Arquitectura de software para exponer datos abiertos de E-commerce aplicando metodología SCRUM y la norma ISO/IEC 25023 en la Cámara Ecuatoriana de Comercio Electrónico

Maestrantes: Ing. Caiza Caizabuano, José Rubén
Ing. Mise Pasuña, José Luis

Tutor: Ing. Quiña Mera, José Antonio, Mgs

1. Antecedentes

La comunidad de desarrolladores ha visto como alternativa para solucionar algunos problemas de las RESTFUL API a las API GraphQL por los tantos beneficios que presenta el modelo de funcionamiento de esta tecnología.

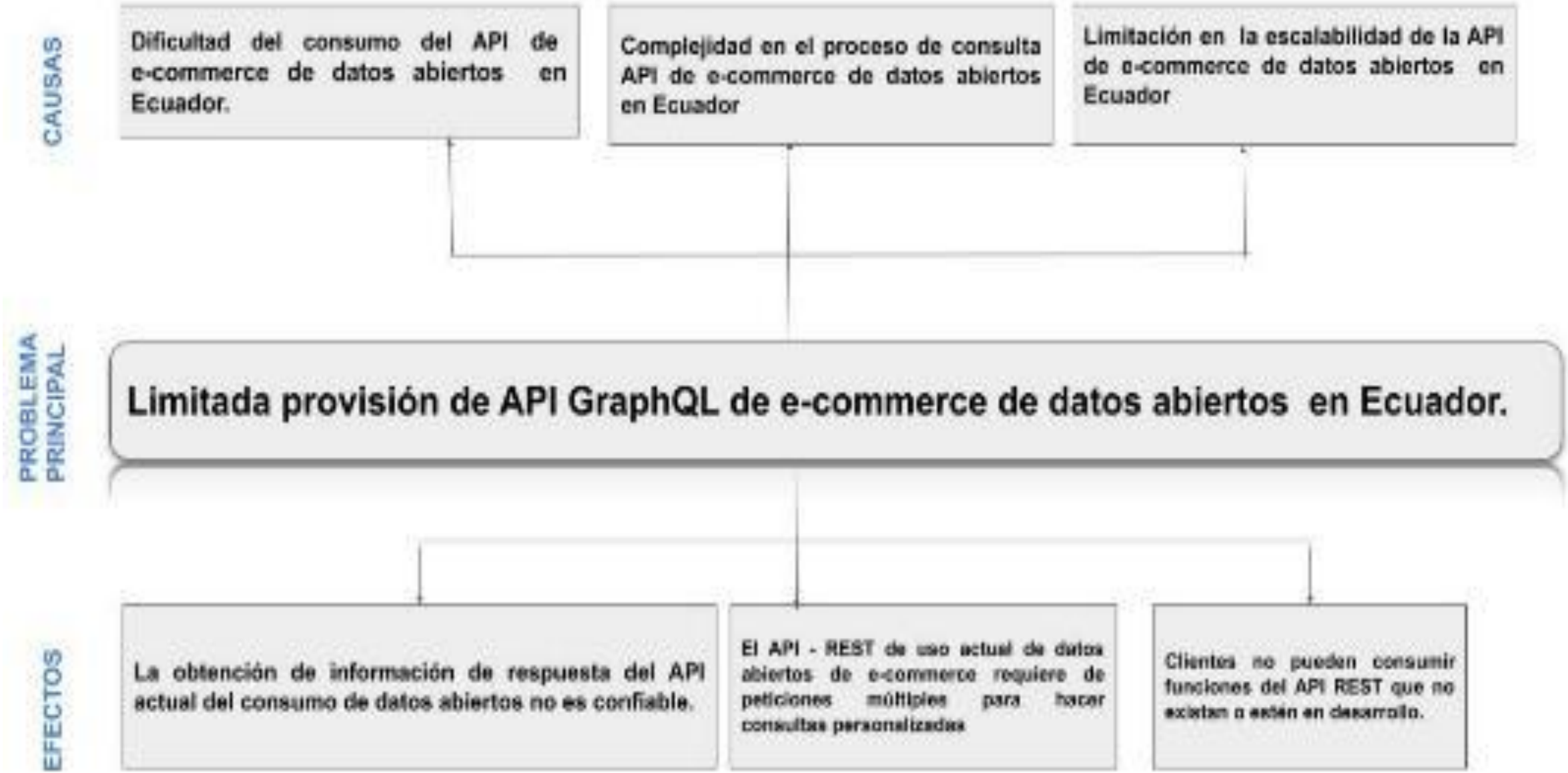
En el caso de la herramienta generada para el consumo de datos abiertos, al tener la arquitectura REST; presenta los inconvenientes que tiene este tipo de arquitectura en términos de optimización y flexibilidad; además el consumo se vuelve algo tedioso del lado del cliente ya que debe realizar varias peticiones si desea obtener información personalizada (información de varios EndPoint).

2. Planteamiento del problema

Las limitaciones y la dificultad de consumo de Datos abiertos en la Cámara Ecuatoriana de Comercio Electrónico provoca malestar al cliente al no poder consultar alguna información.

La falta de una herramienta (Api GraphQL) que sea eficiente e intuitiva en el consumo de datos abiertos para el cliente origina el desconocimiento de la interacción de la tecnología y el comercio electrónico en el Ecuador.

De acuerdo al estudio de investigación realizada por la Universidad Espíritu Santo (UES) manifiesta: “Hay cada vez más internautas, especialmente a través de teléfonos inteligentes en el país. Ecuador es un país con un camino por recorrer en términos de E-commerce y las barreras en las que tiene que trabajar más están centradas en educación, seguridad y confianza de los usuarios para un acelerado crecimiento en esta relación comercial”. (UES)


CAUSAS
Dificultad del consumo del API de e-commerce de datos abiertos en Ecuador.
Complejidad en el proceso de consulta API de e-commerce de datos abiertos en Ecuador
Limitación en la escalabilidad de la API de e-commerce de datos abiertos en Ecuador
PROBLEMA PRINCIPAL
Limitada provisión de API GraphQL de e-commerce de datos abiertos en Ecuador.
EFECTOS
La obtención de información de respuesta del API actual del consumo de datos abiertos no es confiable.
El API - REST de uso actual de datos abiertos de e-commerce requiere de peticiones múltiples para hacer consultas personalizadas
Clientes no pueden consumir funciones del API REST que no existan o estén en desarrollo.

3. Objetivo general y específicos

Desarrollar una arquitectura de software para exponer datos abiertos de E-commerce aplicando metodología SCRUM y la norma ISO/IEC 25023 en la Cámara Ecuatoriana de Comercio Electrónico

- Establecer el marco teórico del estado del arte acerca de la creación de envoltorios GraphQL.
- Implementar un envoltorio GraphQL del API REST de datos abiertos del E-commerce de la cámara de comercio de ecuador utilizando la metodología SCRUM.
- Evaluar la arquitectura del API GraphQL desarrollado, basándose en la calidad externa de la norma ISO/IEC 25023..
- Analizar estadísticamente los resultados obtenidos..

4. Materiales y métodos

Fundamentación Teórica

- Caracterización del Proceso de desarrollo de software
- Caracterización del Modelo de Calidad Externa
- Caracterización Tecnológica

Desarrollo del modelo

- Objetivo del modelo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa

Medición del modelo

- Desarrollo de la aplicación móvil
- Medición del modelo de calidad externa

Evaluación del modelo

- Establecer los requisitos
- Especificar la evaluación
- Ejecutar la evaluación
- Concluir la evaluación

4. Materiales y métodos

Fundamentación Teórica

Proceso de desarrollo	Modelo de calidad externa	Tecnología
• Desarrollo de software • Proceso de desarrollo de software • Garantía de calidad del software SQA • Garantía de calidad • Control de calidad	• Calidad • Calidad de software • Modelos de calidad de software • Modelos a nivel de productos	• Arquitectura de software • Arquitectura de microservicios • GraphQL • React Native

4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa

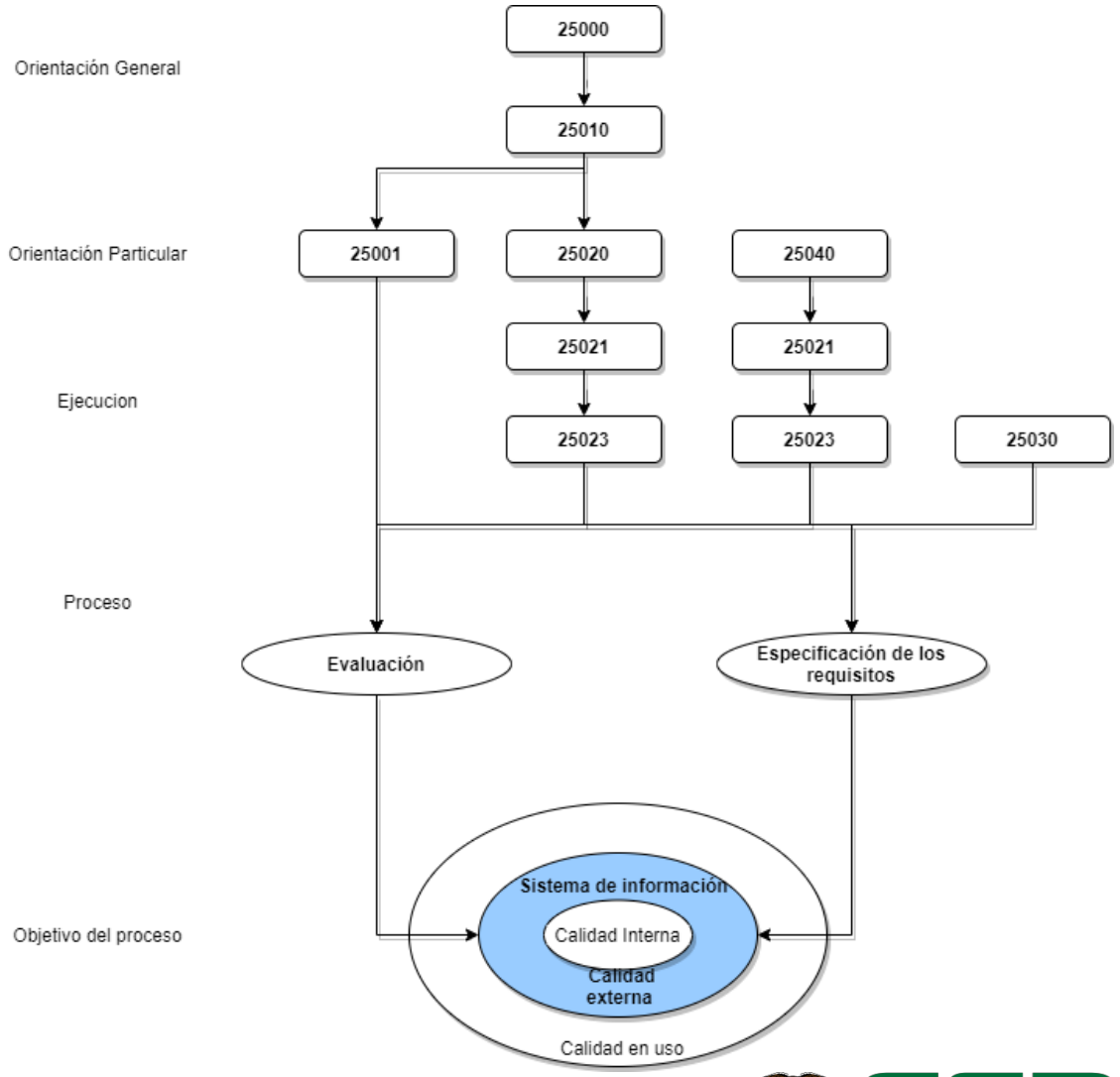
El modelo de calidad externa de software abreviado MCES, servirá como instrumento para llevar a cabo la evaluación de los productos de software.

4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa

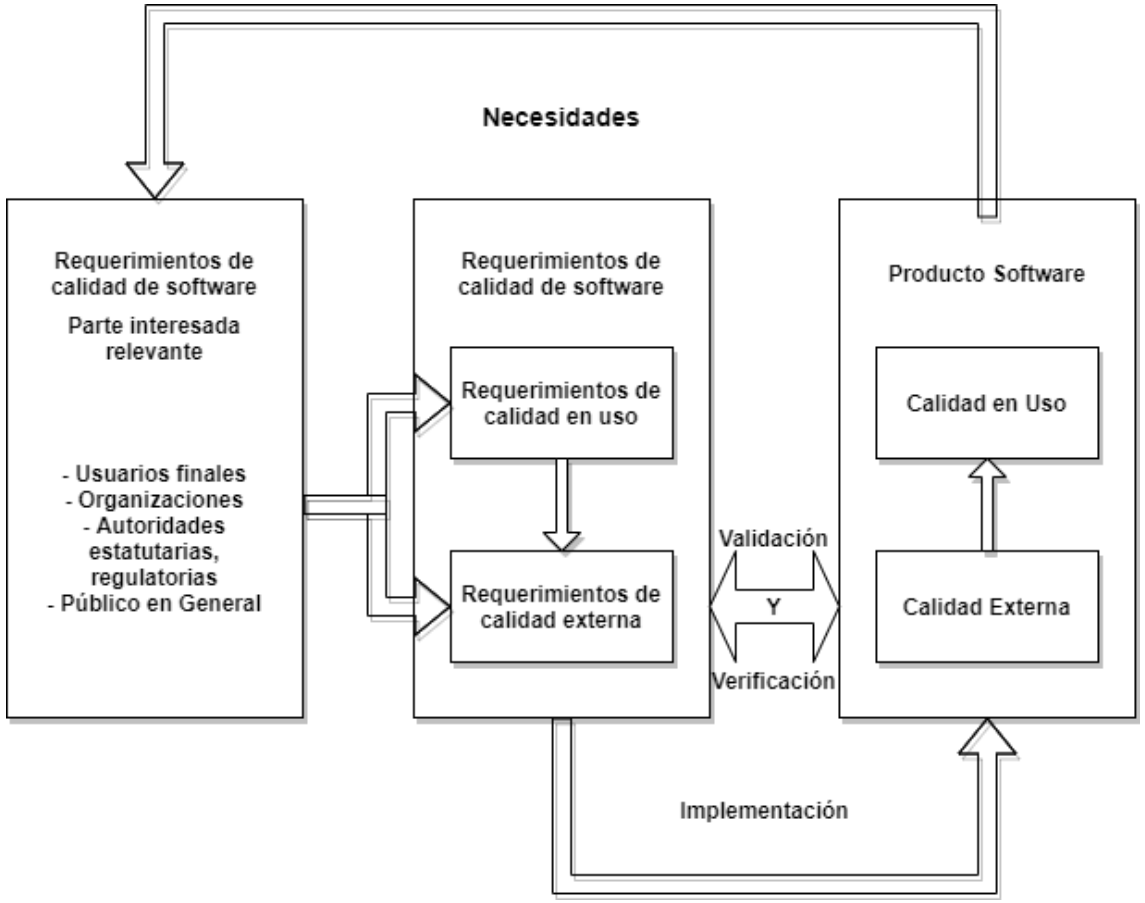


4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa

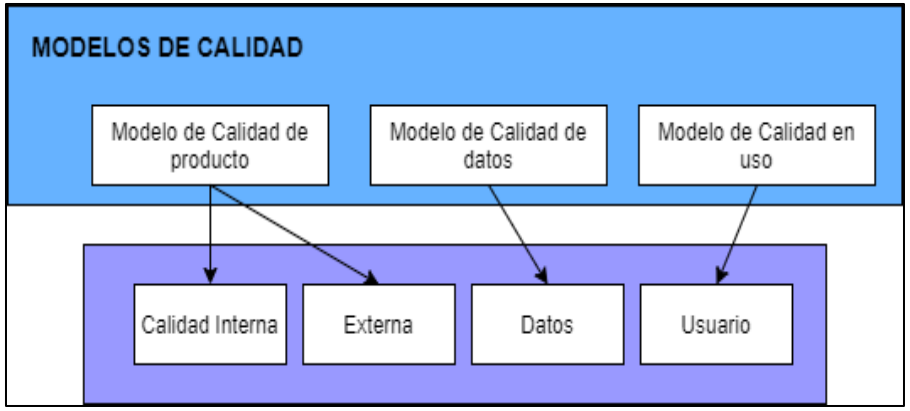


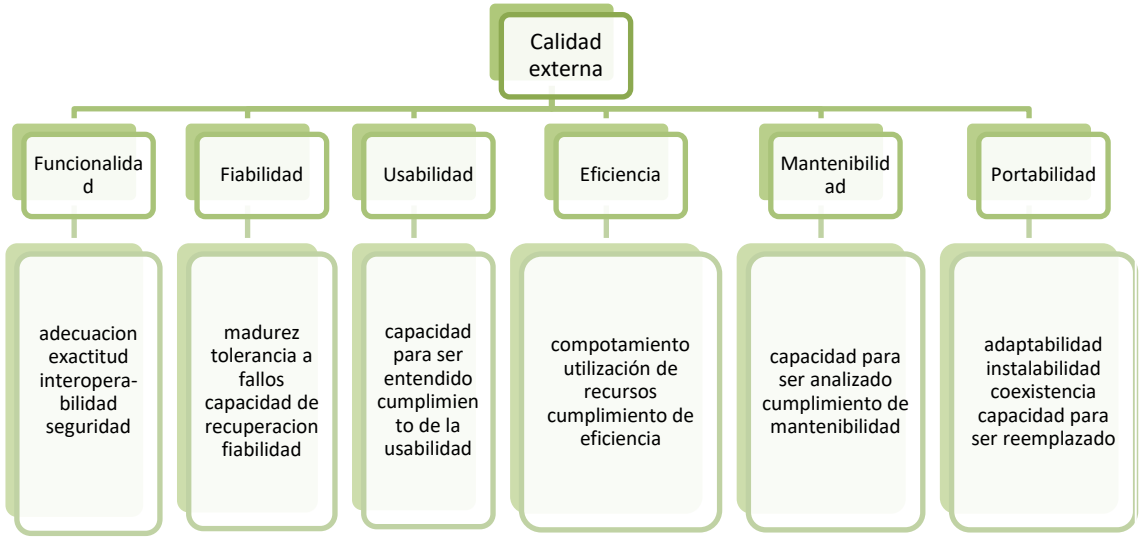
4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa



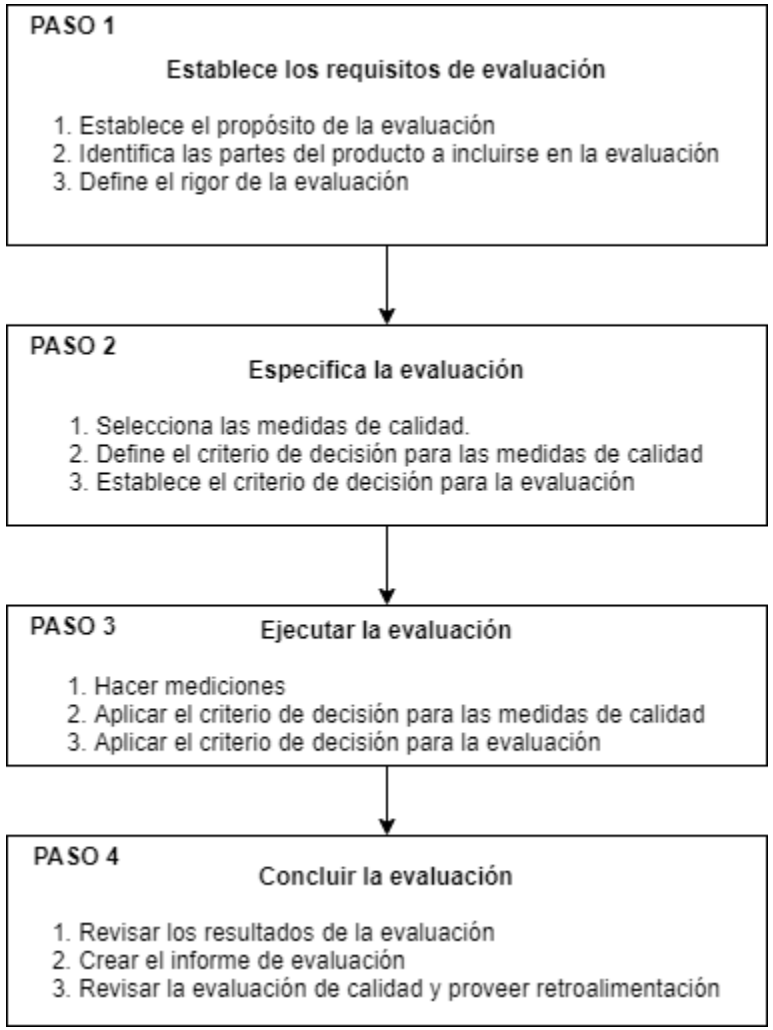


4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa



4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa
  1. Adecuación
  2. Fiabilidad
  3. Eficiencia en el desempeño
  4. Facilidad de uso
  5. Seguridad
  6. Compatibilidad
  7. Mantenibilidad
  8. Portabilidad

Subcaracterísticas	Métricas	Nivel de recomendación	Peso característica	Peso subcaracterística
Completitud funcional	Completitud de la implementación funcional.	MR	5%	3.00%
Exactitud funcional	Exactitud	MR		1.00%
	Precisión computacional	MR		1.00%

4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa
  1. Adecuación
  2. Fiabilidad
  3. Eficiencia en el desempeño
  4. Facilidad de uso
  5. Seguridad
  6. Compatibilidad
  7. Mantenibilidad
  8. Portabilidad

Subcaracterísticas	Métricas	Nivel de recomendación	Peso característica	Peso subcaracterística
Madurez	Disipación del fallo	R	30%	2.00%
	Suficiencia de las pruebas	R		1.00%
	Tiempo medio entre fallos	MR		10.00%
Disponibildad	Tiempo de servicio	MR		5.00%
	Tiempo medio de inactividad	R		2.00%
Tolerancia a fallos	Prevención de fallas	MR		4.00%
	Redundancia	R		1.00%
	Anulación de operación incorrecta	DU		0.00%
Recuperabilidad	Tiempo medio de recuperación	MR		5.00%

4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa
  1. Adecuación
  2. Fiabilidad
  3. Eficiencia en el desempeño
  4. Facilidad de uso
  5. Seguridad
  6. Compatibilidad
  7. Mantenibilidad
  8. Portabilidad

Subcaracterísticas	Métricas	Nivel de recomendación	Peso característica	Peso subcaracterística
Comportamiento temporal	Tiempo de respuesta	MR	20%	4.00%
	Tiempo de espera	R		4.00%
	Rendimiento	R		3.00%
Utilización de recursos	Líneas de código	R		2.00%
	Utilización de CPU	MR		1.00%
	Utilización de la memoria	R		1.00%
	Utilización de los dispositivos de E/S	R		1.00%
Capacidad	Número de peticiones	R		2.00%
	Numero de accesos simultáneos	R		2.00%
	Sistema de transmisión de ancho de banda	DU		0.00%

4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa
  1. Adecuación
  2. Fiabilidad
  3. Eficiencia en el desempeño
  4. Facilidad de uso
  5. Seguridad
  6. Compatibilidad
  7. Mantenibilidad
  8. Portabilidad

Subcaracterísticas	Métricas	Nivel de recomendación	Peso característica	Peso subcaracterística
Capacidad de reconocer su adecuación	Integridad de descripción	MR	20%	3.00%
	Capacidad de demostración.	DU		1.00%
Capacidad de ser entendido	Funciones evidentes	DU		1.00%
	Efectividad de la documentación del usuario o ayuda del sistema	DU		1.00%
Operatividad	Recuperabilidad de error operacional	R		2.00%
	Claridad de mensajes	R		2.00%
	Consitencia operacional	MR		3.00%
	Posibilidad de personalización	DU		0.00%
Protección contra errores del usuario	Verificación de entradas válidas	MR		3.00%
	Prevención del uso incorrecto	MR		3.00%
Estética de la interfaz del usuario	Personalización de la apariencia de la interfaz del usuario	DU		0.00%
Accesibilidad técnica	Accesibilidad física	R		1.00%

4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa
  1. Adecuación
  2. Fiabilidad
  3. Eficiencia en el desempeño
  4. Facilidad de uso
  5. Seguridad
  6. Compatibilidad
  7. Mantenibilidad
  8. Portabilidad

Subcaracterísticas	Métricas	Nivel de recomendación	Peso característica	Peso subcaracterística
Confidencialidad	Capacidad de control de acceso	MR	15%	3.00%
	Encriptación de datos	R		1.00%
Integridad	Prevención de corrupción de datos	MR		3.00%
No repudio	Utilización de firma digital	R		1.00%
Responsabilidad	Capacidad de auditoría de acceso	MR		3.00%
Autenticidad	Métodos de autenticación	MR		4.00%

4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa
  1. Adecuación
  2. Fiabilidad
  3. Eficiencia en el desempeño
  4. Facilidad de uso
  5. Seguridad
  6. Compatibilidad
  7. Mantenibilidad
  8. Portabilidad

Subcaracterísticas	Métricas	Nivel de recomendación	Peso característica	Peso subcaracterística
Co – existencia	Co – existencia disponible	MR	3%	1.50%
Interoperatividad	Conectividad con sistemas externos	MR		1.00%
	Capacidad de intercambiar de datos	R		0.50%

4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa
  1. Adecuación
  2. Fiabilidad
  3. Eficiencia en el desempeño
  4. Facilidad de uso
  5. Seguridad
  6. Compatibilidad
  7. Mantenibilidad
  8. Portabilidad

Subcaracterísticas	Métricas	Nivel de recomendación	Peso característica	Peso subcaracterística
Modularidad	Capacidad de condensación	DU	4%	0.00%
	Acoplamiento de clases	R		1.00%
Reusabilidad	Ejecución de reusabilidad	MR		1.00%
Capacidad de ser analizado	Capacidad de pistas de auditoria	MR		1.00%
	Diagnóstico de funciones suficientes	R		0.50%
Capacidad de ser modificado	Complejidad ciclomática	DU		0.00%
	Profundidad de herencia	DU		0.00%
	Grado de localización de corrección de impacto	DU		0.00%
	Complejidad de modificación	R		0.50%
	Índice de éxito de modificación	R		0.00%
Capacidad de ser probado	Completitud funcional de funciones de pruebas	R		0.00%
	Capacidad de prueba autónoma	DU		0.00%
	Capacidad de reinicio de pruebas	DU		0.00%

4. Materiales y métodos

Desarrollo del modelo

Componentes:

- Objetivo
- Modelo de referencia
- Ciclo de vida
- Estructura y características
- Evaluación
- Modelo de calidad externa
  1. Adecuación
  2. Fiabilidad
  3. Eficiencia en el desempeño
  4. Facilidad de uso
  5. Seguridad
  6. Compatibilidad
  7. Mantenibilidad
  8. Portabilidad

Subcaracterísticas	Métricas	Nivel de recomendación	Peso característica	Peso subcaracterística
Adaptabilidad	Adaptabilidad en entorno hardware	MR	3%	1.00%
	Adaptabilidad en entorno de software	MR		1.00%
	Adaptabilidad en entorno organizacional	DU		0.00%
Capacidad de ser instalado	Eficiencia en el tiempo de instalación	R		0.25%
	Facilidad de instalación	R		0.25%
Capacidad de ser reemplazado	Consistencia en la función de soporte al usuario	MR		0.50%
	Inclusividad funcional	R		0.00%
	Uso continuo de datos	DU		0.00%

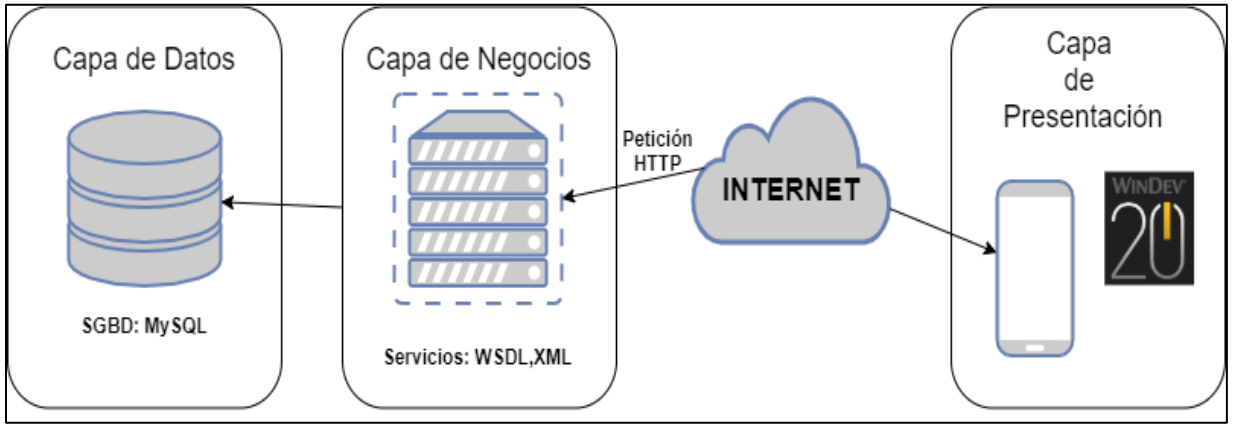
4. Materiales y métodos

Medición del modelo

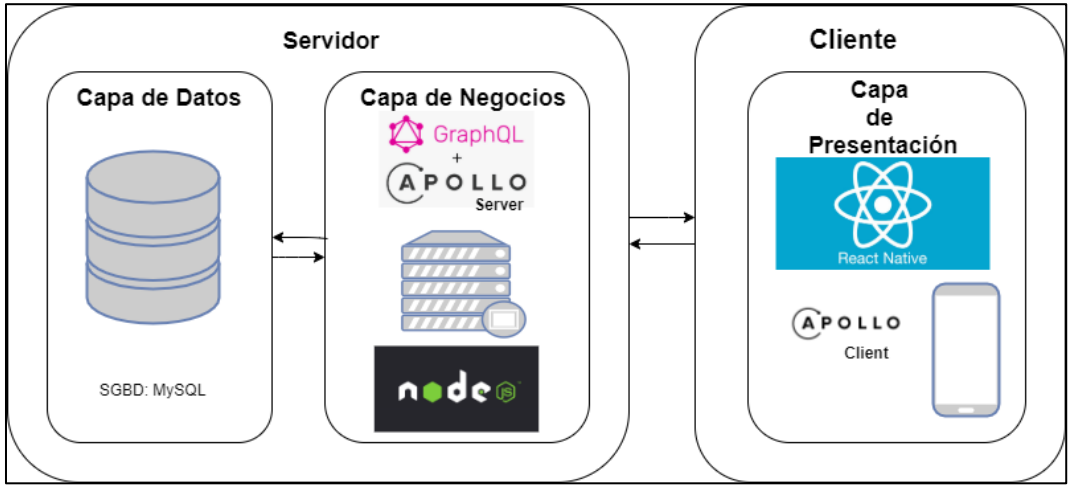
Componentes:

- Desarrollo de la aplicación móvil
- Medición del modelo de calidad externa

APP SUPPORT CONTROL



APP ACTIVITIES



4. Materiales y métodos

Medición del modelo

Componentes:

- Desarrollo de la aplicación móvil
- Medición del modelo de calidad externa
  1. Experimento
  2. Encuesta

1. Alcance

2. Planificación

3. Operación

4. Análisis e interpretación

Prueba de igualdad de medias de grupos

	Lambda de Wilks	F	gl1	gl2	Sig.
APIS_ REST	,976	,622	2	51	,541
APIS_ REST MULTI	,986	,371	2	51	,692
APIS_ GRAPHQL	,808	6,049	2	51	,004
CPU_ REST	,860	4,149	2	51	,021
CPU_ REST MULTI	,872	3,739	2	51	,031
CPU_ GRAPHQL	,911	2,481	2	51	,094

4. Materiales y métodos

Medición del modelo

Componentes:

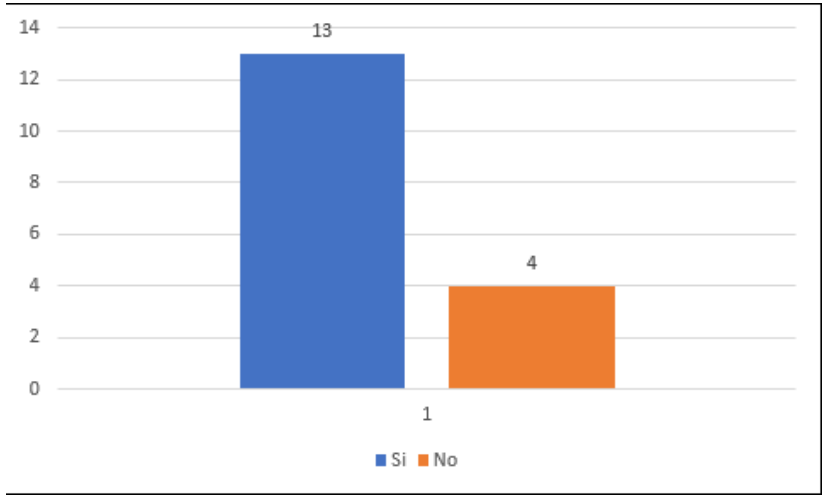
- Desarrollo de la aplicación móvil
- Medición del modelo de calidad externa
  1. Experimento
  2. Encuesta

Análisis de capacidad de ser instalado – facilidad de instalación

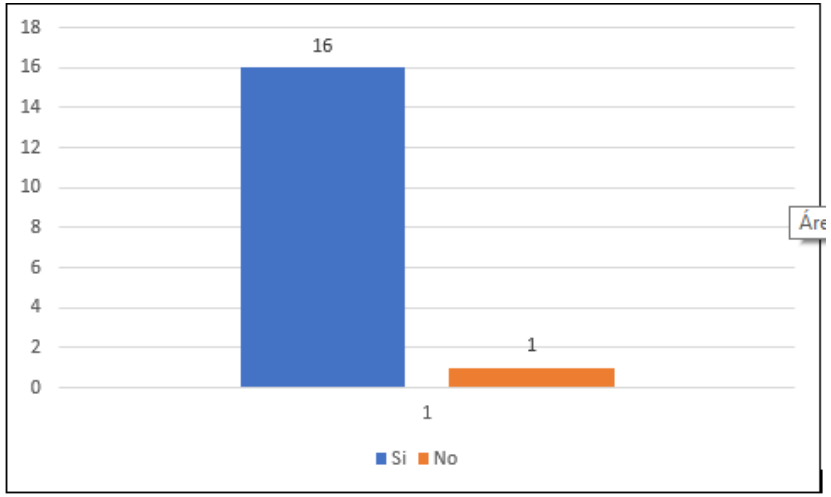
$$X = \frac{A}{B}$$

A: número de casos de éxito
B: número de casos totales

Support Control 0,77



Activities 0,94



4. Materiales y métodos

Medición del modelo

Componentes:

- Desarrollo de la aplicación móvil
- Medición del modelo de calidad externa
  1. Experimento
  2. Encuesta

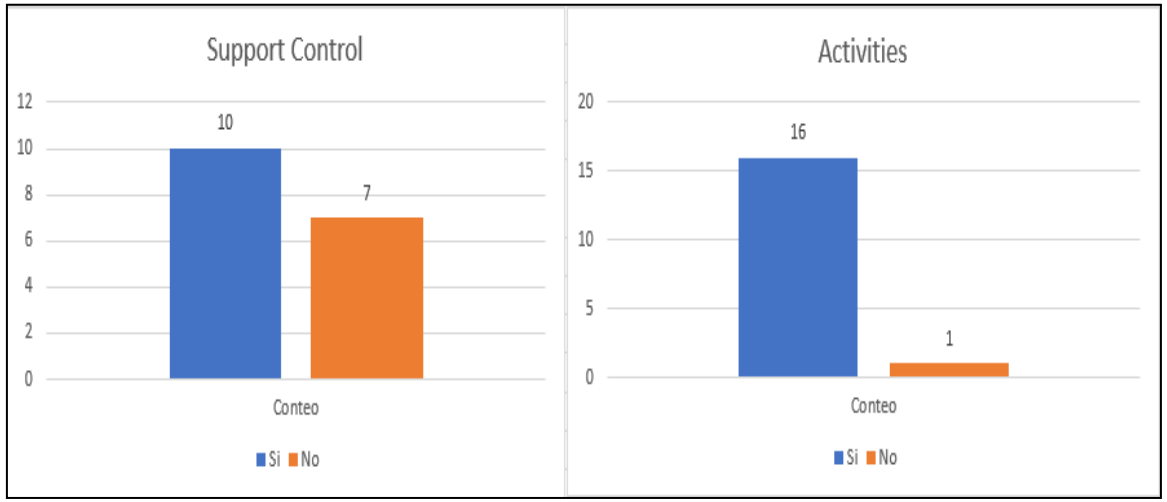
Adaptabilidad – Adaptabilidad al entorno hardware

$$X = 1 - \frac{A}{B}$$

A: número de casos fracasados
B: número de casos totales

Support Control 0,59

Activities 0,94



4. Materiales y métodos

Evaluación del modelo

Componentes:

- Establecer los requisitos
- Especificar la evaluación
- Ejecutar la evaluación
- Concluir la evaluación

Caracteris	Subcaracterist	Metricas	Nivel	% Carac	% Subcar.
Eficiencia	Comportamiento temporal	Tiempo de Respuesta	MR	20	4
	Utilización de recursos	Utilización CPU	MR		1
	Capacidad	Número de peticiones	R		2
Portabilidad	Adaptabilidad	Adaptabilidad entorno hardware	MR	3	1
	Capacidad de ser instalado	Facilidad de instalación	MR		0.25

4. Materiales y métodos

Evaluación del modelo

Componentes:

- Establecer los requisitos
- Especificar la evaluación
- Ejecutar la evaluación
- Concluir la evaluación

Interpretación de Resultados	Rango de Indicadores
EXCELENTE	$0.85 \leq X \leq 1$
BUENO	$0.75 \leq X \leq 0.84$
ACEPTABLE	$0.65 \leq X \leq 0.74$
REGULAR	$0.45 \leq X \leq 0.64$
DEFICIENTE	$X \leq 0.44$

4. Materiales y métodos

Evaluación del modelo

Componentes:

- Establecer los requisitos
- Especificar la evaluación
- Ejecutar la evaluación
- Concluir la evaluación

Características	Subcaracteristicas	Metricas	Valor X
Eficiencia	Comportamiento temporal	Tiempo de Respuesta	0.81
	Utilización de recursos	Utilización CPU	0.86
	Capacidad	Número de peticiones	0.81
Portabilidad	Adaptabilidad	Adaptabilidad entorno hardware	0.94
	Capacidad de ser instalado	Facilidad de instalación	0.94

Caract	Subcar	Metricas	% Carac	% Subcar.	% Md Carac	% Md Subcc
Eficiencia	Comportamiento temporal	Tiempo de Respuesta	87	50	70	40
	Utilización de recursos	Utilización CPU		12		10
	Capacidad	Número de peticiones		25		20
Portabilidad	Adaptabilidad	Adaptabilidad entorno hardware	13	10	12	9
	Capacidad de ser instalado	Facilidad de instalación		3		3

4. Materiales y métodos

Evaluación del modelo

Componentes:

- Establecer los requisitos
- Especificar la evaluación
- Ejecutar la evaluación
- Concluir la evaluación

La evaluación realizada al modelo de calidad externa de software es de 0.87 lo que equivale a un 82% sobre 100 considerando el modelo en un rango de excelente.

5. Conclusiones

- Mediante el marco teórico establecido, se logró realizar un laboratorio experimental controlado para comprobar si la arquitectura de software construida con GraphQL es eficiente. La realización del experimento se la realizó con la metodológica de la guía de (Wohlin et al., 2003) & (Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B., 2012), y para medir eficiencia nos basamos en las métricas de calidad externa de la normativa internacional ISO / IEC 25023: 2016.
- El uso del marco de trabajo SCRUM sirvió como una guía metodológica para el desarrollo ágil, ya que al hacerlo por iteraciones o Sprints, ayudó a alcanzar los objetivos de los requerimientos de forma rápida y eficiente en la implementación del envoltorio GraphQL del API REST de datos abiertos del E-commerce de la Cámara de Comercio del Ecuador.
- El estudio de las Normas ISO/IEC 25000, ayudó a entender la estructura y los procesos aplicados en las arquitecturas de software, del mismo exponer datos abiertos eficiente de E-commerce de la Cámara Ecuatoriana de Comercio Electrónico.

- Con el laboratorio experimental diseñado en este trabajo y el análisis estadístico se logró establecer, luego de comparar las APIs REST y GraphQL que exponen los datos públicos del E-commerce, se pudo evaluar que la arquitectura de software más eficiente a nivel de BackEnd es la arquitectura basada en el API-GraphQL (wrapper) que envolvió al API-REST. Los lenguajes de programación utilizados fueron javascript con la librería Apollo Server para la gestión de consulta de datos con GraphQL. La combinación de ambas herramientas alcanzó una correlación positiva de la evaluación luego de ejecutar 4 casos de uso, y así se pudo aceptar la hipótesis alternativa que menciona: es posible desarrollar una arquitectura orientada a microservicios, utilizando el lenguaje de consultas GraphQL, que sea más eficiente que otras.
- En la validación de la arquitectura de software utilizando el Wrapper API GraphQL, se pudo ejecutar el proceso de evaluación de calidad, donde se evaluó dos variables, el tiempo de respuesta: 897,8643867 milisegundos lo cual se considera excelente y la variable de rendimiento: 0.003 mientras más bajo mejor de acuerdo a la consideración en lambda de Wilks, lo que evidencia que el envoltorio del API GraphQL al exponer datos abiertos mejora la eficiencia y el desempeño al realizar las consultas.

6. Recomendaciones

- En primera instancia, se recomienda el uso de los envoltorios API GraphQL, ya que como se ha comprobado, al consumir datos, posee muchas ventajas frente al API REST, por tal razón se posiciona como la mejor opción en la actualidad.
- Considerando el alcance de la presente investigación es recomendable para futuros trabajos, evaluar los demás criterios de calidad enmarcados en la ISO/IEC 25023, que no son medidos en esta oportunidad, de tal manera de que sean complementarios en sus análisis.
- En este estudio se configuraron 3 arquitecturas con aplicativos asociados a 1 lenguaje que es compatible con la herramienta GraphQL y que registran mayor demanda de uso por las comunidades de programadores, este lenguaje es: JavaScript, por lo tanto, se recomienda realizar ensayos con otras opciones, manteniendo un ambiente de pruebas de controlado.

- Además, se recomienda aplicar este tipo de envoltorios em diferentes ámbitos, no solo para el E-commerce, como es este el caso, sino a nivel empresarial, comercial, etc., y así beneficiarse de las bondades de este servicio web.
- Dada la metodología de prueba de software utilizada, uno de los aspectos que permita la validación de los experimentos es tener un ambiente controlado, y esto se caracteriza por el hecho de que permita registrar los eventos que se presenta durante la ejecución de las pruebas para su posterior análisis, por ello se recomienda, el esquema presentado por Wohilin et al. (2012), para la ejecución y control de las fases de experimentales de los aplicativos.
- Este proyecto contribuye a la generación de oportunidades de investigación con datos reales y que profesionales pueden usar la herramienta desarrollada para el consumo de datos abiertos de fácil acceso.