



**EDUCACIÓN**  
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO  
NACIONAL DE MÉXICO

# Tecnológico Nacional de México

**Centro Nacional de Investigación  
y Desarrollo Tecnológico**

## **Tesis de Maestría**

**“Estudio de Métricas y Patrones de Seguridad en  
Microservicios”**

presentada por

**Ing. Juana Victoria Penagos Sánchez**

como requisito para la obtención de grado de  
**Maestra en Ciencias de la Computación**

Director de Tesis  
**Dr. Juan Carlos Rojas Pérez**

Cuernavaca, Morelos, México. Enero de 2023.

Cuernavaca, Morelos **26/octubre/2022**

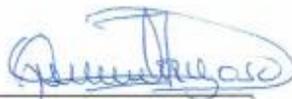
OFICIO No. DCC/086/2022  
Asunto: Aceptación de documento de tesis  
CENIDET-AC-004-M14-OFCIO

**DR. CARLOS MANUEL ASTORGA ZARAGOZA**  
SUBDIRECTOR ACADÉMICO  
PRESENTE

Por este conducto, los integrantes de Comité Tutorial de la C. Juana Victoria Penagos Sánchez, con número de control M20CE068, de la Maestría en Ciencias de la Computación, le informamos que hemos revisado el trabajo de tesis de grado titulado "ESTUDIO DE MÉTRICAS Y PATRONES DE SEGURIDAD EN MICROSERVICIOS"

y hemos encontrado que se han atendido todas las observaciones que se le indicaron, por lo que hemos acordado aceptar el documento de tesis y le solicitamos la autorización de impresión definitiva.

  
\_\_\_\_\_  
**DR. JUAN CARLOS ROJAS PÉREZ**  
Director de tesis

  
\_\_\_\_\_  
**DRA. OLIVIA GRACIELA FRAGOSO DÍAZ**  
Revisor 1

  
\_\_\_\_\_  
**M.C. MARIO GUILLÉN RODRÍGUEZ**  
Revisor 2



Interior Internado Palmira S/N, Col. Palmira, C. P. 62430, Cuernavaca, Morelos  
Tel. 01 (777) 3627770, ext. 3201, e-mail: dcc@tecnm.mx | cenidet.tecnm.mx



Cuernavaca, Mor., **18/noviembre/2022**  
No. De Oficio: **SAC/166/2022**  
Asunto: **Autorización de impresión de tesis**

**JUANA VICTORIA PENAGOS SÁNCHEZ**  
**CANDIDATO(A) AL GRADO DE MAESTRO(A) EN CIENCIAS**  
**EN INGENIERÍA MECÁNICA**  
**PRESENTE**

Por este conducto, tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado "ESTUDIO DE MÉTRICAS Y PATRONES DE SEGURIDAD EN MICROSERVICIOS", ha informado a esta Subdirección Académica, que están de acuerdo con el trabajo presentado. Por lo anterior, se le autoriza a que proceda con la impresión definitiva de su trabajo de tesis.

Esperando que el logro del mismo sea acorde con sus aspiraciones profesionales, reciba un cordial saludo.

**ATENTAMENTE**  
Excelencia en Educación Tecnológica®  
"Educación Tecnológica al Servicio de México"

**DR. CARLOS MANUEL ASTORGA ZARAGOZA**  
**SUBDIRECTOR ACADÉMICO**

C. c. p. Departamento de Ingeniería Mecánica  
Departamento de Servicios Escolares

CMAZ/CHG



## *Dedicatoria*

*“Como no voy a vencer si nací para ganar”  
Anónimo.*

### *Esta tesis está dedicada:*

*A Mis padres Dolores y Rafael, quienes con su amor, paciencia y esfuerzo me han permitido llegar a cumplir hoy un sueño más, proporcionándome las herramientas necesarias en cada paso de este largo camino, gracias por inculcar en mí el ejemplo de esfuerzo y valentía, de no temer a las adversidades porque Dios está siempre conmigo. Ellos son sin duda la parte más importante tanto de mi vida como de mi carrera, porque sin su apoyo incondicional no sería quien soy ahora. Supieron darme sabios consejos, corrigiéndome cuando fue necesario, escuchándome siempre y amándome cuando más lo necesite. Aunque mi papá no se encuentra presente físicamente, sé que está siempre a mi lado; estoy muy segura de que está orgulloso de mí, que desde el cielo puede ver este momento y desde ahí me está cuidando.*

*A mis hermanas María Guadalupe y Mónica, a mis sobrinos Jesús Alfonso, José Ángel, Aura Ximena, Yael Uriel y Fernando Emmanuel, gracias a todos ellos, por sus consejos, por sus palabras de aliento, brindándome siempre su afecto y comprensión, ahora sé que verán en mí un ejemplo de perseverancia y constancia para que así puedan lograr lo que se propongan, alentándose para dar lo mejor de sí mismos, pudiendo saborear sus propios logros. A toda mi familia, porque con sus oraciones y los múltiples consejos hicieron de mí una mejor persona, una gran mujer, sé que de una u otra forma me acompañan en todos mis sueños y metas.*

*A mis amigos que me han estado apoyando en cada momento, en especial a mi mamá postiza Verito Jardines que siempre contado con sus consejos y apoyo, agradezco mucho a la familia de Vi-Ana y Jesús por el apoyo brindado en los últimos meses, mil gracias.*

*Finalmente, quiero dedicar mi tesis a mi amigo Oswaldo Vargas Martínez, por apoyarme cuando más lo necesite, por extender su mano en los momentos más difíciles, por su amistad, paciencia, cariño y comprensión brindados cada día.*

*“A todos ellos de verdad mil gracias, siempre los llevo en el corazón.”*

## ***Agradecimiento***

*Mi profundo agradecimiento al CONACYT por proporcionarme el apoyo necesario para poder alcanzar una de mis metas y poder culminar mis estudios. Así mismo expreso mi agradecimiento al Tecnológico Nacional de México, Campus CENIDET (Centro Nacional de Investigación y Desarrollo Tecnológico), a la Directora, Dra. Yesica Imelda Saavedra, por confiar en mí y permitirme realizar todo el proceso investigativo dentro de esta institución, así mismo a toda la comunidad de este centro de estudios, a mis profesores en especial a la Dra. Blanca Dina Valenzuela Robles, Dra. Olivia Graciela Fragoso Díaz, Lic. Patricia Armas León, Dr. Moisés González, Dr. René Santaolaya Salgado y al Mtro. Mario Guillen Rodríguez, quienes con sus valiosos conocimientos me brindaron la oportunidad de crecer día a día como profesional, por su paciencia, dedicación, apoyo incondicional y amistad, gracias a cada uno.*

*Quiero expresar mi más grande y sincero agradecimiento al Dr. Juan Carlos Rojas Pérez, quien con su dirección, conocimiento, enseñanza y colaboración permitió el desarrollo de este trabajo, motivándome cada momento durante todo este proceso.*

*Finalmente, agradezco infinitamente al Lic. Oswaldo Vargas Martínez, por brindarme su valioso tiempo, quien, con su conocimiento, apoyo y experiencia en cada etapa de este proceso.*

## *Resumen*

Los microservicios son una tecnología relativamente nueva para diseñar aplicaciones de software como conjuntos de servicios escalables, modulares y centrados en el cliente, que se pueden desplegar de forma independiente. Sin embargo, al ser una tecnología reciente también presenta algunos problemas como la seguridad, área de investigación en la que se han propuesto algunas soluciones, aunque no acordes con la arquitectura de microservicios. En este trabajo se presenta un estudio de mapeo sistemático, cuyo objetivo se centra en evidenciar el estado actual del problema de la seguridad en microservicios. Los mapeos sistemáticos MS consideraron las fechas entre 2011 y 2021 donde se identificaron 51 artículos relevantes, de los cuales se eligieron 19 artículos con base a criterios de clasificación, de los cuales 9 se emplearon para métricas de seguridad en microservicios y 10 para patrones de seguridad en microservicios, se observó que las métricas y los patrones seguridad para microservicios son aún temas nuevos que están en desarrollo, se propone utilizar métricas y patrones derivados de otras arquitecturas (monolíticas, SOA, DeOps), los cuales se han modificado para ser empleados en los microservicios no en todos los casos se aplican por sí mismos algunos de los ejemplos utilizan atributos de calidad en casos de pruebas. De los dos MS se analizaron y clasificaron métricas y patrones de acuerdo a la información obtenida, así como un ejemplo del funcionamiento de un microservicio, así como la propuesta de una guía que permita medir la funcionalidad de un microservicio enfocado a la seguridad.

## *Abstract*

Microservices are a relatively new technology for designing software applications as sets of scalable, modular, client-centric services that can be deployed independently. However, being a recent technology, it also presents some problems such as security, an area of research in which some solutions have been proposed, although not in accordance with the microservices architecture. In this paper we present a systematic mapping study, whose objective is focused on evidencing the current state of the problem of security in microservices. The MS systematic mappings considered the dates between 2011 and 2021 where 51 relevant articles were identified, from which 19 articles were chosen based on classification criteria, of which 9 were used for microservices security metrics and 10 for microservices security patterns, It was observed that security metrics and patterns for microservices are still new topics under development, it is proposed to use metrics and patterns derived from other architectures (monolithic, SOA, DeOps), which have been modified to be used in microservices, but not in all cases they apply by themselves, some of the examples use quality attributes in test cases. From the two MS, metrics and patterns were analyzed and classified according to the information obtained, as well as an example of the operation of a microservice, as well as the proposal of a guide that allows measuring the functionality of a microservice focused on security.

# Contenido

<b>Lista de Figuras</b> .....	xi
<b>Lista de Ilustraciones</b> .....	xii
<b>Lista de Tablas</b> .....	xii
<b>Capítulo 1.</b> Introducción.....	1
1.1. Antecedentes. ....	2
1.2. Definición del problema.....	2
1.3. Objetivos. ....	2
1.3.1. Objetivo general. ....	2
1.3.2. Objetivos específicos.....	3
1.4. Justificación.....	3
1.5. Alcances y limitaciones.....	3
1.5.1. Alcances. ....	3
1.5.2. Limitaciones. ....	3
<b>Capítulo 2.</b> Marco Conceptual .....	4
2.1. Definiciones .....	4
2.2. Microservicios.....	4
2.3. Métricas de seguridad.....	4
2.3.1. Entidad .....	5
2.3.2. Atributos.....	5
2.3.3. Medición .....	5
2.3.4. Medida.....	5
2.3.5. Escala .....	5
2.3.6. Mecanismos de Seguridad.....	5
2.3.7. Seguridad Informática .....	6
2.3.8. Autenticación .....	6
2.3.9. Arquitectura de Microservicios.....	6
2.3.10. Modelo Vista Controlador (MVC).....	6
2.3.10.1. Modelo .....	6
2.3.10.2. Vista O interfaz de usuario.....	6
2.3.10.3. Controlador.....	6
2.3.11. Servicio Web.....	7
2.3.12. API .....	7

2.3.13.	SOA.....	7
2.3.14.	DevOps.....	7
2.4.	Mapeo Sistemático SM (Systematic Mapping).....	8
2.5.	Metodología .....	8
2.6.	Extracción y Síntesis de Datos .....	8
<b>Capítulo 3. MS Métricas para Microservicios .....</b>		<b>10</b>
3.1.	Mapeo Sistemático para Métricas de Seguridad en Microservicios.....	11
3.2.	Formulación de Preguntas para Métricas de Seguridad en Microservicios.....	11
3.3.	Selección de Fuentes .....	11
3.4.	Realización de Búsquedas y Palabras Clave sobre Métricas de seguridad .....	12
3.5.	Cadenas de Búsqueda para Métricas de Seguridad en Microservicios .....	12
3.6.	Criterios de Inclusión y Exclusión Métricas .....	13
3.7.	Extracción y Síntesis de Datos .....	13
3.8.	Resultados de MS de Métricas para seguridad en microservicios .....	14
3.9.	Análisis de métricas para microservicios .....	16
3.10.	Análisis de las Preguntas de Investigación Métricas en Microservicios.....	17
3.10.2.	Taxonomías en Métricas Aplicadas a Sistemas Software. ....	18
3.10.3.	Análisis de métricas aplicadas a diversas arquitecturas .....	19
3.10.4.	Resumen de Trabajos relacionados de Métricas de seguridad para Microservicios .....	20
<b>Capítulo 4. MS Patrones para Microservicios.....</b>		<b>22</b>
4.1.	Formulación de Preguntas para Patrones en Microservicios.....	22
4.2.	Selección de Fuentes .....	22
4.3.	Realización de Búsquedas y Palabras Clave Patrones .....	23
4.4.	Cadenas de Búsqueda para Patrones de seguridad en microservicios.....	23
4.5.	Criterios de Inclusión y Exclusión Patrones.....	23
4.6.	Extracción y Síntesis de Datos .....	24
4.7.	Resultados de MS de Patrones para seguridad en microservicios.....	24
4.8.	Análisis de las preguntas de investigación patrones de seguridad en microservicios .....	28
4.9.	Resumen de Trabajos Relacionados a Patrones en Microservicios.....	28
<b>Capítulo 5. Análisis de los MS.....</b>		<b>30</b>
5.1.	Clasificación de métricas para seguridad en diversas arquitecturas.....	30
5.2.	Clasificación de Métricas Aplicadas a Microservicios .....	31
5.3.	Patrones de seguridad aplicables a diferentes arquitecturas.....	32

5.4.	Análisis de Métricas y Patrones para aplicación a Microservicios .....	34
<b>Capítulo 6.</b>	<b>Guía de referencia para microservicios .....</b>	<b>36</b>
6.1.	Objetivo.....	36
6.2.	Actividades.....	36
6.3.	Introducción .....	36
6.4.	Características comunes de los Microservicios.....	37
6.5.	HTTP REST .....	37
6.6.	Ventajas y Desventajas de los microservicios.....	38
6.7.	El ciclo de vida del desarrollo de software.....	38
6.7.1.	Planificación.....	38
6.7.2.	Análisis.....	39
6.7.3.	Requisitos .....	39
6.7.3.1.	Tipos de requisitos .....	39
6.7.4.	Diseño .....	39
6.7.5.	Implementación.....	40
6.7.6.	Pruebas .....	41
6.7.7.	Revisiones .....	41
6.7.8.	Instalación o despliegue .....	41
6.7.9.	Uso y Mantenimiento.....	42
6.7.10.	Nivel de soporte .....	42
6.7.11.	Cuidado y Administración .....	42
6.8.	Factor de Calidad Norma ISO/IEC 9126: 2001 .....	42
6.8.1.	Características de Calidad de un software según la ISO/IEC 9126: 2001.....	43
6.8.2.	Funcionalidad.....	43
6.8.3.	Diagrama de Norma ISO 9126.....	43
6.8.4.	Primer Paso .....	45
6.8.5.	Segundo paso Valores de funcionalidad de microservicios .....	50
<b>Capítulo 7.</b>	<b>Conclusión.....</b>	<b>53</b>
7.1.	Conclusión.....	53
7.2.	Referencias .....	55
<b>Capítulo 8.</b>	<b>Anexos.....</b>	<b>59</b>
8.1.	Ejemplo de Microservicio .....	59
a.	Introducción .....	59

b.	Objetivo.....	59
c.	Descripción general.....	59
d.	Limitaciones.....	60
e.	Estructura del ejemplo.....	60
f.	Interfaz de usuario.....	60
g.	Diagrama de caso de usos.....	61
h.	Modelo de negocios BPMN.....	63
i.	Diagrama de Secuencias.....	63
j.	Diagrama Despliegue.....	64
k.	Diagrama ER de Agencia de Viajes ACME.....	64
ñ.	Sitio web Viajes ACME.....	65
l.	Herramientas de desarrollo.....	70
m.	Wapserver.....	70
n.	Boostrap.....	70
o.	Css.....	70
p.	Html.....	71
q.	Php.....	71
8.2.	Ejemplo de un microservicio con Python.....	71
8.3.	Resumen.....	73

## Lista de Figuras

<i>Figura 1. Proceso del mapeo sistemático (Petersen Kai, 2008)</i> .....	11
<i>Figura 2. Proceso General de Selección</i> .....	15
<i>Figura 3. Distribución de estudios por fuentes de información.</i> .....	15
<i>Figura 4. Nube de palabras clave de Estudios Seleccionados.</i> .....	16
<i>Figura 5. Artículos que responden las preguntas de Investigación.</i> .....	17
<i>Figura 6. Resultados de búsquedas por fuente digital</i> .....	25
<i>Figura 7. Proceso General de Selección.</i> .....	25
<i>Figura 8. Distribución de artículos por año de publicación (Tabla9).</i> .....	26
<i>Figura 9. Nube de palabras clave de Patrones de Seguridad.</i> .....	27
<i>Figura 10. Artículos relacionados a patrones en microservicios.</i> .....	27
<i>Figura 11. Beneficio de la arquitectura de los microservicios, Roldan Martínez David, P. J. (2018).</i> .....	37
<i>Figura 12. Norma ISO 9126 (Verity, 2021).</i> .....	44
<i>Figura 13. Microservicios y ciclo de vida de desarrollo</i> .....	49
<i>Figura 14. Diagrama de valores de funcionalidad de microservicios</i> .....	50
<i>Figura 15. Pantalla de inicio del sitio web.</i> .....	60
<i>Figura 16. Diseño de destino vuelos</i> .....	61
<i>Figura 17. Diagrama de clases de selección de destinos</i> .....	61
<i>Figura 18. Caso de uso general del sitio web</i> .....	62

Figura 19. Diagrama BPMN de destinos de vuelos	63
Figura 20. Diagrama de secuencia de vuelos	64
Figura 21. Diagrama Despliegue de opción de vuelos	64
Figura 22. Estructura de Base de Datos	65
Figura 23. página principal de agencia de viajes	65
Figura 24. Página de registro de usuarios	66
Figura 25. Página de registro de datos	66
Figura 26. Reservación de vuelos	67
Figura 27. Alta de vuelos	67
Figura 28. Página principal de alojamiento del sistema	68
Figura 29. Entorno de desarrollo	68
Figura 30. Herramientas de Desarrollo	72
Figura 31. Estructura WSGI	72
Figura 32. Estructura del microservicio simple	73
Figura 33. Terminal Python	73
Figura 34. Ejecución del servicio en el navegador	73

## Lista de Ilustraciones

Ilustración 1. Elementos de la planificación	45
Ilustración 2. Requisitos funcionales y no funcionales	46
Ilustración 3. Estructuras, técnicas, algoritmos	46
Ilustración 4. Tipos de lenguajes de programación	46
Ilustración 5. Pruebas de funcionalidad	47
Ilustración 6. Revisión por caso de uso	47
Ilustración 7. Instalación de software	47
Ilustración 8. Mantenimiento de software	48
Ilustración 9. Administración del software	48

## Lista de Tablas

Tabla 1. Tipos de enfoque de clasificación de Investigación (Wieringa Roel, 2006)	9
Tabla 2. Fuentes de Información	12
Tabla 3. Criterios Inclusión y Exclusión de datos	13
Tabla 4. Publicaciones Seleccionadas para Métricas en Microservicios	16
Tabla 5. Análisis de Taxonomías de Seguridad en diferentes arquitecturas	19
Tabla 6. Análisis de trabajos sobre métricas de seguridad en diferentes arquitecturas	20
Tabla 7. Fuentes de Información	22
Tabla 8. Criterios Inclusión y Exclusión de datos	23
Tabla 9. Publicaciones seleccionadas	26
Tabla 10. Métricas para Diversas Arquitecturas	31
Tabla 11. Clasificación de Mecanismos de Seguridad	32
Tabla 12. Clasificación de patrones con atributos de calidad	33
Tabla 13. Mecanismos de Patrones para Microservicios	34
Tabla 14. Valores de Funcionalidad para medir los Microservicios	51
Tabla 15. Check List-Formato de medición de Microservicios	52

# Capítulo 1.

# Introducción

*“El término "Arquitectura de microservicio" ha surgido en los últimos años para describir una forma particular de diseñar aplicaciones de software como conjuntos de servicios desplegados de forma independiente. Si bien no existe una definición precisa de este estilo arquitectónico, existen ciertas características comunes en torno a la organización, la capacidad empresarial, la implementación automatizada, la inteligencia en los puntos finales y el control descentralizado de idiomas y datos”. (Martin, 2014).*

Los microservicios son la nueva tecnología en el desarrollo de aplicaciones, el término microservicio ha reemplazado a DevOps y RESTfull como la nueva palabra de moda. En los últimos 10 años y gracias entre otros factores a IOT (Internet de las cosas), los paradigmas de programación han sufrido, más que una evolución, una revolución que ha llevado a las arquitecturas basada en microservicios (Roldan Martínez David, 2018) . Se prefiere el uso de contenedores para implementar microservicios debido a su simplicidad, menor costo, su rápida inicialización y ejecución dentro de las plataformas, pueden implementarse, replicarse, reemplazarse y destruirse fácilmente de forma independiente sin afectar la disponibilidad de los sistemas. De acuerdo a (Chondamrongkul Nacha & Jing Sun, 2020.) actualmente no existe un enfoque que pueda analizar las vulnerabilidades de seguridad en los microservicios y que hasta ahora proporcionen un resultado eficaz. De acuerdo a (Márquez Gastón,2018) las plataformas de repositorios de código proporcionan a la comunidad de desarrolladores ideas y ejemplos sobre sistemas de microservicios, pero, dado que se encuentran en una fase temprana de adopción, todavía no existe una noción clara de qué sistemas de microservicios reales encarnan patrones arquitectónicos, si es que hay alguno, lo que reduce el uso de marcos y la obtención de atributos de calidad. Los sistemas basados en microservicios son un estilo arquitectónico que concibe los sistemas como conjuntos de servicios modulares, centrados en el cliente, independientes y escalables (Astudillo, 2018). Sin embargo, a pesar de las ventajas que ofrece la adopción de arquitecturas de microservicios al desarrollo de sistemas complejos, la seguridad es uno de los serios desafíos que se deben abordar. Actualmente no hay forma de encontrar vulnerabilidades en microservicios y lograr un resultado efectivo (Chondamrongkul Nacha & Jing Sun, 2020.).En este trabajo presenta un Mapeo Sistemático (MS) en dos aspectos el primero en patrones y el segundo en métricas de seguridad utilizados en los microservicios. El objetivo principal de este MS fue obtener un panorama del estado actual de métricas y patrones de seguridad que pueden aplicarse a los microservicios y realizar un análisis de cómo se están aplicando. El primer MS de patrones de seguridad en microservicios se obtuvieron 51 artículos a través de diferentes fuentes de información, siguiendo como referencia el modelo (Petersen Kai, 2008). Después de un proceso de refinación se obtuvieron 9 artículos que abordan el problema de la seguridad a través de patrones de diseño. De los artículos obtenidos se recopilaron 49 patrones que se aplican en diversas arquitecturas y se han adaptado a los microservicios.

El segundo MS nos proporcionó un total de 3308 artículos, relacionados a métricas, por las diferentes fuentes de información de los cuales se realizó el proceso de refinación de información aplicando “*snowballing*” (Streeton, 2004), el cual permitió trabajar con un total de 10 artículos en relación a métricas de seguridad para microservicios, dentro de este MS se empleó un análisis de manera similar que en el primer mapeo sistemático donde se clasificaron patrones aplicados a la seguridad en microservicios. Cabe señalar que no existe una única definición de microservicio dado que cada autor tiene un criterio diferente en cada trabajo y la aplicación de métricas y patrones va de acuerdo a las experiencias de arquitecturas anteriores. Algunos de los artículos hacen comparaciones con SOA (Service Oriented Architecture), pero no en todos los casos estas soluciones se adaptan por sí mismas, sino que recurren a la utilización de modelos de calidad del software para poder resarcir el problema de seguridad de manera provisional. Esta tesis se llevó a cabo un estudio donde se implementen o adapten las métricas y patrones de seguridad en microservicios, efectuando un mapeo sistemático que permitiera ampliar el panorama actual sobre la seguridad en estos, desarrollando un análisis de los datos obtenidos mediante tablas y diagramas que permitieran comprender mejor dicha información, así mismo se realizó una guía que permite mejorar el desarrollo de los microservicios y su seguridad utilizando un estándar de calidad ISO/IEC 9126-1 basado en el atributo de funcionalidad y sus subatributos en conjunto de la utilización del ciclo de vida del software mediante un check List de atributos medible de acuerdo a información obtenida del estudio.

## 1.1. Antecedentes.

En CENIDET (Centro de Investigación y Desarrollo Tecnológico) no se han desarrollado algún proyecto relacionado con seguridad en microservicios que anteceda al problema planteado en este trabajo de investigación.

## 1.2. Definición del problema.

La seguridad y protección de la información es una cualidad que deben presentar los servicios Web, el problema radica en que cuando no se tiene el conocimiento o experiencia para llevar a cabo esta tarea, el software se vuelve vulnerable, por lo cual se requiere de estudios que puedan dar un indicio de cómo afrontar este problema.

En este sentido, los Microservicios considerados una tecnología relativamente nueva debe afrontar esta propiedad, por lo cual se plantea un estudio de mapeo sistemático para conocer qué mecanismos de seguridad para arquitecturas de microservicios existen, cómo aplicarlos, cómo medirlos.

## 1.3. Objetivos.

### 1.3.1. Objetivo general.

Realizar un estudio del estado actual de métricas y patrones en el área de seguridad para el desarrollo de microservicios.

### 1.3.2. Objetivos específicos.

- a) Definir o evidenciar que métricas son aplicables en mecanismos de seguridad de microservicios.
- b) Definir o evidenciar que patrones de seguridad son aplicables o adaptables en el desarrollo de microservicios.
- c) Elaborar una guía que proporcione al usuario la información necesaria para implementar la seguridad en microservicios.

## 1.4. Justificación.

La presente tesis se desarrolló con la finalidad de contribuir al conocimiento mediante un estudio que demuestre que métricas y patrones de seguridad se pueden asociar a alguno de los medios de seguridad válidos para poder identificar los más adecuados en el problema de seguridad para el desarrollo de microservicios y aporte al estado actual del conocimiento de esta arquitectura la cual sigue evolucionando conforme pasa el tiempo ampliando las brechas permitiendo que en un futuro se puedan definir métricas y patrones en específicos para los microservicios que permitan evaluar su calidad.

## 1.5. Alcances y limitaciones

### 1.5.1. Alcances.

- a) Realizar un mapeo sistemático de la implementación de seguridad en microservicios empleando a métricas.
- b) Realizar un mapeo sistemático de la implementación de la seguridad en microservicios empleando patrones.
- c) Realizar microservicios (servicio web, SOAP, REST) para diferenciar la comunicación entre ellos y cómo afecta la seguridad.

### 1.5.2. Limitaciones.

- a) Se consideraron atributos o métricas y patrones, las más importantes, que sean causantes o deban ser considerados para cuantificar la seguridad en el desarrollo de microservicios.
- b) La implementación de microservicios fue con fines de ejemplo, no se realizaron microservicios complejos.

# Capítulo 2.

## Marco Conceptual

### 2.1. Definiciones

### 2.2. Microservicios

Los sistemas basados en microservicios son un estilo arquitectónico que concibe los sistemas como conjuntos de servicios modulares, centrados en el cliente, independientes y escalables(Astudillo, 2018).

### 2.3. Métricas de seguridad

La ingeniería de software es una disciplina que está aún aprendiendo a medir y estimar para mejorar la calidad de sus productos y procesos, puesto que es uno de los primeros pasos para progresar en las ciencias es tomar medidas e interpretarlas (Rodríguez, 2012).

La teoría de la representación de la medición establece tanto los principios generales de la medición como su validez. Esta teoría trata de expresar de forma numérica (mundo formal) las entidades del mundo real (o mundo empírico) y la correspondencia entre ambos mundos. Las entidades en la ingeniería de software son los procesos, los recursos (personal, oficinas, etc.) y todos los artefactos (código, documentación, etc.) generados durante el ciclo de vida del software (Rodríguez, 2012).

El término “**métricas de seguridad**” se refiere a las medidas relacionadas con la seguridad las describen como el proceso de medir los costos y beneficios de un Programa de Protección de Activos, así como sus éxitos y fracasos (Kovacich y Halibozeck 2006).

**Métrica.** En el campo de la ingeniería del software una **métrica** es un estándar de medida de un grado en el que un sistema o proceso de software posee alguna propiedad. Aunque una métrica no es una medida (las métricas son roles o funciones, mientras que las mediciones son números obtenidos por aplicación de tales métricas), con frecuencia ambos términos son usados como si fueran sinónimos. Ya que las mediciones cuantitativas son esenciales en todas las ciencias, hay un continuo esfuerzo de parte de practicantes de la informática y teóricos para lograr acercamientos similares para el desarrollo de software. La meta es obtener mediciones objetivas, reproducibles y cuantificables, que posibilitan tener valiosas y numerosas aplicaciones, en planificación de calendarios y presupuestos, planificación presupuestaria, aseguramiento de calidad, pruebas, depuración de software, optimización del rendimiento del software y asignaciones óptimas de tareas del personal (Centre, 2011).

### 2.3.1. Entidad

Se denomina **entidad** a un objeto que va a ser caracterizado mediante una medición de sus atributos (Rodríguez, 2012).

### 2.3.2. Atributos

Los atributos son las características de las entidades. Por ejemplo, algunos atributos del código fuente pueden ser líneas de código o su complejidad. Un **atributo** es una característica medible de una entidad (Rodríguez, 2012).

Para el estudio de la medición es importante clarificar la diferencia entre *medición* y *medida* los términos que se utilizan dentro de la ingeniería de software (Rodríguez, 2012).

### 2.3.3. Medición

**Medición** es el proceso por el que se asignan números o símbolos a los atributos de entidades del mundo real para describirlo según unas reglas definidas de antemano (Rodríguez, 2012).

### 2.3.4. Medida

La definición clásica de (Fenton y Pfleeger, 1998 como se citó en Rodriguez, 2012), quienes definen medida así: **Medida** es la asignación de un símbolo o número de resultado de una medición a una entidad para caracterizar un atributo.

### 2.3.5. Escala

Una **escala** de medición es un conjunto de valores que permite establecer relaciones entre medidas. Con frecuencia dicho conjunto es continuo, está ordenado y viene delimitado por un punto inicial y otro final (Rodríguez, 2012).

### 2.3.6. Mecanismos de Seguridad

Un mecanismo de seguridad informática es una técnica o herramienta que se utiliza para fortalecer la confidencialidad, la integridad y/o la disponibilidad de un sistema informático (Rios, 2020).

### 2.3.7. Seguridad Informática

La seguridad informática la define como cualquier medida que impida la ejecución de operaciones no autorizadas sobre un sistema o red informática cuyos efectos puedan conllevar daños sobre la información, equipo o software (Gómez 2006).

### 2.3.8. Autenticación

Es el proceso que debe seguir un usuario para tener acceso a los recursos de un sistema o de una red de computadores. Este proceso implica identificación (decirle al sistema quién es) y autenticación (demostrar que el usuario es quien dice ser). La autenticación por sí sola no verifica derechos de acceso del usuario; estos se confirman en el proceso de autorización (Telecomunicaciones, 2006).

### 2.3.9. Arquitectura de Microservicios

(En inglés, *Micro Services Architecture*, MSA) es una aproximación para el desarrollo de software que consiste en construir una aplicación como un conjunto de pequeños servicios, los cuales se ejecutan en su propio proceso y se comunican con mecanismos ligeros (normalmente una API de recursos HTTP). Cada servicio se encarga de implementar una funcionalidad completa del negocio. Cada servicio es desplegado de forma independiente y puede estar programado en distintos lenguajes y usar diferentes tecnologías de almacenamiento de datos (Martin, 2014).

### 2.3.10. Modelo Vista Controlador (MVC)

Es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Se trata de un modelo muy maduro y que ha demostrado su validez a lo largo de los años en todo tipo de aplicaciones, y sobre multitud de lenguajes y plataformas de desarrollo (Alicante, 2022).

#### 2.3.10.1. Modelo

Un modelo es una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia (Alicante, 2022).

#### 2.3.10.2. Vista O interfaz de usuario

La vista o interfaz de usuario que compone la información que se envía al cliente y los mecanismos que interaccionan con éste (Alicante, 2022).

#### 2.3.10.3. Controlador

El controlador es el que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno (Alicante, 2022).

### 2.3.11. Servicio Web.

Un servicio web (en inglés, Web service o Web services) es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos. Las organizaciones OASIS y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web (w3c, 2004).

### 2.3.12. API

Una API es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones. API significa interfaz de programación de aplicaciones. Las API's permiten que sus productos y servicios se comuniquen con otros, sin necesidad de saber cómo están implementados. Esto simplifica el desarrollo de las aplicaciones y permite ahorrar tiempo y dinero. Las API le otorgan flexibilidad; simplifican el diseño, la administración y el uso de las aplicaciones, y proporcionan oportunidades de innovación, lo cual es ideal al momento de diseñar herramientas y productos nuevos (o de gestionar los actuales). A veces, las API se consideran como contratos, con documentación que representa un acuerdo entre las partes: si una de las partes envía una solicitud remota con cierta estructura en particular, esa misma estructura determinará cómo responderá el software de la otra parte (Hat, 2017).

### 2.3.13. SOA

La arquitectura orientada a los servicios (SOA) es un tipo de diseño de software que permite reutilizar sus elementos gracias a las interfaces de servicios que se comunican a través de una red con un lenguaje común (IBM, 2017).

### 2.3.14. DevOps

Es una combinación de los términos en inglés development (desarrollo) y operations (operaciones), designa la unión de personas, procesos y tecnología para ofrecer valor a los clientes de forma constante. DevOps permite que los roles que antes estaban aislados (desarrollo, operaciones de TI, ingeniería de la calidad y seguridad) se coordinen y colaboren para producir productos mejores y más confiables. Al adoptar una cultura de DevOps junto con prácticas y herramientas de DevOps, los equipos adquieren la capacidad de responder mejor a las necesidades de los clientes, aumentar la confianza en las aplicaciones que crean y alcanzar los objetivos empresariales en menos tiempo (Azure, s.f.).

## 2.4. Mapeo Sistemático SM (Systematic Mapping)

Los estudios de mapeo sistemático o estudios de alcance están diseñados para brindar una descripción general de un área de investigación a través de la clasificación y el conteo de contribuciones en relación con las categorías de esa clasificación. Implica buscar en la literatura para saber qué temas se han cubierto en la literatura y dónde se ha publicado la literatura. Un mapeo sistemático emplea los mismos métodos que una revisión sistemática, pero suele ser un estudio más sencillo de realizar ya que no tiene como objetivo analizar los resultados después de los artículos que conforman el mapeo. Los estudios de mapeo también son revisiones, pero no discuten los hallazgos, se basan en el concepto de que los artículos publicados no solo representan hallazgos, sino que, indirectamente, representan actividades relacionadas con el mismo (Petersen Kai, 2008). A medida que un área de investigación madura, suele aumentar el número de informes y resultados disponibles, por lo que es importante resumirlos y ofrecer un panorama general. Muchos campos de investigación cuentan con metodologías específicas para este tipo de estudios secundarios, y se han utilizado ampliamente como, por ejemplo, en la medicina basada en la evidencia se ha utilizado mapeos sistemáticos ampliamente. Hasta hace poco, no era el caso de la ingeniería del software, sin embargo, existe una tendencia general hacia una ingeniería del software más basada en la evidencia (Kitchenham et al. 2004) ha llevado a centrarse más en métodos de investigación nuevos, empíricos y sistemáticos. Estudios de mapeos sistemáticos en ingeniería de software requieren considerable esfuerzo, así como en las revisiones sistemáticas estas se han centrado en estudios cuantitativos y empíricos (Dixon-Woods et al. 2005). Un estudio de mapeo sistemático proporciona una estructura del tipo de informes de investigación y resultados que se han publicado, clasificándolos suele ofrecer un resumen visual del mapa de sus resultados, requiere menos esfuerzo a la vez que proporciona una visión general, anteriormente los estudios sistemáticos en la ingeniería del software se han recomendado sobre todo para las áreas de investigación en las que se carece de estudios primarios relevantes (Kitchenham, 2007).

## 2.5. Metodología

Una metodología es una revisión sistemática de literatura de investigación que se desarrolla para obtener y evaluar la evidencia disponible que sea pertinente sobre un tema específico y se usa en diversas disciplinas (PearlBrereton, 2007).

## 2.6. Extracción y Síntesis de Datos

Para (Paulo Anselmo da Mota Silveira Neto, 2010), sugiere la exploración de los documentos y realizar la lectura completa de los artículos, así como responder las preguntas planteadas, esto debido a que en muchas ocasiones el título y el resumen no indican nada relevante, por ende, no responde a las interrogantes. Para ello se emplea el enfoque que categoriza cinco

tipos de investigación y es aplicado para ambos MS como se muestra en la Tabla 1 (Wieringa Roel, 2006).

*Tabla 1. Tipos de enfoque de clasificación de Investigación (Wieringa Roel, 2006).*

Categoría	Descripción
<b>Investigación Exploratoria</b>	Son estudios que suelen ser el primer acercamiento científico a un problema. Se utiliza cuando éste aún no ha sido abordado o no ha sido suficientemente estudiado y las condiciones existentes no son aún determinantes.
<b>Investigación Explicativa</b>	Con este tipo de investigación es posible encontrar la relación existente entre la causa y consecuencia de un problema. De esta forma es posible conocer el porqué de este y cómo ha llegado a su estado actual.
<b>Investigación Aplicativa</b>	La investigación trata de resolver un determinado problema o planteamiento específico, enfocándose en la búsqueda y consolidación del conocimiento para su aplicación con un grupo de herramientas computacionales existentes para un determinado fin
<b>Artículos Teóricos</b>	Estos artículos tienen como objetivo la obtención de conocimiento sin importar su posterior adaptación del modelo al que sea aplicado.
<b>Artículos de Revisión</b>	Consiste en un grupo de trabajos elaborados a partir de artículos originales previamente publicados. A partir de un tema que el autor seleccionó para investigar, buscar, identificar, recopilar y revisar los trabajos más recientes.

# Capítulo 3. MS

## Métricas para

### Microservicios

*Los estudios de mapeo sistemático se utilizan para estructurar un área de investigación, mientras que las revisiones sistemáticas se centran en recopilar y sintetizar evidencia. Las pautas más recientes para el mapeo sistemático son de 2008. Desde entonces, se han hecho muchas sugerencias sobre cómo mejorar las revisiones sistemáticas de literatura (SLR). Existe la necesidad de evaluar cómo los investigadores llevan a cabo el proceso de mapeo sistemático e identificar cómo se deben actualizar las pautas en función de las lecciones aprendidas de los mapas sistemáticos existentes y las pautas SLR. (Petersen Kai, 2008).*

En esta sección se presentan el proceso para el desarrollo de un Mapeo Sistemático (MS) para métricas y patrones de seguridad basándose en el modelo de (Petersen Kai, 2008). A continuación, se detalla el proceso que se utilizó, así como trabajos relacionados de ambos casos y los resultados de cada MS de los trabajos obtenidos relacionadas a la seguridad de microservicios, así como el desarrollo de análisis de la investigación.

Un estudio de mapeo sistemático incluye los siguientes pasos: a) una definición de preguntas de investigación, b) búsqueda de artículos relevantes, c) selección de artículos encontrados, d) proponer o utilizar un esquema de clasificación existente, e) extracción de datos y mapeo de estudios (Petersen Kai, 2008). Los pasos esenciales del proceso del estudio realizado se listan a continuación (ver Figura 1):

- Formulación de preguntas de investigación.
- Selección de fuentes.
  - ✚ Realizar búsquedas.
  - ✚ Palabras claves usando resúmenes.
  - ✚ Cadenas de búsquedas.
  - ✚ Criterios de inclusión y exclusión.
  - ✚ Extracción y Síntesis de Datos.
  - ✚ Síntesis de datos.
- Resultado del mapeo.
  - ✚ Resumen de estudios seleccionados.

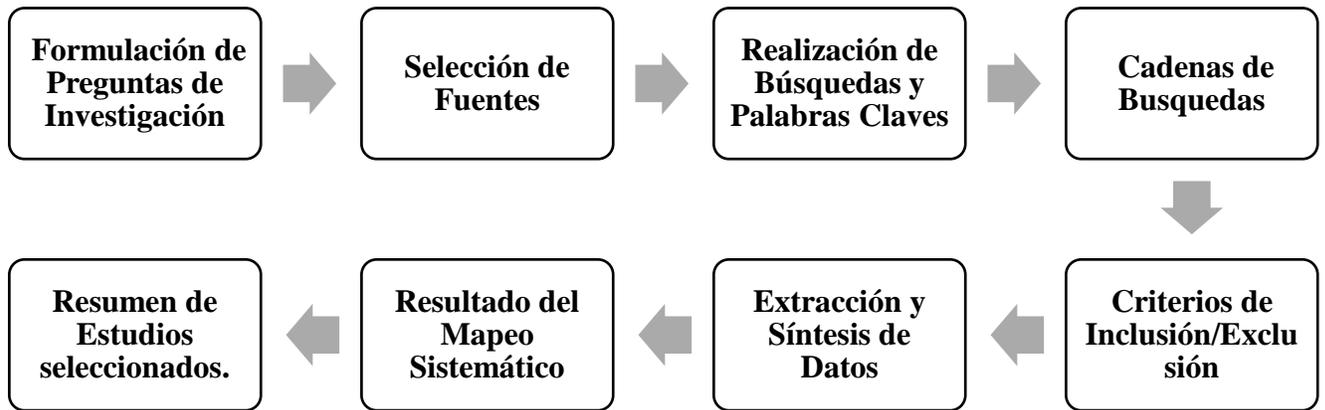


Figura 1. Proceso del mapeo sistemático (Petersen Kai, 2008)

### 3.1. Mapeo Sistemático para Métricas de Seguridad en Microservicios

El objetivo principal de este MS es mostrar el estado actual de métricas en el área de seguridad para el desarrollo de microservicios. Tomando como referencia el proceso del mapeo sistemático antes mencionado en la Figura 1 a continuación, se muestran las etapas para el desarrollo de ambos MS.

### 3.2. Formulación de Preguntas para Métricas de Seguridad en Microservicios

El objetivo de estas tres preguntas de investigación es poder conocer la existencia de estudios relacionados a métricas de seguridad que se apliquen a los microservicios y conocer cómo se están aplicando a los microservicios si existen para estos.

**RQ1.** ¿Existe algún estudio de mapeo sistemático de seguridad en microservicios?

**RQ2.** ¿Qué métricas de seguridad se aplican a los contenedores para microservicios?

**RQ3.** ¿Cuáles son las métricas de seguridad implementadas en el desarrollo de microservicios?

### 3.3. Selección de Fuentes

Las fuentes de información digitales en línea consultadas para este MS se muestran en la Tabla2.

Tabla 2. Fuentes de Información.

Fuente	Dirección Url
IEEEExplore	<a href="https://www.ieee.org/">https://www.ieee.org/</a>
Elsevier	<a href="http://www.elsevier.com">www.elsevier.com</a>
EBSCO host	<a href="https://www.ebsco.com/es">https://www.ebsco.com/es</a>
ACM Digital	<a href="https://dl.acm.org/">https://dl.acm.org/</a>
Google Scholar	<a href="https://scholar.google.com/">https://scholar.google.com/</a>

### 3.4. Realización de Búsquedas y Palabras Clave sobre Métricas de seguridad

Las cadenas de búsquedas fueron diseñadas de acuerdo con las preguntas de investigación previamente mencionadas, utilizando palabras clave. De acuerdo al objetivo de este MS y a las preguntas de investigación se obtuvieron las siguientes palabras:

- Seguridad.
- Microservicios.
- Mapeo sistemático.
- Métricas.
- Contenedor.
- Taxonomía.
- Estudio.
- Arquitectura.

### 3.5. Cadenas de Búsqueda para Métricas de Seguridad en Microservicios

Se efectuó una búsqueda con respecto a MS orientados a métricas para seguridad en microservicios, la cual no arrojó resultados específicos. A continuación, se muestran las cadenas generales de búsqueda en lo que se refiere a las preguntas de investigación de acuerdo a cada fuente digital:

**Cadena de Búsqueda 1.** Systematic security mapping study for microservices **or** security study for microservices **or** systematic mapping of security metrics for microservices.

**Cadena de Búsqueda 2.** Container security metrics for microservices o security metrics applied to containers for microservices **or** study of metrics applied to containers for microservices **or** taxonomy of container security metrics for microservices.

**Cadena de Búsqueda 3.** Security metrics implemented in microservices development o implementation of security metrics for microservices **or** metrics applied to microservices security **or** application of security for microservices with metrics.

**Cadena de Búsqueda1.** Systematic mapping study of security in microservices **or** microservices security study **or** systematic mapping of security metrics for microservices.

**Cadena de Búsqueda2.** Security metrics in microservices containers **or** security metrics apply microservice containers **or** study of metrics applied to microservices containers **or** taxonomy of microservices container security metrics.

**Cadena de Búsqueda3.** Security metrics implemented in microservices development **or** implementation of security metrics in microservice **or** metrics applied to the security of microservices **or** application of security in microservices with metrics.

### 3.6. Criterios de Inclusión y Exclusión Métricas

El rango inicial del período de búsqueda se definió a partir del 2011, (Pendleton, 2016), ya que se desconocía el tema sobre microservicios antes de este año, y este se extendió hasta el 2021. En la Tabla3 se presentan los criterios de inclusión y exclusión los cuales fueron utilizados para filtrar los resultados obtenidos a través de las cadenas de búsqueda.

*Tabla 3. Criterios Inclusión y Exclusión de datos.*

ID	Criterios
<b>Inclusión</b>	
I1	✚ Artículos publicados del 2011-2021.
I2	✚ Artículos publicados en idioma inglés.
I3	✚ Artículos que aborden problemas de métricas para seguridad en microservicios.
I4	✚ Artículos que apliquen métricas para seguridad en microservicios.
I5	✚ Artículos que realicen mapeos sistemáticos en microservicios.
<b>Exclusión</b>	
E1	✚ Artículos incompletos.
E2	✚ Documentos que estén en formatos no adecuados (Noticias, presentaciones, Tesis, tesinas, patentes).
E3	✚ Artículos que mencionan el término de seguridad en microservicios, pero no es el tema principal abordado por el artículo.
E4	✚ Trabajos que no se relacionan con seguridad para microservicios.
E5	✚ Trabajos que se enfocan en Revisiones Sistemáticas de la literatura.

### 3.7. Extracción y Síntesis de Datos

Para completar el MS se clasificaron los trabajos seleccionados sobre métricas de acuerdo a (Wieringa Roel, 2006), el cual define los criterios de clasificación: investigación exploratoria, investigación explicativa, investigación aplicada, artículos teóricos y artículos de revisión como se muestran anteriormente (Tabla 1). Encontramos estas categorías fáciles

de interpretar y usar para la clasificación sin evaluar cada un artículo en detalle (como se hace para una revisión sistemática).

Se realizó una clasificación de estudios por tipo de investigación, la cual permitió validar, analizar y clasificar los artículos finales como se muestra a continuación:

**La investigación exploratoria:** tiene como fin obtener estudios iniciales que aporten una visión general al problema de mapeos sistemáticos de métricas para seguridad en microservicios.

**La investigación explicativa:** tiene como fin obtener estudios que evidencien los efectos sobre métricas de seguridad para microservicios.

**La investigación aplicativa:** tiene como fin obtener estudios que aborden el problema de métricas para seguridad en microservicios, así como la aplicación.

**Artículos teóricos:** tiene como fin obtener estudios que aborden el problema de mapeos sistemáticos de seguridad en microservicios, así como las métricas aplicadas en diseño o desarrollo de microservicios.

**Artículos de revisión:** tiene como fin obtener estudios que aborden el problema de métricas de seguridad para microservicios por medio de mapas, mapeos, revisiones, etc.

### 3.8. Resultados de MS de Métricas para seguridad en microservicios

El proceso de búsqueda de información se realizó desde el año 2011 hasta enero del 2021 y aplicando “*snowballing*” (Streeton, 2004), mostrando un total de 3308 artículos publicados, relacionados con métricas, por las diferentes fuentes de información de los cuales se eliminaron 6 artículos duplicados, esto redujo el número a 3302. Al seleccionar los títulos y resúmenes de los artículos se excluyeron 3260 artículos por su irrelevancia con el tema de investigación obteniendo un total de 42, una vez realizado esto, se aplicaron los criterios de inclusión y exclusión y se eliminaron 32. El total de artículos seleccionados fueron 10, como se puede observar en la Figura 2.

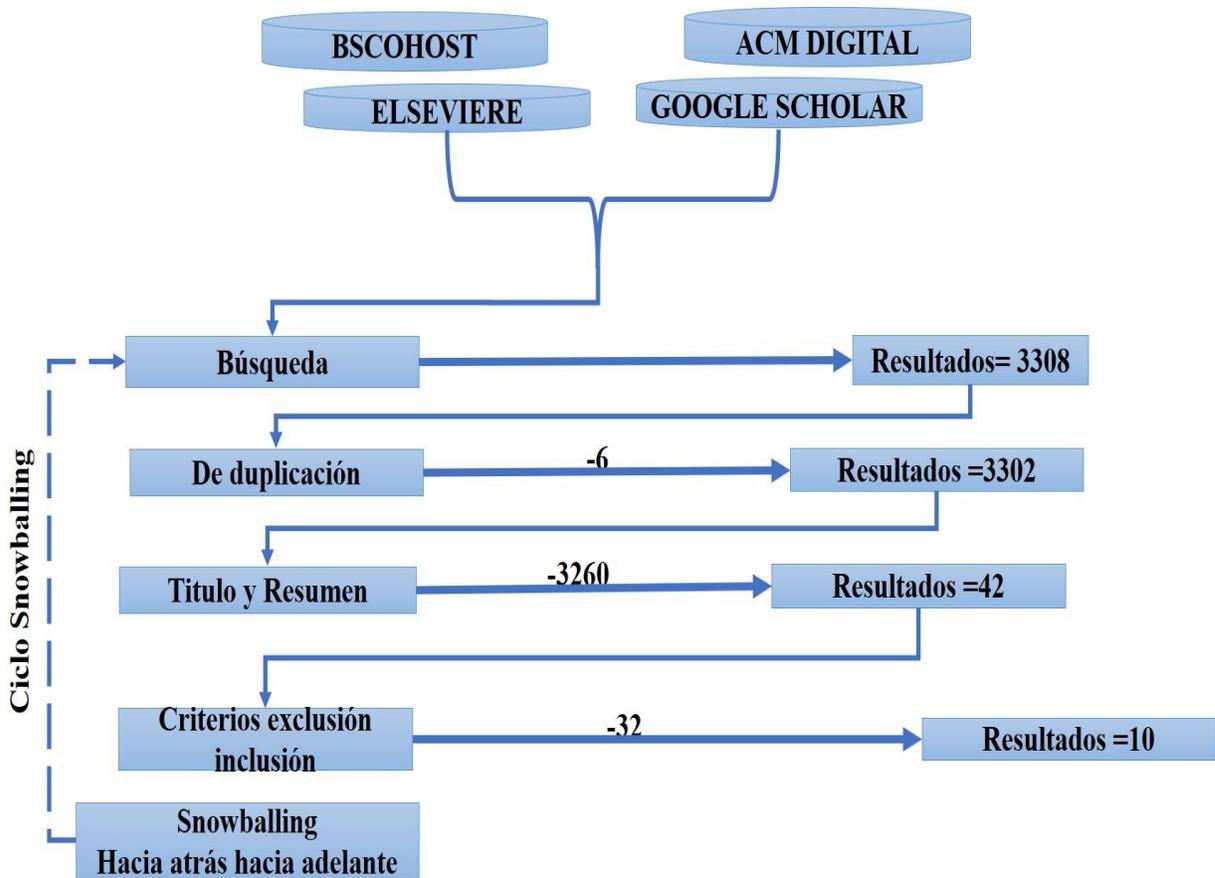


Figura 2. Proceso General de Selección

En la Figura 3 se muestran los resultados obtenidos de la distribución de búsquedas por fuentes de información.

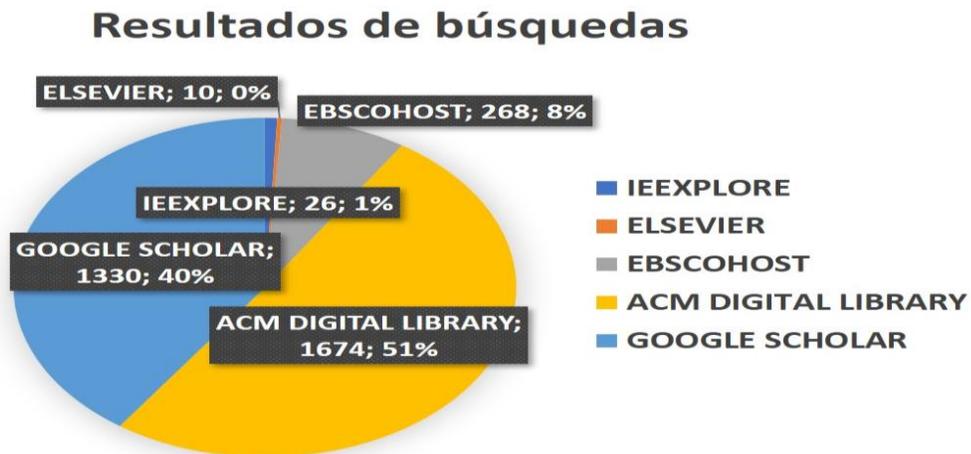


Figura 3. Distribución de estudios por fuentes de información.

La Figura 4 muestra las palabras claves utilizadas en artículos seleccionados, como las palabras más utilizadas son microservicios, seguridad, contenedores, métricas, taxonomía y privacidad.



En la Figura 5, se muestran las preguntas de investigación y el total de los estudios que dan respuesta a estas preguntas de investigación (Tabla 4): RQ1 0%, RQ2 20% y para RQ3 un 80%. El análisis de estas preguntas se presenta en la siguiente sección.

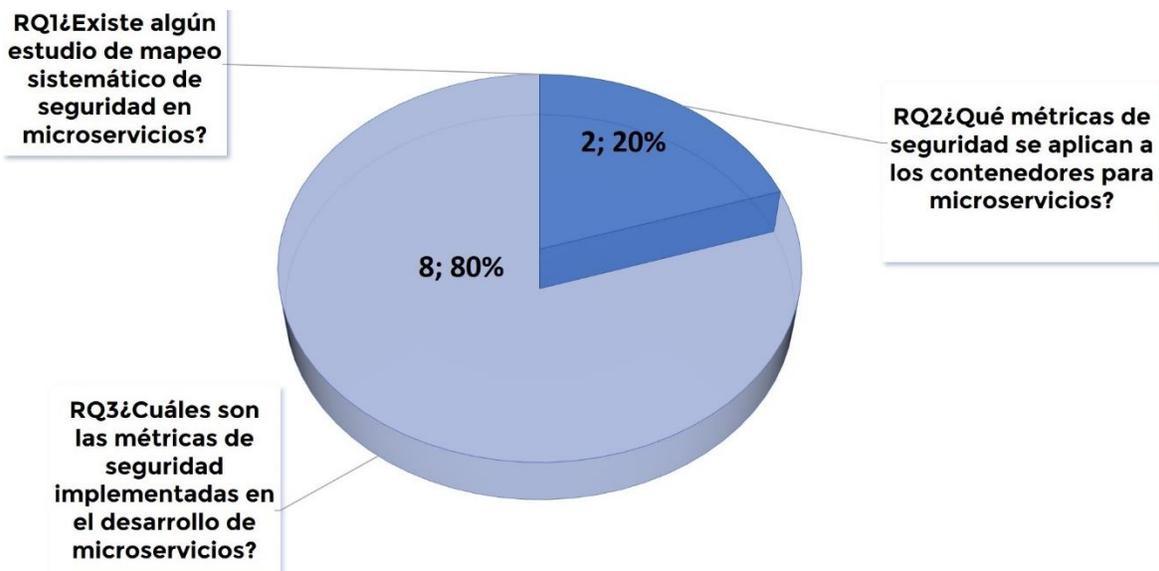


Figura 5. Artículos que responden las preguntas de Investigación.

### 3.10. Análisis de las Preguntas de Investigación Métricas en Microservicios

**RQ1.** ¿Existe algún estudio de mapeo sistemático de seguridad en microservicios?

Existen MS enfocados al análisis de su funcionamiento y características de estos, (Alshuqayran Nuha, 2016).

Algunos otros mencionan mapeos sistemáticos enfocándose a mecanismos de seguridad donde evalúan vulnerabilidades como en el caso de (Anelis Pereira Vale\*, 2019), pero ninguno está relacionado con métricas de seguridad (Muhammad Waseema, 2021).

**RQ2.** ¿Qué métricas de seguridad se aplican los contenedores para microservicios?

Se encontraron artículos como el de (Nancy Loja Mora, 2017), donde mencionan el análisis realizado para identificar métricas, que se aplican en diferentes servicios como son los contenedores.

En el trabajo de (Guerron, 2020), que proporciona una serie de clasificaciones dependiendo el servicio, tamaño para transferencia de recursos.

**RQ3.** ¿Cuáles son las métricas de seguridad implementadas en el desarrollo de microservicios?

Se muestran trabajos relacionados en métricas, que se aplican a microservicios tomadas de otras arquitecturas como son SOA y que se están aplicando a modelos que los autores trabajan de forma experimental.

En el caso de (Sandoval, 2009) que proporciona un modelo taxonómico para seguridad en software especializados, es decir, para usuarios u organizaciones en específico, basadas en sus características propias.

El trabajo de (Pendleton, 2016) realiza un estudio basado en cómo medir la seguridad de un sistema mediante la propuesta de un marco de seguridad dividido en submétricas.

(Thomas Engel, 2018) propone un enfoque de evaluación de arquitecturas de microservicios basados en investigación y prácticas sobre el acoplamiento flexible.

El autor (Guerron, 2020), realiza una taxonomía de las métricas de calidad para el servicio en la nube.

En el trabajo de (Mohammad S. Aslanpour, 2020) explora la literatura y presenta una taxonomía de distintas métricas para la evaluación del rendimiento de la computación en la nube.

Para (Bandar Alshammari, 2010) diseña una aplicación orientada a objetos definiendo una serie de métricas de seguridad para su comparación con base a la calidad.

En el trabajo de (Sebastiano Panichella, 2021) propone una forma de calcular y visualizar el acoplamiento de código abierto.

Para (Mohamed Almorsy, 2013), realiza un análisis de la seguridad de arquitecturas utilizando escenarios y métricas de seguridad.

En el caso de (Muhammad Waseema, 2021), comprende los microservicios mediante supervisión de métricas de monitorización y gestión de registro probados en la industria.

### 3.10.1. Taxonomías en Métricas Aplicadas a Sistemas Software.

A continuación, se muestran en la Tabla 5 tres taxonomías que realizan una descripción conceptual sobre métricas para seguridad en diferentes arquitecturas. En el trabajo de (Sandoval, 2009) realizaron un análisis de sistemas especializados de los cuales combinan la utilización de la gestión de seguridad y desarrollo para el software utilizando algunas normas de calidad ISO/IEC 21827, sobre los sistemas de software especializados, es decir, es el software que es dedicado a un usuario o empresa en específico, las otras dos taxonomías están dedicadas a sistemas basados en la nube en el trabajo de (Guerron, 2020) menciona la descomposición del sistema basado en la nube mediante la descripción de un metamodelo basándose en algunas normas de calidad ISO. Los artículos presentan un nivel diferente de conceptualizar las métricas dependiendo del uso de diagramas UML, así como herramientas para medir rendimiento de estas (MAPE-K loop), las taxonomías presentan una referencia en

común que son la utilización de métricas de calidad del software y normas de calidad (Mohammad S. Aslanpour, 2020).

*Tabla 5. Análisis de Taxonomías de Seguridad en diferentes arquitecturas*

Nombre del Artículo	Características
<b>A Security Metrics Taxonomization Model for Software-Intensive Systems (SMOS (Sandoval, 2009)).</b>	Descomposición de requisitos de acuerdo a las características del sistema especializado basado en la Norma (SSE-CMM) Norma ISO/IEC 21827 que trata de las actividades de ingeniería de la seguridad, teniendo dos puntos de vista uno en la gestión de la seguridad del software y en la gestión para el proyecto de software.
<b>A Taxonomy of Quality Metrics for Cloud Services (Guerron, 2020).</b>	Descomposición mediante metamodelo 1. Taxonomía del metamodelo de las métricas de los servicios en la nube. 2. Taxonomía de las métricas de calidad del servicio en nube. Sistemas basados en la nube Norma ISO/IEC 15939 que define un proceso de medición aplicable a los sistemas basados en sistemas en la nube.
<b>Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research (Mohammad S. Aslanpour, 2020).</b>	Divide las capas de la nube en 1. IoT. 2. Capa de Edge Computing 3. Capa de Fog Computing 4. Capa de Cloud Computing 5. MAPE-K loop Evaluación de rendimiento, algunas métricas que son iguales, pero en cada capa es diferente su medición y son investigadas para evaluar elementos fundamentales y tiene diferentes niveles de importancia.

### 3.10.2. Análisis de métricas aplicadas a diversas arquitecturas

En la Tabla 6 se muestran 7 trabajos sobre métricas de seguridad aplicadas a diferentes arquitecturas utilizando normas de calidad del software, se muestran métricas de forma general, así también clasifican algunas de estas métricas por sus vulnerabilidades, tipo de defensa y ataques, además muestran la clasificación para medirlas: normales, ordinarias e intervalos, analizando ventajas y desventajas sobre las mismas.

Actualmente, no existen métricas específicas para la seguridad en microservicios, pero se realizan adaptaciones utilizando atributos de calidad del software que solo se enfocan en casos de prueba, así como análisis y conceptualización. En estos trabajos no se han realizado pruebas de manera directa a microservicios, sino que son propuestas, algunos de ellos se desarrollan en entornos virtuales. En estos trabajos (Tabla 6) se auxilian de herramientas de diseño de diagramas como (UML) y gráficas que les permiten tener una interpretación de los datos para una posible hipótesis que quizás a futuro pueda funcionar específicamente para los microservicios, pero hasta el día de hoy no existen soluciones de seguridad probadas y evaluadas exhaustivamente, actualmente utilizan aquellas derivadas de aplicaciones monolíticas y arquitectura SOA, que si bien son propuestas adaptables para su posible aplicación, esto no soluciona el problema de seguridad.

Tabla 6. Análisis de trabajos sobre métricas de seguridad en diferentes arquitecturas

Nombre del Artículo	Características
A Survey on Systems Security Metrics (Pendleton, 2016).	<ul style="list-style-type: none"> <li>• Las miden por escalas: Nominal, Ordinaria, Intervalo.</li> <li>• Definen como métricas bases exploratorias, impacto, temporales y medioambientales.</li> <li>• Para sistemas empresariales de ataque defensa, analizan ventajas y desventajas o limitaciones análisis de cada una de ellas. Centrado en el estado de seguridad actual de un sistema dado.</li> </ul>
Structural Coupling for Microservices (Sebastiano Panichella, 2021).	<ul style="list-style-type: none"> <li>• La mayoría de estas métricas puede ser adecuadas para microservicios guiada por el acoplamiento y cohesión</li> <li>• Estas métricas se basadas en mediciones manuales con base a un conjunto de propiedades y no están validadas empíricamente.</li> </ul>
Quality metrics for web application development (Nancy Loja Mora, 2017).	<ul style="list-style-type: none"> <li>• Muestras métricas orientas a aplicaciones web para proveer el desarrollo a empresas y desarrolladores, basados en encuestas y revisiones bibliográficas.</li> <li>• El análisis muestra para medir las aplicaciones Web se utilizan los mismos modelos que para el software tradicional. Pero para algunos autores ciertas características son más importantes.</li> </ul>
Evaluation of Microservice Architectures: A Metric and Tool-Based Approach (Thomas Engel, 2018).	<ul style="list-style-type: none"> <li>• Realizan derivaciones de principios y métricas para realizar evaluaciones, utilizan el enfoque de ingeniería inversa. Método de evaluación basándose en datos de comunicación dinámica.</li> <li>• Para evaluar el modelo de dependencia (caso específico). consideran atributos de calidad como acoplamiento, cohesión y granularidad.</li> </ul>
Design, Monitoring, and Testing of Microservices Systems: The Practitioners' Perspective (Mohammad S. Aslanpour, 2020)	<ul style="list-style-type: none"> <li>• Realizó un estudio de métodos mixtos como encuestas y entrevistas. Una combinación de diseño impulsado por el dominio y la estrategia de negocio para la descomposición de una aplicación.</li> </ul>
Automated Software Architecture Security Risk Analysis using Formalized Signatures (Almorsy, 2013)	<ul style="list-style-type: none"> <li>• Utilizando el lenguaje de restricción de objetos (OCL). Analiza un sistema para localizar coincidencias para los escenarios de ataque.</li> <li>• Capturan los detalles de seguridad usando UML y SecDSVL.</li> <li>• Puede aplicarse a nivel diseño y código.</li> </ul>
Security Metrics for Object-Oriented Designs (Bandar Alshammari, 2010)	<ul style="list-style-type: none"> <li>• Estudio de cada propiedad y su relevancia en el diseño del software seguro</li> <li>• Métricas orientadas objetos basadas en las propiedades de diseño de calidad del software.</li> <li>• Utilizando UMLsec y SPARK.</li> </ul>

### 3.10.3. Resumen de Trabajos relacionados de Métricas de seguridad para Microservicios

En el trabajo de (Sandoval, 2009), la principal contribución es introducir un nuevo modelo para la taxonomía en relación a la descomposición de requisitos basado en software especializado donde sistematiza y organiza el desarrollo de métricas de seguridad basadas en el riesgo.

Para (Pendleton, 2016), su trabajo está centrado en cómo puede evolucionar el estado de seguridad de un sistema como resultado de las interacciones entre ciberataque y defensa, para ello este estudio propone como medir la seguridad de un sistema mediante un marco métrico basado en cuatro submétricas proponiendo una ontología jerárquica.

En el trabajo de (Guerron, 2020), clasifica métricas basadas en la nube y compara las métricas de calidad de servicio (QoS) en el ámbito de la computación en la nube.

Por otro lado, (Mohamed Almorsy, 2013), introducen un nuevo enfoque para apoyar el análisis de seguridad de las arquitecturas utilizando escenarios y métricas de seguridad basado en la formalización de escenarios de ataque, utilizando lenguaje de restricción de objetos (OCL), analizando un sistema para localizar las coincidencias para los escenarios de ataque.

El trabajo de (Muhammad Waseema, 2021), se centra en un análisis profundo para comprender como los sistemas de microservicios son diseñados, supervisados y aprobados en la industria mediante encuestas y entrevistas a profesionales de los microservicios.

El trabajo de (Thomas Engel, 2018), proporciona un enfoque de evaluación de las arquitecturas de microservicios basándose en los principios de investigación y prácticas como el tamaño reducido de los servicios, el diseño basado en el dominio o el acoplamiento flexible.

Dentro del trabajo (Nancy Loja Mora, 2017), analiza los diferentes modelos y estándares de calidad orientados al producto de software, mediante recopilación de datos bibliográficos y comparaciones descriptivas para la identificación de métricas eficaces en el desarrollo de aplicaciones Web.

(Mohammad S. Aslanpour, 2020), presenta una taxonomía de las distintas métricas para evaluar el rendimiento de la computación en la nube para reconocer las métricas comunes y sus aplicaciones.

(Bandar Alshammari, 2010), se centra en diseñar una aplicación orientada a objetos definiendo una serie de métricas de seguridad de la información de los elementos de diseño de un programa, estas métricas permiten descubrir y corregir vulnerabilidades de seguridad en una etapa temprana permitiendo comparar la seguridad potencial de varios diseños alternativos con la presentación de métricas basadas en acoplamiento, extensibilidad, herencia y el tamaño del diseño de un programa orientado a objeto.

El trabajo de (Sebastiano Panichella, 2021), propone formas de calcular el acoplamiento entre microservicios, extendiendo y adaptando los conceptos detrás del cálculo del acoplamiento estructural tradicional validado con casos de estudios de código abierto proporcionando un enfoque automático para medirlas.

# Capítulo 4. MS

## Patrones para

### Microservicios

El objetivo principal es mostrar estado actual de los patrones que existen en el área de seguridad para el desarrollo de microservicios aplicando la misma adaptación del MS de métricas para microservicios del capítulo 3.

#### 4.1. Formulación de Preguntas para Patrones en Microservicios

El objetivo principal de este MS fue obtener un panorama sobre el estado actual de los patrones de seguridad en microservicios y analizar cómo se están aplicando, para esto se plantearon tres preguntas de investigación:

**RQ1** ¿Qué patrones de seguridad se están aplicando a nivel diseño en microservicios?

**RQ2** ¿A nivel desarrollo en microservicios, qué patrones de seguridad se están aplicando?

**RQ3** ¿Qué métricas existen para medir los patrones de seguridad en microservicios?

#### 4.2. Selección de Fuentes

Las fuentes de información digitales en línea consultadas para este MS se muestran en la Tabla 7.

Tabla 7. Fuentes de Información.

Fuente	Dirección Url
IEEEExplore	<a href="https://ieeexplore.ieee.org/Xplore/home.jsp">https://ieeexplore.ieee.org/Xplore/home.jsp</a>
Elsevier	<a href="https://www.elsevier.com/es-mx">https://www.elsevier.com/es-mx</a>
ACM Digital	<a href="https://dl.acm.org/">https://dl.acm.org/</a>
Springer	<a href="https://link.springer.com/">https://link.springer.com/</a>
Google Scholar	<a href="https://scholar.google.es/schhp?hl=es">https://scholar.google.es/schhp?hl=es</a>

### 4.3. Realización de Búsquedas y Palabras Clave Patrones

Para el MS de patrones se obtuvieron las siguientes palabras clave:

- Patrones
- Microservicios
- Métricas
- Seguridad
- Diseño
- Servicios
- Mapeo Sistemático

### 4.4. Cadenas de Búsqueda para Patrones de seguridad en microservicios

Se efectuó una búsqueda inicial con respecto a mapeos sistemáticos orientados a patrones y seguridad en microservicios, la cual no arrojó resultados específicos. A continuación, se muestran las cadenas generales de búsqueda en relación con las preguntas de investigación antes mencionadas. Estas cadenas fueron modificadas de acuerdo a cada fuente de información:

**Cadena de Búsqueda1.** Security patterns applied to the microservices design level **or** security patterns adapted to the microservices design level **or** security patterns in microservices **or** application of security patterns in microservices.

**Cadena de Búsqueda2.** Security patterns applied at the microservices development level **or** security patterns adapted to the microservices development level **or** security patterns in microservices **or** security in microservices **or** security in microservices application of patterns.

**Cadena de Búsqueda3.** Existing metrics to measure security patterns in microservices **or** adapted metrics to measure security patterns in microservices **or** application of security metrics with patterns in microservices **or** security metrics and patterns in microservices.

En estas búsquedas se encontraron artículos relacionados con microservicios de manera general y en relación a los patrones en otras arquitecturas y diferentes aplicaciones.

### 4.5. Criterios de Inclusión y Exclusión Patrones

El rango inicial del período de búsqueda se definió a partir del 2011, (Pendleton, 2016), ya que se desconocía del tema sobre microservicios antes de este año y este se extendió hasta el 2021.

En la Tabla 8 se presentan los criterios de inclusión y exclusión utilizados para filtrar los resultados obtenidos a través de las cadenas de búsqueda.

*Tabla 8. Criterios Inclusión y Exclusión de datos.*

ID	Criterios
<b>Inclusión</b>	
I1	Artículos publicados del 2011-2021.
I2	Artículos publicados en idioma inglés.
I3	Artículos que aborden el problema de patrones en seguridad para microservicios.
I4	Artículos que apliquen patrones de seguridad.
I5	Artículos que realicen mapeos sistemáticos aplicando patrones a microservicios.
<b>Exclusión</b>	
E1	Artículos incompletos.
E2	Documentos que estén en formatos no adecuados (Noticias, presentaciones, Tesis, tesinas, patentes).
E3	Artículos que mencionan el término de seguridad en Microservicios, pero no es el tema principal abordado por el artículo.
E4	Trabajos que no se relacionan con seguridad para microservicios.
E5	Trabajos que se enfocan en Revisiones Sistemáticas de la literatura.

## 4.6. Extracción y Síntesis de Datos

Para completar el MS se clasificaron los trabajos seleccionados sobre patrones de acuerdo a (Wieringa Roel, 2006), el cual define los criterios de clasificación: investigación exploratoria, investigación explicativa, investigación aplicada, artículos teóricos y artículos de revisión mencionados en la (Tabla 1).

## 4.7. Resultados de MS de Patrones para seguridad en microservicios

El objetivo principal de este estudio fue desarrollar un MS que permitiera dar un panorama del estado actual de los patrones de seguridad para microservicios y analizar cómo se están aplicando, por este motivo se plantearon tres preguntas de investigación:

El proceso de búsqueda de información mostró datos desde el año 2011 hasta mediados del 2021 y se obtuvieron un total de 51 artículos publicados, la Figura 6 muestra la distribución de búsquedas por fuentes de información.

## RESULTADOS DE BÚSQUEDAS

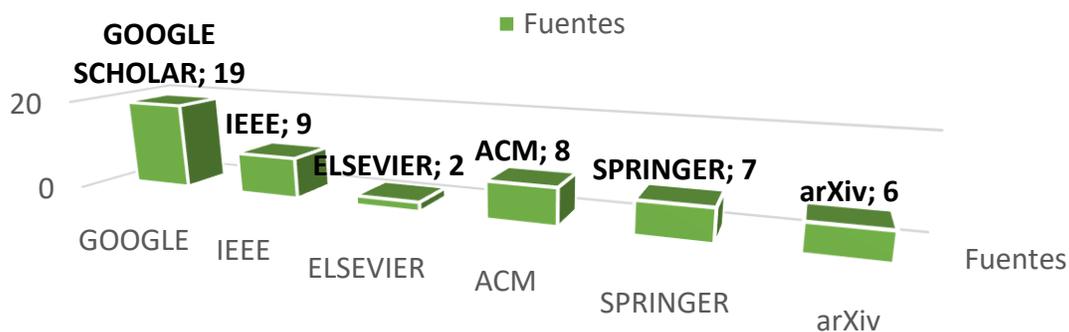


Figura 6. Resultados de búsquedas por fuente digital

Los resultados obtenidos del proceso general de selección del MS de patrones para seguridad en microservicios que responde las tres preguntas de investigación antes mencionadas, obteniendo de la búsqueda general un total de 51 artículos, de los cuales se descartaron 8 duplicados, reduciendo el número a 43 trabajos, de estos se realizó la selección de los títulos y resúmenes excluyendo 17 por su irrelevancia con el tema de investigación obteniendo 26 artículos a los cuales se le aplicaron los criterios de inclusión y exclusión eliminando 17 de ellos. El total de artículos seleccionados fueron 9 se aprecia en la Figura 7.

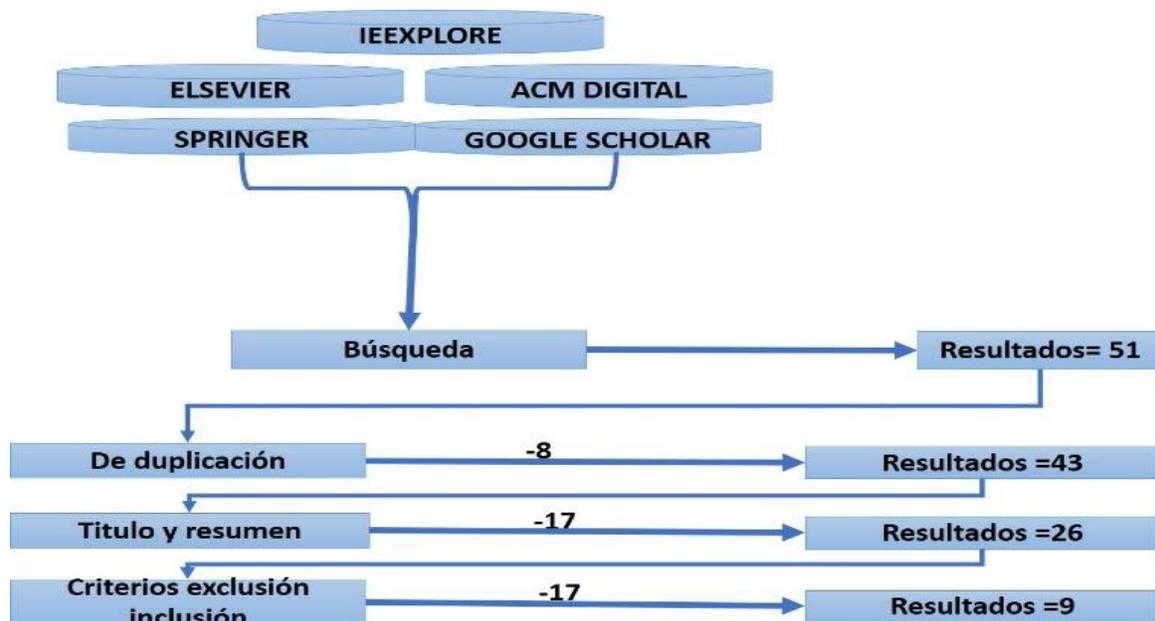


Figura 7. Proceso General de Selección.

En la Tabla 9 se muestran los artículos seleccionados dichos trabajos utilizan patrones de seguridad adoptados de otras arquitecturas para realizar adaptaciones para microservicios. Cabe señalar que no se encontró algún trabajo específico en seguridad, además estos se enfocan a diferentes arquitecturas, los cuales son adaptados y aplicados a microservicios.

Tabla 9. Publicaciones seleccionadas

ID	Nombre del Artículo	Año	Preguntas	Tipo
Astudillo, M. G. (2018)	Actual Use of Architectural Patterns in Microservices-based Open-Source Projects	2019	RQ2	C
Barabanov Alexander, M. D. (2020)	Authentication and authorization in microservice-based systems: survey of architecture patterns	2020	RQ1	A
Bogner, J. F. (2021)	Industry practices and challenges for the evolvability assurance of microservices	2021	RQ3	J
Stocke Mirko, e. a. (2018)	Interface Quality Patterns Communicating and Improving the Quality of Microservices APIs	2018	RQ1	C
Torkura Kennedy A, e. a. (2017)	Leveraging Cloud Native Design Patterns for Security-as-a-Service Applications	2017	RQ3	C
Richardson, C. (2019)	Microservices Patterns	2019	RQ3	L
Barabanov Alexander, M. D. (2021)	Security audit logging in microservice-based systems: survey of architecture patterns	2019	RQ1	J
Rudrabhatla, C. K. (2020).	Security Design Patterns in Distributed Microservice Architecture	2020	RQ1	J
Tihomir Tenev, D. B. (2018)	Security patterns for microservices located on different vendors	2018	RQ1	J

**C: Conferencia, J: Revista, A: Artículo, S: Simposio, L: Libro**

En la Figura 8 se muestra la distribución de los estudios seleccionados dependiendo su año de publicación, obtenidos de la búsqueda anteriormente mencionada de la cual se observó que existen antecedentes relevantes a partir del 2018, sin embargo, algunos documentos de años anteriores hacen referencias de los patrones en cuanto al desarrollo de Microservicios.



Figura 8. Distribución de artículos por año de publicación (Tabla9).

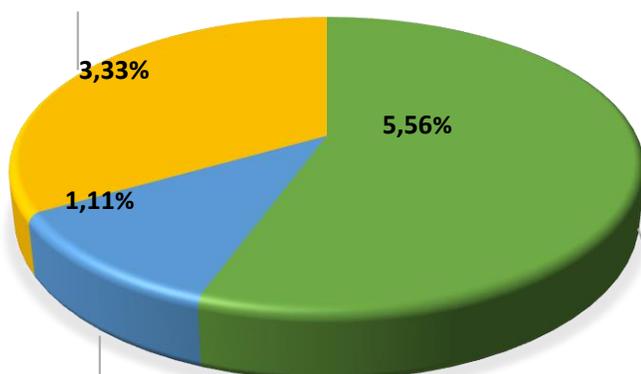
La Figura 9 muestra las palabras clave utilizadas en los artículos seleccionados, como las palabras clave como lo es patrones, seguridad, microservicios.



Figura 9. Nube de palabras clave de Patrones de Seguridad.

En la Figura 10 se muestran las preguntas de investigación y el total de los estudios que dan respuesta a estas (Tabla 9): RQ1 56%, RQ2 11% y para RQ3 un 33%.

**RQ3 ¿Qué métricas existen para medir los patrones de seguridad en microservicios?**



**RQ1 ¿Qué patrones de seguridad se están aplicando a nivel diseño en microservicios?**

**RQ2 ¿Qué patrones de seguridad se están aplicando a nivel desarrollo en microservicios?**

Figura 10. Artículos relacionados a patrones en microservicios.

## 4.8. Análisis de las preguntas de investigación patrones de seguridad en microservicios

**RQ1.** ¿Qué patrones de seguridad se están aplicando a nivel diseño en microservicios?

Se encontraron 5 artículos relacionados, donde (Stocke Mirko, 2018), muestran cinco patrones que se centran en la calidad para la seguridad de una API.

(Makrushin, 2021), realiza un análisis para las mejoras en los registros de patrones de auditoría.

(Birov, 2018), conceptualiza la utilización de patrones para la implementación de aplicaciones analiza vulnerabilidades utilizando STRIDE y muestra recomendaciones de patrones

(Barabanov A., 2020), menciona las mejores acciones para la aplicación, descentralización, ventajas y recomendaciones de los patrones.

(Rudrabhatla, 2020), se enfoca en la protección de los sistemas distribuidos conceptualizando elementos de MSA.

**RQ2.** ¿A nivel desarrollo en micro servicios, qué patrones de seguridad se están aplicando?

(Astudillo, 2018) muestra un estudio de código abierto para la aplicación de patrones basados en SOA y que se aplican a microservicios.

**RQ3.** ¿Qué métricas existen para medir los patrones de seguridad en microservicios?

Se encontraron 3 artículos.

(Torkura Kennedy A, 2017), realiza un análisis sobre aplicaciones nativas en la nube y aplicación de métricas en tiempo real.

Para (Justus Bogner, 2021), realiza un análisis de métricas y encuestas obtenidas de la literatura gris.

Para (Chris, 2019), muestras métricas de monitorización y alerta, todas relacionadas con JWT (JSON Web Token).

## 4.9. Resumen de Trabajos Relacionados a Patrones en Microservicios

En el trabajo de (Astudillo, 2018), se explora qué patrones arquitectónicos se utilizan en sistemas reales de código abierto basados en microservicios, sometiendo treinta proyectos a una revisión exhaustiva de código y diseño con criterios múltiples.

(Barabanov A., 2020), realiza un análisis de artículos de investigación sobre los patrones de autorización, así como autenticación y su aplicabilidad en función al entorno, tomando en cuenta ventajas y desventajas.

El trabajo de (Justus Bogner, 2021), presenta una revisión de la literatura gris y un sistema de entrevistas para obtener un análisis de descentralización y estandarización como los principios arquitectónicos para garantizar la coherencia y automatización con el uso de herramientas, métricas y patrones para la integración de microservicios.

Por otro lado, (Stocke Mirko, 2018), cubre el espacio de diseño de API (Application Programming Interface) proponiendo cinco patrones para la calidad de interfaz orientados a la eficiencia, seguridad y gestión.

(Torkura Kennedy A, 2017), aborda los problemas de seguridad rediseñando y desplegando una aplicación SecaaS (Security as a Service) monolítica utilizando patrones de diseño nativos de la nube, este prototipo puede manejar eficazmente aplicaciones SecaaS.

En su libro (Chris, 2019), explica los patrones de arquitectura de microservicios, recopilando 44 formas que resuelven problemas con la descomposición de servicios, gestión de transacciones, consultas y la comunicación entre servicios.

En el trabajo de (Makrushin, 2021), modelan las amenazas de la seguridad; identificando las mejores prácticas en cuanto a patrones de auditoría de registros y su aplicación en función a entornos; proporcionando un modelo de amenazas para el patrón de arquitecturas basados en microservicios.

En el artículo de (Rudrabhatla, 2020), se conceptualizan elementos de la seguridad de MSA (Microservice Architecture), en las necesidades de seguridad y protección de servicios distribuidos.

(Tihomir, 2018), realiza un análisis de vulnerabilidades utilizando STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege), que es un acrónimo que resume 6 categorías de amenazas, proporciona recomendaciones de patrones de seguridad adecuados a la función del escenario para mitigar la amenaza de seguridad como la divulgación o la manipulación de esta.

# Capítulo 5.

## Análisis de los MS

### 5.1. Clasificación de métricas para seguridad en diversas arquitecturas

En la investigación se llevó a cabo un análisis de los resultados obtenidos del MS. Se realizó una clasificación de los diferentes estudios, los cuales muestran métricas enfocadas a diferentes arquitecturas, algunos de ellos se enfocan en servicios en la nube; donde realizan conceptualizaciones, entrevistas, encuestas, la industria, así como orientadas a objetos, también hacen referencia a modelos de calidad del desarrollo del software y Normas ISO.

Estos estudios proporcionan recursos útiles para arquitectos y desarrolladores de seguridad de aplicaciones sobre métricas existentes y explorando la evolución de los microservicios, se presentan las 55 métricas, que se obtuvieron de 10 estudios seleccionados y mencionados anteriormente (Tabla 4).

Estas métricas se enfocan en el diseño de arquitecturas como SOA, DevOps y Orientadas a Objetos, para la seguridad en algunos artículos refieren la utilización de OCL y JWT para java, así como servicios de código abierto.

Para los microservicios se llevan a cabo conceptualizaciones y adaptaciones mencionando que esperan realizar pruebas para trabajos futuros, para tener mayor información y extender las investigaciones en microservicios y seguridad.

En la Tabla 10 se muestran las métricas identificadas y presentadas de acuerdo a los atributos que cada autor especifica en sus trabajos, cabe mencionar que son recomendaciones que cada uno propone para su utilización, las cuales se muestran como guía para su empleo en microservicios de acuerdo a sus investigaciones, encuestas y experiencias propias, en la parte superior se encuentran dos normas de calidad que llegan a emplearse para el desarrollo.

Tabla 10. Métricas para Diversas Arquitecturas

Clasificación	IOT	Software Especializado	Normas de Calidad	Virtualizaciones	Sistemas	Casos de usos
<b>Métricas</b>	<ul style="list-style-type: none"> <li>❖ IoT/Mist Metrics.</li> <li>❖ Common Metrics.</li> <li>❖ Fog Metrics.</li> <li>❖ Clouds Metrics.</li> <li>❖ Edge Metrics.</li> </ul>	<ul style="list-style-type: none"> <li>❖ Base Metric.</li> <li>❖ Derived Metric.</li> <li>❖ Indicator</li> </ul>	<ul style="list-style-type: none"> <li>❖ ISO/IEC 25000.</li> <li>❖ ISO/IEC 9126</li> </ul>	<ul style="list-style-type: none"> <li>❖ Virtualización de funciones de red (NFV).</li> <li>❖ Métricas del kernel MetricBeat.</li> </ul>	<ul style="list-style-type: none"> <li>❖ Status.</li> <li>❖ cpuUsage.</li> <li>❖ memory Usage.</li> <li>❖ Diskspace.</li> <li>❖ requestArrivalRate.</li> <li>❖ errorRequestRateAverageResponseTime</li> </ul>	<ul style="list-style-type: none"> <li>❖ Composite-Part Classes (CPCC).</li> <li>❖ Coupling (DCC).</li> <li>❖ Critical Classes Coupling (CCC).</li> <li>❖ Critical Classes Extensibility (CCE).</li> <li>❖ Classified Methods Extensibility (CME).</li> <li>❖ Critical Superclasses Proportion (CSP).</li> <li>❖ Critical Superclasses Inheritance (CSI).</li> <li>❖ Classified Methods Inheritance (CMI).</li> <li>❖ Classified Attributes Inheritance (CAI).</li> <li>❖ Critical Design Proportion (CDP).</li> </ul>

## 5.2. Clasificación de Métricas Aplicadas a Microservicios

En la Tabla 11 se muestran los mecanismos como calidad y sus atributos, nativos de la nube y orientados a objetos obtenidos de los estudios mencionados anteriormente, que no son propias de los microservicios, ya que se definieron en arquitecturas monolíticas y SOA para ser implementadas en los microservicios. La mayoría están relacionadas con modelos de calidad, en específico con ISO/IEC 9126 para productos de software.

Tabla 11. Clasificación de Mecanismos de Seguridad.

Mecanismos	Calidad	Atributos, Métricas	
		<ul style="list-style-type: none"> <li>❖ Autenticación</li> <li>❖ Confidencialidad</li> <li>❖ Integridad</li> <li>❖ Disponibilidad</li> <li>❖ Funcionalidad</li> <li>❖ Fiabilidad</li> <li>❖ Usabilidad</li> <li>❖ Eficiencia</li> <li>❖ Mantenibilidad</li> <li>❖ Portabilidad</li> <li>❖ Cloud native</li> </ul>	
	Cloud Native	<ul style="list-style-type: none"> <li>❖ Métricas vulnerabilidades</li> </ul>	<ul style="list-style-type: none"> <li>❖ Usuario</li> <li>❖ Interfaz</li> <li>❖ Software</li> </ul>
		<ul style="list-style-type: none"> <li>❖ Métricas de fuerzas de defensas</li> <li>❖ Métricas de ataque</li> </ul>	
		<ul style="list-style-type: none"> <li>❖ Métricas de situación</li> </ul>	<ul style="list-style-type: none"> <li>❖ Métricas de seguridad</li> </ul>
	Orientadas a Objetos	<ul style="list-style-type: none"> <li>❖ Composición</li> <li>❖ Acoplamiento</li> <li>❖ Extensibilidad</li> <li>❖ Herencia</li> </ul>	

### 5.3. Patrones de seguridad aplicables a diferentes arquitecturas

En la investigación, se llevó a cabo un análisis de los resultados obtenidos del MS; se realizó una clasificación de los diferentes estudios, los cuales muestran patrones enfocados a diferentes arquitecturas, algunos de ellos se enfocan en servicios en la nube, donde realizan conceptualizaciones, entrevistas, encuestas de fuentes académicas y de la industria, así como también hacen referencia a modelos de calidad del desarrollo del software y Normas ISO.

Estos estudios proporcionan recursos útiles para arquitectos y desarrolladores de seguridad de aplicaciones sobre patrones existentes y exploran la evolución de los microservicios.

Se encontraron un total de 49 patrones, de los cuales se obtuvieron 9 estudios, seleccionados y mencionados en la Tabla 9.

Estos patrones se clasificaron de acuerdo a los atributos de calidad con los que cada uno de ellos se relaciona, realizando una selección de acuerdo a la seguridad, autenticación y autorización para el diseño de arquitectura como: SOA, DevOps (development operations) y O.O.A. (Object-Oriented Architecture).

Principalmente utilizan la autenticación y autorización, para la seguridad y JWT (JSON Web Tokens) para java, así como servicios de código abierto, para los microservicios se llevan a cabo conceptualizaciones y adaptaciones mencionando que esperan realizar pruebas para trabajos futuros, ya que no existen patrones específicos para estos.

En la Tabla 12 se muestran los patrones identificados de acuerdo al análisis realizado, el cual clasifica patrones con los mecanismos de seguridad asociados, que cada autor especifica en sus trabajos. Cabe mencionar que son recomendaciones que cada autor propone para su utilización de acuerdo a sus investigaciones y experiencias propias.

Esta serie de patrones podrían ser un guía para su empleo en microservicios, en la parte superior se encuentran el nombre de la clasificación y dentro de cada una de estas los patrones asociados, se consideran específicos para la seguridad.

Se puede observar que autenticación y autorización se utilizan como mecanismos y podrían asociar atributos de calidad en conjunto de herramientas que permitan resolver la seguridad dentro de los servicios, donde se emplea la puerta de enlace o API Gateway principalmente, autenticación, autorización, framework como OAuth 2.0.

Es importante resaltar que todos estos trabajos constituyen propuestas de solución, ningún trabajo presenta resultados de su aplicación en problemas industriales. Aunado a lo anterior, tampoco mencionan como medir la calidad de estos patrones, no indican alguna métrica para obtener un estatus de la seguridad para dichos atributos, lo cual representa una brecha para realizar futuros trabajos de investigación en seguridad para microservicios.

Tabla 12. Clasificación de patrones con atributos de calidad

Enfocados a Seguridad			
	Authentication	Authorization	Security
Patrones	<ul style="list-style-type: none"> <li>✚ Decentralized model</li> <li>✚ Centralized pattern with a single policy decision point</li> <li>✚ Centralized pattern with embedded policy decision point</li> <li>✚ Send external entity identity as a clear or self-signed data structure using a data structure signed by a trusted sender</li> <li>✚ Design patterns arcadia framework JSON web token (JWT), OAuth 2.0</li> </ul>	<ul style="list-style-type: none"> <li>✚ Event-based messaging</li> <li>✚ Service log throttling</li> <li>✚ Backends for Frontends</li> <li>✚ Consumer-oriented contracts</li> <li>✚ Reader tolerant</li> <li>✚ API Gateway</li> <li>✚ Request-Reaction</li> <li>✚ Stand-alone systems</li> <li>✚ Event Sourcing</li> </ul>	<ul style="list-style-type: none"> <li>✚ API Key</li> <li>✚ Wish List</li> <li>✚ Rate limit</li> <li>✚ Rate Plan</li> <li>✚ Service Level Agreement</li> <li>✚ Authorization failures (access control)</li> <li>✚ Application errors and system events,</li> <li>✚ Use of high-risk functionality</li> <li>✚ Input validation failures</li> <li>✚ Application/microservice state changes</li> <li>✚ Design patterns arcadia framework JSON web token (JWT), OAuth 2.0</li> </ul>

Los mecanismos más utilizados han sido autenticación, autorización y la aplicación de algunos estándares, siendo la autenticación el mecanismo más utilizado para atender el problema de la seguridad. Este mecanismo comprende patrones descentralizados, políticas de patrones de decisión, patrones de logging, clave API, estructura de datos firmada, entidades externas firmadas, patrón centralizado con punto de decisión de política incrustado. Estas soluciones han sido aplicadas a los microservicios, sin embargo, no todas constituyen soluciones específicas para esta arquitectura.

En la Tabla 13 se muestra una clasificación de patrones organizados de acuerdo al análisis de los documentos analizados, enfocados a lo que es autenticación, autorización y el desarrollo en la nube.

Tabla 13. Mecanismos de Patrones para Microservicios

Patrones	
<b>Autenticación y Autorización</b>	
	<ul style="list-style-type: none"> <li>• Decentralized pattern</li> <li>• Centralized pattern with single policy decision point</li> <li>• Logging pattern</li> <li>• Api key</li> <li>• Using a data structure signed by trusted issuer</li> <li>• Send the external entity identity as a clear or self-signed data structures</li> <li>• centralized pattern with embedded policy decision point.</li> </ul>
<b>Mecanismos</b>	<b>API Gateway</b>
	<b>Cloud Native Desing Patterns</b>
	<ul style="list-style-type: none"> <li>• OAuth2</li> <li>• OpenID</li> <li>• JWT (Métricas de monitorización y alertas)</li> <li>• Single sign-on (SSO)</li> </ul>
	<ul style="list-style-type: none"> <li>• SecaaS</li> <li>• SaaS</li> </ul>

#### 5.4. Análisis de Métricas y Patrones para aplicación a Microservicios

En el Diagrama 1 se muestra el análisis de las métricas y patrones para microservicios donde se clasifican de acuerdo a la perspectiva de la información analizada, dichos elementos no son propios de los microservicios como ya se mencionó anteriormente, son adaptaciones que se pretenden utilizar para solventar la seguridad en microservicios.

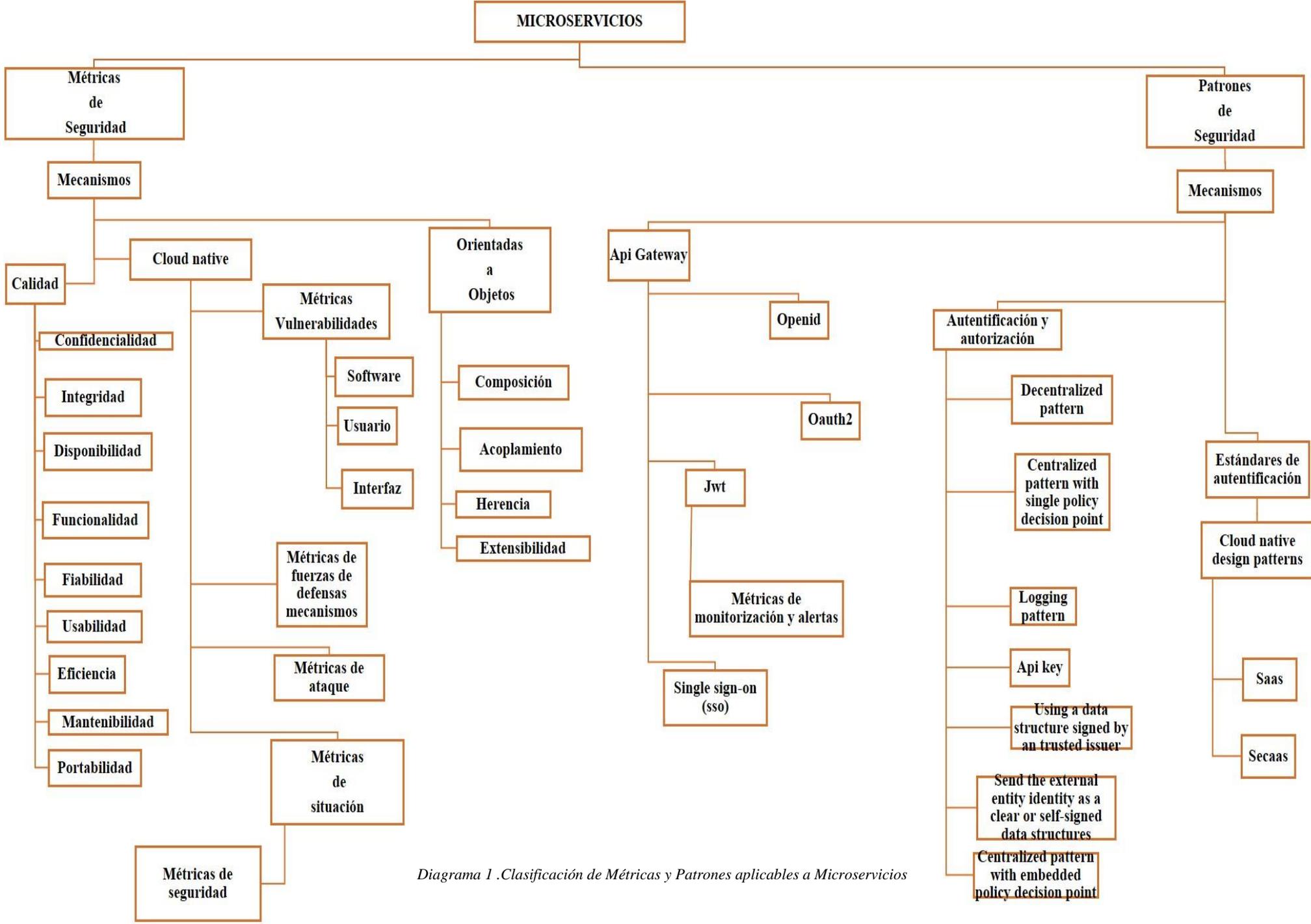


Diagrama 1 .Clasificación de Métricas y Patrones aplicables a Microservicios

# Capítulo 6. Guía de referencia para microservicios

**Resumen:** La guía surge como propuesta para elaborar microservicios de forma segura, para los desarrolladores de software, con el objetivo fundamental de señalar la base para la elaboración de los microservicios a través de la integración del ciclo de vida del software y el modelo de calidad ISO/ IEC 9126-1: 2001 que es un estándar; el cual está estructurado para mejorar las cualidades del software.

## 6.1. Objetivo

Proporcionar un instrumento técnico que pueda normalizar la elaboración de los microservicios, que permita la optimización de ellos, con la adopción y manejo de los conceptos y elementos vertidos en esta guía.

## 6.2. Actividades

Seleccionar los elementos necesarios para la elaboración de un microservicio mediante el ciclo de vida del software y la norma de calidad ISO/IEC 9126:200 (Verity, 2021).

## 6.3. Introducción

El estándar ISO/IEC 9126:2008 (Verity, 2021) especifica atributos y subatributos que se pueden utilizar en conjunto con el ciclo de vida del software. Para esta guía se considera el atributo de funcionalidad, el contenido de esta guía consiste en proponer un modelo para la arquitectura de microservicios en donde se puedan medir las características con las que se están desarrollando, para ello se consideró el atributo o característica de la “funcionalidad”, la cual consiste en determinar la capacidad del software en ejecución en términos de lo que el usuario necesita.

Esta guía consta de definiciones básicas como microservicio, ciclo de vida del software, así como normas de calidad, para mejorar, garantizar el desarrollo de los microservicios y aumentar la calidad desde la planificación hasta la implementación. Presentando una tabla de valores para la medición de elementos del desarrollo del software basados en la funcionalidad con valores propuestos a medir, así como el desarrollo de diagramas definidos para explicar y demostrar cada elemento.

## 6.4. Características comunes de los Microservicios

En resumen, en esta guía las características básicas de una arquitectura de microservicios son las siguientes:

- **Orientación hacia las capacidades de negocio:** los microservicios flexibilizan el desarrollo de aplicaciones al proporcionar un enfoque modular.
- **Independencia de los microservicios entre sí:** cada microservicio contiene su propia lógica de negocio y se despliega de manera separada al resto, permitiendo así realizar actualizaciones en algunos microservicios sin necesidad de un corte en el servicio global.
- **Gestión descentralizada de datos:** cada microservicio puede tener su propia base de datos (incluso de distinta tecnología).
- **Tolerancia a fallos:** los microservicios están débilmente acoplados por lo que los fallos no se propongan en la cadena de servicios, contribuyendo en un sistema global más estable y robusto (Roldan Martínez David, 2018).

En la Figura 11 se pueden observar los beneficios de la arquitectura de microservicios.

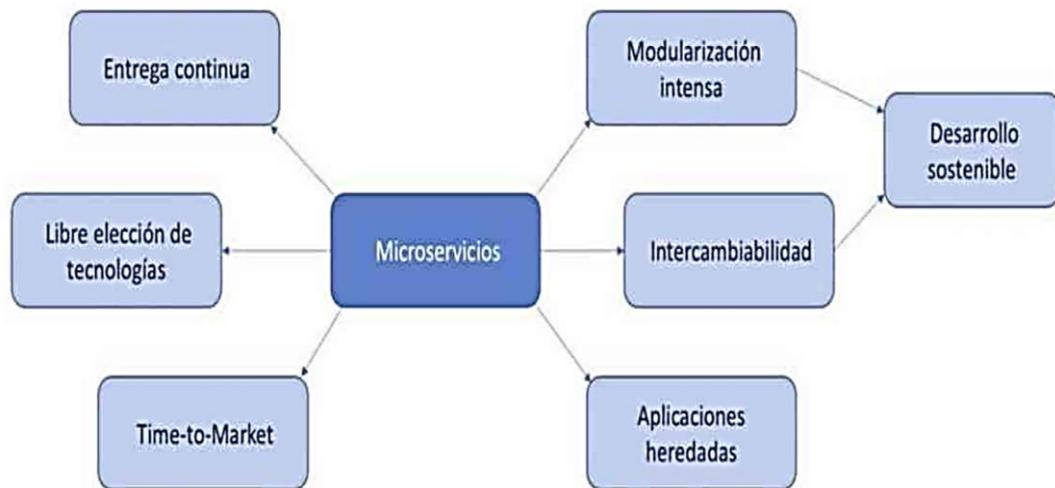


Figura 11. Beneficio de la arquitectura de los microservicios, Roldan Martínez David, P. J. (2018).

## 6.5. HTTP REST

REST (*Representational State Transfer*) es un protocolo basado en http orientado a desarrollo de aplicaciones web de manera muy sencilla y especialmente adecuada para el diseño de API, se trata de un protocolo sin estado, es decir, que no debe guardarse información de estado en el servidor, sino que toda la información necesaria debe contener en la consulta del cliente (Roldan Martínez David, 2018).

## 6.6. Ventajas y Desventajas de los microservicios

Algunas de sus ventajas y desventajas (Roldan Martínez David, 2018).

### **Ventajas de los microservicios:**

1. Equipo de trabajo mínimo
2. Escalabilidad
3. Funcionalidad modular, módulos independientes.
4. Libertad del desarrollador de desarrollar y desplegar servicios de forma independiente
5. Uso de contenedores permitiendo el despliegue y el desarrollo de la aplicación rápidamente

### **Desventajas de los microservices**

- Alto consumo de memoria
- Necesidad de tiempo para poder fragmentar distintos microservicios
- Complejidad de gestión de un gran número de servicios
- Necesidad de desarrolladores para la solución de problemas como latencia en la red o balanceo de cargas
- Pruebas o testeos complicados al despliegue distribuido

## 6.7. El ciclo de vida del desarrollo de software

La norma ISO 12207 define el Ciclo de Vida del Software como Un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso. Existen distintos modelos de Ciclo de Vida, que determinan cuáles son y de qué manera se ejecutan las etapas del desarrollo de software (Mauricio Diéguez, 2011). Sus fases son: ***Planificación, Análisis, Diseño, Implementación, Pruebas, Instalación o despliegue, Uso y Mantenimiento.***

### 6.7.1. Planificación

Es la estructuración de una serie de acciones que se llevan a cabo para cumplir determinados objetivos. La planificación es entonces, en términos generales, la definición de los procedimientos y estrategias a seguir para alcanzar ciertas metas (Westreiche, 2020).

#### **Etapas de la planificación**

Dentro de la planificación podemos identificar tres etapas:

- Identificación del problema a resolver y/o de los objetivos que se desean cumplir.
- Proponer soluciones y estrategias que se deban seguir para resolver el problema identificado o para cumplir con las metas planteadas.
- Después de analizar todas las opciones, determinar cuáles son las acciones más eficientes para cumplir con los objetivos propuestos, estructurando un plan.

## 6.7.2. Análisis

El análisis de requerimientos permite al ingeniero de sistemas especificar las características operacionales del software (función, datos y rendimientos), indica la interfaz del software con otros elementos del sistema y establece las restricciones que debe cumplir el software (Pressman, 2002).

El análisis de requisitos del software se puede subdividir en cinco áreas de esfuerzo:

1. Reconocimiento del problema
2. Evaluación y síntesis
3. Modelado
4. Especificación
5. Revisión

## 6.7.3. Requisitos

Un **requisito de software** es la capacidad que debe de alcanzar o poseer un sistema o componente de un sistema para satisfacer un contrato estándar, especificación u otro documento formal (Rodríguez, 2012).

### 6.7.3.1. Tipos de requisitos

Tradicionalmente se han identificado 2 tipos de requisitos atendiendo a criterios de funcionalidad: *requisitos funcionales* y *requisitos no funcionales*. Los primeros definen aquellas funciones que el sistema o alguno de sus componentes debería ser capaz de hacer; los segundos, por el contrario, describen restricciones y exigencias de calidad del sistema o de los procesos realizados por el mismo. El glosario IEEE de ingeniería del software (IEEE, 1990) define el primero de estos términos del siguiente modo:

Un **requisito funcional** especifica una función que un sistema o componente de un sistema debe ser capaz de llevar a cabo.

Por otra parte, es posible definir requisitos no funcionales de la siguiente manera:

Los **requisitos no funcionales** son aquellos que especifican aspectos técnicos que debe incluir el sistema, ya que pueden clasificarse en restricciones y calidades (Rodríguez, 2012).

## 6.7.4. Diseño

El momento del diseño implica una representación mental y la posterior implementación de dicha idea en algún formato gráfico (visual) para exhibir cómo será la obra que se planea realizar. El diseño, por lo tanto, puede incluir un dibujo o trazado que anticipe las características de la obra. Se aplica habitualmente en el contexto de la industria, ingeniería, arquitectura, comunicación, marketing y otras disciplinas que requieren creatividad (Pérez Porto Julián, 2008).

Los conceptos fundamentales del software son:

- Abstracción
- Acoplamiento
- Cohesión

- Descomposición y modularización
- Encapsulamiento y ocultamiento de información
- Separación de interfaz e implementación
- Suficiencia, comprensión
- Sencillez

### 6.7.5.Implementación

**Implementación:** La implementación constituye la realización de determinados procesos y estructuras en un sistema. Representa así la capa más baja en el proceso de paso de una capa abstracta a una capa más concreta (Voigtmann GmbH, 2005).

**Construcción:** Define un conjunto de actividades que engloban fundamentalmente la codificación, pero también la verificación del código su depuración y ciertos tipos de pruebas estas actividades parten de una especificación de requisitos detallados que fueron elaborados durante las actividades del diseño y dan como resultado un software libre de errores que cumple con dicha especificación (Rodríguez, 2012).

**Lenguajes de construcción:** El glosario y IEEE de términos de ingeniería de software define lenguaje de computadora como un lenguaje diseñado para permitir a los humanos comunicarse con las computadoras dentro de esta categoría tan genérica existen diferentes subcategorías tales como los lenguajes de consulta los lenguajes de especificación a los lenguajes de construcción los lenguajes de construcción incluyen todas las posibles formas de comunicación mediante las cuales un humano puede especificar a una computadora a una solución ejecutable a un problema (Rodríguez, 2012).

**Lenguajes de configuración:** Lenguajes que, a partir de un conjunto de opciones, permiten especificar cómo se configurara una determinada instalación de un software. Es habitual que las especificaciones de entrada se proporcionen en forma de fichero de texto, legible por tanto con cualquier editor (Rodríguez, 2012).

**Lenguajes de las cajas de herramientas (toolkits):** más complejos que los anteriores, se emplean para construir aplicaciones a partir de bloques predefinidos en las cajas de herramientas, bien utilizando una interfaz específica para ello o bien a través de un lenguaje de programación simplificado incorporado en la propia caja de herramientas (Rodríguez, 2012).

**Lenguajes de programación:** Son los más flexibles potentes y ampliamente utilizados los lenguajes de programación no son solo la forma en que los programadores implementan prácticamente el diseño detallado realizado por los diseñadores, la elección de un determinado lenguaje determina inequívocamente el modo en que el proceso de construcción se llevará a cabo (Rodríguez, 2012).

Según (Rodríguez, 2012) Los principios fundamentales de la construcción del software son los siguientes:

1. Minimizar la complejidad (obtener código simple).
2. Anticipar los cambios (métodos dirigidos).
3. Construir para verificar (pruebas).
4. Utilizar estándares (especificaciones).

Complejidad (Rodríguez, 2012). Expresa el grado en que un sistema o componente tiene un diseño o implementación difícil de entender o verificar (IEEE, 1990).

### 6.7.6. Pruebas

Es todo proceso orientado a comprobar la calidad del software mediante la identificación de fallos en el mismo la prueba implica necesariamente la ejecución del software. Se denomina probar un software al proceso de mostrar la presencia de un error en el mismo, consiste en descubrir en qué lugar exacto se encuentra un error y modificar el software para eliminar dicho error (Rodríguez, 2012).

#### Pruebas de caja blanca y de caja negra

El concepto de **caja negra** de uso frecuente en muchas disciplinas se refiere al hecho de pensar en un mecanismo o sistema como si la entrada de datos para su desempeño se conocían, pero su funcionamiento interno se ignore, debe tenerse en cuenta que la mayoría de nosotros utiliza a diario mecanismos según este enfoque (Rodríguez, 2012).

Se considera **caja blanca** a todo componente para cuyo uso es necesario conocer su funcionamiento interno; en el contexto de la prueba de software se habla de pruebas de caja blanca cuando la propia naturaleza de la prueba impone la observación del código del módulo a probar (Rodríguez, 2012).

### 6.7.7. Revisiones

Las revisiones ofrecen la oportunidad de poder ejecutar en la fase de desarrollo medidas para asegurar la calidad. Con la ayuda de una revisión completa se detectan alrededor de 60-90 % de todos los errores. En una revisión participan, por lo menos, el autor del programa, un perito, un redactor de actas y un moderador. Con frecuencia, se emplea para la revisión una lista de control estandarizada. El "walkthrough" es una variante de revisión con menos formalismos y menos participantes (Voigtmann GmbH, 2005).

### 6.7.8. Instalación o despliegue

La instalación del software se efectúa bien en los servidores del cliente, en los ordenadores de trabajo de los usuarios o también paralelamente. Para aplicaciones de bases de datos, se realiza además frecuentemente la migración de datos de soluciones de aplicación más antiguas (Voigtmann GmbH, 2005).

La implementación de software incluye, entre otras, las siguientes actividades:

1. Crear y mantener paquetes de software actualizados y listos para instalar
2. Configurar los equipos de destino antes de instalar o desinstalar el paquete
3. Instalar o desinstalar el software en los equipos de destino

4. Configurar los equipos de destino después de la instalación o desinstalación
5. Actualizar el software existente.

### 6.7.9. Uso y Mantenimiento

Después de la puesta en operación de una solución de software, se requiere, y es usual, una asistencia continúa. Esta abarca tanto el apoyo de los usuarios durante el servicio operativo, así como aplicaciones y actualizaciones del software en caso necesario (Voigtmann GmbH, 2005).

### 6.7.10. Nivel de soporte

Se distingue entre soporte de primer nivel y de segundo nivel. El soporte de primer nivel (también llamado Helpdesk) es el primer punto de contacto para todas las solicitudes de soporte realizadas y todos los informes sobre problemas. Problemas graves se remiten al soporte de segundo nivel; por ejemplo, para software estandarizado, al fabricante del producto (Voigtmann GmbH, 2005).

### 6.7.11. Cuidado y Administración

**Cuidado y administración** (Voigtmann GmbH, 2005). La adaptación continua del software a requerimientos y condiciones cambiantes se designa como "cuidado de software".

La Gestión de Cambio es una actividad que se desarrolla durante todo el proceso de desarrollo, ya que no sabemos en qué momento se originará un cambio, las actividades en este proceso se desarrollan para:

1. Identificar el Cambio
2. Controlar el Cambio
3. Garantizar que el cambio se realizara de manera adecuada
4. Reportar los cambios a todos los interesados

## 6.8. Factor de Calidad Norma ISO/IEC 9126: 2001

Previo a la creación de esta norma existían diferentes estándares ISO relacionados con la calidad de software, incluyendo la ISO/IEC 9001:2000 y la ISO/IEC 12207:1995 las cuales buscan la adopción de un enfoque basado en procesos para mejorar la eficacia de un sistema de gestión de calidad y definir los procesos del ciclo de vida del software respectivamente. Sin embargo, estas normas no permitían resolver la dificultad que hay para cuantificar y calificar la mayoría de las características que definen un software. Es allí donde recae la importancia de la norma ISO/IEC 9126:2001, debido a que al basarse en el modelo de McCall establece una guía acerca de los elementos que deben considerarse al evaluar un software

para así generar métricas propias que guíen tanto el desarrollo como su valoración. (Patrik Berander, 2005).

### 6.8.1. Características de Calidad de un software según la ISO/IEC 9126: 2001

**Funcionalidad:** se evalúa la adecuación, el cumplimiento funcional, idoneidad, corrección, interoperabilidad, conformidad y seguridad de acceso. Por lo que es posible afirmar que la funcionalidad determina la capacidad del software de funcionar en términos de lo que el usuario necesita, de interactuar con otros sistemas y que permita el acceso de diferentes personas, pero que cumpla con las regulaciones de las leyes de protección de datos (Verity, 2021).

### 6.8.2. Funcionalidad

**Funcionalidad:** se evalúa la adecuación, el cumplimiento funcional, idoneidad, corrección, interoperabilidad, conformidad y seguridad de acceso. Por lo que es posible afirmar que la funcionalidad determina la capacidad del software de funcionar en términos de lo que el usuario necesita, de interactuar con otros sistemas y que permita el acceso de diferentes personas pero que cumpla con las regulaciones de las leyes de protección de datos (Verity, 2021).

- ✚ **Idoneidad:** atributo del software que se relaciona con la presencia y adecuación de un conjunto de funciones para tareas específicas (Verity, 2021).
- ✚ **Precisión:** Atributos del software que se basan en la provisión de resultados o efectos correctos o acordados (Verity, 2021).
- ✚ **Seguridad:** Atributos del software que se relacionan con su capacidad para prevenir el acceso no autorizado, ya sea accidental o deliberado, a programas y datos (Verity, 2021).
- ✚ **Interoperabilidad:** Atributos del software que se relacionan con su capacidad para interactuar con sistemas específicos (Verity, 2021).
- ✚ **Cumplimiento:** Atributos del software que hacen que el software se adhiera a los estándares, convenciones o regulaciones relacionados con la aplicación en las leyes y prescripciones similares (Verity, 2021).

### 6.8.3. Diagrama de Norma ISO 9126

En la Figura 12 se presenta el diagrama de norma ISO 9126 (Verity, 2021), donde se muestra los elementos de esta norma que son atributos y sub atributos enmarcando la funcionalidad, la cual es la que se está tomando de referencia para este trabajo.

Para medir este atributo se consideró asignar el 20% a cada subatributo, que en este caso contiene 5 elementos: idoneidad, predicción, seguridad, interoperabilidad y cumplimiento; los cuales al sumarlos indicaran el total del valor de la funcionalidad.

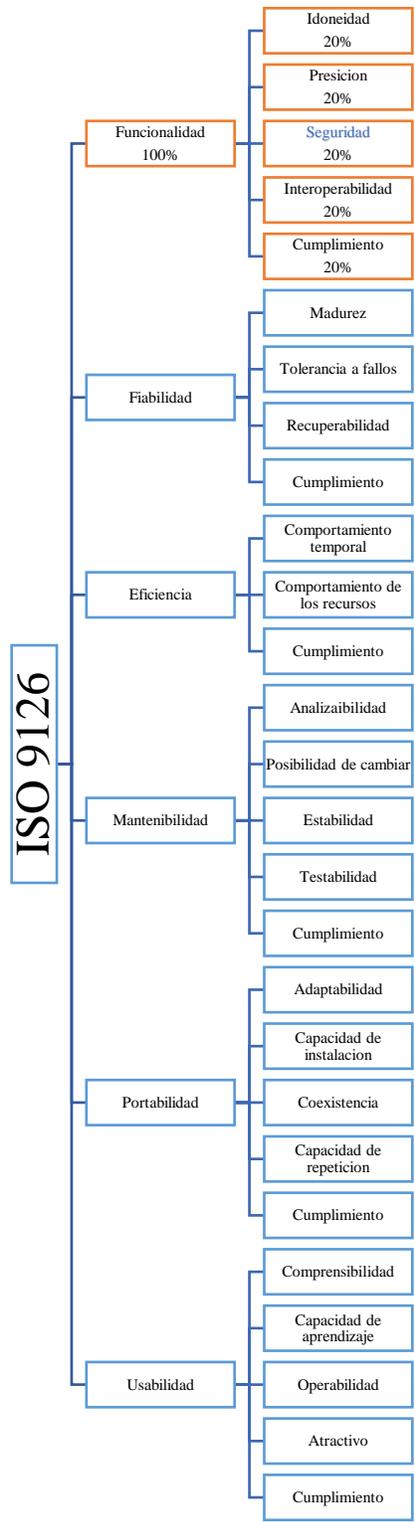


Figura 12. Norma ISO 9126 (Verity, 2021).

## 6.8.4. Primer Paso

En la Figura 13 se muestra el diagrama planteado para el desarrollo del microservicio aplicando los elementos del ciclo de vida del desarrollo del software: Planificación, Análisis, Diseño, Implementación, Pruebas, Revisión, Instalación, Mantenimiento y Administración.

Considerando cada elemento importante para el desarrollo, ya que cada etapa va ligada a la siguiente no podría existir un diseño sin un análisis previo o una instalación sin una revisión y no podría dejar de lado ningún elemento, sabemos que cada desarrollador de software maneja de forma diferente cada elemento, pero de manera general considero que el método es el mismo en el orden que deberían de ir estructurando cada parte lo que cambia es la técnica de aplicación.

A continuación, se enlistan algunas actividades o sugerencias a la hora de realizar un producto.

- a. **Planificación:** para este paso el desarrollador tendrá que tomar en cuenta 3 elementos importante como marco referencial como primer punto es el planteamiento del problema para ello se puede considerar el uso de mapas mentales o lluvias de ideas, que permitan denotar palabras claves o los objetos que estarán interactuando en el desarrollo. Para el segundo punto posteriormente al denotar los elementos primordiales se emplean un o varias propuestas solución que permitan determinar una solución al problema y el último paso es el análisis de las diferentes propuestas para determinar cuál sería la mejor opción para ello podrían utilizar diagramas (UML) que permitan dar una solución óptima al problema (Ilustración1).



Ilustración 1. Elementos de la planificación

- b. **Análisis:** para esta etapa se consideran dos elementos importantes como son los requisitos funcionales que nos permitirán determinar los servicios que presentara nuestro desarrollo como interoperabilidad, seguridad, precisión, cumplimiento, esto podría realizarse mediante alguna técnica empleada por los desarrolladores como son formularios de requisitos y los no funcionales se basa en lo que debería ser un sistema estos explican aspectos enfocados en calidad que se va a construir como lo es el rendimiento (Ilustración2).



Ilustración 2. Requisitos funcionales y no funcionales

- c. **Diseño:** para esta parte una vez que se analizó los primeros requerimientos desde el punto de vista del cliente se toma en cuenta el pliego de condiciones para elaborar un concepto, en el que se define la estructura desde programación, técnicas y algoritmos donde se explican cada elemento (Ilustración3).

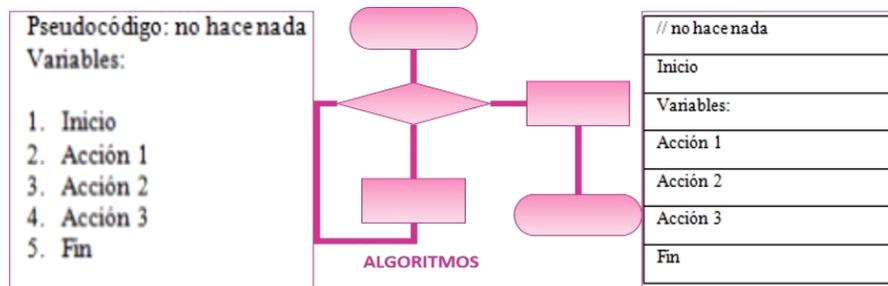


Ilustración 3. Estructuras, técnicas, algoritmos

- d. **Implementación:** para esta etapa de nuestro desarrollo debemos tomar en cuenta la construcción definitiva donde se adaptan o añaden elementos para ajustarse al desarrollo esto depende del paso anterior, tomando en cuenta el tipo de lenguaje de programación (Ilustración 4).



Ilustración 4. Tipos de lenguajes de programación

- e. **Pruebas:** en la siguiente etapa se toman pruebas de caja negra estas son para verificar la funcionalidad del software para ello se pueden utilizar herramientas y plataformas externas que verifican el funcionamiento como son Nagios, Sensu, Heartbeat (Elastic Stack) = Uptime application, Pingdom (HTTP Checks), Stackdriver (HTTP Checks), Statuscake (HTTP Checks). Para las pruebas de caja blanca es sobre la estructura interna el diseño la codificación del software para verificación de entradas esto implica probar código como agujeros sobre seguridad internos, rutas mal estructuradas, las entradas específicas, rendimiento que se espera para ello se revisa el código fuente, posterior a esto es poder crear casos de prueba o pruebas de error y uso de herramientas (Ilustración 5).

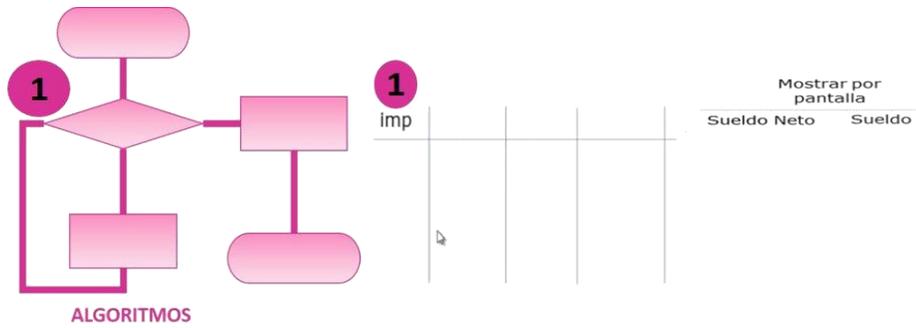


Ilustración 5. Pruebas de funcionalidad

- f. **Revisiones:** en esta parte se puede emplear el recorrido de los diagramas de casos de uso o walkthrough para detectar errores en el código fuente y encontrar algún conflicto.

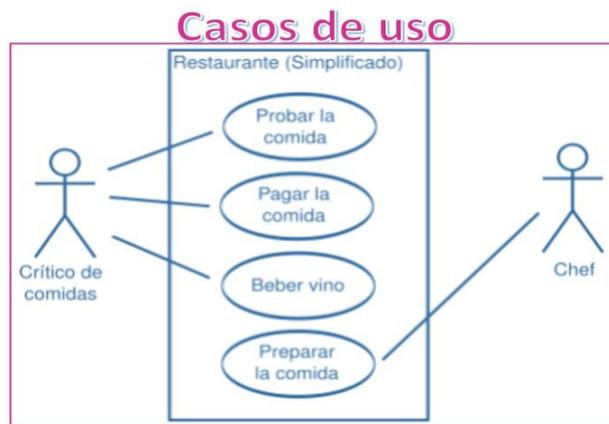


Ilustración 6. Revisión por caso de uso

- g. **Instalación:** para esta etapa se realiza cuando el software está terminado y se empieza a realizar la actualización y configuración de equipos que tendrán contenido el software, después se procede a la instalación o desinstalación de paquetes de acuerdo a las especificaciones de los clientes y ya instalado se realiza la configuración propia del producto.



Ilustración 7. Instalación de software

- h. **Mantenimiento:** para esta parte se debe de brindar ayuda al cliente en cuanto a la solución de algún problema no grave del software esto entraría dentro del primer nivel para esta parte se podrían dejar una serie de recomendaciones o quizás algunas demostraciones de posibles soluciones por escrito al usuario, para el segundo nivel es cuando quizás el producto no se adapte al cliente y hay que realizar un cambio directamente, dependerá de la documentación desarrollada del producto dejada al cliente/usuario.



*Ilustración 8. Mantenimiento de software*

- i. **Administración:** en este punto se trata que el software desarrollado y que se está empleando sea capaz de resistir cambios, esto realizando una planificación a futuro de lo que puede esperarse con el producto desarrollado y que esto a su vez pueda involucrar a ambas partes clientes/usuarios y desarrolladores para que esto sea una oportunidad de crecimiento.



*Ilustración 9. Administración del software*

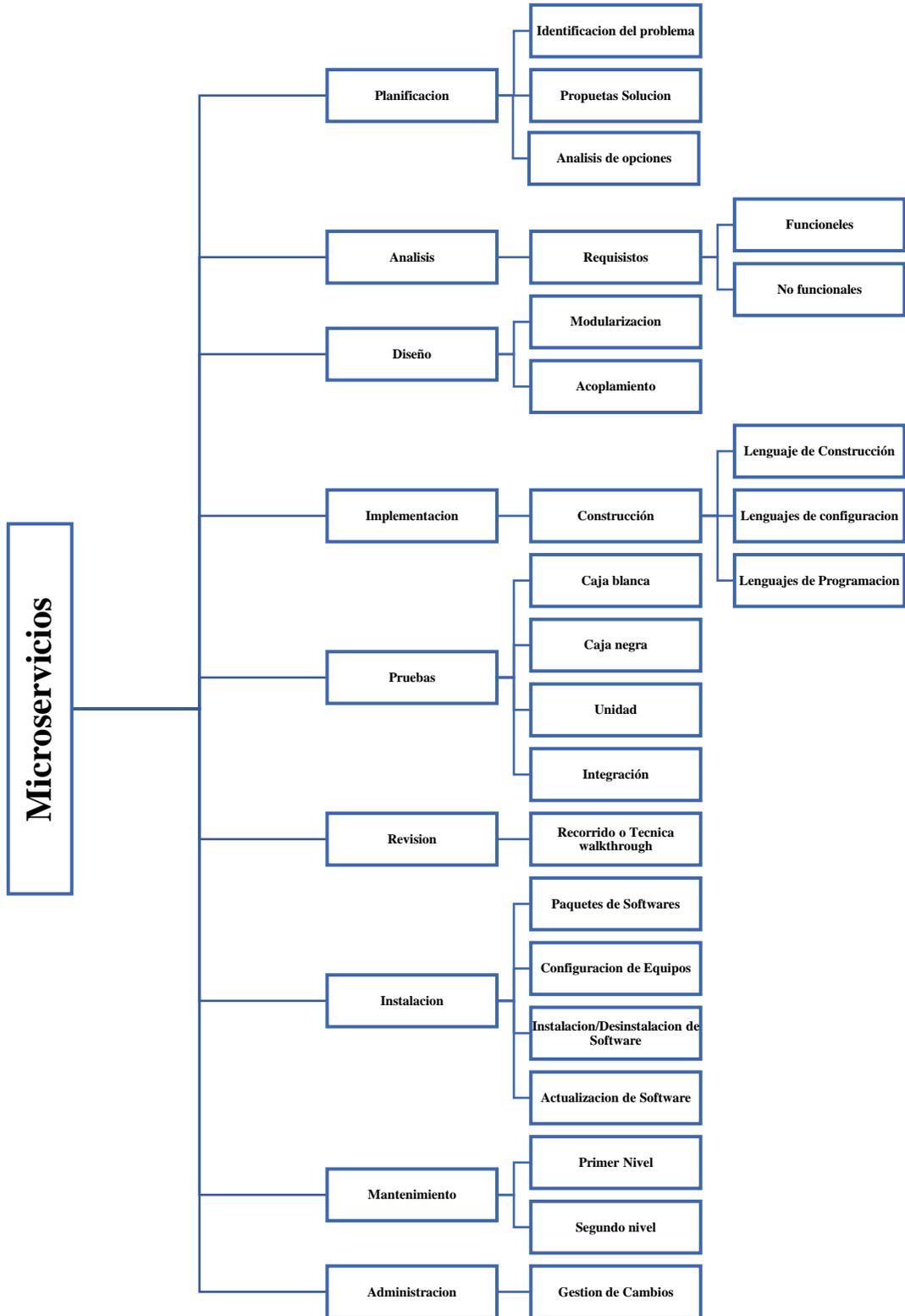


Figura 13. Microservicios y ciclo de vida de desarrollo

## 6.8.5. Segundo paso Valores de funcionalidad de microservicios

En la Figura 14 se muestra el diagrama del atributo de Funcionabilidad y sus subatributos con los valores para medir la calidad del desarrollo de los microservicios, así como la tabla de valores para cada uno con el valor máximo, valor medio y el valor mínimo. Estos valores de acuerdo a la clasificación propia de un valor de 100% como total de la funcionalidad como atributo que debe contener el desarrollo de microservicios y los subatributos con un 20% estimado a cada uno de ellos, y de ahí desglosamos cada elemento conteniente para darle un valor. Para cada atributo de calidad se evalúa la adecuación, el cumplimiento funcional, idoneidad, corrección, interoperabilidad, conformidad y seguridad de acceso, si cumple con todos estos elementos podemos indicar que nuestro microservicio estará funcionando y ofreciendo un mejor servicio. Se puede observar también que se enmarcó la parte de seguridad, ya que se está considerando como parte fundamental en las pruebas, instalación e implementación, puesto que aplicando la seguridad estamos garantizando que nuestro producto será confiable, podemos anticiparnos a ciertos problemas a la hora de estar desarrollando un producto, así como el verificar que se lleven a cabo cada etapa de forma correcta y desacuerdo al ciclo de vida del software.

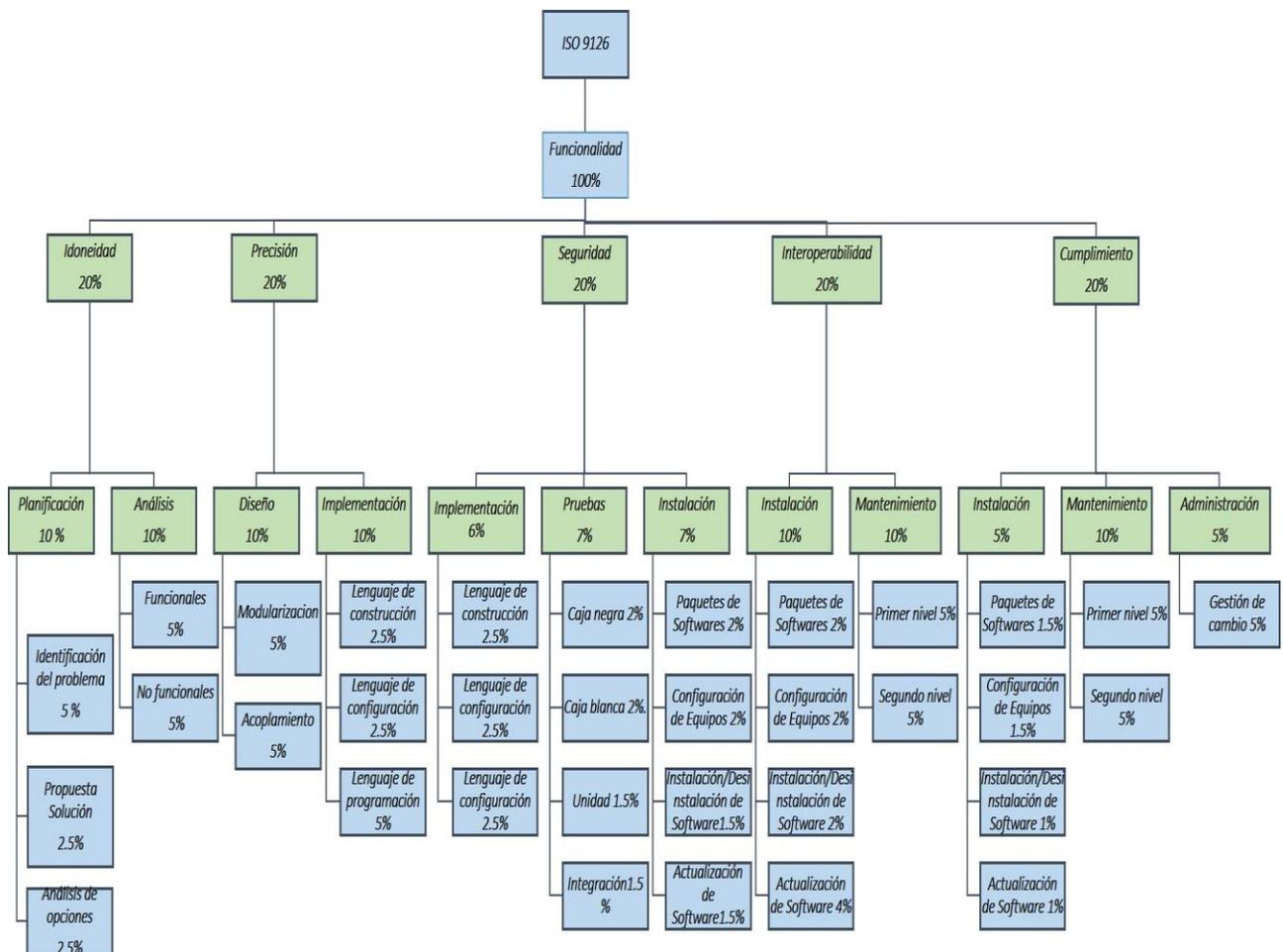


Figura 14. Diagrama de valores de funcionalidad de microservicios

En la Tabla 14 se muestran los valores antes mencionados en la Figura 14 para medir la calidad en el desarrollo de un microservicio, este sea un check lis (Tabla 15) que permita evaluar los elementos contenidos en su elaboración. Para realizar la medición (Tabla 14) de los atributos usaremos la siguiente operación para medir su valor de cada elemento, estos valores son propuestas propias.

$$F = \text{Idn} + \text{Pre} + \text{Seg} + \text{Int} + \text{Cum} = 100\%$$

$$F = 20 + 20 + 20 + 20 + 20 = 100 \%$$

**Donde:**

**Idn**=Idoneidad

**Pre**=Precisión

**Seg**=Seguridad

**Int**= Interoperabilidad

**Cum**= Cumplimiento

Si el microservicio cuenta con todos los elementos especificados dentro de la tabla de valores, entonces se podrá realizar esta operación donde el mejor valor será el 100% que contiene todos los elementos, el valor medio será si contiene el 50% de los elementos y el valor más bajo será 25% donde los elementos no son suficiente para que este microservicio sea de calidad, en esos casos no sería un producto que pueda cumplir con las expectativas de un usuario o cliente.

*Tabla 14. Valores de Funcionalidad para medir los Microservicios*

<b>Funcionabilidad Microservicios y ciclo de vida</b>						
<b>Funcionabilidad 100%</b>		<b>Ciclo de Vida del Software</b>	<b>Valor Máximo</b>	<b>Valor Medio</b>	<b>Valor Mínimo</b>	
	<b>Idoneidad 20%</b>		Planificación	10%	5	2.5
			Análisis	10%	5	2.5
	<b>Precisión 20%</b>		Diseño	10%	5	2.5
			Implementación	10%	5	2.5
	<b>Seguridad 20%</b>		Implementación	7%	3.5	1.75
			Pruebas	6%	3	1.5
			Instalación	7%	3.5	1.75
	<b>Interoperabilidad 20%</b>		Instalación	10%	5	2.5
			Mantenimiento	10%	5	2.5
	<b>Cumplimiento 20%</b>		Instalación	5%	2.5	1.25
			Mantenimiento	10%	5	2.5
			Administración	5%	2.5	1.25
<b>Total</b>			100%	50%	25%	

Tabla 15. Check List-Formato de medición de Microservicios

<b>FUNCIONABILIDAD MICROSERVICIOS Y CICLO DE VIDA</b>				
<b>Funcionabilidad 100%</b>		<b>Ciclo de Vida del Software</b>	<b>Valor Esperado</b>	<b>Valor Real</b>
	<b>Idoneidad 20%</b>	Planificación	10pts	
		Análisis	10pts	
	<b>Precisión 20%</b>	Diseño	10pts	
		Implementación	10pts	
	<b>Seguridad 20%</b>	Implementación	7pts	
		Pruebas	6pts	
		Instalación	7pts	
	<b>Interoperabilidad 20%</b>	Instalación	10pts	
		Mantenimiento	10pts	
	<b>Cumplimiento 20%</b>	Instalación	5pts	
		Mantenimiento	10pts	
		Administración	5pts	
<b>Total</b>		100%		

# Capítulo 7.

# Conclusión

## 7.1. Conclusión

De acuerdo a los resultados presentados en el Capítulo 5 se concluye que se cumplió con el objetivo general, mencionado en la sección 1.3.1. Objetivo general del Capítulo 1.

*“Realizar un estudio del estado actual de métricas y patrones en el área de seguridad para el desarrollo de microservicios.”*

Así mismo, se concluye que los objetivos específicos se cumplieron satisfactoriamente de acuerdo a lo siguiente:

- Se desarrolló un MS para métricas de seguridad en microservicios para evidenciar la información que existe actualmente sobre ello.
- Se obtuvieron conceptualizaciones de métricas y clasificaciones aplicadas a otras arquitecturas que en un futuro puedan ser aplicadas a los microservicios.
- Se realizó una clasificación de las métricas listadas en cada artículo de investigación, dividiéndolas en mecanismos enfocados a la calidad, Cloud native, orientados a objetos (Diagrama 1).
- Se desarrolló un segundo MS para patrones de seguridad en microservicios para evidenciar de igual forma la información actual de ellos.
- De los diferentes estudios obtenidos y analizados se obtuvo patrones enfocados a diferentes arquitecturas y enfocados a servicios en la nube, haciendo referencia a normas de calidad, utilizando la autenticación y autorización para la seguridad, la mayoría utilizando JWT para java, así como servicios de código abierto.
- Se realizó una clasificación de los patrones mencionados en cada investigación, dividiendo cada uno de los mecanismos enfocados en lo que son API Gateway, autenticación y autorización y estándares de autenticación (Diagrama 1).
- Se desarrolló una, guía que sirva como referencia para la elaboración de microservicios basándose en el ciclo de vida del software y la norma de calidad ISO 9126:200, para evaluar su funcionalidad y sus subatributos y permita mejorar el desarrollo de nuestros microservicios detectando los elementos que no se estaban considerando al desarrollarlo.

En esta investigación es un acercamiento a la identificación de mecanismos de seguridad que en un futuro pueden ser aplicados a los microservicios (MS) para una investigación más detallada. Se puede argumentar que aún los microservicios son temas nuevos que están evolucionando, los MS que se desarrollaron nos dieron un panorama de las métricas y

patrones de seguridad que se emplean en las arquitecturas de MS, sin embargo, son aún pruebas de software especializado, y de forma específica no existen mecanismos específicos para la seguridad en microservicio.

Los mecanismos que se emplean para los microservicios se basan y se adaptan de arquitecturas anteriores, como los orientados a objetos cuyas soluciones en cuestión de seguridad no necesariamente se pueden mapear de forma directa a una arquitectura de microservicios.

Por lo anterior, esto representa brechas de investigación para futuros trabajos sobre los microservicios en cuanto a su funcionamiento, la seguridad y el desarrollo ya que se consideran, tanto en métricas como en patrones, y su aplicación directa a estos; y se puedan realizar comprobaciones prácticas en la industria, permitiendo que en un futuro se puedan definir métricas y patrones en específicos para los microservicios que permitan evaluar su calidad.

# Referencias

## 7.2. Referencias

1. Alicante, U. d. (2022). *Modelo vista controlador (MVC)*. Obtenido de *Modelo vista controlador (MVC)*: <https://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-vista-controlador-mvc.html>
2. Almorsy, M. G. (2013). *Automated Software Architecture Security Risk Analysis using Formalized Signatures*. 662-671.
3. Anelis Pereira Vale\*, A. P. (2019). *Security Mechanisms Used in Microservices-Based Systems: A Systematic Mapping*. 2019 XLV Latin American Computing Conference (CLEI). Panama, Panama: Universidad Técnica Federico Santa María, Valparaíso, Chile.
4. Astudillo, M. G. (2018). *Actual Use of Architectural Patterns in Microservices-based Open Source Projects*. 2018 25th Asia-Pacific Software Engineering Conference (APSEC). Nara, Japón.
5. Azure, M. (s.f.). *¿Qué es DevOps? Obtenido de ¿Qué es DevOps?:* <https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-devops/>
6. Bandar Alshammari, C. F. (2010). *Security Metrics for Object-Oriented Designs*. 21st Australian Software Engineering Conference (págs. 1-10). Australian: IEEE.
7. Barabanov A., M. D. (2020). *Authentication and Authorization in Microservice-Based Systems: Survey of Architecture Patterns*. *Cybersecurity issues*, 32-43.
8. Birov, T. T. (2018). *SECURITY PATTERNS FOR MICROSERVICES LOCATED ON DIFFERENT VENDORS*. © *Journal of the Technical University - Sofia "Fundamental Sciences and Applications"*, 105-108.
9. Centre, L. S. (2011). *Descriptive Information (DI) Metric Thresholds*.
10. Chondamrongkul Nacha & Jing Sun, a. I. (2020.). *Automated Security Analysis for Microservice Architecture*. *IEEE International Conference on Software Architecture Companion (ICSA-C)*, *IEEE Explore*, 79-82,.
11. Chris, R. (2019). *Microservices Patterns*. Shelter Island, NY: MANNING.
12. Content, R. (2021). *¿Qué es Bootstrap? Obtenido de ¿Qué es Bootstrap?:* <https://rockcontent.com/es/blog/bootstrap/>

13. Corporation's, M. (2022). *CSS|MDN. Obtenido de CSS:*  
<https://developer.mozilla.org/es/docs/Web/CSS>
14. Corporation's, M. (2022). *HTML: Lenguaje de etiquetas de hipertexto. Obtenido de HTML: Lenguaje de etiquetas de hipertexto:*  
<https://developer.mozilla.org/es/docs/Web/HTML>
15. Foundation., P. S. ( 2001-2022). *wsgiref-Utilidades WSGI e implementación de referencia. Obtenido de wsgiref-Utilidades WSGI e implementación de referencia:*  
<https://docs.python.org/es/3/library/wsgiref.html#:~:text=La%20Interfaz%20de%20Pasarela%20del,WSGI%20con%20diferentes%20servidores%20web>.
16. Group, C. ©.-2. (2022). *PHP. Obtenido de PHP:¿Qué puede hacer PHP?:*  
<https://www.php.net/manual/es/intro-whatcando.php>
17. Guerron, X. A.-D.-L.-D. (2020). *A Taxonomy of Quality Metrics for Cloud Services. IEEE Access, 131-461.*
18. Hat, R. (2017). *¿Qué es una API? Obtenido de ¿Qué es una API?:*  
<https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces>
19. hectorivanortegamartinez. (2020). *¿QUE ES WAMP SERVER? Obtenido de ¿QUE ES WAMP SERVER?:* <https://abdolflores.wordpress.com/2013/11/19/que-es-wamp-server/>
20. Justus Bogner, J. F. (2021). *Industry practices and challenges for the evolvability assurance of microservices. Empirical Software Engineering, 1-39.*
21. Kitchenham, B. a. (2007). *Guidelines for Performing Systematic Literature Reviews in Software Engineering Technical Report EBSE 2007-001. Keele University and Durham University Joint Report .*
22. Makrushin, B. A. (2021). *Security Audit Logging in Microservice-Based Systems: Survey of Architecture Patterns. problemas de ciberseguridad, 71-80.*
23. Martin, F. (25 de Marzo de 2014). *Microservices. Recuperado el 22 de Agosto de 2021, de* <https://martinfowler.com/articles/microservices.html>
24. Mauricio Diéguez, C. C. (2011). *De la Gestión de Seguridad en el Ciclo de Vida del Software . Universidad de la Frontera (UFRO), 1-5.*
25. Mohamed Almorsy, J. G. (2013). *Automated Software Architecture Security Risk Automated Software Architecture Security Risk. 2013 35th International Conference on Software Engineering (ICSE), (págs. 662-671). San Francisco, CA, USA.*
26. Mohammad S. Aslanpour, S. S. (2020). *Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research. Elsevier, 1-20.*

27. Muhammad Waseema, P. L. (2021). *Design, Monitoring, and Testing of Microservices Systems: The Practitioners' Perspective*. Preprint submitted to *Journal of Systems and Software*, 1-75.
28. Nancy Loja Mora, F. R. (2017). *Quality metrics for web application development*. *ARJE*, 1-26.
29. Parnas, D. (1972). *On the Criteria to Be Used in Decomposing Systems Into Modules*. *Communications of the ACM*.
30. Patrik Berander, L.-O. D. (2005). *Software quality attributes and trade-offs*. Blekinge Institute of Technology.
31. Paulo Anselmo da Mota Silveira Neto, I. d. (2010). *A regression testing approach for software product lines architectures*. *Fourth Brazilian Symposium on Software Components, Architectures and Reuse (págs. 41-50)*. Brazilian: IEEE.
32. PearlBrereton. (2007). *Lessons from applying the systematic literature review process within the software engineering domain*. *Journal of Systems and Software*, 571-583. *Obtenido de Lessons from applying the systematic literature review process within the software engineering domain*.
33. Pendleton, M. G.-l. (2016). *A Survey on Systems Security Metrics*.
34. Petersen Kai, e. a. (2008). *Systematic Mapping Studies in Software Engineering*. *splc*, 1-10.
35. Rios, J. (2020). *Monografías*. *Obtenido de Seguridad Informática: <https://www.monografias.com/trabajos82/la-seguridad-informatica/la-seguridad-informatica2>*
36. Rodríguez, S. S. (2012). *Ingeniería del Software un enfoque desde la guía SWEBOK*. Madrid, España: Alfaomega.
37. Roldan Martínez David, P. J. (2018). *Microservicios un Enfoque Integrado*. Madrid: *rama*.
38. Rudrabhatla, C. K. (2020). *Security Design Patterns in Distributed Microservice Architecture*. *International Journal of Computer Science and Information Security (IJCSIS)*, 72-75.
39. Sandoval, R. (2009). *A Security Metrics Taxonomization Model for Software-Intensive Systems*. 197-206.
40. Sebastiano Panichella, S. P. (2021). *Structural Coupling for Microservices*. *11th International Conference on Cloud Computing and Services Science, CLOSER 2021*.
41. Services, A. W. (2022). *¿Qué es Python?|guía python para principiantes en la nube| AWS*. *Obtenido de ¿Qué es Python?: <https://aws.amazon.com/es/what-is/python/>*

42. SIMON, T. F. (2021). *A Comparison Between Microservice Security Design Patterns for Authentication and Authorization Flows*. STOCKHOLM SWEDEN: KTH ROYAL INSTITUTE OF TECHNOLOGY SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE.
43. sistemas.com. (s.f.). *Sistemas . Obtenido de Definición de Encapsulamiento:*  
<https://sistemas.com/encapsulamiento.php>
44. Stocke Mirko, e. a. (2018). *Interface Quality Patterns — Communicating and Improving the*. EuroPLoP '18: Actas de la 23a Conferencia europea sobre lenguajes de patrones de programas, (págs. 1-16). Irsee, Germany.
45. Streeton, R. C. (2004). *Researching the researchers: using a snowballing technique*. Nurse researcher. *Researching the researchers: using a snowballing technique*. Nurse researcher, 35-47.
46. Telecomunicaciones, C. I. (2006). *Info@citel*. Obtenido de Info@citel:  
[https://www.oas.org/en/citel/infocitel/2006/junio/seguridad\\_e.asp#:~:text=Autenticaci%C3%B3n%20es%20el%20proceso%20que,usuario%20es%20quien%20dice%20ser](https://www.oas.org/en/citel/infocitel/2006/junio/seguridad_e.asp#:~:text=Autenticaci%C3%B3n%20es%20el%20proceso%20que,usuario%20es%20quien%20dice%20ser).
47. Thomas Engel, M. L. (2018). *Evaluation of Microservice Architectures:A Metric and Tool-Based Approach*. Springer International Publishing, 74-89.
48. Tihomir, T. &. (2018). *SECURITY PATTERNS FOR MICROSERVICE DATA MANAGEMENT*. Conference Proceedings, YOUNG RESEARCHERS (pág. 575). Sofia University “St Kliment Ohridski.
49. Torkura Kennedy A, e. a. (2017). *Leveraging Cloud Native Design Patterns for*. IEEE International Conference on Smart Cloud (págs. 90-97). Potsdam, Germany: Hasso Plattner Institute, University of Potsdam.
50. Verity. (23 de junio de 2021). *Todo sobre ISO/IEC 9126: 2001 y su importancia en las empresas*. Obtenido de La ISO/IEC 9126: 2001: Características de la calidad de software:  
<https://www.verity.cl/blog/que-es-norma-iso-iec-9126-2001#:~:text=El%20est%C3%A1ndar%20ISO%2D9126%20establece,trav%C3%A9s%20de%20un%20conjunto%20de>
51. w3c. (11 de febrero de 2004). *Servicio web*. Obtenido de Servicio web:  
<https://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/#webservice>
52. Westreiche, G. (25 de agosto de 2020). *Economipedia.com*. Obtenido de Planificación. Economipedia.com:  
<https://economipedia.com/definiciones/planificacion.html#:~:text=La%20planificaci%C3%B3n%20es%20la%20estructuraci%C3%B3n,seguir%20para%20alcanzar%20ciertas%20metas>.
53. Wieringa Roel, M. N. (2006). *Requirements engineering paper classification and evaluation criteria: a proposal and a discussion*. Conferencia de Ingeniería de Requisitos (RE) del IEEE, 102-107.

# Capítulo 8.

## Anexos

### 8.1. Ejemplo de Microservicio

#### a. Introducción

La especificación de requisitos de software (ERS) se utiliza para describir completamente el comportamiento del sistema que se va a desarrollar. Aquí se van a describir los requisitos funcionales y las características de diseño de interfaz y de uso que tendrá nuestro servicio. El presente ejemplo es tomado de la nube para conocer el funcionamiento de un microservicio con el contexto de una agencia de viajes con el funcionamiento de tres módulos principales.

#### b. Objetivo

El principal objetivo es desarrollar un ejemplo de un microservicio y conocer su funcionamiento, así como saber en donde se están implementando la seguridad, la realización de este proyecto será desarrollar un sitio web para la recomendación de destinos turísticos que permita:

- Ofrecer al usuario la posibilidad de filtrar búsquedas según sus necesidades
- Consultar sus búsquedas
- Recomendar viajes a los destinos seleccionados

#### c. Descripción general

El sistema proporciona al usuario una amplia gama de destinos según sus gustos, cualidades o necesidades. La aplicación estará alojada en un servidor web (000 webhost). La base de datos utilizada será MySQL por ser esta de uso libre, una base de datos robusta y de gran almacenamiento que nos permitirá almacenar toda la información perteneciente a los destinos, las últimas búsquedas, así como los usuarios registrados.

Los usuarios que accedan al sitio web serán personas con una conexión a internet y que tengan las mínimas nociones sobre cómo funciona la red. En nuestro sitio web existen dos tipos de usuarios:

- **Usuarios registrados:** Son los usuarios que se han registrado en el sitio web y, por tanto, tienen una cuenta de usuario. Pueden consultar destinos, ver los destinos aleatorios y recuperar sus últimas búsquedas.

• **Usuario administrador:** Es el usuario que se encarga de gestionar y mantener el sitio web. Pueden hacer lo mismo que cualquier usuario registrado en cuanto a consultas y búsquedas. Gozan de más privilegios que los usuarios registrados y se les permite actualizar y ampliar los destinos desde el sitio web.

#### d. Limitaciones

La finalidad de este proyecto es puramente académica, por lo tanto, el servicio presentado puede no corresponder con la realidad o ser más limitado de lo que serían en cualquier sistema a la disposición del público. Puede que los datos no sean exactos o que aparezcan menos resultados debido al tamaño de la base de datos.

#### e. Estructura del ejemplo

#### f. Interfaz de usuario

En la parte del encabezado (Figura 15) contiene el título y el menú de nuestro sitio donde se puede observar el nombre del sitio con el eslogan de la agencia de viajes, el menú contiene las opciones de vuelos.

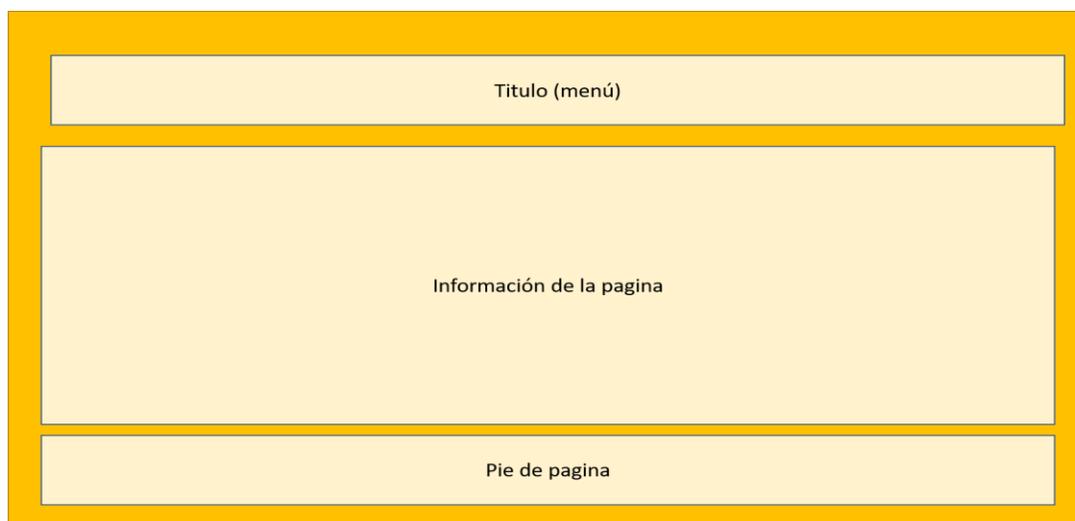


Figura 15. Pantalla de inicio del sitio web.

En la Figura 16. Se muestra el menú y las opciones, así como vuelos, hoteles y opción de alquilar autos. Este trabajo se centró en la parte de vuelos.



Figura 16. Diseño de destino vuelos

### g. Diagrama de caso de usos

Se desarrolló un diagrama de uso, la mejor comprensión del sistema y sus elementos que contiene, como se muestra a continuación (Figura 17).

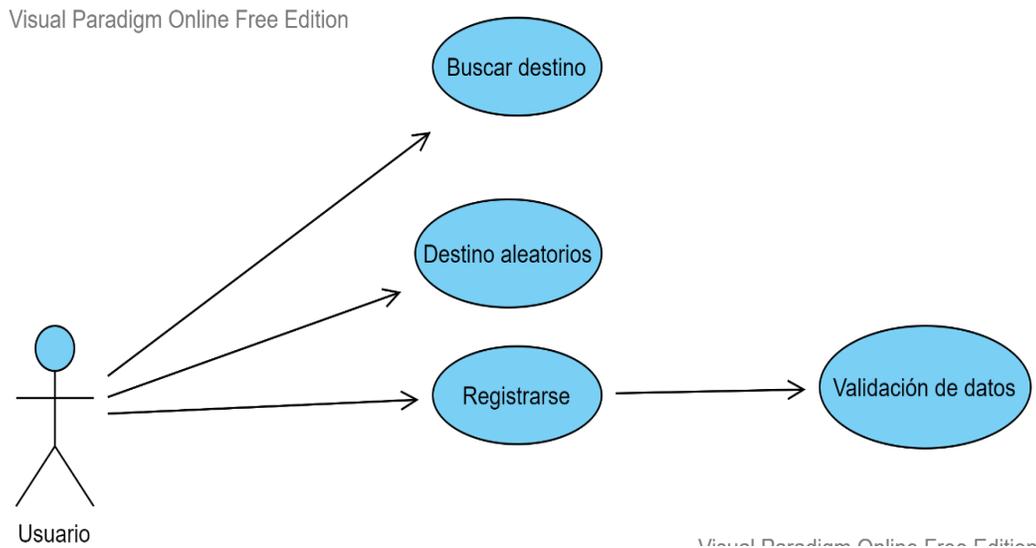


Figura 17. Diagrama de clases de selección de destinos

1	ID	C1		
2	Nombre del caso de uso:	DESTINOS		
3	Autor de creación		Autor última modificación:	
4	Fecha de creación:	01/11/2021	Fecha de modificación	17/11/2021
5	Actores	Usuario		
6	Descripción	El sistema muestra al usuario la página principal la opción de vuelos para que realice su selección de búsqueda.		
7	Precondición	<ol style="list-style-type: none"> <li>1. Tener un dispositivo móvil o computadora</li> <li>2. Tener conexión a internet</li> <li>3. Tener el navegador abierto</li> </ol>		

8	Postcondición	<ol style="list-style-type: none"> <li>1. El usuario visualiza el listado de destinos que puede elegir y buscar.</li> <li>2. El usuario elija fechas correctamente para que pueda buscar.</li> </ol>
9	Escenario principal de éxito	<ol style="list-style-type: none"> <li>1. El usuario ingrese la URL correctamente en el navegados</li> <li>2. El sistema muestre la página principal de los vuelos</li> <li>3. El usuario elija fechas correctas</li> </ol>
10	Escenario alternativo	
11	Escenario de fracaso	<ol style="list-style-type: none"> <li>1. El usuario no ingrese la URL correctamente</li> <li>2. El navegador muestre un mensaje de error</li> <li>3. El sitio muestre mensajes de error</li> </ol>
12	Prioridad	Alta

Usuario registrado: el actor usuario es un invitado que se ha registrado para que pueda acceder a las acciones como:

- ✚ Acceso al sistema
- ✚ Guardar todas las búsquedas realizadas
- ✚ Y revisar las búsquedas

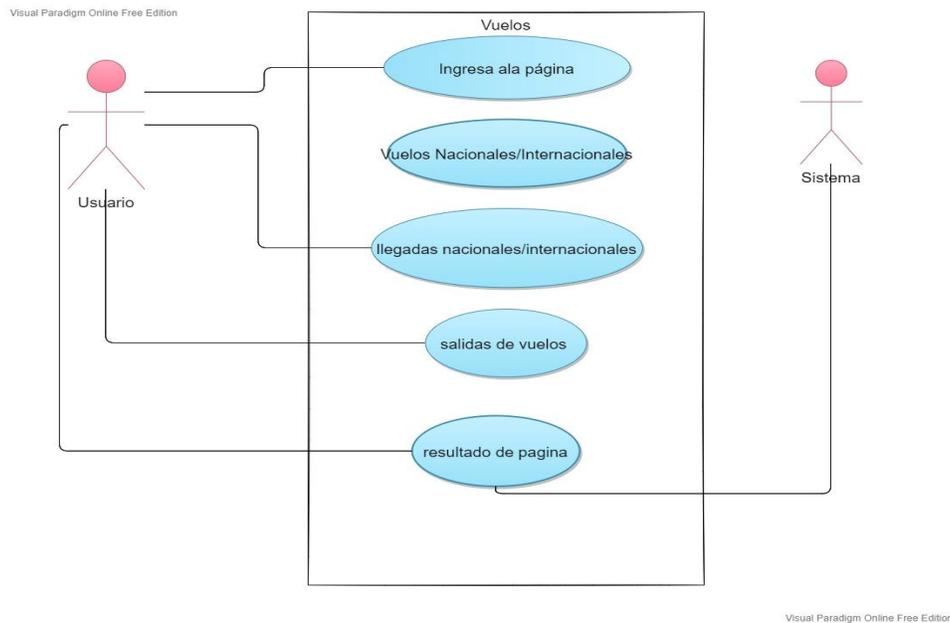


Figura 18. Caso de uso general del sitio web

1	ID	C2		
2	Nombre del caso de uso:	Sitio principal		
3	Autor de creación		Autor última modificación:	
4	Fecha de creación:	01/11/2021	Fecha de modificación	17/11/2021
5	Actores	Usuario		
6	Descripción	El sistema muestra al usuario la página principal para que pueda seleccionar del menú la que el crea.		
7	Precondición	<ol style="list-style-type: none"> <li>4. Tener un dispositivo móvil o computadora</li> <li>5. Tener conexión a internet</li> <li>6. Tener el navegador abierto</li> </ol>		
8	Postcondición	3. El usuario visualiza el listado de destinos que puede elegir y buscar		

9	Escenario principal de éxito	4. El usuario ingrese la URL correctamente en el navegados 5. El sistema muestre la página principal.
10	Escenario alterno	
11	Escenario de fracaso	4. El usuario no ingrese la URL correctamente 5. El navegador muestre un mensaje de error 6. El sitio muestre mensajes de error
12	Prioridad	Alta

El usuario registrado puede acceder a todas las opciones dentro del sistema e interactuar con el sistema y revisar los resultados, de manera general tiene acceso al sitio para revisar y reservar.

#### h. Modelo de negocios BPMN

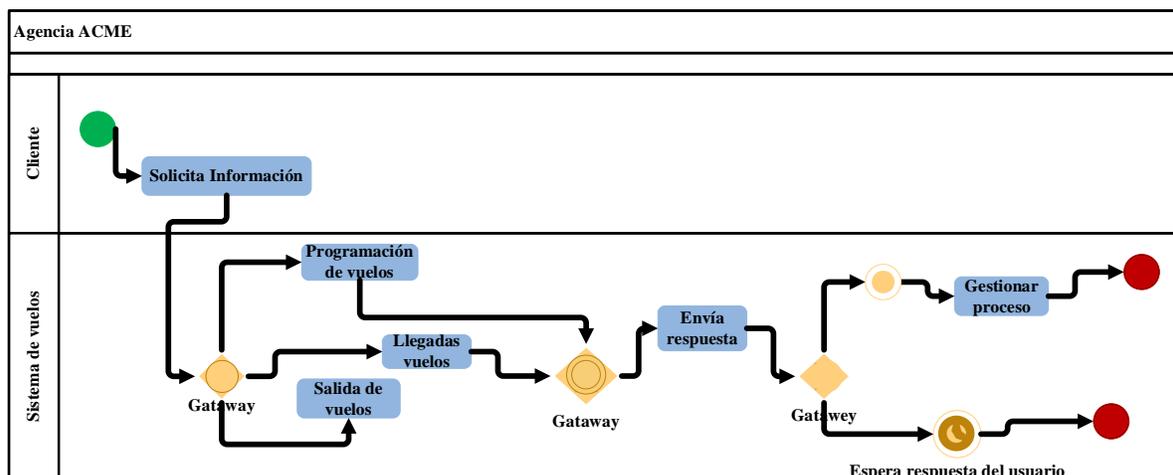


Figura 19. Diagrama BPMN de destinos de vuelos

En este diagrama (Figura 19) se muestra el modelado del proceso que se lleva para elegir un vuelo o un destino de viaje primero el cliente entra al sitio Web, selecciona la opción que desea realizar, selecciona el destino, y las opciones que requiere esa sección, como son fechas de salida y regreso destinos y realizar la búsqueda

1. El cliente selecciona el sitio
2. El cliente selecciona la opción de vuelos
3. El cliente selecciona el origen
4. El cliente selecciona el destino
5. El sitio solicita la fecha de salida
6. El sitio solicita la fecha de llegada
7. El sitio solicita la hora
8. El sitio envía respuesta al usuario

#### i. Diagrama de Secuencias

El usuario (Figura 20) realiza una petición al sistema mediante la interfaz, la solicitud se envía al sistema donde verifica el servicio y la información, el sistema envía un mensaje para solicitar información al cliente para seguir con la solicitud de vuelos.

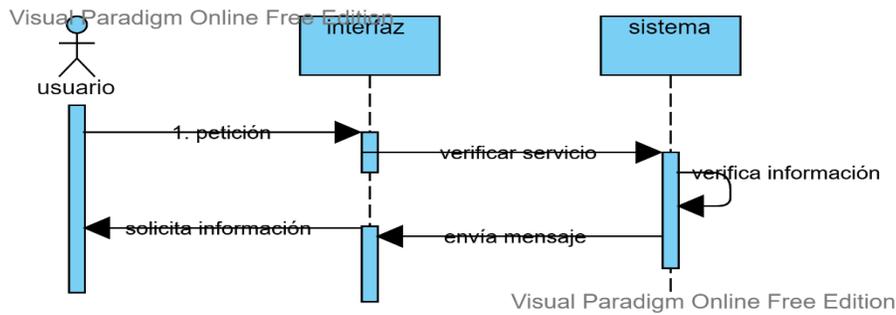


Figura 20. Diagrama de secuencia de vuelos

### j. Diagrama Despliegue

El diagrama de despliegue muestra la interacción del sitio web con el servidor y el navegador del cliente, de cómo interactúan el dispositivo del cliente accede al sitio Web, este sitio recibe la petición para enviarle al servidor que envía una respuesta al sitio donde indica que está en comunicación y el usuario realiza peticiones al sitio para buscar vuelos o reservas.

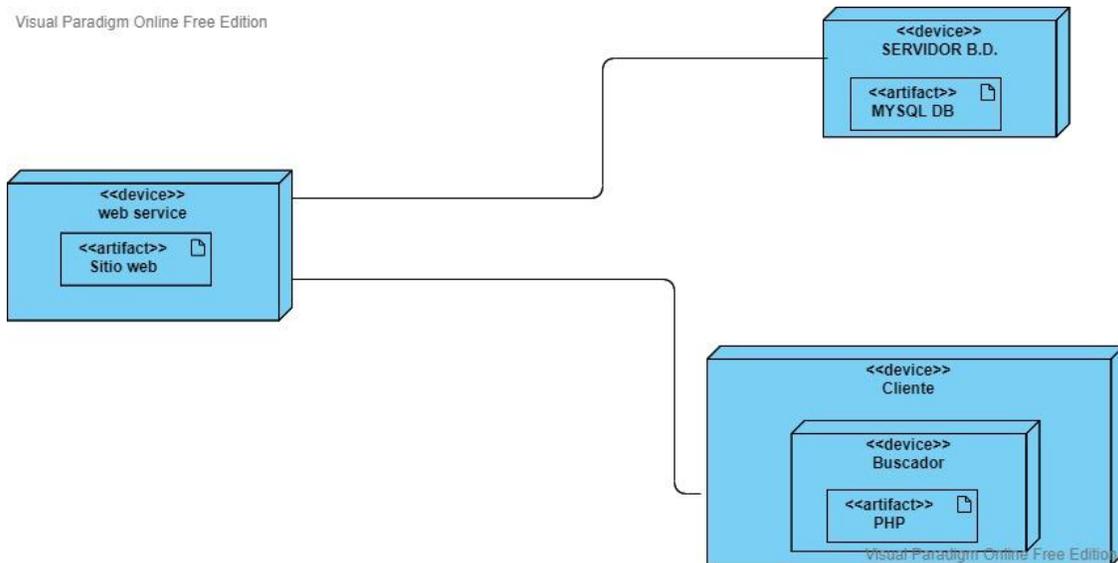


Figura 21. Diagrama Despliegue de opción de vuelos

### k. Diagrama ER de Agencia de Viajes ACME

En la capa de datos, es donde todos los vuelos y reservas registrados en el sistema se almacenarán a la base de datos, en este caso utilizaremos MySQL, lo cual se trabajará desde donde se encuentra alojado el sistema y el gestor de base de datos (Figura 22).

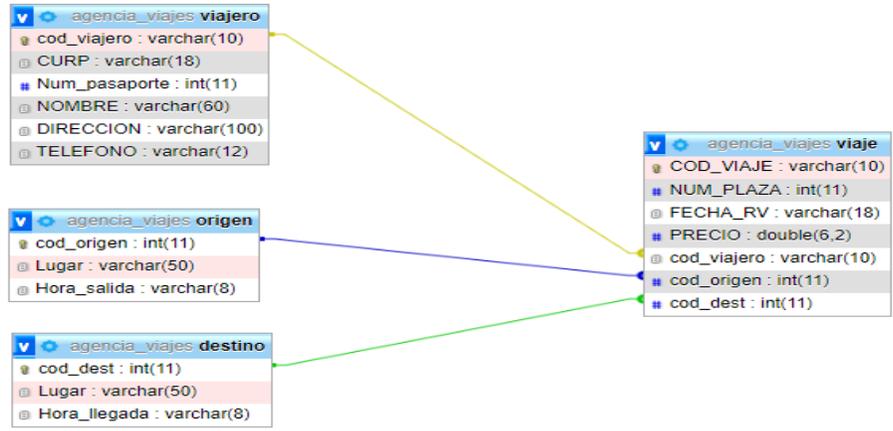


Figura 22. Estructura de Base de Datos

## ñ. Sitio web Viajes ACME

En este ejemplo se estructura un microservicio y visualizar donde se aplica la seguridad que se aplica en la parte de ingreso a la aplicación para poder realizar cambios y agregar elementos como administrador y en la parte de usuario puede ingresar a registrar un vuelo si está dado de alta en el sistema, a continuación, se muestra en la Figura 23 se muestra la pantalla principal del sitio web que contiene los elementos principales para el usuario.

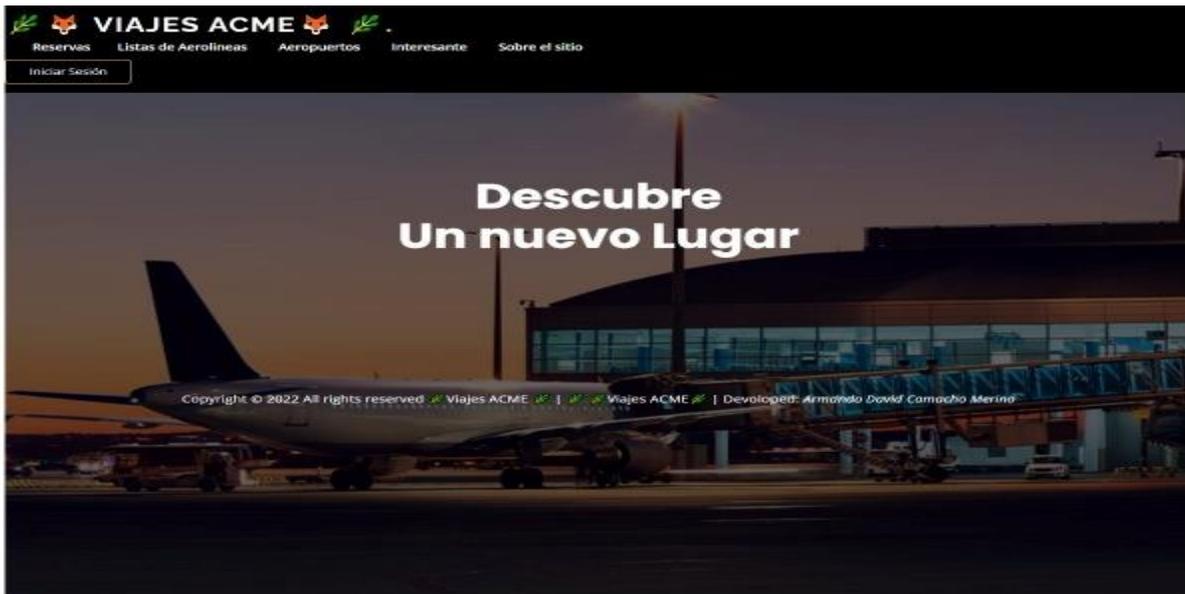


Figura 23. página principal de agencia de viajes

En la Figura 24 se muestra la pantalla de inicio de sesión, esta opción es cuando el usuario ya se encuentra registrado para que pueda acceder al sitio y existe la cuenta de administrador quien es el que tiene el acceso total del sistema.

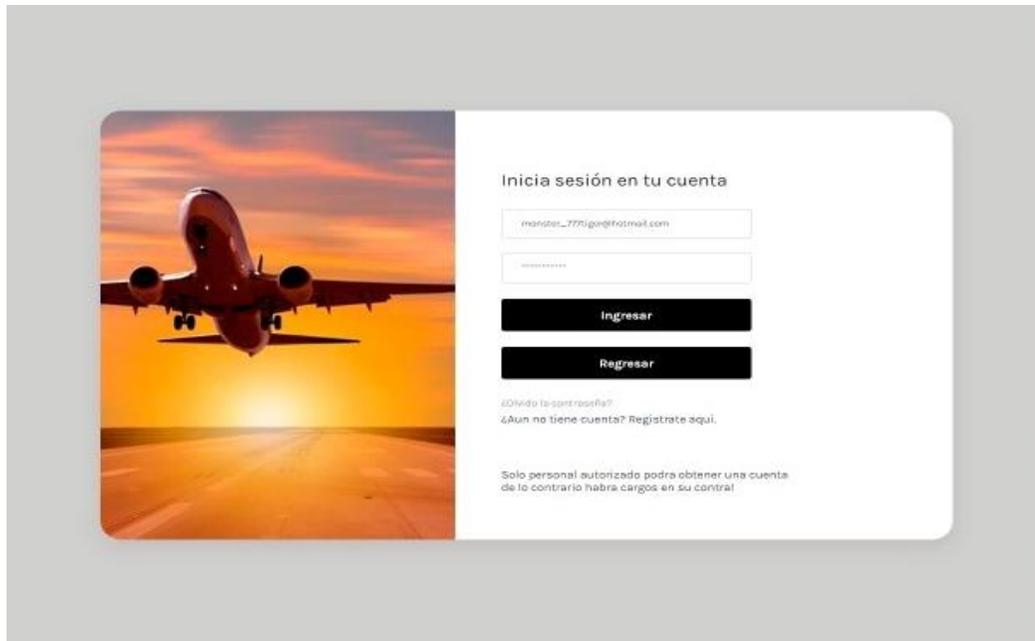


Figura 24. Página de registro de usuarios

En la Figura 25 se muestra la pantalla de bienvenida cuando el usuario ya se encuentra registrado y tiene alguno de los permisos, sea usuario del sistema o administrador del mismo.



Figura 25. Página de registro de datos

En la Figura 26 de se muestra el inicio para la reserva de vuelos por primera vez para que posterior a eso aparezcan las búsquedas relacionadas dependiendo el destino.

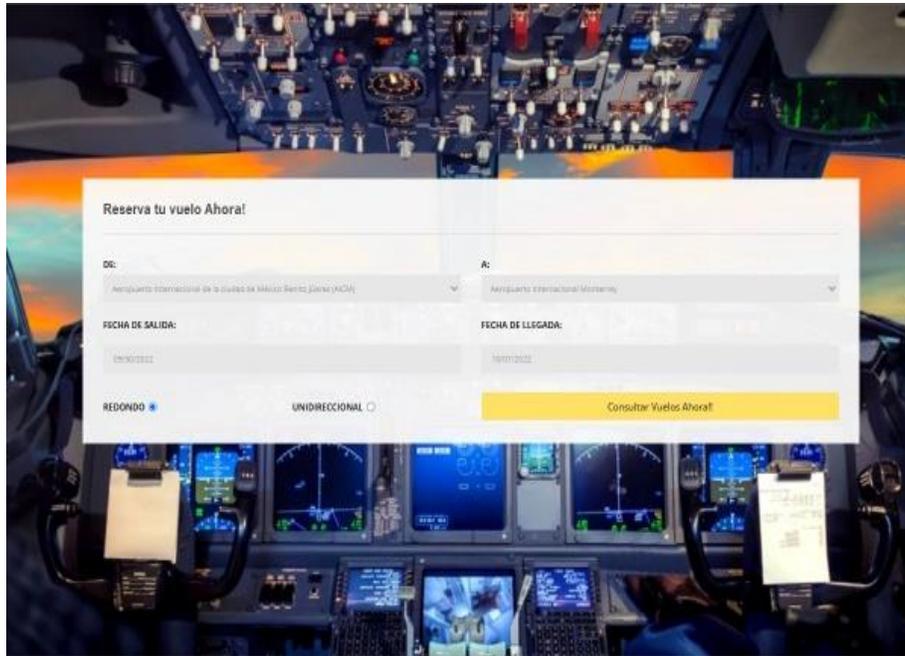


Figura 26. Reservación de vuelos

En la Figura 27 se muestra el formulario para crear vuelos, esto se realiza directamente el administrador del sistema quien tiene acceso total a este sistema para dar de alta cada vuelo.

Figura 27. Alta de vuelos

En la Figura 28 se muestra el hosting donde se almacena el servicio, para ello se realizó un inicio de sesión y se procedió a crear el espacio para el sistema el 000 webhost de forma gratuita, al igual se muestra el entorno donde se almacenan las carpetas del sistema (Figura 29).

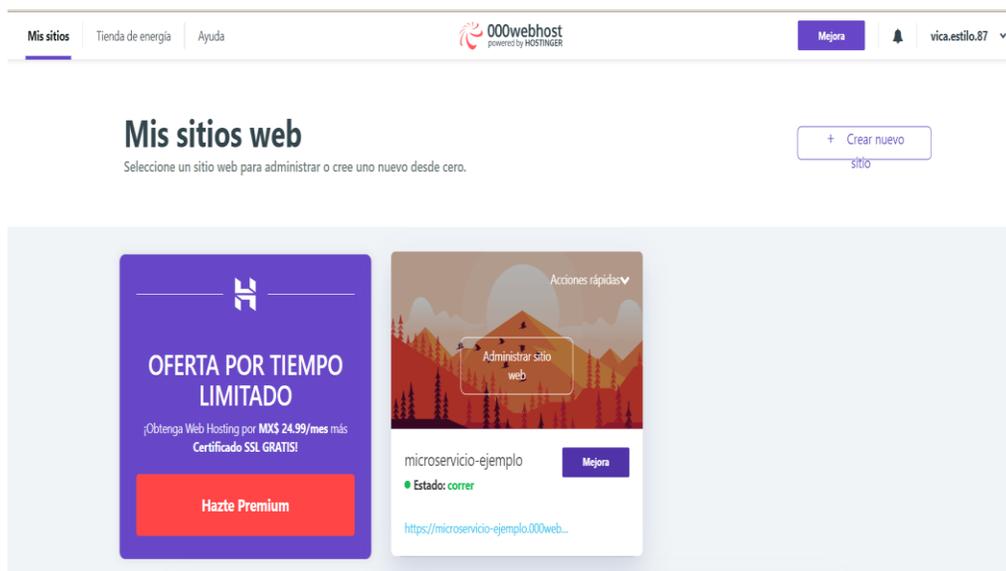


Figura 28. Página principal de alojamiento del sistema

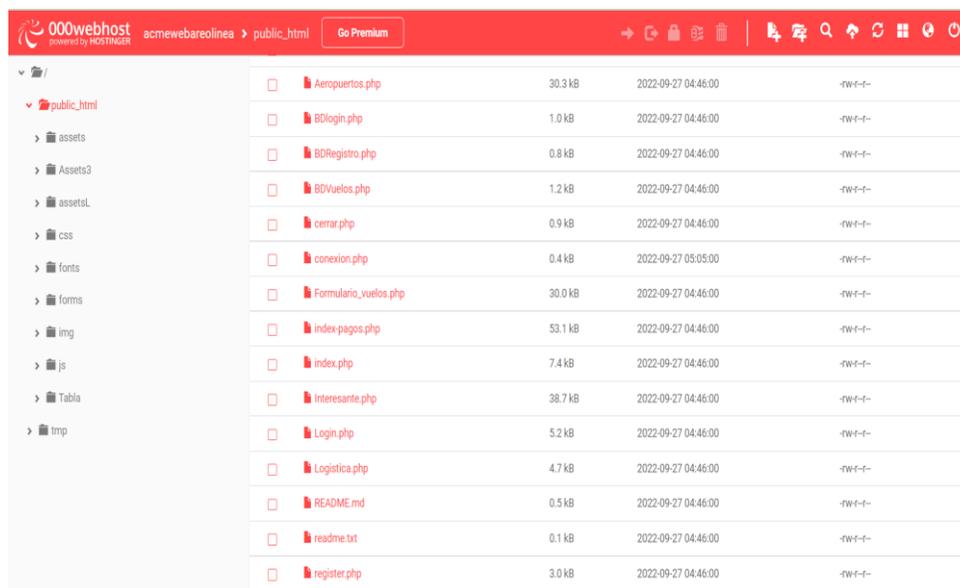


Figura 29. Entorno de desarrollo

A continuación, se muestra el fragmento de código que se utiliza para la conexión a la base de datos, está definido los valores de servidor, usuario, contraseña, así como el nombre de la base de datos.

```
<?php
define("DB_SERVER", "localhost");
define("DB_USER", "id19630285_rootadminbts");
define("DB_PASSWORD", 'j0TsS(wYSuu7Wpei');
define("DB_DATABASE", "id19630285_aeropuerto");
$conexion = mysqli_connect(DB_SERVER , DB_USER, DB_PASSWORD,
DB_DATABASE);
// Verificamos que haga la conexion
if($conexion){
    echo ("Conexión Exitosa");
}else{
    echo("Conexion Denegada");
}
?>
```

Se muestra también el fragmento de código que se utiliza para el inicio de sesión del sistema, si el usuario está registrado lo dejará iniciar sesión y si no enviara un mensaje.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta content="width=device-width, initial-scale=1.0" name="viewport">
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Iniciar sesión</title>
    <!-- Favicons -->
    <link href="assets/Imagenes que se pueden ocupar/airport-MMAA.png" rel="icon">
    <link href="assets/Imagenes que se pueden ocupar/protesis.jpg" rel="apple-touch-icon">
    <!--End Favicon-->
    <!-- Vendor CSS Files -->
    <link href="assets/vendor/aos/aos.css" rel="stylesheet">
    <link href="assets/vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
```

```

<link href="assets/vendor/bootstrap-icons/bootstrap-icons.css" rel="stylesheet">
<link href="assets/vendor/boxicons/css/boxicons.min.css" rel="stylesheet">
<link href="assets/vendor/glightbox/css/glightbox.min.css" rel="stylesheet">
<link href="assets/vendor/remixicon/remixicon.css" rel="stylesheet">
<link href="assets/vendor/swiper/swiper-bundle.min.css" rel="stylesheet">
<link href="https://fonts.googleapis.com/css?family=Karla:400,700&display=swap"
rel="stylesheet">
<link href="https://cdn.materialdesignicons.com/4.8.95/css/materialdesignicons.min.css">
<link href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
<link rel="stylesheet" href="assetsL/css/login.css">

```

## l. Herramientas de desarrollo

### m. Wapserver

WampServer es un entorno de desarrollo web para Windows con el que se puede crear aplicaciones web con Apache, PHP y bases de datos MySQL database. También incluye PHPMyAdmin y SQLiteManager para manejar tus bases de datos en un plis plas (hectorivanortegamartinez, 2020).

Características. Provee a los desarrolladores con los cuatro elementos necesarios para un servidor web: un Sistema Operativo (Window), un manejador de base de datos (MySQL), un software para servidor web (Apache) y un software de programación script Web (PHP (generalmente), Python o PERL), debiendo su nombre a dichas herramientas. Lo mejor de todo es que WAMPServer es completamente gratuito. WAMP incluye, además de las últimas versiones de Apache, PHP Y MySQL, versiones anteriores de las mismas, para el caso de que se quiera testear en un entorno de desarrollo particular (hectorivanortegamartinez, 2020).

### n. Bootstrap

Bootstrap 4 es la última versión de Bootstrap, el framework de CSS, HTML y JavaScript más popular, que nos permite desarrollar webs que se ajustan a cualquier resolución y dispositivo. Bootstrap es un kit de herramientas de código abierto para desarrollos web responsive con HTML, CSS y JavaScript. Con él puedes darle forma a tu sitio web a través del uso de sus librerías CSS y JavaScript. Incluye diferentes componentes: ventanas modales, menús, cuadros, botones, formularios... Es decir, los elementos que necesitas para maquetar tu página (Content, 2021).

### o. Css

Hojas de Estilo en Cascada (del inglés Cascading Style Sheets) o CSS es el lenguaje de estilos utilizado para describir la presentación de documentos HTML o XML (en-

US) (incluyendo varios lenguajes basados en XML como SVG, MathML o XHTML). CSS describe como debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios. CSS es uno de los lenguajes base de la Open Web y posee una especificación estandarizada por parte del W3C. Anteriormente, el desarrollo de varias partes de las especificaciones de CSS era realizado de manera sincrónica, lo que permitía el versionado de las recomendaciones. Probablemente habrás escuchado acerca de CSS1, CSS2.1, CSS3. Sin embargo, CSS4 nunca se ha lanzado como una versión oficial (Corporation's, CSS|MDN, 2022).

## p. Html

HTML (Lenguaje de Marcas de Hipertexto, del inglés HyperText Markup Language) es el componente más básico de la Web. Define el significado y la estructura del contenido web. Además de HTML, generalmente se utilizan otras tecnologías para describir la apariencia/presentación de una página web (CSS) o la funcionalidad/comportamiento (JavaScript). "Hipertexto" hace referencia a los enlaces que conectan páginas web entre sí, ya sea dentro de un único sitio web o entre sitios web. Los enlaces son un aspecto fundamental de la Web. Al subir contenido a Internet y vincularlo a las páginas creadas por otras personas, te conviertes en un participante activo en la «World Wide Web» (Red Informática Mundial) (Corporation's, 2022).

## q. Php

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. PHP es un lenguaje de programación de uso general que se adapta especialmente al desarrollo web (Group, 2022).

## 8.2. Ejemplo de un microservicio con Python

Se realizó a través del estándar WSGI (Web Server Gateway Interface), que es una interfaz estándar entre el servidor web y aplicaciones web escritas en Python.

Con una interfaz estándar es más sencillo usar una aplicación que soporte WSGI con diferentes servidores web (Foundation., 2001-2022).

Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML). Los desarrolladores utilizan Python porque es eficiente y fácil de aprender, además de que se puede ejecutar en muchas plataformas diferentes. El software Python se puede descargar gratis, se integra bien a todos los tipos de sistemas y aumenta la velocidad del desarrollo (Services, 2022).

Es por esto que en el mundo de la seguridad de la informática dicho lenguaje se ha convertido en referencia gracias a la gran cantidad de herramientas y librerías que existen para este tema.

Primero se instala Visual Studio Cod, Python y el navegador Google Chrome.



Figura 30. Herramientas de Desarrollo

1. En la figura 2 se muestra el diseño de WSGI.

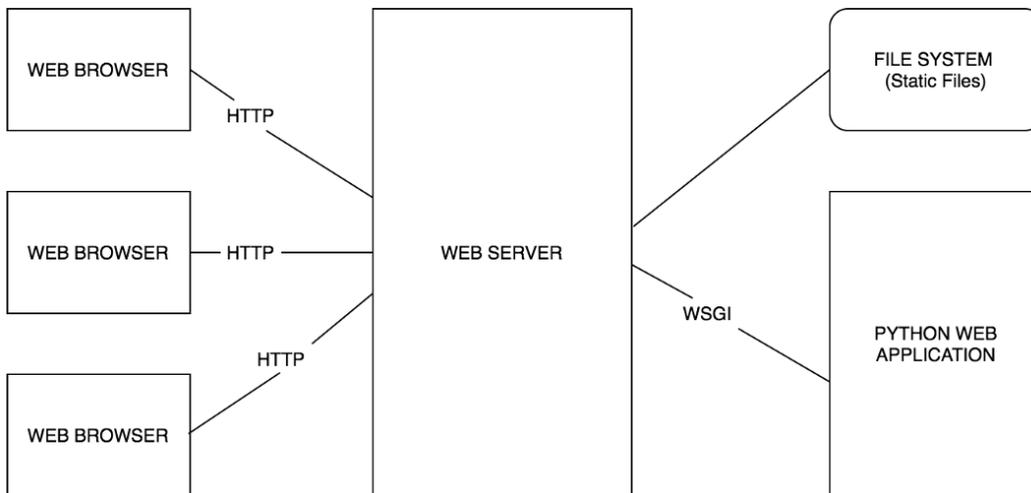
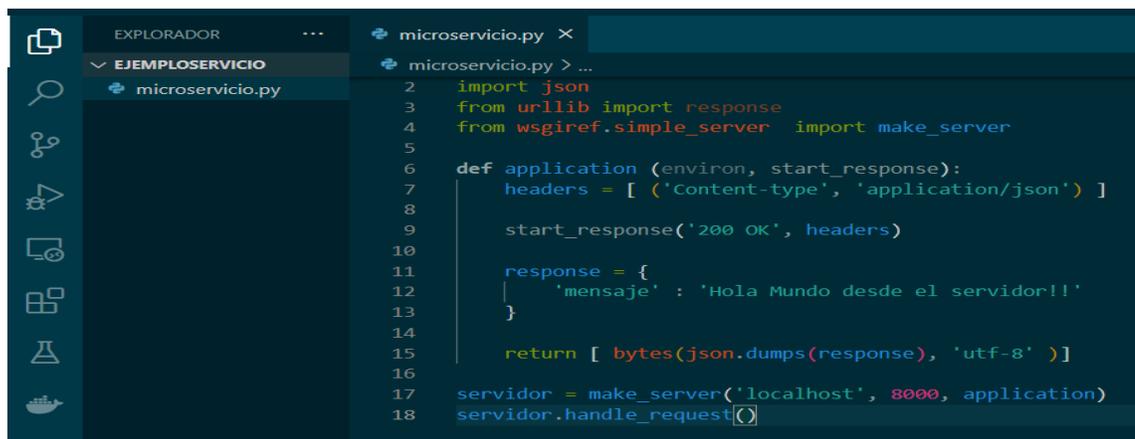


Figura 31. Estructura WSGI

2. Se codificó una aplicación para que responda al cliente con un mensaje, para lo cual se creó un archivo llamado `microservicio.py` como se muestra en la Figura 32. Se generó una función que recibe la petición al cliente utilizando Json, con un simple mensaje.

The image is a screenshot of a code editor window. The file explorer on the left shows a folder named 'EJEMPLOSERVICIO' containing a file named 'microservicio.py'. The main editor area shows the following Python code:

```
1 import json
2 from urllib import response
3 from wsgiref.simple_server import make_server
4
5
6 def application (environ, start_response):
7     headers = [ ('Content-type', 'application/json') ]
8     start_response('200 OK', headers)
9
10
11     response = {
12         'mensaje' : 'Hola Mundo desde el servidor!!'
13     }
14
15     return [ bytes(json.dumps(response), 'utf-8' ) ]
16
17 servidor = make_server('localhost', 8000, application)
18 servidor.handle_request()
```

Figura 32. Estructura del microservicio simple

3. A través de una terminal se revisa el funcionamiento del servidor, si en la terminal no se presenta algún mensaje es indicio que está en función el servidor de nuestro equipo.

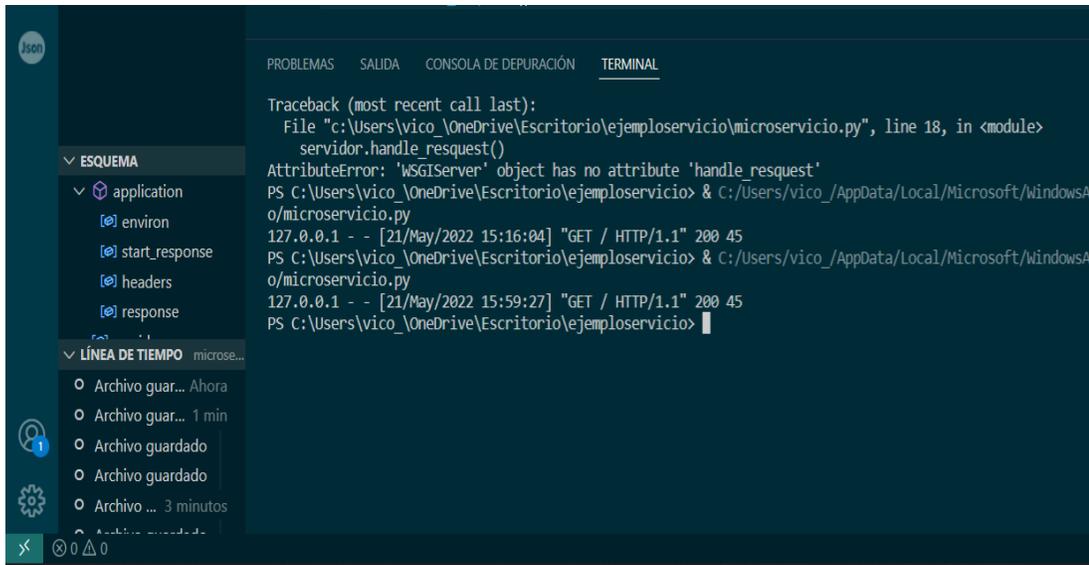


Figura 33. Terminal Python

4. Posterior a esto se visualiza la respuesta dentro del navegador como se muestra en la figura 34.

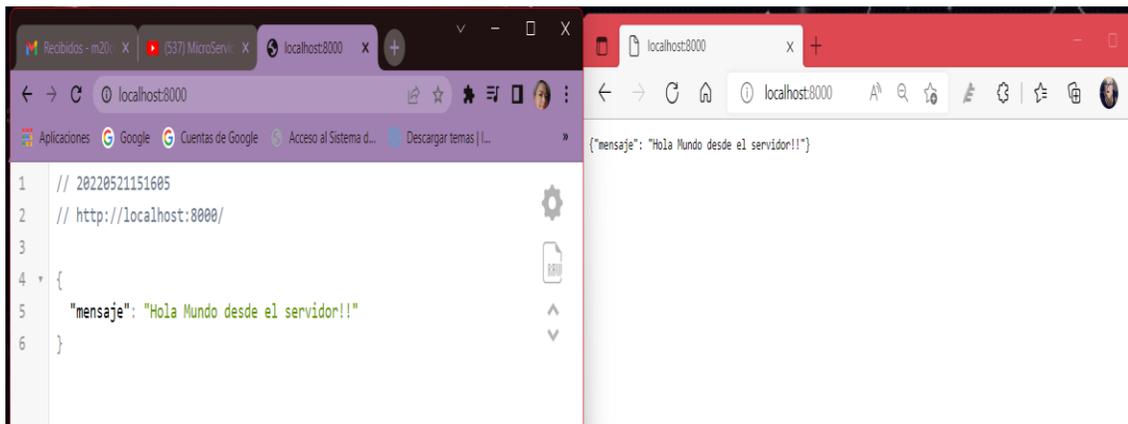


Figura 34. Ejecución del servicio en el navegador

### 8.3. Resumen

La mayoría de herramientas para realizar pruebas de seguridad están en Python, permiten su uso para ampliar las funcionalidades. Además, permite automatizar tareas de forma sencilla. Entre sus características destaca que es un lenguaje interpretado, es decir, se escribe una única

vez y funciona en todas las máquinas que dispongan del intérprete. Tiende a estar instalado en la mayoría de máquinas por defecto. Es decir, una vez instalado, casi cualquier tipo de máquina puede ejecutar cualquier código. Sabemos que es un lenguaje de alto nivel, se puede utilizar para el desarrollo de software mediante frameworks, así como una variedad de librerías extensas, así como herramientas como son Sqlmap que permite buscar vulnerabilidades en SQL de forma automática. El desarrollo de script o aplicaciones en el área de seguridad de la información, son ámbitos en los que habitualmente Python es empleado como un lenguaje de programación principal, integrando componentes escritos en diferentes lenguajes de programación.

## 8.4. Conclusión

Para este sencillo microservicio que ejecuta una sola petición desde el localhost a simple vista no existe seguridad como tal, pero el que proporciona para poder acceder a esta información es el servidor llamando y dando la ruta específica donde se quiere acceder como se observa en la (figura 32), se ejecuta mediante el servidor y la siguiente instrucción `servidor = make_server('localhost', 8000, application)` recibe los parámetros necesarios para resolver la petición del cliente. Este servidor es limitado a una sola petición a la vez. Se puede decir que cada uno de los microservicios creados en cualquier herramienta o lenguaje de programación contienen su seguridad de manera sencilla, ya que todos colocan la dirección IP o el localhost para poder acceder y el puerto a utilizar junto al nombre de su aplicación.