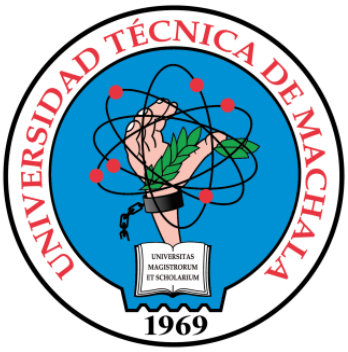


UTMACH

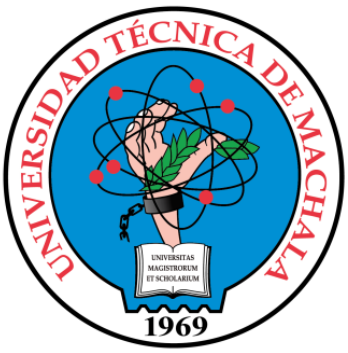
FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

ESTUDIO DE INDICADORES DE CALIDAD EN DESARROLLO DE SOFTWARE BASADO EN PROGRAMACIÓN ORIENTADA A OBJETOS

PIÑA OROZCO JOSIAS ISRAEL
INGENIERO DE SISTEMAS

MACHALA
2019



UTMACH

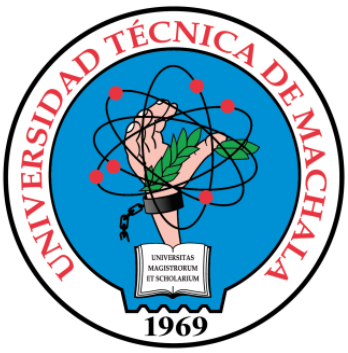
FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

ESTUDIO DE INDICADORES DE CALIDAD EN DESARROLLO DE SOFTWARE BASADO EN PROGRAMACIÓN ORIENTADA A OBJETOS

PIÑA OROZCO JOSIAS ISRAEL
INGENIERO DE SISTEMAS

MACHALA
2019



UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

EXAMEN COMPLEXIVO

ESTUDIO DE INDICADORES DE CALIDAD EN DESARROLLO DE SOFTWARE BASADO EN PROGRAMACIÓN ORIENTADA A OBJETOS

PIÑA OROZCO JOSIAS ISRAEL
INGENIERO DE SISTEMAS

HONORES TAPIA JOOFRE ANTONIO

MACHALA, 23 DE AGOSTO DE 2019

MACHALA
23 de agosto de 2019

Nota de aceptación:

Quienes suscriben, en nuestra condición de evaluadores del trabajo de titulación denominado Estudio de indicadores de calidad en desarrollo de software basado en programación orientada a objetos, hacemos constar que luego de haber revisado el manuscrito del precitado trabajo, consideramos que reúne las condiciones académicas para continuar con la fase de evaluación correspondiente.

HONORES TAPIA JOOFRE ANTONIO
0704811751
TUTOR - ESPECIALISTA 1

CÁRDENAS VILLAVICENCIO OSCAR EFRÉN
0703935312
ESPECIALISTA 2

MOROCHO ROMAN RODRIGO FERNANDO
0703820464
ESPECIALISTA 3

Fecha de impresión: viernes 23 de agosto de 2019 - 14:29

URKUND

Urkund Analysis Result

Analysed Document: Caso_Practico_Josias_Piña.docx (D54802925)
Submitted: 8/13/2019 5:48:00 PM
Submitted By: jipina_est@utmachala.edu.ec
Significance: 0 %

Sources included in the report:

Instances where selected sources appear:

0

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

El que suscribe, PIÑA OROZCO JOSIAS ISRAEL, en calidad de autor del siguiente trabajo escrito titulado Estudio de indicadores de calidad en desarrollo de software basado en programación orientada a objetos, otorga a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tiene potestad para otorgar los derechos contenidos en esta licencia.

El autor declara que el contenido que se publicará es de carácter académico y se enmarca en las dispociones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

El autor como garante de la autoría de la obra y en relación a la misma, declara que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asume la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

Machala, 23 de agosto de 2019

PIÑA OROZCO JOSIAS ISRAEL
0750182867

DEDICATORIA

Para empezar con este trabajo me remito a agradecer a todas las personas que, a lo largo de mi vida han sido un pilar fundamental para mi desarrollo, tanto académico, como humanístico:

A mis padres, sin los cuales la existencia y constancia de mis logros no se habrían podido desarrollar, los cuales han cuidado de mi durante el trayecto fortuito que he recorrido que, con mano firme estuvieron cuando los problemas y vicisitudes aquejaban mi mente aún en crecimiento, inspirándome para resistir con fuerza los tropiezos y golpes que la vida puso en mi camino.

A mis compañeros, por ser la mano amiga que se dieron su apoyo y tiempo en el trayecto que compartimos, aquellas personas con las que las risas y dificultades se nos ha otorgado, haciendo de las duras batallas para alcanzar las metas, una carcajada fugaz.

Por terminar y no menos importante, a mis maestros, por ser fuente inequívoca de sabiduría que han orientado a sus estudiantes y a mí, con sus consejos, anécdotas y conocimiento, ofrecieron una visión de lo que cada uno deseaba desarrollar en el futuro, dando sus bases académicas no solo para llegar al final de la Carrera de Ingeniería de Sistemas, sino también en el proceso de este trabajo, que pondrá a prueba mis capacidades adquiridas.

Sr. Piña Orozco Josías Israel.

AGRADECIMIENTO

En primer lugar, quisiera agradecer a la gran institución universitaria “Universidad Técnica de Machala”, por otorgar la oportunidad de estudiar en sus establecimientos con la mejor calidad de infraestructura y gestión.

Así mismo quisiera dar una mención especial al Señor Ing. Joofre Honores, tutor académico del presente trabajo de titulación, con su ayuda y conocimiento me ha proporcionado el camino y perspectiva necesaria para la culminación con éxito de mi documento, requerido para la culminación de la carrera de Ingeniería de Sistemas. Me considero muy afortunado por tener la oportunidad de trabajar con un experto de su calibre. Trabajar con usted fue una gran experiencia de aprendizaje para mí.

Para concluir, quisiera expresar de forma especial a mis familiares, amigos y compañeros que, sin su presencia día a día, no habrían dado a pie a mantener mis fuerzas para continuar en la carrera, con lucha y constancia para cumplir los objetivos propuestos al iniciar mi vida como estudiante universitario.

RESUMEN

ESTUDIO DE INDICADORES DE CALIDAD EN DESARROLLO DE SOFTWARE WEB BASADO EN PROGRAMACIÓN ORIENTADA A OBJETOS

Piña Orozco Josias Israel, 0750182867

La evolución de las Tecnologías de la información y Comunicación en la cotidianeidad de la sociedad ha provocado que, dentro de los estratos empresariales más bajos o equipos de desarrollo inexpertos en el campo del desarrollo de software, las tareas de controlar y proveer una óptima calidad, sea un proceso cada vez más complejo. Dentro del ámbito de desarrollo web, la calidad formal y medible a base de pruebas, no representa una prioridad para el personal no experimentado, siendo así, que la aplicación de las mismas resulta casi innecesaria. Por estas razones el propósito de la investigación es "*Estudiar los indicadores de calidad en el desarrollo de software web basado en programación orientada a objetos utilizando diferentes métricas para la construcción de un modelo base evaluativo de idoneidad".* Con este fin, la cuestión central de la investigación es el siguiente: ¿Cuáles serían los indicadores potenciales que estarían incidiendo dentro en la etapa de desarrollo para la creación de un sistema? Siendo el punto central para evaluar, los indicadores presentes durante la creación de software web aplicada a la programación a objetos. Obteniéndose de forma precisa, mediante una comparativa entre las metodologías y sus ciclos de vida utilizados, un conjunto de métricas que ayudarán a determinar la cualificación cuantitativa y cualitativa, respecto a la calidad del desarrollo presente en ciclo de vida del software, aplicando un enfoque basado en el estándar de calidad ISO/IEC-9126. Además de un modelo base para la documentación de errores e indicador empleado para dar solución.

PALABRAS CLAVES: Metodologías, Indicadores, Métricas, Estándares, Calidad, Software basado en la Web.

ABSTRACT

TUDY OF QUALITY INDICATORS IN WEB SOFTWARE DEVELOPMENT PROCESS BASED ON OBJECT- ORIENTED PROGRAMMING

Piña Orozco Josias Israel, 0750182867

The evolution of Information and Communication Technologies in the daily life of society has led to the tasks of controlling and providing quality, be an increasingly complex process. Within the field of web development, formal and measurable quality based on evidence, does not represent a priority for inexperienced personnel, being thus the application of them is almost unnecessary. For these reasons the purpose of the research is to "*Study quality indicators in the development of web software based on object-oriented programming using different metrics for the construction of an evaluative base model of suitability*". To this end, the central question of research is this: What would be the potential indicators to affect in the development stage for the creation of a system? Being the central point to evaluate, the indicators present during the creation of web software applied to object programming. Accurately obtaining, through a comparison between the methodologies and their life cycles used, a set of metrics that will help determine the quantitative and qualitative qualification, regarding the quality of development present in the life cycle applying an approach based on the ISO/IEC-9126 quality standard. In addition to a base model for error documentation and indicator used to provide solution.

KEYWORDS: Methodologies, Indicators, Metrics, Standards, Quality, Web-based Software.

CONTENIDO

CONTENIDO DE TABLAS

1. INTRODUCCIÓN

Actualmente la web es uno de los recursos más utilizados tanto por personas comunes como profesionales de la industria del desarrollo del software, debido a la gran capacidad de difundir información, recibir y enviar datos, realizar transacciones e incluso como centro de entretenimiento [1]. La gran diversidad de contenido que soportan ha dado cabida a que existan sistemas completos que antes solo podrían existir dentro de un sistema operativo, ocupando recursos en disco duro, complementos, programas de terceros, etc. Por ello, los navegadores son cada vez más actualizados y mejorados en rendimiento, como en velocidad de respuesta; gracias a ello se disminuye el valor de despliegue y mejora continua, pues su contenido está enfocado en la accesibilidad multiplataforma que cuenten con un programa informático capaz de acceder a enlaces específicos de carácter hipertextual, sea con conexión a internet o no.

Los softwares soportados en un medio web, han dado pie al uso y distribución de información en forma precisa e inmediata, lo que hace de la intercomunicación entre personas de una demografía o alrededor del mundo, una experiencia mucho más natural y ágil; así mismo se logra un control más exacto de cuándo y cómo esa información será entregada, es decir, actuar sólo cuando se demanda. Una de las grandes ventajas es que se ha eliminado el uso de registros o grandes colecciones de datos (inventarios) que deban almacenarse al momento de ejecutar una orden. A medida que el mundo esté mucho más interconectado por el acceso al internet y el desarrollo de nuevas tecnologías que permitan a todos los seres humanos compartir de un servicio, que hoy en día es algo tan frecuente que se usa en cualquier ámbito de trabajo profesional o incluso para el ocio; las páginas web no serán sino un medio que adquiera cada vez más relevancia, debido a que posee un enfoque publico principalmente global, siendo su acceso descentralizado y sin fronteras [2].

La innovación en los entornos web cada vez más complejos, posibilita a los expertos e investigadores en la materia de gestión y desarrollo de software, plantear nuevas técnicas y métodos para dar sostén, mejorar, y controlar la creación de softwares web, combinando metodologías existentes, así como también propuestas únicas que se enfocan en nuevos parámetros de calidad, seguridad, accesibilidad y estabilidad [2]. Con lo anterior mencionado, se ha propuesto para este trabajo, Estudiar los indicadores de calidad en el desarrollo de software web basado en programación orientada a objetos utilizando diferentes métricas para la construcción de un modelo base evaluativo de idoneidad.

Para desarrollar el presente trabajo, se ha constituido la siguiente estructura detallada a continuación:

Capítulo 1: Descripción del marco contextual del problema, formulación del problema y pregunta de investigación, objetivo general del documento.

Capítulo 2: Fundamentación teórica de del trabajo, donde se detallan la recopilación de perspectivas de diferentes autores que han propuesto soluciones a problemas similares, así como también el marco metodológico y resultados obtenidos, que presentan una posible solución a la pregunta planteada en el Capítulo 1.

Capítulo 3: Se describen las conclusiones de trabajo, las cuales dependen de los resultados del Capítulo 2. Además, se documentará la bibliografía que ha sido citada a lo largo del documento.

1.1. Marco Contextual

El ámbito de desarrollo del software se ha convertido en un proceso que, con el paso de las décadas, ha enfocado su soporte más allá soporte de un sistema operativo específico. El empleo de los navegadores como una opción eficiente de soporte del software, debido a la existencia de Frameworks y las mejoras en sus lenguajes de codificación, mayor compatibilidad, facilidad de desarrollo, rápida accesibilidad al cliente desde cualquier dispositivo que disponga de un navegador y una comunidad que dispone sus esfuerzos a mejorar las técnicas de desarrollo, ha provisto de una reducción el gasto de recursos de equipo para soportar su procesamiento y labor del personal en la obtención de un producto final de software web. Aun así, incluso con las mejores tecnologías, resulta necesaria la aplicación de técnicas y procedimientos que permitan gestionar las actividades, con el fin de controlar que toda tarea asignada al personal se esté llevando con eficiencia y normalidad, esto con el fin de cumplir ya sea con entregables de forma periódica sin retrasos o inconvenientes; dichas actividades son reunidas en una sola guía de trabajo que, de forma ordenada y sencilla, son explicadas para que un grupo independiente o empresa dedicada las utilice para mejorar la calidad dentro de sus procesos y producto final [3]. En las metodologías de desarrollo de software, es necesario un adecuado seguimiento de todo el ciclo de vida, por lo tanto resulta oportuno la verificación de todas las etapas que conllevan, el cual va desde la fase de Análisis hasta la de Mantenimiento; por lo tanto, la aplicación de herramientas indicadoras que evalúen si se ha culminado con aceptación, se hace casi imprescindible la existencia de una síntesis serie de indicadores, que permitan obtener la calidad del Software pero, debido a la utilización de las mismas y a la existencias de herramientas, modelos y estándares de calidad neta del sistema o aplicación a nivel de producto, se ha de enfatizar la investigación en las fases del ciclo de vida dedicadas al diseño y codificación.

1.2. Problema

El uso arbitrario, generalizado e ineficiente de guías métricas dentro del proceso de desarrollo de software basados en la web por parte de las empresas o equipos, lo cual convierte a la medición de calidad del software en un trabajo arduo y complejo; por lo tanto, es necesario la recolección y construcción de una base, que permita evaluar la calidad del proceso de desarrollo del software desde el inicio al fin de su ciclo de vida, enfatizando en sus fases de diseño y codificación. Con esto, se plantea el problema con la siguiente interrogante: ¿Cuáles serían los indicadores potenciales que estarían incidiendo dentro en la etapa de desarrollo para la creación de un sistema?

1.3. Objetivo General

Estudiar los indicadores de calidad en el desarrollo de software web basado en programación orientada a objetos utilizando diferentes métricas para la construcción de un modelo base.

2. DESARROLLO

2.1. Marco Teórico

2.1.1. Programación Orientada a Objetos

La POO pretende encapsular un pequeño número de clases evitando la redundancia de código, debido a las posibles existencias de que se puedan necesitar servicios específicos en varias clases determinadas. Además, estos encapsulamientos dan origen a módulos concentrados en los componentes para evitar un gasto innecesario de trabajo en las pruebas de cumplimiento y productividad, mejorando el grado de calidad en la culminación del software. La utilización del código repetitivo disminuye, dando acceso eficiente a las clases requeridas por los diferentes módulos [4].

2.1.2. La Web en el Desarrollo del Software

El proceso de desarrollo web es la adopción de estrategias, herramientas y técnicas necesarias para llevar a cada la creación de un software destinado a funcionar dentro de un navegador, este proceso puede clasificarse en dos ámbitos, las cuales pueden interactuar entre sí, siempre y cuando el equipo de desarrollo así lo necesite: El lado del cliente y de servidor [5][6]. Dentro del lado cliente, el proceso se centra directamente en el modelamiento y diseño de lo que se visualizará por el cliente. Por otro lado, el servidor funciona con código fuente de carácter complejo, constituyendo el llamado Back-End, componiendo parte de la web con el que el usuario no interactúa.

2.1.3. Ciclo de Vida del Software

Es la especificación y manejo adecuado de los procesos del desarrollo del software, desenvolviéndose desde la fase inicial hasta la finalización, detallando las tareas y actividades que contienen fases para que el software sea culminado satisfactoriamente, avalando el cumplimiento de los requisitos del usuario y sistema, así como también comprobar que todos los métodos del desarrollo sean adecuados. Entre las fases más utilizadas tenemos: Especificación, Análisis, Diseño, Construcción Pruebas y Mantenimiento [7] [8].

2.1.4. Calidad del Software

La calidad del software es un conjunto de procedimientos que utilizan métodos para comprobar que el software cumpla con una serie de parámetros idóneos acorde a un producto satisfactorio, así como también mantener gestionado la evolución a lo largo del ciclo de vida. Existen ciertos indicadores que pueden probar de manera general que el desarrollo ha sido correcto, estos pueden ser: El porcentaje de cumplimiento de un procedimiento, en ciertas condiciones sobre un tiempo determinado, Cuán apto es el software de ser comprendido, agradable, asimilable y empleabilidad flexible por el usuario, Rendimiento y Mantenibilidad adecuada [8] [9].

2.1.4.1. Metodologías de Desarrollo Web

Al igual que las metodologías tradicionales y ágiles, dentro del desarrollo web existen actividades y etapas que constituyen la creación del software, dando como resultado un resultado final de calidad. Principalmente las metodologías web se enfocan en las etapas iniciales, debido a que los requisitos deben ser especificados con exactitud para que en la fase de diseño el usuario pueda obtener la mejor usabilidad, navegabilidad, accesibilidad y facilidad de uso en la interacción del con el sistema [10] [11].

2.1.4.2. Métricas e Indicadores de Calidad

Las métricas de calidad son directrices que el equipo de desarrollo sigue para controlar, evaluar y cumplir con los requerimientos. La calidad para el software puede explicarse con métricas internas y externas, siendo la interna explicada como reportes de cumplimiento en el proceso de desarrollo en la culminación de un hito; las métricas externas son a diferencia de la interna, pruebas de calidad que verifican si el producto de software cumple con atributos como la Rendimiento, Funcionalidad, Confiabilidad, etc. [12] [13].

2.2. Marco Metodológico

Para el desarrollo de la investigación se han designado las siguientes etapas de análisis:

a) Metodologías aplicables al desarrollo web

1. En la **Tabla 1**, se ha realizado una recopilación de metodologías propuestas por diversos autores e investigadores para el desarrollo de software web, en función del paradigma de la programación utilizado, siendo estos: Orientado a Objetos (POO), Basado en Aspectos (PBA) o Basado en Componentes (PBC), su ciclo de vida en relación al ciclo de vida tradicional de 6 etapas: Especificación (ES), Análisis (AN), Diseño (DS), Codificación (CD), Pruebas (PB) y Mantenimiento (MT).
2. Se enfatiza la comparativa en las fases de diseño y codificación, debido a las prácticas existentes sobre las fases de especificación, análisis, pruebas y mantenimiento, por ello posteriormente los indicadores de calidad estarán desglosados únicamente en dichas fases; y se definirán métricas o modelos generalizados para la calidad en las otras áreas.

Metodologías / Abreviatura		Paradigma			Ciclo de vida del software					
		POO	PBA	PBC	ES	AN	DS	CD	PB	MT
Unified Process	UP	X			X	X	X	X	X	
Object Oriented Pattern	OOP	X	X	X		X	X	X	X	
Object Oriented Method	OOM	X					X	X	X	x
Relationship Navigational Analysis	RNA	X				X			X	
Web Site Design Method	WSDM			X		X	X	X		
Scenario Based Object-Oriented Hypermedia Design Method	SOOHDM	X		x	X	X	X	X	X	
Hypermedia Flexible Process Modeling	HFPM	X	x		X	X	X	X	X	X
Object Oriented Hypermedia Design Model	OOHDM	X	X	X			X	X		
Web Application Extension	WAE	X					X	X		
Relationship Management Methodology	RMM			X			X	X	X	
Service-Oriented Development Method	SODM	X			X	X	X	X		
Model Driven Architecture	MDA	X	X		X	X	X			
Navigational Development Tecniques	NDT	X		X						
UML-Based Web Engineering	UWE	X		X		X	X	X		
Enhanced Object Relationship Methodology	EORM	X		x			X	X		
Software Native Architecture Iterative-Logical	SNAIL	X			X	X	X	X	X	X

Tabla 1. Metodologías de desarrollo web en función al paradigma y la influencia en el ciclo de vida tradicional.
Elaboración: Autor, fuente: [14][15][16][17]

Se puede apreciar que la mayoría se ha orientado al Paradigma Orientado a Objetos, esto puede deberse a muchos factores como la facilidad de reutilización de código y

modelado simple pero eficiente en sistemas complejos, etc. Además se aprecia como las metodologías orientadas a la web no tienen un ciclo de vida que abarque todos los aspectos propuestos en las fases tradicionales, podría ser consecuencia de que la mayoría no enfatiza el esfuerzo en procesos de corrección de errores por ciclos (iteraciones), es decir, basar enteramente su proceso de desarrollo a través de un proceso en cascada, siendo HFPM quien mayormente enfatiza esfuerzos por cubrir todo este ciclo de vida y la necesidad de documentar procesos, pruebas y el modelado. Si bien se afirma que la robustez permite crear sistemas más complejos, consumen una gran cantidad de tiempo y costos debido a sus procesos lineales y poco flexibles. Por lo tanto, es importante que el ciclo de vida que permita abarcar proyectos de todo tamaño y permita un desarrollo ágil, procedural e iterativo, siendo SNAIL la que destaca en dichos aspectos debido a su arquitectura de desarrollo basada en la mitigación del riesgo en la fase de especificación, diseño de interfaz abstracta, acorde la programación adaptada a la POO, y pruebas itérales de producto de calidad.

b) Métricas del paradigma orientado a objetos

1. Se obtienen las métricas tradicionales de dentro del paradigma POO, tradicionalmente se propusieron 15 métricas adaptadas a diferentes niveles de análisis y características del paradigma, las cuales se detallan a continuación.

Características POO	**Nivel de Análisis**				
	Sistema	**Acoplamiento**	**Herencia**	**Clases**	**Métodos**
Localización				RFC, WMC	
Encapsulamiento	MHF, AHF		DIT, NOC		
Ocultamiento		CBO			
Herencia	MIF, AIF		CBO	SIX	
Polimorfismo	PF	CBO			
Abstracción	CF	CBO,			
No específico				LCOM	LOC, NOM,

Tabla 2. Métricas tradicionales de la POO.
Elaboración: Autor, fuente: [18]

2. Siendo el desarrollo del software un procedimiento evolutivo, se prioriza la necesidad de encontrar nuevas formas para la medición de calidad dentro del proceso de desarrollo de software, para el año 2017 [18], existen alrededor de 190 diferentes métricas, utilizadas para diferentes aspectos de la calidad en el desarrollo de software orientado a objetos, evidenciando una concentración de 10 métricas utilizadas frecuentemente desde el año 2010 [19], con ello se procede a realizar una comparativa con las métricas tradicionales (ver **Tabla 3**).

Métrica / Abreviatura		Relevancia	
		Tradicional	Actual
Method Hiding Factor	MHF	X	
Attribute Hiding Factor	AHF	X	
Method Inheritance Factor	MIF	X	
Attribute Inheritance Factor	AIF	X	
Polymorphism Factor	POF	X	
Coupling Factor	COF	X	
Weighted Methods per Class	WMC	X	X
Coupling Between Objects	CBO	X	X
Lack of Cohesion in Methods	LCOM	X	X
Depth of Inheritance Tree	DIT	X	X
Lines of Code	LOC	X	X
Number of Children	NOC	X	X
Response for a Class	RFC	X	X
Number of Methods	NOM	X	X
Specialisation Index per Class	SIX	X	
Cyclomatic Complexity	V(G)		X
Number of Attributes	NOA		X

Tabla 3. Comparación de métricas tradicionales y su relevancia actual.
Elaboración: Autor, fuente: [18]

En la **Tabla 3**, se evidencia poca variabilidad en la utilización de las métricas tradicionales para la validación en la calidad del proceso de desarrollo del software, de hecho se aprecia que dichas métricas siguen teniendo relevancia significativa para medir la calidad en muchos estudios, proyectos y/o pruebas [18]. Dentro de los cambios apreciables, se destaca el uso de las métricas Cyclomatic Complexity V(G) y Number of Attributes (NOA), dejando de lado la utilización de las métricas a nivel de sistema como indicadores de calidad dentro del paradigma orientado a objetos (ver **Tabla 2**).

c) Enfoque de calidad en el proceso de desarrollo de software

1. Es necesario identificar la incidencia de los indicadores de calidad basado en los enfoques que persigue evaluar dentro del desarrollo del software, para ello, se han identificado dos contextos principales de aplicación y medición en la calidad de todo software, siendo éstas: Calidad de Proceso, Calidad de Producto y Calidad en Uso [20]. Estos contextos también están relacionados intrínsecamente en el proceso de desarrollo de software, en la cual la calidad del proceso se basa en los niveles iniciales del ciclo de vida y la calidad del producto al igual que la calidad de uso, puede aplicarse directamente a las fases de pruebas o mantenimiento del software.
2. Se plantean a nivel de proceso (interna) y producto - uso (externa) distintas métricas que permiten establecer indicadores de calidad basados en factores de evaluación del estándar ISO/IEC 9126 – 2014, así como también el ciclo de vida de la metodología SNAIL.

2.3. Resultados

Una vez completado la recopilación y análisis de investigaciones propuestas por diversos autores acerca de las diferentes metodologías de desarrollo aplicables a los softwares soportados en la web, el enfoque o paradigma utilizado por los mismos autores, se obtuvo el paradigma más utilizado, resultando que, la programación orientada a objetos es comúnmente utilizada debido a la facilidad con la que ofrece modularidad de sus componentes y la reutilización eficiente del código.

Para validar la selección de las métricas internas orientadas a objetos, se ha realizado una búsqueda exhaustiva de modelos predictivos de fallos e impacto en la calidad de desarrollo del software a nivel de producto (ver **Tabla 4**).

Estudio	Modelo aplicado
An empirical study on software defect prediction with a simplified metric set. [19]	Modelo basado en escenarios de precisión por ANOVA y correlación.
Fault prediction considering threshold effects of object-oriented metrics. [21]	Validación y predicción de fallas a través de comparaciones de modelos y uso de Value of an Acceptable Risk Level (VARL).
Transitive-based object-oriented lack-of-cohesion metric. [22]	Desviación estándar de fallos reportados en tiempo de codificación.
Software bug prediction using object-oriented metrics. [23]	Pronóstico de errores en una clase cuando se prueba cualquier nuevo sistema en ella, algoritmo propio del autor.
Using Software Metrics Thresholds to Predict Fault-Prone Classes in Object-Oriented Software. [24]	Curva de características de funcionamiento del receptor (ROC).
A Combined-Learning Based Framework for Improved Software Fault Prediction.[25]	Curva de características de funcionamiento del receptor (ROC) y correlación.
Early software reliability analysis using reliability relevant software metrics. [26]	Lógica difusa y las métricas de software relevantes de confiabilidad de artefactos.
Design Of Software Fault Prediction Model Using BR Technique. [27]	Algoritmo de Regularización Bayesiana (BR).

Tabla 4. Estudios predictivos de fallos y mejora de la calidad, usando las métricas POO.
Elaboración: Autor.

Con lo anterior mencionado, se propone un modelo métrico o guía métrica de indicadores de calidad, basada en la programación orientada a objetos, metodologías y el estándar ISO/IEC 9126-2014, detallada a continuación (ver **Tabla 4**):

ISO/IEC 9126 – 2014	Calidad Interna				Calidad Externa	
	Ciclo de vida SNAIL					
	ES	AN	DS	CD	PB	MT
Funcionalidad	NR, SRR CR, IMS, FSS,	FP	DIT, CBO, NOC, SIX, CF, V(G),	WMC, NOM, NOA, LCOM MHF, AHF, FCI	AAL IADn	
Confiabilidad	CR, IMS, FSS		V(G), CBO, DIT, NOC, SIX, CF	LOC, LCOM,	FR VitR TUR	
Usabilidad	IMS	FP	DIT, NOC, AIF, MIH, POF, CF		EFL PERUs MC	MECT
Eficiencia	IMS, FSS	FP	DIT, V(G), CF		MRsT Throughput ThrComp IOU	
Mantenibilidad	NR SRR	FP	NOM, CBO, DIT, NOC, AIF, MIH, POF, CF, V(G),	LOC, RFC, LCOM, WMC, NOA		DSCp, CSR, MFR
Portabilidad		FP	DIT, NOM, NOC, CF, V(G),	RFC, LCOM, WMC	PorComp	PMAc, AAcOE
Efectividad					TskEfc	TSCR, AAcOE
Productividad					PP, PC	PP, PC
Seguridad					USafL, SWDL	
Satisfacción					USatL, USatC	

Tabla 5. Matriz de trazabilidad sobre los indicadores de calidad.
Elaboración: Autor.

Dentro del modelo en la **Tabla 5,** como parámetros de calidad se han seleccionado los indicadores que con mayor frecuencia pueden aplicarse a cualquier proyecto de desarrollo de software web, así como también, indicadores a nivel de especificación, pruebas y mantenimiento basados en la reformación del estándar ISO/IEC 9126 – 2014. Este estándar pretende el análisis completo del software a nivel de calidad interna, externa y de uso, permitiendo dar el mayor control de calidad al momento de ser presentados los entregables al cliente. Por otro lado, el empleo de las métricas a nivel de desarrollo neto del software, es decir la fase de diseño y codificación, permite

no solo garantizar la efectividad con la que el equipo, reduce la aplicación de pruebas de software a nivel de producto y corrección de errores.

3. CONCLUSION

El uso de las metodologías de desarrollo de software dentro de cualquier proyecto de desarrollo de web, ofrece la adecuada gestión de sus procesos para obtener un producto de software de calidad, aplicando una comparativa entre sus diferentes propuestas se pudo determinar que la orientación al diseño y codificación de calidad es imprescindible pero, si bien cada proceso es controlado para que sus requisitos sean desarrollados con eficiencia y rapidez, no todas constan de métodos que permiten obtener una buena especificación de requerimientos, por esto la necesidad de usar métricas e indicadores de calidad durante la creación del software basado en la web es importante para compensar la ineficacia en el tratamiento a nivel de requisitos del software. Se han utilizado investigaciones que reflejan modelos predictivos y de calidad basados en métricas de desarrollo orientadas a objetos y se pudo concluir cuales son los que provocan una incidencia mayor sobre el desarrollo, así como también el uso de métricas que impactan en la calidad externa del sistema a nivel de producto y de uso, lo cual pretende la mitigación y control de las falencias ocurridas durante los procesos internos del proyecto.

BIBLIOGRAFIA

[1] D. Fogli and G. Guida, “A practical approach to the assessment of quality in use of corporate web sites,” *J. Syst. Softw.*, vol. 99, pp. 52–65, 2015.

[2] N. Kumar, R. Dadhich, and A. Shastri, “Quality Models for Web-based Application: A Comparative Study,” *Int. J. Comput. Appl.*, vol. 125, no. 2, pp. 25–32, 2015.

[3] C. A. Sam-Anlas and Y. Stable-Rodríguez, “Evaluación de la accesibilidad web de los portales del Estado en Perú,” *Rev. española Doc. Científica*, vol. 39, no. 1, p. e120, Mar. 2016.

[4] M. Callejas-Cuervo, A. C. Alarcón-Aldana, A. M. Álvarez-Carreño, M. Callejas-Cuervo, A. C. Alarcón-Aldana, and A. M. Álvarez-Carreño, “Software quality models, a state of the art,” *Entramado*, vol. 13, no. 1, pp. 236–250, 2017.

[5] R. N. Martínez, *El Proceso de Desarrollo de Software: 2da Edición*, 2nd ed. CreateSpace Independent Publishing Platform, 2017.

[6] D. R. Cardozzo, *Desarrollo de Software: Requisitos, Estimaciones y Análisis. 2da Edición*, 2nd ed. CreateSpace Independent Publishing Platform, 2016.

[7] J. L. B. GÓMEZ, *UF1844 - Desarrollo de aplicaciones web en el entorno servidor*. Paraninfo, 2015.

[8] L. Marcela and A. Constanzo, “Comparación de modelos de calidad, factores y metricas en el ambito de la ingeniería del software,” *Inst. Tecnol. Apl.*, vol. 78, no. 1, pp. 1–36, 2014.

[9] J. L. Asociación Interciencia., F. Torres-Castillo, S. Alcaraz-Corona, and F. Banda-Muñoz, “CALIDAD, TIEMPO Y COSTO EN PROYECTOS DE DESARROLLO DE SOFTWARE,” *Interciencia*, vol. 43, no. 10, 2018.

[10] J. Molina, M. Zea, M. Contento, and F. García, “State of Art : Development Methodologies in Web,” *3C Tecnol. glosas innovación Apl. a la pyme*, vol. 6, no. 2254–4143, pp. 54–71, 2017.

[11] B. Zhao and Y. Zhu, “Formalizing and validating the web quality model for web source quality evaluation,” *Expert Syst. Appl.*, vol. 41, no. 7, pp. 3306–3312, 2014.

[12] S. Verona Marcos, Y. Pérez Díaz, L. Torres Pérez, M. Delgado Dapena, and C. Yáñez Márquez, “Pruebas de rendimiento a componentes de software utilizando programación orientada a aspectos,” *Ing. Ind.*, vol. 37, no. 3, pp. 278–285, 2016.

[13] I. Moreno-Montes-de-Oca, A. Rodríguez-Morffi, M. Snoeck, R. Moreno-Rodríguez, G. Casas-Cardoso, and L. González-González, “Directrices prácticas y métricas de calidad en la modelación de procesos de negocio: un caso de estudio,” *Rev. Cuba. Ciencias Informáticas*, vol. 8, no. 2, pp. 1–18, 2014.

[14] J. R. Molina Ríos and Z. Ordóñez, “METODOLOGÍAS DE DESARROLLO EN APLICACIONES WEB,” *Rev. Arje*, vol. 11, no. 3, pp. 245–270, 2017.

[15] A. N. del V. Rodríguez, “Propuesta para lograr especializacion en tiae título: metodologías de diseño usadas en ingeniería web, su vinculación con las ntics,” UNIVERSIDAD NACIONAL DE LA PLATA, 2009.

[16] J. R. Molina Ríos, M. P. Zea Ordóñez, F. F. Redrován Castillo, N. M. Loja Mora, M. R. Valarezo Pardo, and J. A. Honores Tapia, *SNAIL, Una metodología híbrida para el desarrollo de aplicaciones web*, I. Machala: ÁREA DE INNOVACIÓN Y DESARROLLO, S.L., 2018.

[17] J. Molina, M. Zea, J. Honores, and A. Gómez, “Analysis Methodologies Web Application Development,” *Int. J. Appl. Eng. Res.*, vol. 11, no. 0973–4562, pp. 9070–9078, 2016.

[18] A. S. Nuñez-Varela, H. G. Pérez-Gonzalez, F. E. Martínez-Perez, and C. Soubervielle-Montalvo, “Source code metrics: A systematic mapping study,” *J. Syst. Softw.*, vol. 128, pp. 164–197, 2017.

[19] P. He, B. Li, X. Liu, J. Chen, and Y. Ma, “An empirical study on software defect prediction with a simplified metric set,” *Inf. Softw. Technol.*, vol. 59, pp. 170–190, 2015.

[20] D. Galin, “Software Product Quality Metrics,” *Softw. Qual. Concepts Pract.*, pp. 346–374, 2018.

[21] R. Malhotra and A. J. Bansal, “Fault prediction considering threshold effects of object-oriented metrics,” *Expert Syst.*, vol. 32, no. 2, pp. 203–219, 2015.

[22] J. Al Dallal, “Transitive-based object-oriented lack-of-cohesion metric,” *Procedia Comput. Sci.*, vol. 3, pp. 1581–1587, 2011.

[23] D. L. Gupta and K. Saxena, “Software bug prediction using object-oriented metrics,” *Sadhana - Acad. Proc. Eng. Sci.*, vol. 42, no. 5, pp. 655–669, 2017.

[24] A. Boucher and M. Badri, “Using Software Metrics Thresholds to Predict Fault-Prone Classes in Object-Oriented Software,” *Proc. - 4th Int. Conf. Appl. Comput. Inf. Technol. 3rd Int. Conf. Comput. Sci. Appl. Informatics, 1st Int. Conf. Big Data, Cloud Comput. Data Sci.*, pp. 169–176, 2017.

[25] C. W. Yohannese and T. Li, “A Combined-Learning Based Framework for Improved Software Fault Prediction,” *Int. J. Comput. Intell. Syst.*, vol. 10, no. 1, p. 647, 2017.

[26] H. B. Yadav and D. K. Yadav, “Early software reliability analysis using reliability relevant software metrics,” *Int. J. Syst. Assur. Eng. Manag.*, vol. 8, no. 1992, pp. 2097–2108, 2017.

[27] R. Mahajan, S. K. Gupta, and R. K. Bedi, “Design of software fault prediction model using BR technique,” *Procedia Comput. Sci.*, vol. 46, no. Icict 2014, pp. 849–858, 2015.

[28] D. I. George and P. H. Maitheen, “Analysis of Object Oriented Metrics on a Java Application,” *Int. J. Comput. Appl.*, vol. 123, no. 1, pp. 32–39, 2015.

[29] T. J. Vijay, M. G. Chand, and H. Done, “Software Quality Metrics in Quality Assurance to Study the Impact of External Factors related to Time,” *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 7, no. 1, pp. 221–224, 2017.

[30] N. Padhy, S. Satapathy, and R. P. Singh, “State-of-the-art object-oriented metrics and its reusability: A decade review,” *Smart Innov. Syst. Technol.*, vol. 77, pp. 431–441, 2018.

[31] R. Jabangwe, J. Börstler, D. Šmite, and C. Wohlin, “Empirical evidence on the link between object-oriented measures and external quality attributes: A systematic literature review,” *Empir. Softw. Eng.*, vol. 20, no. 3, pp. 640–693, 2015.

[32] T. Hariprasad, G. Vidhyagaran, K. Seenu, and C. Thirumalai, “Software complexity analysis using halstead metrics,” *Proc. - Int. Conf. Trends Electron. Informatics, ICEI 2017*, vol. 2018-Janua, pp. 1109–1113, 2018.

[33] R. Sehgal and D. Mehrotra, “Predicting faults before testing phase using Halstead’s metrics,” *Int. J. Softw. Eng. its Appl.*, vol. 9, no. 7, pp. 135–142, 2015.

[34] A. H. G. Alena González Reyes, Margarita André Ampuero, “Análisis comparativo de modelos y estándares para evaluar la calidad del producto de software,” *Rev. Cuba. Ing.*, vol. 6, no. 3, pp. 43–52, 2015.

[35] K. Rahul, “A Metrics Design For Evaluation of Component’s Performance During Software Development Process,” *Int. J. Adv. Res. Comput. Sci.*, vol. 8, no. 5, pp. 106–111, 2017.

[36] P. Morrison, D. Moye, R. Pandita, and L. Williams, “Mapping the field of software life cycle security metrics,” *Inf. Softw. Technol.*, vol. 102, pp. 146–159, 2018.

ANEXOS

Tabla de Métricas de Calidad

Sufijo	Nombre	Forma de cálculo	Consideraciones
MHF	Method Hiding Factor	$\boldsymbol{MHF} = \frac{\sum_{i=1}^{TC} M_h(\boldsymbol{C_i})}{\sum_{i=1}^{TC} M_d(\boldsymbol{C_i})}$	$\boldsymbol{M_h(C_i)}$ = Métodos ocultos en la clase $\boldsymbol{C_i}$. $\boldsymbol{M_d(C_i)} = \boldsymbol{M_v(C_i)} + \boldsymbol{M_h(C_i)}$, Métodos definidos en la clase $\boldsymbol{C_i}$. $\boldsymbol{M_v(C_i)}$ = Métodos visibles en la clase $\boldsymbol{C_i}$. $\boldsymbol{TC}$ = Número total de clases.
AHF	Attribute Hiding Factor	$\boldsymbol{AHF} = \frac{\sum_{i=1}^{TC} \boldsymbol{A_h(C_i)}}{\sum_{i=1}^{TC} \boldsymbol{A_d(C_i)}}$	$\boldsymbol{A_h(C_i)}$ = Atributos ocultos en la clase $\boldsymbol{C_i}$. $\boldsymbol{A_d(C_i)} = \boldsymbol{A_v(C_i)} + \boldsymbol{A_h(C_i)}$, Atributos definidos en la clase $\boldsymbol{C_i}$. $\boldsymbol{A_v(C_i)}$ = Atributos visibles en la clase $\boldsymbol{C_i}$. $\boldsymbol{TC}$ = Número total de clases.
MIF	Method Inheritance Factor	$\boldsymbol{MIF} = \frac{\sum_{i=1}^{TC} M_i(\boldsymbol{C_i})}{\sum_{i=1}^{TC} M_a(\boldsymbol{C_i})}$	$\boldsymbol{M_a(C_i)} = \boldsymbol{M_d(C_i)} + \boldsymbol{M_i(C_i)}$ $\boldsymbol{M_i}$ = Métodos heredados. $\boldsymbol{M_d}$ = Métodos definidos. $\boldsymbol{TC}$ = Número total de clases.
AIF	Attribute Inheritance Factor	$\boldsymbol{AIF} = \frac{\sum_{i=1}^{TC} A_i(\boldsymbol{C_i})}{\sum_{i=1}^{TC} A_a(\boldsymbol{C_i})}$	$\boldsymbol{A_a(C_i)} = \boldsymbol{A_d(C_i)} + \boldsymbol{A_i(C_i)}$ $\boldsymbol{A_i}$ = Atributos heredados. $\boldsymbol{A_d}$ = Atributos definidos. $\boldsymbol{TC}$ = Número total de clases.
POF	Polymorphism Factor	$\boldsymbol{POF} = \frac{\sum_{i=1}^{TC} M_o(\boldsymbol{C_i})}{\sum_{i=1}^{TC}[[M_n(\boldsymbol{C_i}) \times DC(\boldsymbol{C_i})]}$	$\boldsymbol{M_o(C_i)}$ = Métodos **override** en la clase $\boldsymbol{C_i}$. $\boldsymbol{M_n(C_i)}$ = Métodos **new** en la clase $\boldsymbol{C_i}$. $\mathbf{DC}$ = Número de descendientes de la clase $\boldsymbol{C_i}$ (Clases derivadas). $\boldsymbol{TC}$ = Número total de clases.
CF	Coupling Factor	$\boldsymbol{COF} = \frac{\sum_{i=1}^{TC}\left\lvert\sum_{j=1}^{TC} is\_client\ (\boldsymbol{C_i}, \boldsymbol{C_j})\right\rvert}{TC^2 - TC}$	$\boldsymbol{is\_client}\ (\boldsymbol{C_c}, \boldsymbol{C_s}) = 1$ si, $\boldsymbol{C_c} \Rightarrow \boldsymbol{C_s}$ significa que $\boldsymbol{C_c}$ (clase de cliente) contiene al menos una referencia de no herencia a una característica (método o atributo) de clase $\boldsymbol{C_s}$ (clase de proveedor), de lo contrario el valor es 0. $\boldsymbol{TC}$ = Número total de clases.
WMC	Weighted Methods per Class	$\boldsymbol{WMC} = \sum_{i=1}^{n} c_i$	$\boldsymbol{n}$ = Número de métodos de la clase. $\boldsymbol{C_i}$ = Complejidad del método, en caso de no ser calculada, el valor es 1.

			$i =$ 1.
CBO	Coupling Between Objects	$CBO = \sum_{i=1}^{n} C_l$	C_l = Número de clases a las que la clase está relacionada sin herencia, se toma en cuenta el uso de métodos o variables externas.
LCOM	Lack of Cohesion in Methods	$LCOM = 1 - \frac{\sum MF}{M_d \times F}$	M_d = Métodos definidos en la clase F = Número de campos instanciados en la clase. MF = Número de métodos de la clase que accede a un campo de instancia particular.
DIT	Depth of Inheritance Tree	$DIT(c) = \sum \lvert Ac \rvert$	Ac = Máximo número **de a**ncestros de una clase hasta su clase padre, el cual le atribuye una herencia.
LOC	Lines of Code	$LOC = \sum_{i=1}^{n} LF$	LF = Líneas de código, nunca inferior a 1 y máximo 1000.
NOC	Number of Children	$NOC(c) = \sum \lvert Dc \rvert$	Dc = Recuento de las sub-clases que se derivan directamente de una clase especificada.
RFC	Response for a Class	$RFC(c) = \sum M_d + \sum M_r$	M_d = Métodos definidos en la clase. M_r = Métodos fuera externos que son llamados dentro de una clase.
NOM	Number of Methods	$NOM = \sum_{i=1}^{n} M_d(C_i)$	M_d = Número de métodos definidos por clase C_i .
SIX	Specialization Index per Class	$SIX = \frac{\sum_{i=1}^{n} M_o(C_i) \times AL}{\sum_{i=1}^{n} M_d(C_i)}$	$M_o(C_i)$ = Métodos **override** en la clase C_i. M_d = Métodos definidos en la clase. AL = Nivel de anidamiento en la jerarquía de clases.
V(G)	Cyclomatic Complexity	$V(G) = e - n + 2p$	$e =$ Número de aristas en el diagrama de flujo. $n =$ Número de nodos en el diagrama de flujo. $p =$ Número de elementos conectados, normalmente siempre será igual a 1.
NOA	Number of Attributes	$NOA = \sum_{i=1}^{n} A_d(C_i)$	A_d = Atributos definidos dentro de una clase C_i.
FP	Function Points	$FP = FC \times VAF$	FC = Número de cada tipo de componente en la aplicación por la fórmula de función cuenta. VAF = Factor de ajuste de valor.
NR	Total Requirements	$N_t = N_f + N_{nf}$	N_f = Número de requisitos funcionales. N_{nf} = Número de requisitos no funcionales.
SRR	Specification Requirements Ratio	$SRR = \frac{N_u}{N_t}$	N_u = Número de requisitos no ambiguos. N_t = Número de requisitos totales.
CR	Completed Requirements	$CR = \frac{N_c}{N_c + Nv}$	N_c = Número de requisitos completados. N_v = Número de requisitos no validados.

IMS	Software Madurity Index	$IMS = \frac{[M_r - (F_a + F_c + F_d)]}{M_r}$	M_r= Número de módulos en la versión actual F_c = Número de módulos en la versión actual que se han cambiado F_a = Número de módulos en la versión actual que se han añadido F_d = Número de módulos de la versión anterior que se han borrado en la versión actual
FSS	Functional specification stability	$FSS = 1 - \frac{NFSC}{NFSR}$	***NFSC*** = Número de especificación de función modificada. ***NFSR*** = Número de especificación de función requerida,
FCI	Functional correct implementation	$NFICI\ FCI = 1 - NFSR$	***NFICI*** = Número de funciones implementadas incorrectamente. ***NFSR*** = Número de especificación de función requerida.
AAL	Average accuracy level	$NACUs\ AAL = TNCUs$	***NACU*** = Número de cálculos precisos por parte de los usuarios. ***TNCU*** = Número total de cálculos por parte de los usuarios.
IADn	Inaccuracy density	$NIAC\ IADn = OT$	***NIAC*** = Número de casos de inexactitud registrados un período de tiempo observado. ***OT*** = Tiempo de observación.
FR	Full Reliability	$FR = \frac{NYSerH - NYFH}{NYSerH}$	***NYFH*** = Número de horas por año cuando falla al menos una función (incluida la falla total del sistema de software).
VitR	Vital Reliability	$VitR = \frac{NYSerH - NYVitFH}{NYSerH}$	***NYVitFH: =*** Número de horas por año cuando falla al menos una función vital (incluida la falla total del sistema de software). ***TYSerH =*** Número de horas por año que el sistema de software está en servicio.
TUR	Total Unreliability	$TUR = \frac{NYTFH}{TYSerH}$	***NYTFH =*** Número de horas por año de fallas totales del sistema (todas las funciones del sistema fallaron). ***TYSerH =*** Número de horas por año que el sistema de software está en servicio.
EFL	Ease to learn function (hours)	$EFL = MTLUF$	***MTLUF*** = Tiempo medio para aprender a usar una función correctamente (horas).
MECT	Mean error (bug) correction time	$MECT = MTCC - MTSC$	***MTCC*** = Tiempo medio hasta que se complete la corrección por el técnico. ***MTSC***: = Tiempo medio hasta que se inicie la corrección por el técnico.
PERUs	Percentage error recovery by user	$PERU = \frac{NCERUs}{TNEC}$	***NCERU*** = Número de casos de recuperación de errores por parte de los usuarios.

			$TNEC$ = Número total de casos de error.
MC	Message clarity	$MC = \frac{NMCEx}{TNM}$	$NMCEx$ = Número de mensajes con explicación clara. TNM = Número total de mensajes.
MRsT	Mean response time (seconds) Throughput (tasks per hour)	$MRsT\ Throughput = \frac{NTP}{OT}$	NTP = Número de tareas realizadas, OT: tiempo de observación (horas).
ThrComp	Throughput compliance	$ThrComp = \frac{NTPTP}{CTPTP}$	$NTPTP$ = Número de tareas realizadas durante un período de tiempo. $CTPTP$ = Tareas de cumplimiento realizadas durante un período de tiempo.
IOU	I/O utilization	$IOU = \frac{TIODOc}{STIOOp}$	$TIODOc$ = Tiempo ocupado por el dispositivo de E / S. STIOOp = Especificado tiempo para la operación de E / S.
DSCp	Diagnostics support capabilty	$DSCp = \frac{NFMDF}{TNF}$	$NFMDF$ = Número de fallas descubiertas por el uso de la función de diagnóstico. TNF = Número total de fallas.
CSR	Change success ratio	$CSR = 1 - \frac{NCnF}{TNCn}$	$NCnF$ = Número de fallas de cambio. $TNCn$ = Número total de cambios.
MFR	Modification failure density	$MFD = \frac{NMF}{OT}$	NMF = Número de fallas de modificación.
PorComp	Portability compliance	$PorComp = \frac{NCIFPComp}{TNF}$	$NCIFPComp$ = Número de funciones implementadas correctamente en cumplimiento de portabilidad. TNF = Número total de funciones.
PMAc	Perfective maintenance acceptability	$PMAc = \frac{NPMFAc}{TNPMF}$	$NPMFAc$ = Número de funciones de mantenimiento perfecto aceptadas. $TNPMF$ = Número total de funciones de mantenimiento perfecto.
AAcOE	Adaptability acceptance to organization's environment	$AAcOE = 1 - \frac{NFAFail}{TNFATest}$	$NFAFail$ = Número de funciones que fallaron las pruebas de adaptabilidad. $TNFATest$ = Número total de funciones que se sometieron a pruebas de adaptabilidad.
TskEfc	Task effectiveness	$TskEfc = 1 - PTMICCom$	PTMICCom = Proporción de tareas faltantes y componentes incorrectos,
TSCR	Tasks successfully completed ratio	$TSCR = \frac{NTSC}{TNTAt}$	$NTSC$ = Número de tareas completadas con éxito. $TNTAt$ = Número total de tareas intentadas.
AAcOE	Adaptability acceptance to organization's environment	$AAcOE = 1 - \frac{NFAFail}{TNFATest}$	$NFAFail$ = Número de funciones que fallaron las pruebas de adaptabilidad. $TNFATest$ = Número total de funciones que se sometieron a pruebas de adaptabilidad.
PP	Productivity proportion	$PP = \frac{MTPT}{MTT}$	$MNPT$ = Tiempo productivo medio de la tarea.

			MTT: = Tiempo medio de la tarea, donde MTT = MNPT + tiempo de espera + tiempo de error + tiempo de búsqueda.
PC	Productivity change	$TSCR = \frac{MTPTP2 - MTPTP1}{MTPTP1}$	***MTPTP2 =*** Tiempo medio de rendimiento de la tarea 2. ***MTPTP1: =*** Tiempo medio de rendimiento de la tarea 1.
USafL	User safety level	$USafL = \frac{NInjU}{1 - TNU}$	***NInjU***= Número de usuarios lesionados. ***TNU*** = Número total de usos.
SWDL	Software corruption level	$SWDL = \frac{NOcSWC}{TNUSit}$	***NOcSWC =***Número de casos de corrupción de software. ***TNUSit =*** Número total de situaciones de uso.
USatL	User satisfaction level	$ASatL = \frac{SURSQ}{TNRSQ}$	***SURSQ =*** Suma de respuestas del usuario al cuestionario de satisfacción. ***TNRSQ =*** Número total de respuestas del usuario al cuestionario de satisfacción.
USatC	User satisfaction change	$ASatL = \frac{USatLP2 - USatLP1}{USatLP1}$	***USatLP2 =***Nivel de satisfacción del usuario período 2. ***USatLP1 =***Nivel de satisfacción del usuario período 1.

Tabla 6. Lista de métricas de calidad basados en la orientación a objetos y la ISO/IEC 9126-2014.
Elaboración: Autor
Fuentes: [18][19] [20][21][23][28]–[36]