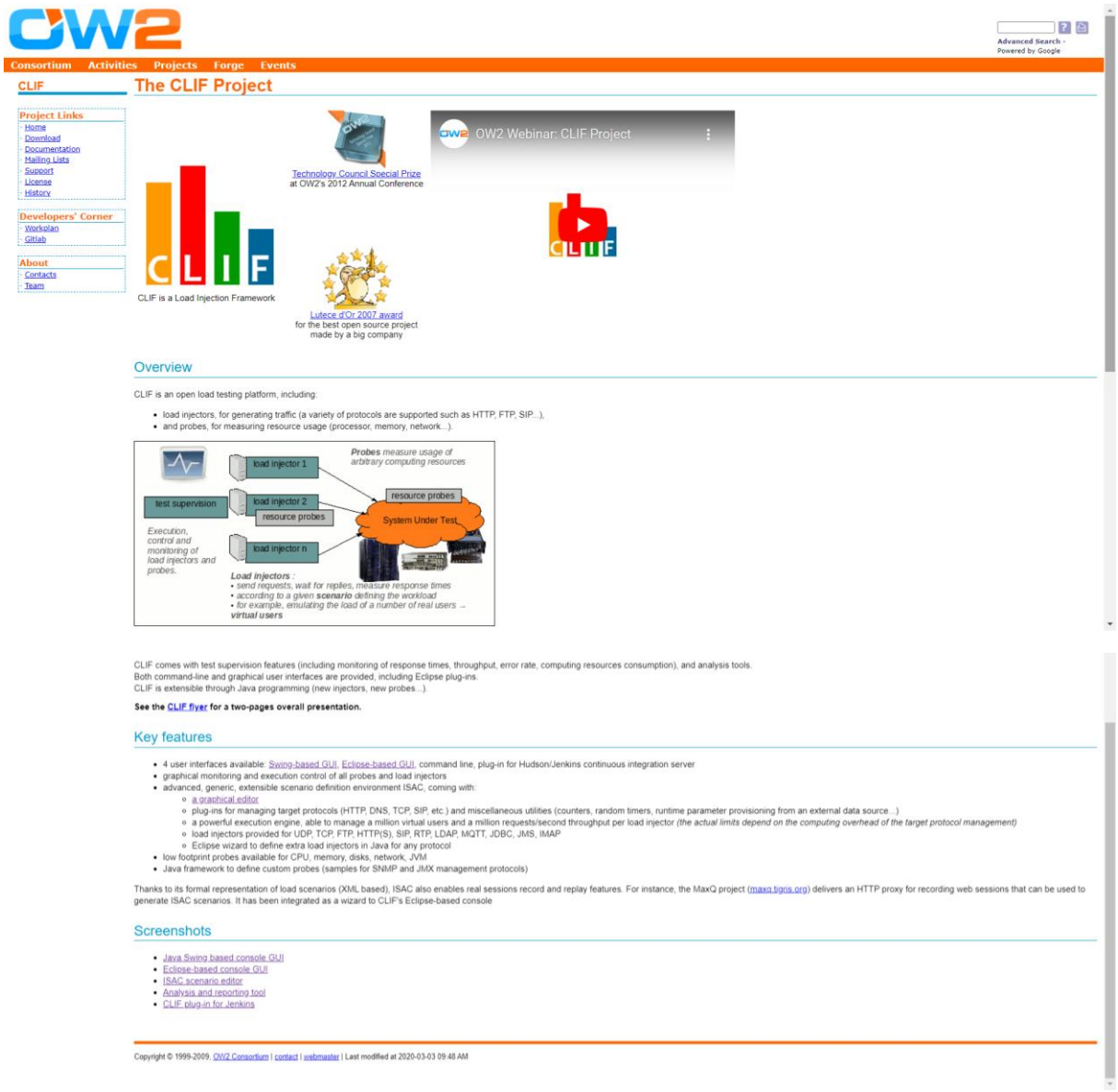
- El usuario cuenta con 4 posibilidades de interfaces.
- Permite una monitorización gráfica y un control de los inyectores de carga.
- Plug-ins para a gestión de protocolos objetivos como son: HTTP, DNS, TCP, etc.
- Un motor de ejecución que permite gestionar millones de usuarios virtuales y un millón de peticiones por segundo.
- Inyectores de carga previstas UDP, TCP, FTP, HTTP (S), SIP, RTP, LDAP, JDBC, JMS, IMAP

En el sitio oficial de la herramienta se puede encontrar toda la documentación necesaria.

Se puede encontrar ejemplos prácticos y manuales como son:

- Guía de instalación.

- Manual de inicio rápido.
- Manual de usuario.
- CLIF consola de Eclipse ayuda en línea.
- ISAC editor de escenarios ayuda en línea.
- ISAC manual de referencia plug-ins.
- Tutorial para CLIF y utilización ISAC.
- Usando CLIF desde servidor de automatización Jenkins.



HERRAMIENTA OPEN SOURCE GATLING

La herramienta Gatling se enfoca en las pruebas de rendimiento de aplicaciones web, estas pruebas consisten en cargar a la aplicación con muchos usuarios con comportamientos complejos.

Otra característica de la herramienta es que permite la revisión de los tiempos de respuesta a los requerimientos que se le hace a la aplicación. Los resultados obtenidos son presentados en informes donde se puede analizar los datos obtenidos con la ayuda de gráficos que permiten una mejor interpretación

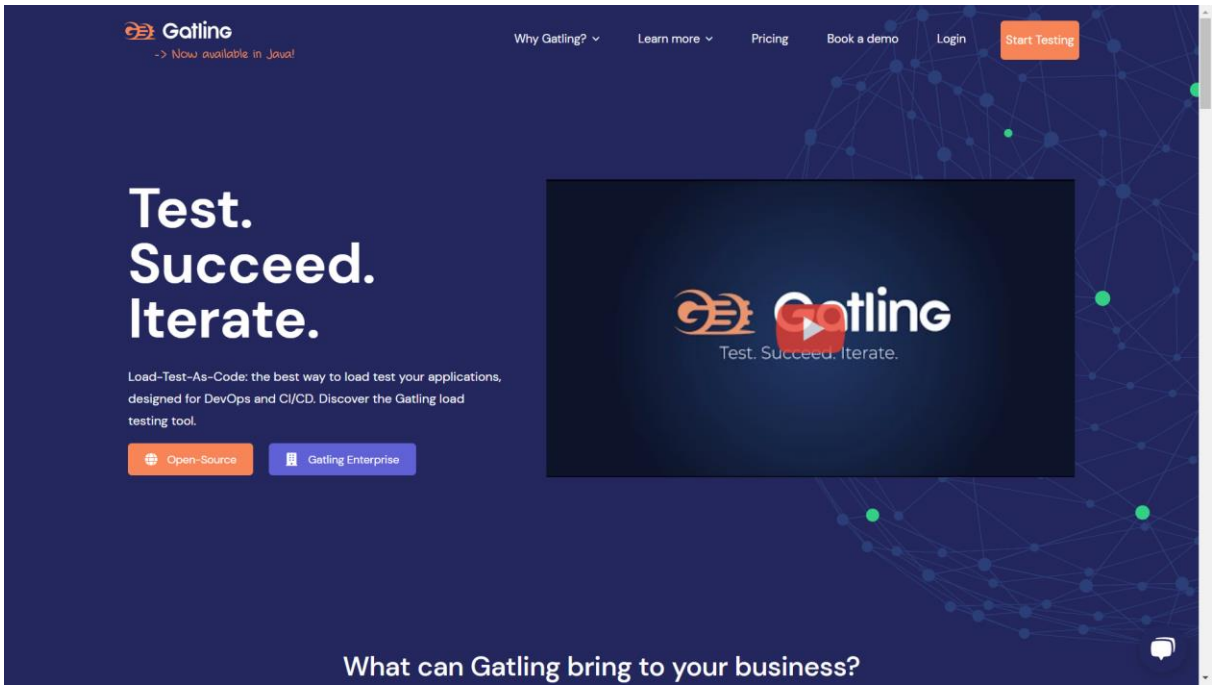
La automatización de pruebas de Gatling es mediante scripts que permiten mantener escenarios y de prueba y a la vez facilita la automatización de las mismas. Gatling permite simular grandes cargas por segundo sobre la aplicación y obtener métricas con un alto margen de precisión.

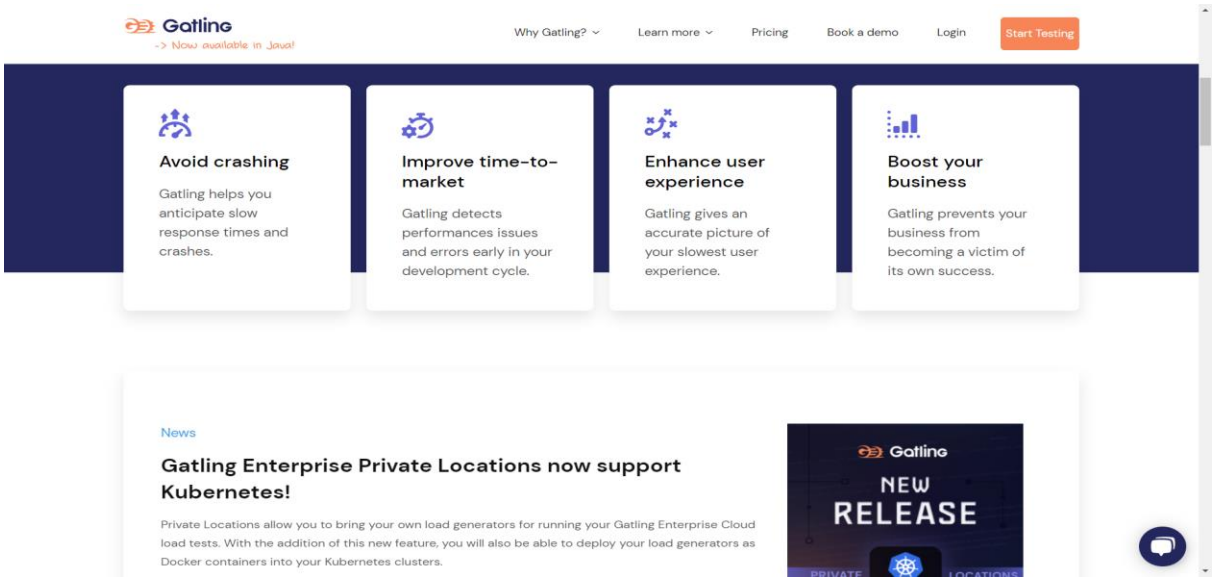
Los desarrolladores de este proyecto, Gatling Corp, han acompañado a la herramienta con buena documentación la cual puede ser revisada en su página oficial (http://gatling.io), ahí se encuentra:

Guía de Usuario

Una guía de inicio rápido con la herramienta Gatling

Un tutorial avanzado para el uso de la herramienta





HERRAMIENTAS SELECCIONADAS CON LAS RESPECTIVAS METRICAS

Los datos de las métricas fueron obtenidos desde los sitios oficiales de cada herramienta

Características	Subcaracteristicas	Herramientas	Clif	Gatling
Rendimiento	Comportamiento en el tiempo	Tiempo de respuesta	0.62	0.62
		Tiempo de espera	0.62	0.62
		rendimiento		
	Utilización de recursos	Utilización de CPU	0.16	0.15
		Utilización de memoria	0.16	0.15
		Utilización de dispositivos E/S	0.16	0.16
	Capacidad	Numero de peticiones online máximo	0.78	0.77
		Numero de accesos simultáneos máximo	0.78	0.78
		Sistema de transmisión de ancho de bandas	0.70	0.70
Fiabilidad	Madurez	Eliminación de fallos	0.50	0.50
		Cobertura de pruebas	0.65	0.64
		Tiempo medio entre fallos	0.60	0.59
	Disponibilidad	Tiempo de servicios	0.50	0.50
		Tiempo medio de inactividad	0.50	0.50
	Tolerancia a fallos	Prevención a fallas	0.70	0.70
		Redundancia	0.70	0.69
	Capacidad de recuperación	Tiempo medio de recuperación	0.50	0.50

Fuente: pagina oficial de cada herramienta

AUTOR: Villalva Rodriguez Wilson Dario.

CONCLUSIONES

Las herramientas Open Source para evaluar rendimiento de aplicaciones web son una buena alternativa frente al software propietario, en este trabajo se pudo obtener una lista de herramientas para realizar pruebas de rendimiento.

No es recomendable usar herramientas Open Source para evaluar fiabilidad, si bien existen las herramientas Open Source para pruebas de fiabilidad, de la lista obtenida las herramientas no cumplen con una documentación adecuada, un instalador o son antiguas lo cual no garantizaría tener un adecuado soporte en caso de presentar problemas en su ejecución.

Existen muchas herramientas Open Source pero si es importante discernir que herramienta cumple ciertos parámetros para poder decir "esta es una herramienta recomendada", en este trabajo se inició con una lista preliminar de diez herramientas pero luego de someterse a los filtros solo quedaron dos.

Al comparar una herramienta con otra se puede verificar la consistencia de datos, si bien las herramientas permiten calcular ciertas métricas de las características de fiabilidad y rendimiento, se pudo verificar que las herramientas pueden hacer esos cálculos de manera distinta como es el caso de Clif y Gatling en la métrica de tiempo de respuesta.

La herramienta Clif es muy versátil, cuenta con una interfaz muy fácil de usar, cuenta con muchas alternativas para realizar pruebas de carga y permite apreciar los resultados en diferentes presentaciones, arboles de resultados, resultados en árbol, de manera individual por cada thread, en informe resumido, en gráficos, etc. Una herramienta recomendada para quienes necesiten realizar pruebas de carga.

Gatling permite medir las métricas; tiempo de respuesta, tiempo de espera, y rendimiento de la subcaracterística comportamiento en el tiempo; número máximo de peticiones, número de accesos simultáneos y el sistema de transmisión de ancho de banda de la subcaracterística capacidad.

BIBLIOGRAFIA

Sánchez Michel, G., González Garay, S., & Ramirez Reyes, M. (2021). Estimación de la fiabilidad de software: Modelo Littlewood-Verall. Revista Cubana de Ciencias Informáticas, 15(4), 61-75.

Maila Maila, E. F. (2018). Evaluación de herramientas Open Source para pruebas de fiabilidad y rendimiento de aplicaciones web (Bachelor's thesis, Quito, 2019.).

Muñoz, C. C., Velthuis, M. G. P., & de la Rubia, M. Á. M. (2010). Calidad del producto y proceso software. Editorial Ra-Ma.

Vivanco Villamar, A. A. (2011). Evaluación de calidad del sistema Integrado para casas de valores SICAV de la bolsa de valores de Quito utilizando la norma ISO/IEC 14598 (Bachelor's thesis, Quito, 2011.).

Florián, B. E., Solarte, O., & Reyes, J. M. (2010). Propuesta para incorporar evaluación y pruebas de usabilidad dentro de un proceso de desarrollo de software. Revista eia, (13), 123-141.

Chiesa, F. (2004). Metodología para selección de sistemas ERP. Reportes técnicos en ingeniería del software, 6(1), 17-37.

RAE, «Real Academia de la Lengua,» [En línea]. Available: http://dle.rae.es/?id=6nVpk8P|6nXVL1Z. [Último acceso: 23 marzo 2017].

ISO/IEC FCD 25000:2004(E), «Software engineering - Software product Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE,» nº 43, 30 noviembre 2014.

ISO/IEC/IEEE 24765:2010(E), «Systems and software engineering - Vocabuary,» 2010.

ISO/IEC WD 25023, «Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Measurement of system and software product quality,» nº 54, 2011.

International Software Testing Qualifications Board, «ISTQB,» [En línea]. Available: http://www.istqb.org/about-as/vision-mission.html. [Último acceso: julio 2017].

International Software Testing Qualifications Board, «Foundation level Syllabus,» 2011. [En línea]. Available: http://www.istqb.org/downloads/send/2-foundation-level-documents/3-foundation-level-syllabus-2011.html4. [Último acceso: mayo 2017].

S. C. B. d. Souza, N. Anquetil y K. M. d. Oliveira, «Universidade Católica de Brasília, Brasilia, Brazil,» noviembre 2005. [En línea]. Available: http://dl.acm.org/citation.cfm?id=1085331. [Último acceso: mayo 2017].

N. Jemutai Kipyegen y W. Korir, «International Journal of Computer Science Issues,» septiembre 2013. [En línea]. Available: https://ijcsi.org/papers/IJCSI-10-5-1-223-228.pdf. [Último acceso: 12 mayo 2017]

ANEXOS.

I. Análisis antiplagio.

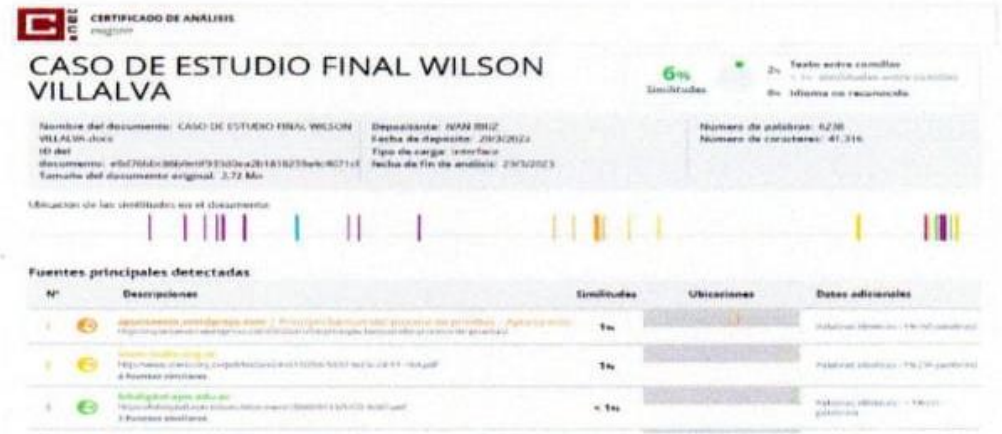
UNIVERSIDAD TÉCNICA DE BABAHOYO
FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA
CARRERA DE INGENIERIA EN SISTEMAS

Babahoyo 29 de Marzo del 2023

CERTIFICACIÓN DE PORCENTAJE DE SIMILITUD CON OTRAS FUENTES EN EL SISTEMA DE ANTIPLAGIO

En mi calidad de Tutor del Trabajo de la Investigación de: el/la, Sr./Sra./ Srta.: **VILLALVA RODRIGUEZ WILSON DARIO**, cuyo tema es: **ANALISIS DE LAS HERRAMIENTAS OPEN SOURCE "CLIF Y GATLING", PARA PRUEBAS DE FIABILIDAD Y RENDIMIENTO EN APLICACIONES EN ENTORNO WEB**, certifico que este trabajo investigativo fue analizado por el Sistema Antiplagio Compilatio, obteniendo como porcentaje de similitud de **[6 %]**, resultados que evidenciaron las fuentes principales y secundarias que se deben considerar para ser citadas y referenciadas de acuerdo a las normas de redacción adoptadas por la institución y Facultad.

Considerando que, en el Informe Final el porcentaje máximo permitido es el 10% de similitud, queda aprobado para su publicación.


CERTIFICADO DE ANÁLISIS

CASO DE ESTUDIO FINAL WILSON VILLALVA

6% Similitudes

Fuentes principales detectadas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1		1%		
2		1%		
3		< 1%		

Por lo que se adjunta una captura de pantalla donde se muestra el resultado del porcentaje indicado.

Ing. Omar Montece Moreno, Msg
DOCENTE DE LA FAFI.