



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO

Tecnológico Nacional de México

**Centro Nacional de Investigación
y Desarrollo Tecnológico**

Tesis de Doctorado

**Capacidad de Entrega de Servicios Web de
Aprendizaje**

presentada por

MC. Sócrates Benítez Domínguez

como requisito para la obtención del grado de
Doctor en Ciencias de la Computación

Director de tesis

Dra. Olivia Graciela Fragoso Diaz

Cuernavaca, Morelos, México. Junio de 2026

	ACEPTACIÓN DE IMPRESIÓN DEL DOCUMENTO DE TESIS DOCTORAL		Código: CENIDET-AC-006-D20
			Revisión: 0
	Referencia a la Norma ISO 9001:2008 7.1, 7.2.1, 7.5.1, 7.6, 8.1, 8.2.4		Página 1 de 1

Cuernavaca, Mor., a 01 de junio de 2026

DR. CARLOS MANUEL ASTORGA ZARAGOZA
SUBDIRECTOR ACADÉMICO
PRESENTE

AT'n: DRA. ANDREA MAGADÁN SALAZAR
PRESIDENTA DEL CLAUSTRO DOCTORAL DEL
DEPARTAMENTO DE CIENCIAS COMPUTACIONALES

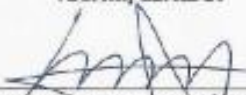
Los abajo firmantes, miembros del Comité Tutorial del estudiante **M.C. SÓCRATES BENÍTEZ DOMÍNGUEZ** manifiestan que después de haber revisado el documento de tesis titulado **"Capacidad de entrega de servicios web de aprendizaje"**, realizado bajo la dirección de la **Dra. Olivia Graciela Fragoso Díaz**, el trabajo se **ACEPTA** para proceder a su impresión.

ATENTAMENTE

Excelencia en Educación Tecnológica®
"Conocimiento y Tecnología al Servicio de México"


DRA. OLIVIA GRACIELA FRAGOSO DÍAZ
TECNM/CENIDET


DR. JUAN CARLOS ROJAS PÉREZ
TECNM/CENIDET


DR. JUAN GABRIEL GONZÁLEZ SERNA
TECNM/CENIDET


DR. RENÉ SANTAOLAYA SALGADO
TECNM/CENIDET


DR. JOSÉ ALEJANDRO REYES ORTIZ
UNIVERSIDAD AUTÓNOMA METROPOLITANA
UNIDAD AZCAPOTZALCO

c.c.p: C Verónica Sotelo Boyas/ Jefa del Departamento de Servicios Escolares
c.c.p: C Née Alejandro Castro Sánchez / Jefe del Departamento de Ciencias Computacionales
c.c.p: Expediente



Centro Nacional de Investigación y Desarrollo Tecnológico
Subdirección Académica

Cuernavaca Mor, 05/junio/2026

Oficio No. SAC/152/2026

Asunto: Autorización de impresión de tesis

SÓCRATES BENÍTEZ DOMÍNGUEZ
CANDIDATO AL GRADO DE DOCTOR EN CIENCIAS
DE LA COMPUTACIÓN
P R E S E N T E

Por este conducto, tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado "Capacidad de entrega de servicios web de aprendizaje", 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

"Conocimiento y Tecnología al servicio de México"




CARLOS MANUEL ASTORGA-ZARAGOZA
SUBDIRECTOR ACADÉMICO

C.P. Departamento de Ciencias Computacionales
Departamento de Servicios Escolares

CMAZ/lmz



Interior Internado Palmira S/N, Col. Palmira,
C.P. 62490, Cuernavaca, Morelos Tel. 01 (777) 3627770, ext. 4106,
e-mail: acad_cenidet@tecnm.mx | cenidet@tecnm.mx

Dedicatoria

A Dios, por su infinito amor, por acompañarme en el camino de la vida y por fortalecer mi fe cada día.

Agradecimientos

A Dios por brindarme la fortaleza, sabiduría y perseverancia necesarias para concluir esta etapa de mi formación.

Al pueblo de México, porque con sus contribuciones fue posible el desarrollo de este trabajo, haciendo viable mi formación y la realización de esta investigación.

A la Secretaría de Ciencia, Humanidades, Tecnología e Innovación (SECIHTI), por el apoyo otorgado para la realización de mis estudios y el desarrollo del presente proyecto.

Al Tecnológico Nacional de México.

Al Centro Nacional de Investigación y Desarrollo Tecnológico (Cenidet), y a todo su personal académico, administrativo y de servicios, por su apoyo, profesionalismo y compromiso, fundamentales en mi formación.

Agradezco profundamente a la doctora Olivia Graciela Fragoso Diaz por su orientación, paciencia y valiosas observaciones durante el desarrollo de este trabajo. Su experiencia y compromiso fueron esenciales para la culminación exitosa de esta investigación.

Al doctor Juan Carlos Rojas Pérez por su valiosa colaboración en el desarrollo de este trabajo.

A los doctores revisores de tesis, Juan Gabriel González Serna, René Santaolalla Salgado y José Alejandro Reyes Ortiz.

Índice

CAPÍTULO I. INTRODUCCIÓN.....	1
1.1 INTRODUCCIÓN.....	2
1.2 PLANTEAMIENTO DEL PROBLEMA.....	3
1.3 COMPLEJIDAD DEL PROBLEMA.....	3
1.4 OBJETIVOS	4
1.5 ALCANCES.....	4
1.6 LIMITACIONES	4
1.7 JUSTIFICACIÓN Y BENEFICIOS	5
1.8 HIPÓTESIS.....	5
CAPÍTULO II. MARCO TEÓRICO Y CONCEPTUAL	7
2.1 SERVICIOS WEB DE APRENDIZAJE	8
2.2 SERVICIOS WEB.....	9
2.3 MÉTRICAS.....	13
2.4 LENGUAJE DE PROGRAMACIÓN JAVA.....	16
2.5 CALIDAD DE SOFTWARE	23
2.6 MEDICIÓN.....	25
CAPÍTULO III. TRABAJOS RELACIONADOS	28
3.1. TRABAJOS RELACIONADOS.....	29
3.2. MODELOS DE QoS.....	30
3.3. APLICACIONES	32
3.4. EVALUACIONES.....	34
3.5. COMPARACIÓN DE LA CALIDAD DE LOS SERVICIOS WEB SOAP Y REST	36
3.6. COMPARACIÓN DE ARQUITECTURAS DE SERVICIOS WEB SOAP Y REST	36
CAPÍTULO IV. METODOLOGÍA DE SOLUCIÓN	40
4.1 METODOLOGÍA DE SOLUCIÓN	41
4.2 DIAGRAMA DEL MÉTODO DE SOLUCIÓN	42
4.3 IDENTIFICACIÓN DE ATRIBUTOS DEL SERVIDOR.....	42
4.4 IDENTIFICACIÓN DE ATRIBUTOS DEL CLIENTE.....	43
4.5 DEFINICIÓN DEL PLAN EXPERIMENTAL	43
4.6 GENERACIÓN DE SWA DE DIFERENTES FORMATOS Y MEDIDAS	46
4.7 EJECUCIÓN DEL PLAN DE PRUEBAS.....	49
4.8 OBTENCIÓN DE LOS RESULTADOS.....	49
4.9 ANÁLISIS DE RESULTADOS	49
4.10 DEFINICIÓN DE MÉTRICAS DE ENTREGABILIDAD	49
4.11 EVALUACIÓN DE RESULTADOS	49
CAPÍTULO V. PRUEBAS Y ANÁLISIS DE RESULTADOS.....	50
5.1 EJECUCIÓN DEL PLAN EXPERIMENTAL	52
5.2 ANÁLISIS DE RESULTADOS	95
5.3 DEFINICIÓN DE MÉTRICAS DE ENTREGABILIDAD	99
5.4 MODELO DE CALIDAD DE ENTREGA DE SWA	101
5.5 EVALUACIÓN DE RESULTADOS	106
5.6 DISCUSIÓN DE RESULTADOS	111
5.7 HALLAZGOS.....	112
CAPITULO VI. CONCLUSIONES Y TRABAJOS FUTUROS	114
6.1 CONCLUSIONES Y APORTACIONES	115

6.2	TRABAJOS FUTUROS.....	117
REFERENCIAS		118
ANEXO A. DESCRIPCIÓN DE LOS TRABAJOS RELACIONADOS		125
7.1	MODELOS DE QoS.....	126
7.2	APLICACIONES	135
7.3	EVALUACIONES.....	141
7.4	COMPARACIÓN DE LA CALIDAD DE LOS SW SOAP Y REST	151
7.5	COMPARACIÓN DE ARQUITECTURAS DE SERVICIOS WEB	153
ANEXO B. ESPECIFICACIONES DE ENTRADA Y SALIDA		160
8.1	ESPECIFICACIONES DE ENTRADA Y DE SALIDA.....	161
ANEXO C. CONFIGURACIÓN DEL EQUIPO DE CÓMPUTO		168
9.1	CONFIGURACIÓN DE EQUIPO DE CÓMPUTO	169
ANEXO D. ESCENARIOS DE PRUEBA.....		171
10.1	ESCENARIOS STANDALONE PARA CASO DE PRUEBAS DE ENTREGABILIDAD DE SWA.....	172
10.2	ESCENARIOS COMPUESTOS PARA CASOS DE PRUEBAS DE ENTREGABILIDAD DE SWA.....	176
ANEXO E. GRANULARIDAD DE SERVICIOS WEB DE APRENDIZAJE		183
11.1	GRANULARIDAD DE LOS SWA	184
ANEXO F. ANÁLISIS ESTADÍSTICO DE LOS RESULTADOS DE ENTREGABILIDAD DE SWA		190
12.1	ANÁLISIS ESTADÍSTICO DE LOS RESULTADOS DE ENTREGABILIDAD DE SWA.....	191
12.2	ANÁLISIS ESTADÍSTICO DE LOS RESULTADOS	196

Índice de Tablas

TABLA 1. OPCIONES DE CONFIGURACIÓN PARA EL PARÁMETRO NewRatio de G1GC.	20
TABLA 2. TRABAJOS RELACIONADOS QUE PROPONEN MODELOS DE QoS.....	31
TABLA 3. TRABAJOS RELACIONADOS QUE PRESENTAN APLICACIONES.	33
TABLA 4. TRABAJOS RELACIONADOS QUE PROPONEN EVALUACIONES DE QoS.	35
TABLA 5. TRABAJOS RELACIONADOS DE EVALUACIÓN Y COMPARACIÓN DE SERVICIOS WEB SOAP Y REST.....	37
TABLA 6. PLANTILLA DE PLAN DE PRUEBAS.....	44
TABLA 7. CASOS DE PRUEBA PARA ENTREGAR SWA.....	46
TABLA 8. SERVICIOS WEB DE APRENDIZAJE GENERADOS EN SW SOAP Y REST.....	47
TABLA 9. SERVICIOS WEB DE APRENDIZAJE PARA PRUEBAS ADICIONALES DE ENTREGA.....	47
TABLA 10. ARTEFACTOS JAVA PARA PRUEBAS DE ENTREGABILIDAD DE SWA.	48
TABLA 11. PARÁMETROS DE JHS PARA PRUEBAS DE DESPLIEGUE.....	53
TABLA 12. PRUEBAS DE DESPLIEGUES DE ARTEFACTOS DE SWA.....	53
TABLA 13. ELEMENTOS DE PRUEBA DEL CASO DE PRUEBA S-TSW1.....	54
TABLA 14. RESULTADOS DEL CASO DE PRUEBA S-TSW1.....	55
TABLA 15. ELEMENTOS DE PRUEBA DEL CASO DE PRUEBA S-TPC1.	57
TABLA 16. RESULTADOS DEL CASO DE PRUEBA S-TPC1.....	58
TABLA 17. ELEMENTOS DE PRUEBA PARA EL CASO DE PRUEBA S-SM1.	60
TABLA 18. RESULTADOS DEL CASO DE PRUEBA S-SM1.....	61
TABLA 19. SWA PARA EL CASO DE PRUEBAS C-C4.....	63
TABLA 20. RESULTADOS DEL CASO DE PRUEBA C-C4.....	64
TABLA 21. SWA UTILIZADOS PARA EL CASO DE PRUEBA C-TF2.	66
TABLA 22. RESULTADOS DE ENTREGABILIDAD DE SWA DEL CASO DE PRUEBA C-TF2.....	67
TABLA 23. CONFIGURACIÓN DE PARÁMETROS XMS Y XMX DE JHS.....	69
TABLA 24. RESULTADOS DE ENTREGABILIDAD DE SWA DE JAVA HEAP SPACE.	70
TABLA 25. ANÁLISIS DE MANEJO DE VOLUMEN EN JHS EN ENTREGABILIDAD DE SWA.....	72
TABLA 26. ALGORITMOS DE GC UTILIZADOS PARA EL CASO DE PRUEBA S-GCA5.....	74
TABLA 27. RESULTADOS DEL CASO DE PRUEBA S-GCA5.....	75
TABLA 28. RESULTADOS DEL CASO DE PRUEBA S-G1R1.....	76
TABLA 29. RESULTADOS DEL CASO DE PRUEBA S-G1R2.....	78
TABLA 30. ELEMENTOS DE PRUEBA DEL CASO DE PRUEBA S-G1P3.	80
TABLA 31. RESULTADOS DEL CASO DE PRUEBA S-G1P3.....	80
TABLA 32. RESULTADOS DEL CASO DE PRUEBA G1DD1.	82
TABLA 33. ELEMENTOS DEL CASO DE PRUEBA C-SC1.....	84
TABLA 34. RESULTADOS DE ENTREGABILIDAD DE SWA CON SERVER CACHE.	84
TABLA 35. ELEMENTOS DEL CASO DE PRUEBA C-CD4.....	87
TABLA 36. RESULTADOS DEL CASO DE PRUEBAS C-CD4.....	88
TABLA 37. ELEMENTOS DEL CASO DE PRUEBA S-JPS1.	90
TABLA 38. RESULTADOS DEL CASO DE PRUEBAS S-JPS1.....	91
TABLA 39. PESOS ASIGNADOS PARA ATRIBUTOS DE ENTREGABILIDAD DEL TRABAJO DE [Benítez, 2018].....	99
TABLA 40. PESO ASIGNADO POR ATRIBUTO DEL SERVIDOR DE ACUERDO AL MODELO DE ENTREGABILIDAD DE SWA.	101
TABLA 41. PESO ASIGNADO A LOS ATRIBUTOS DEL CLIENTE DE ACUERDO AL MODELO DE ENTREGABILIDAD DE SWA.	104
TABLA 42. PESO ASIGNADO AL ATRIBUTO TIPO DE SWA.....	104
TABLA 43. PESO ASIGNADO AL ATRIBUTO PROTOCOLO DE COMUNICACIÓN.....	104
TABLA 44. PESO ASIGNADO A LOS ATRIBUTOS DEL CLIENTE DE ACUERDO AL MODELO DE ENTREGABILIDAD DE SWA.	105
TABLA 45. EVALUACIÓN DE UN SWA DE 15MB.....	106
TABLA 46. EVALUACIÓN DE UN SWA DE 1MB.....	107
TABLA 47. EVALUACIÓN DE UN SWA DE 5MB.....	109
TABLA 48. COMPARACIÓN ENTRE CONTAINER Y VIRTUAL MACHINE [Kropp & Torre, 2020].	126
TABLA 49. DESCRIPCIÓN GENERAL DE LAS CARACTERÍSTICAS PRINCIPALES DE HTTP 1.0, 1.1 Y HTTP/2.....	131
TABLA 50. DIFERENCIAS ENTRE EL PROTOCOLO QUIC CON EL PROTOCOLO HTTP/2.....	132
TABLA 51. TABLA COMPARATIVA ENTRE DOCKER Y MÁQUINAS VIRTUALES [Potdar et al., 2020].	143

TABLA 52. ENFOQUE DE IMPLEMENTACIÓN DE SERVICIOS WEB [TIHOMIROVS & GRABIS, 2016].....	154
TABLA 53. SELECCIÓN DE ENFOQUE DE ACUERDO AL TIPO DE PROYECTO [TIHOMIROVS & GRABIS, 2016].....	154
TABLA 54. CARACTERÍSTICAS DE LOS ENFOQUES DE SERVICIOS WEB [PAVAN ET AL., 2012].....	155
TABLA 55. RECOMENDACIONES DE SOAP VS REST [BLOEBAUM & JOHNSEN, 2015].....	158
TABLA 56. XML PAYLOAD PARA SWA DE TIPO SOAP.....	162
TABLA 57. DESCRIPCIÓN DEL SERVIDOR DEFINIDO COMO SERVICIOS.....	169
TABLA 58. DESCRIPCIÓN DEL SERVIDOR DEFINIDO COMO IS1.....	169
TABLA 59. DESCRIPCIÓN DEL SERVIDOR DEFINIDO COMO IS2.....	169
TABLA 60. DESCRIPCIÓN DEL CLIENTE DEFINIDO COMO LINUX.....	170
TABLA 61. DESCRIPCIÓN DEL CLIENTE DEFINIDO COMO WINDOWS.....	170
TABLA 62. PUERTOS HABILITADOS PARA EL CLÚSTER DE WILDFLY EN IS2.....	177
TABLA 63. PUERTOS HABILITADOS PARA LA ARQUITECTURA CONTENERIZADA DE WILDFLY EN IS2.....	178
TABLA 64. RESOLUCIONES DE ARCHIVOS DE VIDEO DE ACUERDO A GOOGLE INC. (2020).....	185
TABLA 65. VOLUMEN DE ARCHIVOS DE VIDEO DE 1 MINUTO DE ACUERDO A LA RESOLUCIÓN CIRCLEHD (2020).....	185
TABLA 66. FORMATO DE IMAGEN CON CARACTERÍSTICAS DE COMPRESIÓN.....	187
TABLA 67. TAMAÑO DE IMAGEN DE ACUERDO A LA PROFUNDIDAD.....	188
TABLA 68. GRANULARIDAD DE ACUERDO AL VOLUMEN POR TIPO DE RA.....	189
TABLA 69. VALORES DE FUNCIÓN DE DENSIDAD.....	192
TABLA 70. TIPOS DE ASIMETRÍA Y DESCRIPCIÓN.....	194
TABLA 71. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-TSW1.....	197
TABLA 72. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-SM1.....	199
TABLA 73. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-TPC1.....	201
TABLA 74. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA C-SC1.....	203
TABLA 75. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA C-TF2.....	205
TABLA 76. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA C-C4.....	207
TABLA 77. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA C-CD4.....	209
TABLA 78. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-JHS1.....	211
TABLA 79. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-JHS1.....	212
TABLA 80. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-GCA5.....	214
TABLA 81. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-G1P3.....	215
TABLA 82. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-G1R1.....	217
TABLA 83. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-G1R2.....	219
TABLA 84. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-G1DD1.....	221
TABLA 85. RESULTADOS ESTADÍSTICOS DEL CASO DE PRUEBA S-JPS1.....	223

Índice de Figuras

FIGURA 1. ESTRUCTURA DE UN OBJETO DE APRENDIZAJE DE [MUÑOZ ARTEAGA ET AL., 2007].	9
FIGURA 2. FUNCIONAMIENTO DE UN SERVICIO WEB.	10
FIGURA 3. ESTRUCTURA WSDL DE UN SERVICIO WEB DE APRENDIZAJE.	13
FIGURA 4. MODELO DE CALIDAD EXTENDIDO PARA EVALUAR LA ENTREGABILIDAD [BENÍTEZ, 2018].	14
FIGURA 5. DIAGRAMA DE DESPLIEGUE DE JAVA 1.8.	18
FIGURA 6. DIAGRAMA DE SECUENCIA DE ALGORITMOS TEMPORALES DE GC.	18
FIGURA 7. ORGANIZACIÓN DEL HEAP DEL GC G1 FIRST [ORACLE CORP., 2018].	20
FIGURA 8. FUNCIONAMIENTO DE G1 FIRST DEDUPLICATION EN JHS.	21
FIGURA 9. INTERACCIÓN DE UN SWA CON G1 FIRST DEDUPLICATION EN JHS.	22
FIGURA 10. MODELO DE DOS PARTES PARA LA CALIDAD DEL PRODUCTO DE SOFTWARE [ISO/IEC, 2014].	24
FIGURA 11. MODELO DE CALIDAD PARA LA CALIDAD EN USO [ISO/IEC, 2014].	24
FIGURA 12. MODELO DE CALIDAD DE [ZUBROW, 2004]	25
FIGURA 13. MODELO GENERAL DE UN PROCESO [MONTGOMERY, 1991].	26
FIGURA 14. FACTORES DE QoS CONSIDERADOS EN [ABOSAIF & HAMZA, 2020].	39
FIGURA 15. DIAGRAMA DE MÉTODO DE SOLUCIÓN.	42
FIGURA 16. MODELO DE QoS DE ENTREGA DE SWA DEL SERVIDOR.	42
FIGURA 17. MODELO DE QoS DE ENTREGA DE SWA DEL CLIENTE.	43
FIGURA 18. ARQUITECTURA CON SERVIDOR PROXY Y CLIENTE SERVIDOR.	45
FIGURA 19. PROCESO DE GENERACIÓN DE SWA.	48
FIGURA 20. RESULTADOS DEL CASO DE PRUEBA S-TSW1.	56
FIGURA 21. RESULTADOS DEL CASO DE PRUEBA S-TPC1.	59
FIGURA 22. RESULTADOS DEL CASO DE PRUEBA S-SM1.	62
FIGURA 23. RESULTADOS DE PRUEBAS DE ENTREGA DE SWA DE CLUSTERING.	65
FIGURA 24. RESULTADOS DEL CASO DE PRUEBA C-TF2.	68
FIGURA 25. RESULTADOS DEL CASO DE PRUEBA S-JHS1.	71
FIGURA 26. RESULTADOS DEL CASO DE PRUEBA S-GCA5.	75
FIGURA 27. RESULTADOS DEL CASO DE PRUEBA S-G1R1.	77
FIGURA 28. RESULTADOS DEL CASO DE PRUEBA S-G1R2.	79
FIGURA 29. RESULTADOS DEL CASO DE PRUEBA S-G1P3.	81
FIGURA 30. RESULTADOS DEL CASO DE PRUEBA S-G1DD1.	83
FIGURA 31. RESULTADOS DEL CASO DE PRUEBAS C-SC1.	86
FIGURA 32. RESULTADOS DEL CASO DE PRUEBAS C-CD4.	89
FIGURA 33. RESULTADOS DEL CASO DE PRUEBAS S-JPS1.	92
FIGURA 34. RESULTADO DEL CASO DE PRUEBAS S-RC1.	93
FIGURA 35. PARÁMETROS DE RESPUESTA DE LA PRUEBA S-RC1.	94
FIGURA 36. PROCESO DEL ANÁLISIS ESTADÍSTICO PARA MUESTRAS INDEPENDIENTES.	96
FIGURA 37. PROCESO DEL ANÁLISIS ESTADÍSTICO PARA MÁS DE 2 MUESTRAS.	97
FIGURA 38. MÉTODOS DE VIRTUALIZACIÓN DE E/S [MAVRIDIS & KARATZA, 2019].	144
FIGURA 39. COMPARACIÓN ILUSTRADA DE LA COMPLEJIDAD DE LA PILA DE SOFTWARE DE VARIAS ESTRATEGIAS DE IMPLEMENTACIÓN, INCLUIDAS CONFIGURACIONES NATIVAS, MÁQUINAS VIRTUALES, CONTENEDORES, CONTENEDORES DENTRO DE MÁQUINAS VIRTUALES Y UNIKERNELS DE [PLAETH ET AL., 2017].	146
FIGURA 40. OPERACIÓN SOAP DEFINIDA PARA SWA SOAP.	161
FIGURA 41. MÉTODO HTTP GET PARA SOLICITUDES A SWA REST.	162
FIGURA 42. XML PAYLOAD EN SWA SOAP.	163
FIGURA 43. CONFIGURACIÓN DE VARIABLES DEL PLAN EN APACHE JMeter.	164
FIGURA 44. NOMBRE DEL ARCHIVO GENERADO DE LAS PRUEBAS DE ENTREGABILIDAD.	164
FIGURA 45. REPORTE DE SALIDA DE APACHE JMeter.	165
FIGURA 46. ARCHIVO CSV GENERADO DE LA EJECUCIÓN DEL PLAN DE PRUEBAS.	166
FIGURA 47. EJEMPLO DE RESPUESTAS A SOLICITUDES DE SWA EXITOSAS, COMPLETAS Y NO COMPLETAS.	167
FIGURA 48. DIAGRAMA DE DESPLIEGUE PARA LOS CASOS DE PRUEBA STANDALONE.	172
FIGURA 49. DIAGRAMA DE DESPLIEGUE PARA PRUEBAS ENTRE HTTP Y HTTPS.	173

FIGURA 50. DIAGRAMA DE CLASES DEL SWA CON LA IMPLEMENTACIÓN DEL PATRÓN SINGLETON.....	174
FIGURA 51. FUNCIONAMIENTO DEL PATRÓN DE DISEÑO SINGLETON CON EJB EN SWA.....	175
FIGURA 52. DIAGRAMA DE DESPLIEGUE PARA PRUEBAS DE CLUSTERIZACIÓN.....	177
FIGURA 53. DIAGRAMA DE DESPLIEGUE PARA PRUEBAS DE CONTENERIZACIÓN.	178
FIGURA 54. DIAGRAMA DE DESPLIEGUE PARA PRUEBAS DE SERVER CACHE	179
FIGURA 55. DIAGRAMA DE SECUENCIA DEL TIEMPO DE VIDA DE SERVER CACHE.	180
FIGURA 56. DIAGRAMA DE DESPLIEGUE PARA PRUEBAS DE ENTREGABILIDAD DE SWA CON TOLERANCIA A FALLAS.....	181
FIGURA 57. SECUENCIA DE INTERACCIÓN DEL SERVIDOR DE TOLERANCIA A FALLAS.....	182
FIGURA 58. CAMPANA DE GAUSS.	192
FIGURA 59. EJEMPLO DE HOMOCEDASTICIDAD Y HETEROCEDASTICIDAD.....	193
FIGURA 60. TIPOS DE CURTOSIS.	194
FIGURA 61. TIPOS DE ASIMETRÍA ESTADÍSTICA.....	194

Glosario de Términos

Acónimo	Descripción
SW	Servicio Web
SOA	Arquitectura Orientada a Servicios
SWA	Servicio Web de Aprendizaje
RA	Recurso de Aprendizaje
WSDL	<i>Web Service Description Language</i>
CORBA	<i>Common Object Request Broker Architecture</i>
URI	Identificador de Recursos Uniforme
EJB	<i>Enterprise Java Beans</i>
LMS	Sistema de Gestión de Aprendizaje
XML	Lenguaje de Marcado Extendido
UDDI	Universal Description, Discovery, and Integration
XSD	Definición de Esquema XML
MTOM	Mecanismo de Transmisión y Optimización de Mensajes
SwA	SOAP con Adjuntos
SOAP	Protocolo de Acceso Simple al Objeto
REST	Transferencia de Estado Representacional
XOP	Empaquetado Optimizado de Binarios XML
JHS	Java Heap Space
CSV	Archivo separado por comas
XMS	Tamaño inicial de la Java Heap Space
XMx	Tamaño máximo de la Java Heap Space
WAR	Artefacto Java Web Archive
JAR	Artefacto Java Archive
GC	<i>Garbage Collector</i> , o Recolector de Basura
MaGC	<i>Major Garbage Collector</i>
SGC	<i>Serial Garbage Collector</i>
PGC	<i>Parallel Garbage Collector</i>
DEF	Algoritmo por Default de GC
CMS	Algoritmo GC Concurrent Markup Sweep
G1GC	Algoritmo GC G1 First
JVM	Máquina Virtual de Java
SLA	Acuerdo de Nivel de Servicio
Th	Throughput
Tr	Tiempo de Respuesta
Te	Tasa de Éxito
MBPS	Megabytes por Segundo
MV	Máquina Virtual
VoD	Video bajo Demanda
WRAMF	Factorización de Matriz Consciente de Rango Amplio
MF	Matriz de Factorización

AHP	Proceso de Análisis Jerárquico
SAW	Peso Aditivo Simple
IoT	Internet de las Cosas
SSA	Algoritmos de Selección
PLE	ambientes de aprendizaje personal
CK	Chidamber y Kemerer
MUVW	Mundos Virtuales Multiusuario
PLT	Tiempo de Carga de la Página
HTTP	Protocolo de Transferencia de Hipertexto
HTTPS	Protocolo Seguro de Transferencia de Hipertexto
FTP	Protocolo de Transferencia de Archivos
SMTP	Protocolo de Transferencia de Archivos
TLS	Seguro de la Capa de Transporte
SSL	Capa de Conexión Segura
TCP	Protocolo de Control de Transmisión
IP	Protocolo de Internet
QUIC	Protocolo de Comunicaciones Rápidas y Seguras de Internet
ASCII	Estándar de Codificación de Caracteres
WoS	Web of Science
SWS	Servicios Web Semánticos
QoS	Calidad del Servicio
POP3	Protocolo de Correo Electrónico
RL	Aprendizaje por Refuerzo
CPU	Unidad Central de Proceso
NP	Tiempo Polinomio No Determinista
CF	Filtrado Colaborativo
SSF	Smart Service Framework
WMS	Web Map Services
SOCA	adaptación de contenido orientado a servicios
CASS	servicio de adaptación de contenido multicriterio
RAM	Memoria de Acceso Aleatorio
COP	Plataformas Orquestadoras de Contenedores
WSRS	Sistemas de Recomendación en Ambientes Orientados a Servicios
SO	Sistema Operativo
MOEA	Algoritmos Multiobjetivo Evolutivos
SBA	Servicios Basados en Aplicaciones
MCDC	Cobertura de decisión de condición modificada
HMM	Modelos Ocultos de Márkov
SNA	Análisis de Redes Sociales
EVT	Teorema del Valor Extremo Generalizado
DC	DigitalCinema
RTSP	Protocolo de Transmisión de servicios en tiempo real
B2B	Empresa a Empresa

CRUD	Crear, Actualizar, Leer y Eliminar
Wi-Fi	Red Inalámbrica
IS1	Ingeniería de Software 1
IS2	Ingeniería de Software 2
CDN	Red de Entrega de Contenido
VNF	Virtualización de Problemas de Red

Resumen

El *E-Learning* es una de las herramientas que se considera actualmente por parte de las entidades para que las personas puedan acceder a la educación. El confinamiento de la pandemia del COVID-19 modificó y reforzó la manera en la que las personas acceden a la educación hasta la fecha. Como resultado, la utilización de las plataformas educativas se incrementó significativamente. Sin embargo, las plataformas educativas digitales sufrieron un aumento en la demanda de recursos computacionales, lo que causó en muchas ocasiones, el colapso de dichas plataformas, inconsistencia en los programas educativos, defectos en las plataformas y fallas para llevar a cabo las actividades educativas requeridas.

El principal problema de los servicios Web de aprendizaje (SWA) enfocados en la arquitectura de servicios Web, es que no se pueda entregar el contenido, que el usuario solicite el SWA y que este no sea entregado, perdiendo así el propósito del *E-Learning*.

En este trabajo de tesis se presenta un modelo de QoS basado en atributos internos de clientes y servidores que evalúa la capacidad de entrega de SWA, con el objetivo de subsanar la carencia de información acerca de la capacidad de entrega de los elementos que intervienen en la entrega de SWA.

A la fecha, existen diversos atributos de calidad (QoS), así como técnicas, herramientas y plataformas tecnológicas que contribuyen a mejorar la capacidad de entrega de los SWA, permitiendo mejorar las arquitecturas de servicios y del hardware, reduciendo el tráfico de red y aumentando la reutilización de los componentes de software.

Se definieron siete escenarios para diecisiete casos de prueba de entregabilidad de SWA. Además, se realizó el análisis de granularidad de los recursos de aprendizaje para definir los tamaños de estos para ser integrados y utilizados en las pruebas de entregabilidad de servicios Web de aprendizaje. Los resultados y el análisis de los resultados de entregabilidad de SWA permiten determinar la influencia en la capacidad de entrega de los diferentes atributos encontrados.

Los resultados de las pruebas de la capacidad de entrega de SWA demuestran que el modelo permite evaluar de manera efectiva la capacidad de entrega de los SWA, proporcionando información útil a los responsables de las plataformas de servicios Web en E-Learning, a los creadores de contenido de aprendizaje y a los usuarios de dichas plataformas.

Abstract

E-Learning is currently considered one of the main tools used by organizations to provide access to education. The confinement caused by the COVID-19 pandemic changed and reinforced the way people access education, and its effects remain evident today resulting in a significant increase in the adoption of E-Learning platforms. However, digital educational platforms experienced a substantial increase in the demand for computing resources, which in many cases caused platform failures, inconsistencies in educational programs, software defects, and difficulties in carrying out the required educational activities.

The main problem of Learning Web Services (LWS) focused on Web service architectures is the inability to deliver content. When a user requests a Learning Web Service and it is not delivered, the fundamental purpose of E-Learning is compromised.

This thesis presents a Quality of Service (QoS) model based on internal client and server attributes to evaluate the delivery capability of LWS, with the objective of addressing the lack of information regarding the delivery capability of the elements involved in the delivery of LWS.

To date, several quality attributes, techniques, tools, and technological platforms have been developed to improve the delivery capability of LWS's, enabling enhancements in service and hardware architectures, reducing network traffic, and increasing software component reuse.

Seven scenarios comprising sixteen test cases were defined. In addition, a granularity analysis of learning resources was conducted to determine their appropriate sizes for integration and use in LWS deliverability tests. The results and analysis of the LWS's deliverability tests made it possible to determine the influence of the different attributes identified on delivery capability.

The results of the LWS delivery capability tests demonstrate that the proposed model effectively evaluates the delivery capability of LWS's, providing useful information to managers of E-Learning platforms, learning content creators, and users of such platforms.

Capítulo I. Introducción

1.1 Introducción

La educación es uno de los factores que favorecen el desarrollo integral de los individuos y el de las sociedades en las que estos participan, esta debe estar garantizada a cada uno de los individuos. Para impartir la enseñanza y/o capacitación, se deben de extender tanto la cobertura como los horarios por parte de los responsables, ya sean gobiernos, empresas o instituciones educativas. Estas últimos expresan su preocupación y necesidad de mantener el proceso de enseñanza-aprendizaje, así sea de manera presencial o a distancia. Así mismo, la educación a distancia o virtual, es actualmente un tema relevante para las instituciones de educación de todos los niveles. El ***E-Learning*** es a la fecha una modalidad fundamental para el desarrollo de los programas educativos, esta emplea recursos digitales para lograr los objetivos del aprendizaje, que pueden ser de cualquier tipo, tamaño y formato (video, imagen, documento, gráfica, etc.) y ha sido promovido con el rápido crecimiento de internet. Por lo que es importante buscar los medios para llevar a cabo ***E-Learning*** atendiendo a las ventajas de que cada quién aprende en su tiempo, a distancia.

En Cenidet se ha definido el término de SWA como un servicio Web que entrega un recurso de aprendizaje en respuesta a una solicitud. Uno de los retos más importantes en la implementación de SWA, es evaluar las capacidades de entregar el recurso de aprendizaje completo y en el menor tiempo posible.

Aunado a lo anterior, los responsables de ***E-Learning*** deben conocer la capacidad de la tecnología que se emplea para entregar los recursos de aprendizaje, es decir, deben saber cuál es el mejor tipo de implementación de servicios Web de aprendizaje, cuál la mejor plataforma o las mejores herramientas para entregar su capacitación, desde cuales son las más económicas, cuáles son las más rápidas, cuales ofrecen mayor accesibilidad, mayor facilidad de instalación, operación y mantenimiento, si cuentan con seguimiento del aprendizaje, hasta si consideran estrategias para entregar recursos de aprendizaje de gran tamaño, arriba de 30MB por ejemplo. Si los recursos de aprendizaje son administrados y entregados a los usuarios vía servicios Web (WS). Desafortunadamente, información sobre los atributos anteriores aún no se encuentra disponible. Por lo tanto, este trabajo de investigación se enfoca en evaluar la capacidad de entrega de los SWA considerando los atributos de calidad de servicios que intervienen en la entrega de SWA.

1.2 Planteamiento del Problema

Debido a la carencia de información de la capacidad de entrega de SWA, estos últimos no pueden ser clasificados o referenciados de acuerdo a su desempeño de entrega. Esto provoca que el usuario utilice cualquier SWA con la probabilidad de que el recurso no se entregue en tiempo y forma, por lo tanto, los procesos de aprendizaje no puedan ser llevados a cabo como estén definidos y que los objetivos de aprendizaje no se alcancen.

1.3 Complejidad del Problema

A diferencia de los servicios web que normalmente realizan cálculos o transacciones, la responsabilidad de los SWA y el funcionamiento computacional de los mismos son diferentes. Los SWA manejan un recurso de aprendizaje el cuál debe ser procesado, manipulado, instanciado y enviado a una aplicación cliente que solicita dicho recurso. Este funcionamiento junto con su comportamiento son puntos que no están estudiados y que deben de ser analizados para lograr objetivos de aprendizaje que han sido planteados. El modelo para entrega de SWA basado en atributos del cliente - servidor aún no tiene definido algún método que indique cuales son los valores y atributos que permitan referenciar los mejores resultados, por lo tanto, tampoco se han definido ni identificado los atributos de los servidores y de los clientes que permitan o faciliten la entrega de los recursos de aprendizaje.

Los usuarios de los SWA carecen de información que indique cuál servicio tiene mejores resultados de entrega, por lo que también se deben definir cuáles son los valores de entrega para cada uno de los atributos, cuál debe ser el método de evaluación y cuál debe ser la manera en que se debe aplicar cada métrica de manera individual y/o en conjunto.

Cuando se validen los atributos que se incluyan en el modelo de capacidad de entrega de los SWA, se debe de analizar la relación entre atributos, si la relación es positiva, negativa o si estos mantienen una relación neutral en la que ninguno de los dos atributos tenga alguna afectación o mejora en su desempeño.

Adicionalmente, el controlar variables tanto del servidor como del cliente es una tarea compleja. En el primer punto porque se deben de caracterizar los atributos al propio servidor, por ejemplo, lenguaje de programación, virtualización del sistema operativo, **server clustering**, servidor de aplicaciones, etc. En el segundo punto se deben caracterizar los atributos del cliente, el tipo de servicio Web, el protocolo de comunicación, etc.

1.4 Objetivos

1.4.1 Objetivo General

Proponer un modelo de QoS que proporcione al usuario y a los responsables de las plataformas de *E-Learning*, información referente de la capacidad de entrega de SWA con respecto a los atributos de calidad de servidores y de clientes.

1.4.2 Objetivos Específicos

- ◆ Identificar los atributos de QoS de entrega de SWA del servidor
- ◆ Identificar los atributos de QoS de entrega de SWA del cliente
- ◆ Definir las métricas de los atributos
- ◆ Definir el modelo de QoS de entrega de SWA.

1.5 Alcances

Dentro del alcance de este trabajo de investigación se propuso:

- ◆ El modelo de calidad de entrega de SWA con respecto a los atributos de entrega de servidores y clientes.
- ◆ Búsqueda de atributos de calidad que permitan mejorar la calidad de entrega de SWA.
- ◆ Definir los atributos de calidad de servidores y clientes que puedan evaluarse.
- ◆ Definir los métodos de evaluación.
- ◆ Definir plan experimental.
- ◆ Definir los escenarios de pruebas.
- ◆ Evaluación de los resultados en términos de los atributos y métodos definidos.

1.6 Limitaciones

- ◆ El proyecto se limitó a pruebas en servidores locales.
- ◆ Los servicios Web de aprendizaje se implementaron en el lenguaje de programación Java.
- ◆ Los atributos se definieron en función de componentes de hardware y de software.
- ◆ Se identificó un conjunto de valores a partir del conjunto de casos de pruebas establecidos.
- ◆ No se consideraron los elementos internos ni los pedagógicos de los recursos de aprendizaje.
- ◆ Solo se consideraron SWA de los llamados “simples”, los cuales entregan solo un recurso de aprendizaje, exentos de método alguno para organizar estructuras más complejas.

1.7 Justificación y Beneficios

La información de los resultados de entrega que se proporcione de la evaluación de los SWA, apoyará a los usuarios de los SWA a elegir los mejores servicios que apoyen los procesos de enseñanza definidos y alcanzar los objetivos de aprendizaje.

Por otro lado es deseable que los responsables del **E-Learning** conozcan las capacidades computacionales que se deben emplear para entregar los recursos de aprendizaje a través de SW, estas no solo incluyen las capacidades de la arquitectura de servicios, también necesitan conocer cuál es la mejor plataforma o cuáles son las mejores herramientas que se deben utilizar, cuáles son las más económicas, cuáles son las más rápidas, cuáles ofrecen mayor accesibilidad, mayor facilidad de instalación, operación y mantenimiento, si cuentan o no con seguimiento del aprendizaje, hasta si consideran estrategias para entregar recursos de aprendizaje de gran tamaño, arriba de 30MB por ejemplo.

1.8 Hipótesis

Con el propósito de analizar la capacidad de entrega de los SWA, se definieron dos escenarios experimentales de evaluación. Cada escenario representa una configuración específica del entorno de ejecución empleado para la entrega de los SWA. Los dos escenarios se definieron para permitir la comparación respecto del atributo en cuestión, considerando métricas definidas para evaluar la capacidad de entrega de SWA, como es el tiempo de respuesta, el **throughput** y la tasa de éxito. Uno de los ejemplos de estos escenarios se presenta a continuación:

Escenario A. Configuración del GC de JHS sin mecanismos adicionales para la optimización de memoria.

Escenario B. Configuración del GC de JHS con **G1 First Deduplication**, habilitando la deduplicación de objetos **String** para optimizar el uso de memoria.

La definición de estos escenarios facilita la identificación de posibles diferencias en la capacidad de entrega del SWA permitiendo determinar si los atributos evaluados influyen de manera significativa en el desempeño del sistema.

1.8.1 Hipótesis General

En este apartado se definieron dos escenarios, que se describen a continuación:

- ◆ Escenario A. Configuración básica del SWA.
- ◆ Escenario B. Configuración alternativa del SWA.

H₀: No existen diferencias estadísticamente significativas en la capacidad de entrega del servicio Web de aprendizaje entre el Escenario A y el Escenario B.

H₁: Existen diferencias estadísticamente significativas en la capacidad de entrega del servicio Web de aprendizaje entre el Escenario A y el Escenario B.

1.8.2 Hipótesis Específicas

A. Tiempo de respuesta

- ♦ **H₀:** No existe diferencia estadísticamente significativa del tiempo de respuesta entre el Escenario A y Escenario B.
- ♦ **H₁:** Existe diferencia estadísticamente significativa del tiempo de respuesta entre el Escenario A y Escenario B.

B. Throughput

- ♦ **H₀:** No existe diferencia estadísticamente significativa del *throughput* entre el Escenario A y Escenario B.
- ♦ **H₁:** Existe diferencia estadísticamente significativa del *throughput* entre Escenario A y Escenario B.

C. Tasa de éxito

- ♦ **H₀:** No existe diferencia estadísticamente significativa de tasa de éxito entre el Escenario A y Escenario B.
- ♦ **H₁:** Existe diferencia estadísticamente significativa de tasa de éxito entre el Escenario A y Escenario B.

1.8.3 Variables Dependientes

De acuerdo con [Benitez, 2018], las métricas de desempeño que se evalúan para determinar la capacidad de entrega de SWA, son las siguientes:

- A. **Tasa de éxito.** Evalúa el porcentaje de solicitudes respondidas con éxito.
- B. **Tiempo de respuesta.** Tiempo que tarda en responder una solicitud.
- C. **Throughput.** Determina la capacidad de procesamiento de una solicitud de un SWA.

1.8.4 Variables Independientes

En el caso de prueba se definen las condiciones del entorno bajo las cuales se realiza la prueba. De acuerdo al plan de pruebas definido, las variables controladas son las siguientes:

- A. Escenario A: Configuración básica del SWA.
- B. Escenario B: Configuración alternativa de los atributos evaluados.
- C. El número de solicitudes.
- D. El número de grupos de solicitudes.
- E. Tamaño del SWA.

Capítulo II. Marco Teórico y Conceptual

2.1 Servicios Web de Aprendizaje

2.1.1 Servicios Web de Aprendizaje

En [Vossen & Westerkamp, 2003], los autores definen a un SWA como una unidad de aprendizaje construida como Servicio Web que contiene el mecanismo de comunicación hacia el contenido educativo de uno de los cuatro elementos que conforman a un objeto de aprendizaje, funcionando como intermediario entre el usuario y el contenido de aprendizaje. El SWA cuenta con todas las características de un servicio Web, pero su uso es exclusivamente para consultar y desplegar al usuario uno o más contenidos de aprendizaje. En el trabajo de tesis de maestría de [Delgado Fernández, 2012], un SWA es una unidad de aprendizaje construida como servicios Web que contiene el mecanismo de comunicación hacia el contenido educativo, funcionando como intermediario entre el usuario y el contenido de aprendizaje. El SWA cuenta con todas las características de un servicio Web, pero su uso es exclusivamente para consultar y desplegar al usuario uno o más contenidos de aprendizaje.

En [Prieto Méndez et al., 2013], los autores definen a SWA como una actividad individual de un sistema de aprendizaje que puede modelarse como proceso y proporcionarse como servicio.

2.1.2 Objeto de Aprendizaje

En el trabajo de [Polsani, 2003] se menciona que un Objeto de Aprendizaje desde una perspectiva pedagógica está formado por: un objetivo pedagógico, contenido teórico, actividades de aprendizaje, evaluación y metadatos. Para lograr mayor beneficio de los Objetos de Aprendizaje, es necesario que cuenten con ciertas características que son: auto contenido, interoperable, reutilizable, durable y actualizable, de fácil acceso y manejo, secuenciable, breve y sintetizado y que permite la incorporación de la fuente de diversos recursos de autoría. En [Sicilia, 2005], el autor menciona que un objeto de aprendizaje es una unidad didáctica, independiente y auto-contenida predispuesta para su reutilización en diversos contextos educativos. En [Muñoz Arteaga et al., 2007], el autor indica que un objeto de aprendizaje se define como una unidad didáctica de aprendizaje en formato digital, independiente, auto-contenida, perdurable y pre-dispuesta para su reutilización en varios contextos educativos por la inclusión de información auto-descriptiva en forma de metadatos.

En la Figura 1 de acuerdo con [Muñoz Arteaga et al., 2007], se describe la estructura de un objeto de aprendizaje, la cual indica que puede ser de 4 tipos además que todos deben contener los metadatos que describen el contenido del objeto de aprendizaje.

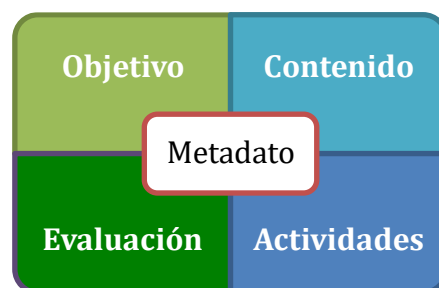


Figura 1. Estructura de un Objeto de Aprendizaje de [Muñoz Arteaga et al., 2007].

2.1.3 Recurso de Aprendizaje

En el trabajo de [Muñoz Arteaga et al., 2011], los autores mencionan que un recurso de aprendizaje digital se define como cualquier entidad, que puede usarse o reutilizarse o referenciarse durante el aprendizaje con soporte tecnológico. El rango va desde un recurso simple como una imagen hasta un curso completo. También agregan que la creación de un recurso de aprendizaje digital podría involucrar material básico, herramientas, una metodología y personas capacitadas. El proceso de creación puede diferir mucho en lo que uno está creando; una imagen puede implicar un simple clic en una cámara digital, un curso completo puede involucrar a un equipo completo de expertos en la materia, artistas gráficos, ingenieros de software, etc.

La descripción de un recurso de aprendizaje proporciona los metadatos del mismo y sigue hoy en día generalmente un estándar o una especificación, más en particular el estándar IEEE LOM.

2.1.4 Sistema de Gestión de Aprendizaje (LMS)

En [Chinnici et al., 2007], Sistema de Gestión de Aprendizaje (LMS por sus siglas en inglés) se define como una aplicación contenida en un servidor de páginas Web que permite el desarrollo de actividades formativas.

2.2 Servicios Web

2.2.1 Cliente – Servidor

De acuerdo a [Fielding, 2000], es un estilo de interacción que más frecuentemente se utiliza en las aplicaciones basadas en redes.

2.2.2 Arquitectura de Software

En el trabajo de [Fielding, 2000], Arquitectura de Software se define como una abstracción de los elementos de tiempo de ejecución de un sistema de software durante alguna fase de su operación. Un sistema puede estar compuesto de muchos niveles de abstracción y muchas fases de operación, cada uno con su propia arquitectura de software.

En el centro de la arquitectura del software está el principio de la abstracción: ocultar algunos de los detalles de un sistema a través de la encapsulación a fin de identificar y sostener mejor sus propiedades. Un sistema complejo contendrá muchos niveles de abstracción, cada uno con su propia arquitectura. Una arquitectura representa una abstracción del comportamiento del sistema a ese nivel, de modo que los elementos arquitectónicos son delineados por las interfaces abstractas que proporcionan a otros elementos a ese nivel. Dentro de cada elemento se puede encontrar otra arquitectura, definiendo el sistema de subelementos que implementan el comportamiento representado por la interfaz abstracta del elemento primario. Esta recursión de arquitecturas continúa hasta los elementos más básicos del sistema: aquellos que no se pueden descomponer en elementos menos abstractos.

La investigación en arquitectura de software busca o intenta encontrar métodos para determinar la mejor manera de dividir un sistema, la manera en que los componentes se identifican y se comunican entre sí, cómo se comunica la información, cómo los elementos de un sistema pueden evolucionar de forma independiente y cómo se puede describir todo lo anterior utilizando notaciones formales e informales.

2.2.3 Servicio Web

En el trabajo de [Erl, 2008], se define el término de servicio como “un recurso abstracto que representa la capacidad de realizar tareas que forman una funcionalidad coherente desde el punto de vista de las entidades proveedoras y entidades solicitantes. Para ser utilizado, un servicio debe realizarse por un agente proveedor concreto”. En [W3C, 2004], el término servicio Web está definido como “un sistema de software diseñado para soportar la interacción entre dos máquinas a través de una red. Posee una interfaz descrita en un formato que puede ser procesado por una máquina. Otros sistemas pueden interactuar con el Servicio Web en la manera prescrita por su descripción utilizando mensajes SOAP, típicamente transportados utilizando HTTP y serializados con XML, en conjunción con otros estándares relacionados con la Web”.

La Figura 2 muestra el método de comunicación entre dos dispositivos a través de una red, desde internet es solicitado un servicio Web y respondida una solicitud.

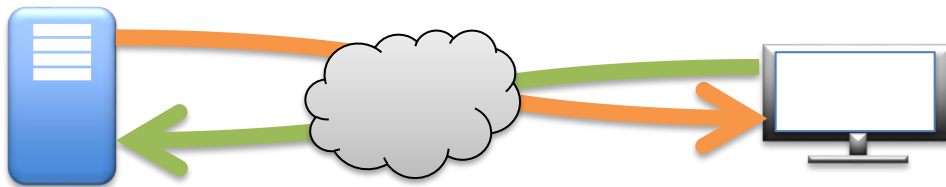


Figura 2. Funcionamiento de un servicio Web.

2.2.4 SOAP

En trabajo de [Newcomer, 2002], el autor describe a SOAP que en español significa "Protocolo de Acceso Simple al Objeto", como un sistema de mensajería para el intercambio de datos XML formateados a través de Internet. En general, se puede decir que SOAP proporciona los mecanismos de comunicación para conectar los servicios Web. SOAP de manera simple, es fácil de desarrollar y es completamente neutral con respecto al sistema operativo, lenguaje de programación o plataforma de computación distribuida. SOAP está destinado a proporcionar un nivel mínimo de transporte sobre el cual se pueden construir interacciones y protocolos más complicados.

De acuerdo con [Tihomirovs & Grabis, 2016b], *"SOAP se desarrolló como una alternativa al estándar CORBA (Common Object Request Broker Architecture). Para garantizar el transporte de datos en SOAP, se utilizan protocolos como HTTP, SMTP, FTP, etc., mientras que los datos se envían en formato XML. El principio principal de este enfoque es el siguiente: un proveedor de servicios publica una descripción del servicio o una interfaz para el registro del servicio, por lo que el solicitante del servicio puede encontrar una instancia de servicio correcta y utilizarla"*.

2.2.5 REST

El estilo REST definido en la tesis de [Fielding, 2000], que en español significa "Transferencia de Estado Representacional", es una abstracción de los elementos arquitectónicos dentro de un sistema hipermedia distribuido. Consiste de un conjunto de principios que definen la interacción entre distintos componentes, un conjunto de operaciones (GET, PUT, POST, DELETE, PURGE) y está formado por tres clases de elementos arquitecturales:

- ◆ Un componente es una unidad abstracta de instrucciones de software y estado interno que proporciona una transformación de datos a través de su interfaz.
- ◆ Un conector es un mecanismo abstracto que media la comunicación, la coordinación o la cooperación entre los componentes.
- ◆ Un **datum** es un elemento de información que es transferido de un componente, o recibido por un componente, a través de un conector.

De acuerdo al trabajo de [Tihomirovs & Grabis, 2016b], REST es un *"enfoque más nuevo, que utiliza el protocolo HTTP para transmitir datos, mientras que los datos están formados por formatos XML, JSON, etc. REST simplifica el acceso a los servicios Web mediante el uso de los estándares ya establecidos en lugar de agregar nuevas capas de procesamiento de datos en la pila de transmisión y comunicación. Por lo tanto, REST tiende a ser una alternativa más ligera al protocolo SOAP. Los servicios Web REST se basan en recursos autodefinidos en los que se utiliza el protocolo HTTP para llegar a ellos. Se proporciona un servicio como un recurso que puede identificarse mediante URI (identificador uniforme de recursos)"*.

2.2.6 UDDI

UDDI es un directorio que almacena información de servicios. Los proveedores de servicios pueden registrar sus servicios en un directorio UDDI. Los usuarios del servicio, pueden buscar en el directorio UDDI para encontrar el servicio que necesiten. Cuando la interfaz del servicio se encuentra, el usuario puede comunicarse con el servicio inmediatamente, [Clement et al., 2004].

2.2.7 WSDL

Es un formato XML para describir servicios de red como un conjunto de **Endpoints** que operan en mensajes que contienen información orientada a documentos u orientada a procedimientos. Las operaciones y los mensajes se describen de forma abstracta y, a continuación, se vinculan a un protocolo de red concreto y a un formato de mensaje para definir un punto final. Los **Endpoints** concretos relacionados se combinan en puntos finales abstractos (servicios). WSDL es extensible para permitir la descripción de los puntos finales y sus mensajes, independientemente de qué formatos de mensaje o protocolos de red se utilizan para comunicarse, [Abdaldhem et al., 2009].

El objetivo de un WSDL es definir la interfaz pública de un servicio Web específico, que puede ser accedido utilizando esta interfaz, [Abdaldhem et al., 2009]. Una descripción del servicio WSDL puede separarse en dos partes:

- ◆ **Descripción Abstracta.** La descripción abstracta proporciona información sobre las características de la interfaz del servicio Web sin ninguna descripción o detalles sobre la tecnología utilizada para implementar los servicios Web o la utilizada para enviar mensajes. Como resultado, la integridad del servicio Web se mantiene a pesar de un cambio en la plataforma tecnológica subyacente. Consta de varias partes, incluidos tipos, mensajes y tipo de puerto.
- ◆ **Descripción Concreta.** La descripción concreta del archivo WSDL define la conexión a la implementación real de los servicios Web donde está su lógica. Esta descripción contiene tres partes: enlace, puerto y servicio.

Un documento WSDL utiliza los siguientes elementos:

- ◆ Tipos: *"un contenedor para las definiciones de tipos de datos que utilizan algún tipo de sistema (como XSD)".*
- ◆ Mensaje: *"una definición abstracta y mecanografiada de los datos que se están comunicando".*
- ◆ Tipo de puerto: *"un conjunto abstracto de operaciones soportadas por uno o más puntos finales".*
- ◆ Operación: *"una descripción abstracta de una acción respaldada por el servicio".*
- ◆ Encuadernación: *"un protocolo concreto y especificación de formato de datos para un tipo de puerto particular".*
- ◆ Servicio: una colección de puntos finales relacionados.

- ♦ Puerto: "un único punto final definido como una combinación de un enlace y una dirección de red".

La Figura 3 muestra un ejemplo de un WSDL de un servicio Web de aprendizaje, en el que se pueden visualizar los elementos listados: tipos, mensaje, operaciones, enlace, servicio y puerto.



Figura 3. Estructura WSDL de un Servicio Web De Aprendizaje.

2.3 Métricas

2.3.1 Reusabilidad

En el trabajo de [Marulanda López, 2014], el autor indica que la reusabilidad es una sub característica de la mantenibilidad que a su vez es una característica de la calidad del producto de software y lo termina definiendo como la capacidad de un activo que permite que sea utilizado en más de un sistema de software o en la construcción de otros activos.

2.3.2 Entregabilidad

En el trabajo de [Benítez Domínguez, 2018], el término entregabilidad está definido como la capacidad que tiene un servicio Web de entregar el recurso solicitado. En la Figura 4 se muestra el modelo de calidad extendido de entregabilidad.

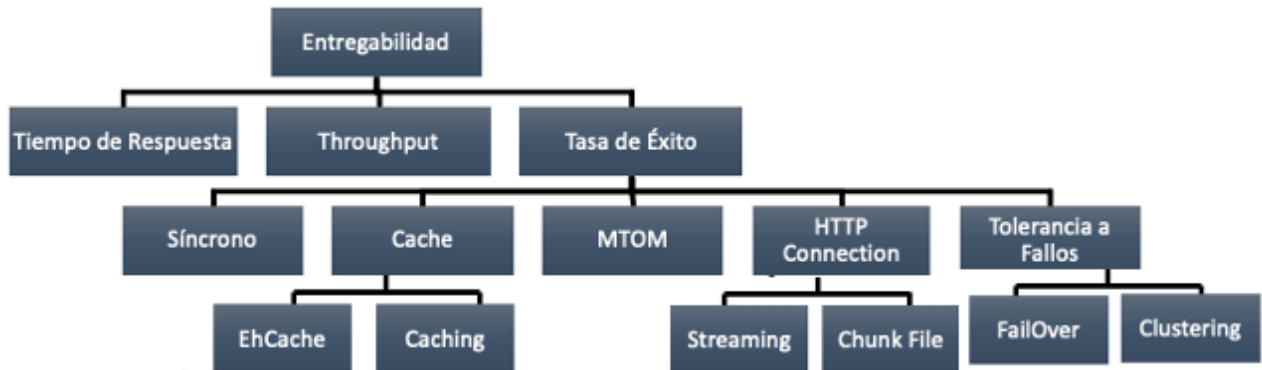


Figura 4. Modelo de calidad extendido para evaluar la entregabilidad [Benítez, 2018].

2.3.3 Throughput

Throughput es el promedio del rango de bits por segundo, enviados como mensaje a través de un canal de comunicación, [Zheng et al., 2010].

2.3.4 Tiempo de Respuesta

El tiempo de respuesta es el tiempo de duración en el que un usuario envía una solicitud y recibe su correspondiente respuesta, [Zheng et al., 2010].

2.3.5 Tasa de éxito

Tasa de éxito se define como el porcentaje de las solicitudes respondidas satisfactoriamente del total de las solicitudes realizadas, [Benítez, 2018]. En contexto de los códigos de respuesta de HTTP, el código 200 indica que la respuesta es satisfactoria o exitosa, el código 500 indica un error del servidor y que no se respondió la solicitud con éxito.

2.3.6 SwA

SwA (SOAP con Adjuntos por sus siglas en inglés) es un mecanismo que permite a los servicios Web adjuntar archivos en su respuesta a la solicitud y enviarlos en el mensaje en formato binario, sin convertir datos binarios a caracteres ASCII utilizando la codificación Base64 [Barton et al., 2000].

2.3.7 MTOM

Mecanismo de Optimización de Transmisión de Mensaje SOAP (MTOM por sus siglas en inglés) es un mecanismo desarrollado para transmitir archivos adjuntos mediante servicios Web basados en SOAP, convirtiendo los datos codificados en Base64Binary en

datos binarios en bruto con el límite MIME, llamado paquete XOP (***XML-binary Optimized Package***), justo antes de la transmisión, [Hunga et al., 2010].

2.3.8 Asíncrono

Los servicios Web se pueden comunicar mediante mensajes, y estos mensajes se pueden enviar de forma síncrona o asíncrona. Cuando se realiza una solicitud síncrona, el emisor de la solicitud suspende el procesamiento hasta que recibe la respuesta, [Al-Humaimedy & Fernández, 2015].

2.3.9 Cache

El Caché HTTP es como un almacén local de mensajes de respuesta y el subsistema que controla el almacenamiento, recuperación y eliminación de mensajes en él, para reducir el tiempo de respuesta, el consumo de ancho de banda de red en el futuro, y de solicitudes que sean equivalentes, [Fielding & Reschke, 2014].

2.3.10 Caching

El almacenamiento en caché es una técnica para mejorar el *throughput* de aplicaciones Web, el almacenamiento en caché es una característica completamente OPCIONAL de HTTP. A la fecha, existen dos formas de implementar en caché: *Client Cache* y *Server Cache*, [Liyanaarachchi & Weerawarana, 2012].

2.3.11 Client Cache

Implementación que mantiene peticiones y respuestas para reusarlas en peticiones posteriores. Puede ser tanto una cache privada como una compartida, [Mozilla, 2023].

2.3.12 Server Cache

Es un mecanismo que permite almacenar una respuesta temporal en el servidor para responder solicitudes similares, [Liyanaarachchi & Weerawarana, 2012]. El objetivo principal de la implementación de Server Cache es eliminar la sobrecarga de invocación del servicio manteniendo una copia en caché del servidor de la respuesta del servicio Web, [He et al., 2017].

2.3.13 Clúster

Es un tipo de sistema de procesamiento paralelo o distribuido que consiste en una colección de computadoras independientes interconectadas que trabajan juntas como un único recurso informático integrado, [López & Baydal, 2017].

2.3.14 Instancia

En arquitectura de software, una instancia es una realización concreta de un componente arquitectónico, módulo, servicio o sistema, en ejecución. Representa una

copia operativa de un diseño, clase o plantilla, con su propio estado, recursos y parámetros configurados, desplegada en un entorno específico, [Feiler et al., 2006].

2.3.15 Nodo

Un nodo es una máquina de trabajo que puede ser una máquina virtual o física, dependiendo del tipo de clúster. Cada nodo está gestionado por el componente máster y contiene los servicios necesarios para ejecutar pods, [The Kubernetes Authors, 2022].

2.3.16 Apache FailOver

Failover o conmutación es un mecanismo que se configura para que en caso de que el servidor no pueda atender la solicitud, este dirija a direcciones alternativas y esta solicitud pueda ser respondida con éxito, [Apache, 2017].

2.3.17 Balanceo de Carga

Load Balancing o Balanceo de Carga son una serie de técnicas ampliamente utilizadas para distribuir el trabajo entre múltiples recursos de acuerdo con una política determinada. El objetivo principal para el balanceo de carga es garantizar la máxima utilización de los recursos disponibles, a través de algoritmos estáticos y dinámicos y estrategias como **Round Robín** o Mínimas Conexiones que pueden permitir el acceso al recurso, [Ewing & Menascé, 2009].

2.3.18 Codificación Fragmentada de HTTP

Es un mecanismo de transferencia de datos, en donde son divididos en fragmentos y se envían y se reciben de forma independiente. Cada fragmento esta precedido por su tamaño en bytes y la transmisión termina cuando se recibe un fragmento de longitud 0. Se menciona en [Belshe & Peon, 2015] que la codificación fragmentada es soportada solo en la versión 1 y 1.1 de HTTP.

2.3.19 HTTP Streaming

Un "**Stream**" es una secuencia independiente y bidireccional de tramas intercambiadas entre el cliente y el servidor dentro de una conexión HTTP/2 [Belshe & Peon, 2015].

2.4 Lenguaje de Programación Java

2.4.1 Enterprise Java Beans (EJB)

EJB es una arquitectura de componentes del lado del servidor para la construcción modular de aplicaciones empresariales. Los EJB permiten el desarrollo simplificado de aplicaciones a gran escala, transaccionales y seguras, proporcionando soporte integrado para concurrencia, seguridad, persistencia, entre otros, [Debu et al., 2007].

2.4.2 Patrón de Diseño Singleton

El autor de [Gamma et al., 1994], indica que el patrón de diseño Singleton se debe utilizar cuando:

- ♦ Debe haber solo una instancia de una clase y esta debe ser accesible para los clientes desde un punto de acceso conocido.
- ♦ Cuando la única instancia debe ser extensible mediante subclases y los clientes deben poder usar una instancia extendida sin modificar su código.

2.4.3 Java Heap Space

Como lo describe [Hall, 2009], la **Java Heap Space** es el espacio asignado en memoria para las aplicaciones Java. Este espacio es controlado por los siguientes parámetros:

- ♦ XMS - eXtended Memory Size. Indica el tamaño inicial de la JHS.
- ♦ MXM - eXtended Memory maXimum size. Indica el tamaño máximo de la JHS.

2.4.4 Artefacto Java

Como artefacto para propósito de este trabajo, se define al producto resultante del proceso de empaquetamiento de la compilación de las clases, librerías, interfaces, metadatos, archivos dinámicos y estáticos de un proyecto Java. A la fecha los archivos Java Archive (JAR) empaquetan aplicaciones Java y los archivos Web Application Archive (WAR) empaquetan aplicaciones Web de Java.

2.4.5 Despliegue

El despliegue es el proceso de instalación de un componente de software en una computadora o servidor y ponerlo disponible para los usuarios finales [Gorski & Wozniak, 2021].

2.4.6 Garbage Collector (GC)

El GC es un proceso de la JVM que se encarga de eliminar objetos en memoria que no están instanciados, [Oracle Corp., 2016]. El GC consiste en un algoritmo que puede implementar diversas metodologías para eliminar objetos no instanciados, entre las metodologías más conocidas están CMS y G1GC, aunque también existen otras como SerialGC y ParallelGC, [Oracle Corp., 2018].

Cada algoritmo funciona de manera diferente, en el caso del enfoque Standard, la JHS elige el algoritmo más adecuado para la aplicación, por lo que no queda a nuestro alcance la elección del algoritmo utilizado. La Figura 5 ilustra el diagrama de despliegue con el componente JVM que incluye a los componentes GC, JHS y **Stack Memory**. La Figura 6 ilustra el funcionamiento de los algoritmos temporales de GC, tales como CMS, SerialGC y ParallelGC. El algoritmo G1GC funciona de manera diferente y más adelante se describe su funcionamiento.

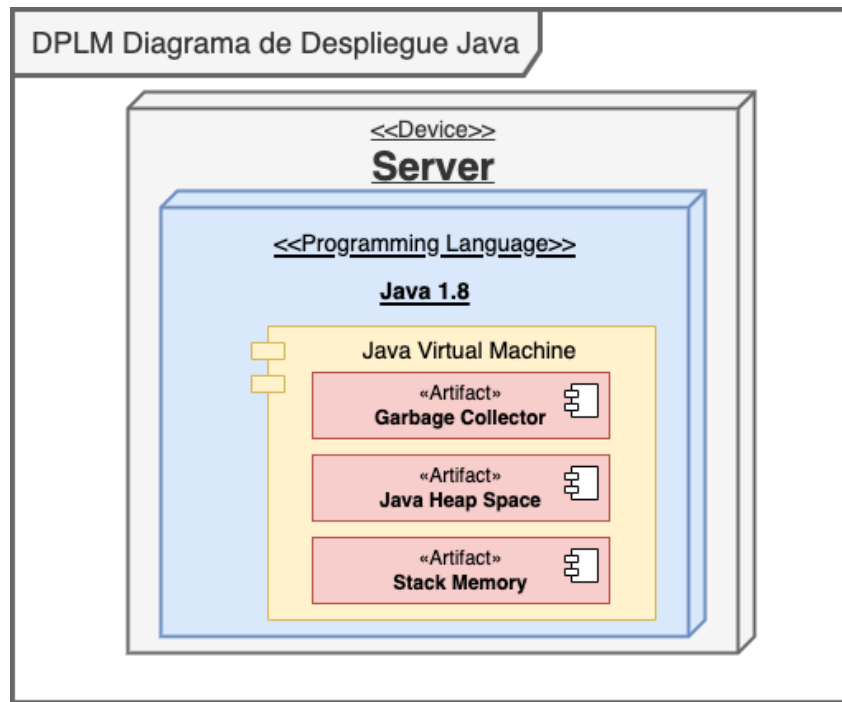


Figura 5. Diagrama de despliegue de Java 1.8.

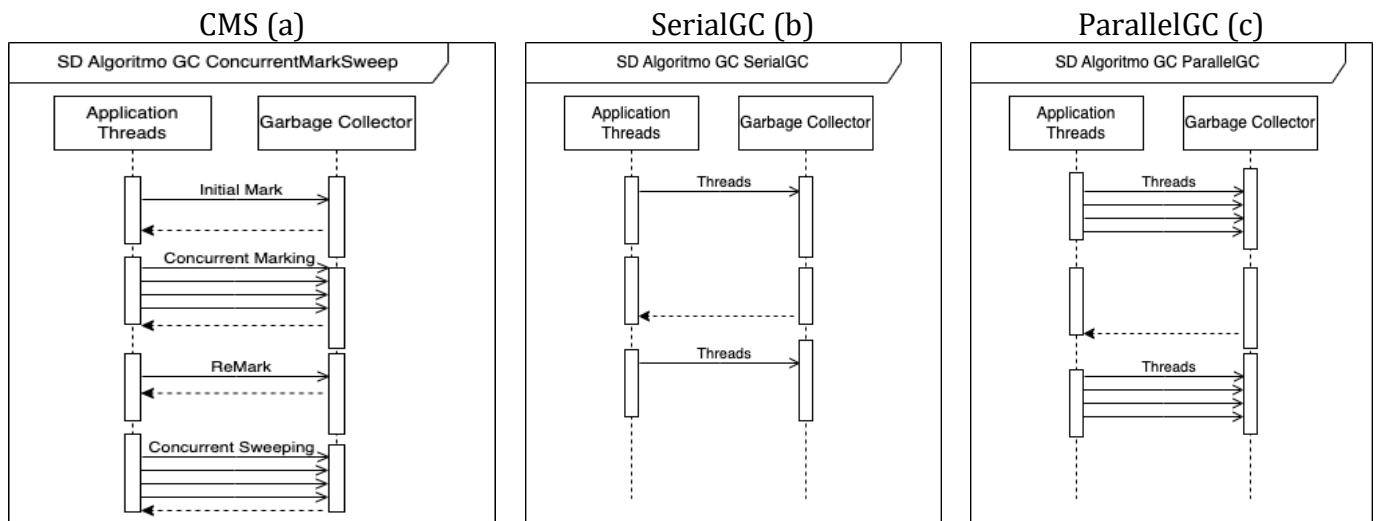


Figura 6. Diagrama de secuencia de algoritmos temporales de GC.

2.4.7 ConcurrentMarkSweep (CMS)

En versiones anteriores a Java 9 se utilizaba la metodología CMS para eliminar objetos del JHS, este proceso consiste en marcar aquellos objetos que no están referenciados, para luego el GC itera sobre estos objetos y los elimina del JHS para preparar estos espacios y reasignar los espacios que están entre objetos instanciados o referenciados y se colocan en posiciones contiguas en memoria, [Oracle Corp., 2016].

2.4.8 SerialGC

El GC serial utiliza un único **thread** o hilo para realizar todo el trabajo de recolección de basura, lo que lo hace relativamente eficiente porque no hay sobrecarga de comunicación entre **threads**. Este algoritmo de GC se adapta mejor a máquinas con un solo procesador porque no puede aprovechar el hardware multiprocesador, aunque también puede ser útil en multiprocesadores para aplicaciones con conjuntos de datos pequeños (hasta aproximadamente 100 MB), [Oracle Corp., 2024].

2.4.9 ParallelGC

ParallelGC es un algoritmo de GC generacional similar al recopilador serial y se recomienda para dispositivos con un procesador de más de 8 threads, hilos o subprocesos, que se utilizan para acelerar la recolección de basura. Es la opción más recomendada para usar en aplicaciones no interactivas donde el rendimiento de la aplicación es más importante que los tiempos de pausa bajos, [Oracle Corp., 2024].

2.4.10 G1 First (G1GC)

De acuerdo con [Oracle Corp., 2018], la metodología G1GC en la versión Java 7 es opcional y a partir de la versión Java 9 en adelante se utiliza por defecto. G1 First se enfoca en aplicaciones que manejan grandes volúmenes de información y plataformas que se instalan en servidores multiprocesador. En esta metodología G1 First los objetos Java dentro del JHS se clasifican en tres tipos de espacios:

- Young Generation: Área de la JHS en la que se almacenan los nuevos elementos, y en esta se hace un menor uso del GC.
- Survivor Space: Área de la JHS que contiene los elementos que han sobrevivido de la Young Generation. Esta área se divide en espacios S0 y S1.
- Old Generation: Área de la JHS que almacena los elementos que han sobrevivido durante un largo periodo de tiempo. Estos elementos se han movido del área Young a esta área.

G1 First no es un GC en tiempo real como lo son SerialGC, ParallelGC y CMS. G1 First maneja espacios de memoria generacional para cumplir con los objetivos de tiempo de pausa de GC con alta probabilidad mientras logra un alto rendimiento con poca necesidad de configuración. G1 First mantiene un equilibrio entre latencia y rendimiento utilizando aplicaciones y entornos de destino actuales.

G1 First divide la JHS en:

- ◆ Young Generation
 - Survivor Space 0
 - Survivor Space 1
 - Eden Space
- ◆ Old Generation

El G1 First funciona en dos fases:

- ◆ Fase solo jóvenes

Marco Conceptual

- Inicio simultáneo
- Observación
- Limpieza
- ♦ Fase de recuperación de espacio
 - Evacua espacios

La Figura 7 ilustra el funcionamiento de la GC de tipo generacional, dividida en Young Generation con regiones identificadas como E de Eden y S de Survivor, además del Old Generation identificadas las regiones en color Azul con la letra O. Las regiones en gris son espacios libres o no asignados, [Oracle Corp., 2018].



Figura 7. Organización del Heap del GC G1 First [Oracle Corp., 2018].

2.4.11 Pausas G1GC

El tiempo de pausa es la duración la cual el GC detiene la aplicación y recupera el espacio que ya no está en uso, [Oracle Corp., 2024].

2.4.12 Ratio G1GC

El parámetro NewRatio permite controlar el tamaño del espacio de la generación Young. El espacio de JHS se divide en generación Young y Old. La generación Young consta del espacio Eden y dos espacios de supervivientes (S0 y S1). Un espacio superviviente está vacío en cualquier momento y sirve como destino de los objetos activos del espacio Eden y el otro espacio superviviente durante la recolección de basura; después de la recolección de basura, el espacio Eden y el espacio de supervivencia de origen están vacíos. La Tabla 1 describe algunas opciones para el parámetro NewRatio para establecer el tamaño del espacio de la generación Young.

Tabla 1. Opciones de configuración para el parámetro NewRatio de G1GC.

Valor NewRatio	Descripción
1	La generación Young puede ser la mitad del JHS.
2	La generación Young puede ser la tercera parte del JHS.
3	La generación Young puede ser la cuarta parte del JHS.

2.4.13 Region G1GC

El tamaño de una región de JHS se determina de acuerdo con el tamaño máximo de la JHS, calculado para representar aproximadamente 2048 regiones y el tamaño debe ser una potencia de 2, y los valores válidos son de 1 a 32 MB a menos que se establezcan utilizando la opción -XX:G1HeapRegionSize, [Oracle Corp., 2024].

2.4.14 JHS Deduplication

La **deduplication** de cadenas de caracteres es una técnica que minimiza el espacio de almacenamiento y el ancho de banda mediante la detección de la redundancia de datos [N. Kumar et al., 2016]. GC G1 First permite que múltiples objetos String que contienen el mismo valor compartan el mismo arreglo internamente, a este mecanismo se le llama **deduplication** de cadenas de caracteres, [Oracle Corp., 2024].

El funcionamiento de la **deduplication** se muestra en la Figura 8, en donde el objeto Java.lang.String genera un string en la JHS, cuando se crea un segundo Java.lang.String idéntico al primero, este segundo solo hace referencia al primer String generado, el segundo String ya no es generado en JHS, por lo que ese espacio no es ocupado, de esta manera se puede evitar el desbordamiento de memoria. El diagrama de secuencia que se ilustra en la Figura 9, muestra la interacción cuando se habilita esta opción del algoritmo G1GC, desde el cliente que envía una solicitud, el servidor recibe la solicitud y redirige al servidor de aplicaciones, que a su vez dirige a la JVM y esta administra a la JHS, donde se crea el String y envía la respuesta, en la segunda solicitud solo se solicita la respuesta, esto quiere decir que el String BASE64 de la respuesta del SWA ya no se genera internamente.

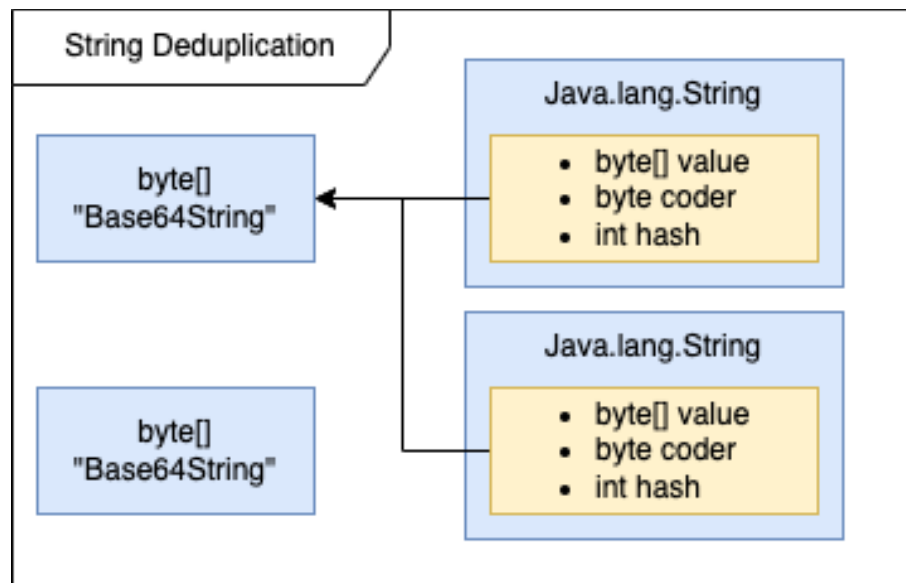


Figura 8. Funcionamiento de G1 First Deduplication en JHS.

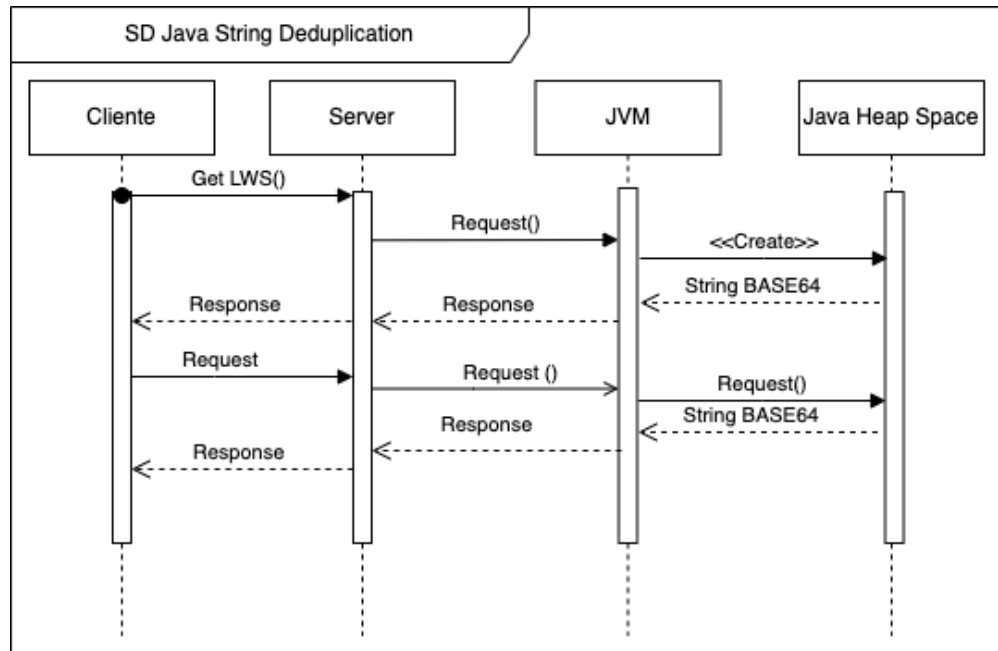


Figura 9. Interacción de un SWA con G1 First Deduplication en JHS.

2.4.15 Acuerdo de Nivel de Servicio (SLA)

De acuerdo con [Shang et al., 2020], el acuerdo de nivel de servicio es un contrato entre un proveedor de servicios y un cliente que define el nivel de servicio que se espera del proveedor. Los proveedores de servicio suelen tener SLA que cubren el tiempo de actividad de las máquinas virtuales, el hardware y las redes.

2.4.16 Elasticidad

De acuerdo con [Rossi et al., 2020], una aplicación elástica puede asignar y desasignar dinámicamente recursos informáticos para manejar su carga de trabajo, por lo que indica que existen dos tipos de implementación de la elasticidad:

- ◆ Elasticidad Vertical. Escalar la cantidad de recursos informáticos asignados a cada uno de ellos frente a variaciones de carga de trabajo.
- ◆ Elasticidad Horizontal. Escalar el número de contenedores frente a variaciones de carga de trabajo.

2.4.17 Contenerización

La contenerización es un mecanismo de virtualización liviano para respaldar la carga e implementación rápida de aplicaciones. Una aplicación en contenedor se basa en el kernel del sistema operativo *host*. Compartir el kernel y evitar la necesidad de un sistema operativo más agresivo generalmente produce un mejor *performance* y una huella reducida, [Simonsson et al., 2019]. Por *footprint* o huella se entiende como la utilización de los recursos de cómputo que ocupa una aplicación o un software [Whatism.com, 2005].

Docker es una tecnología de habilitación clave para microservicios, un estilo arquitectónico que promueve servicios simples, poco acoplados y actualizables individualmente organizados en torno a productos comerciales [Ramadan & Mohamed, 2016]. Un contenedor Docker encapsula una aplicación y sus dependencias de software; se puede ejecutar en diferentes máquinas Linux con el motor Docker. Se basa en las características del kernel de Linux, como los espacios de nombres y los grupos de control, para aislar los contenedores independientes que se ejecutan en la misma instancia del sistema operativo, [Plauth et al., 2017].

2.4.18 Tolerancia a Fallas

La tolerancia a fallas es una estrategia para crear aplicaciones de software más confiables y resilientes contra fallas u eventos inesperados, [Hosseini et al., 2020]. Para el caso de las pruebas de capacidad de entrega de SWA, el error más común es el Java OutOfMemoryError, que indica que no hay espacio en memoria para generar más objetos Java. En el trabajo de tesis de [Benítez, 2018], se mencionan los mensajes de error por tipo de implementación de SWA.

2.5 Calidad de Software

2.6.1 Modelo de Calidad

El modelo de calidad es el conjunto de características y las relaciones entre las mismas, que proporcionan la base para especificar requerimientos de calidad y para evaluar la calidad, [ISO/IEC, 2000].

2.6.2 Calidad Interna

Es la totalidad de las características del producto de software desde una vista interna. La calidad interna se mide y se evalúa en función de los requisitos de calidad internos. Los detalles de la calidad del producto de software se pueden mejorar durante la implementación, revisión y prueba del código, pero la naturaleza fundamental de la calidad del producto de software representada por la calidad interna permanece sin cambios a menos que se rediseñe.

Puede ser medida y evaluada por medio de atributos estáticos de documentos tales como:

- ◆ Especificación de requerimientos.
- ◆ Arquitectura o diseño.
- ◆ Piezas de código fuente, entre otros.

En etapas tempranas del ciclo de vida del software es posible medir, evaluar y controlar la calidad interna de estos productos. Sin embargo, asegurar la calidad interna no es generalmente suficiente para asegurar la calidad externa.

2.6.3 Calidad Externa

Es la totalidad de las características del producto de software desde una visión externa. Es la calidad cuando se ejecuta el software, que suele medirse y evaluarse mientras se prueba en un entorno simulado con datos simulados utilizando métricas externas. Durante las pruebas, la mayoría de los fallos deben ser descubiertos y eliminados. Sin embargo, algunas fallas pueden permanecer después de la prueba. Dado que es difícil corregir la arquitectura del software u otros aspectos fundamentales del diseño del software, el diseño fundamental suele permanecer inalterado a lo largo de las pruebas. En la Figura 10 se ilustra el diagrama de calidad interna y calidad externa que se especifica en [ISO/IEC, 2014], el cual consta de 6 atributos con sus respectivos sub atributos.

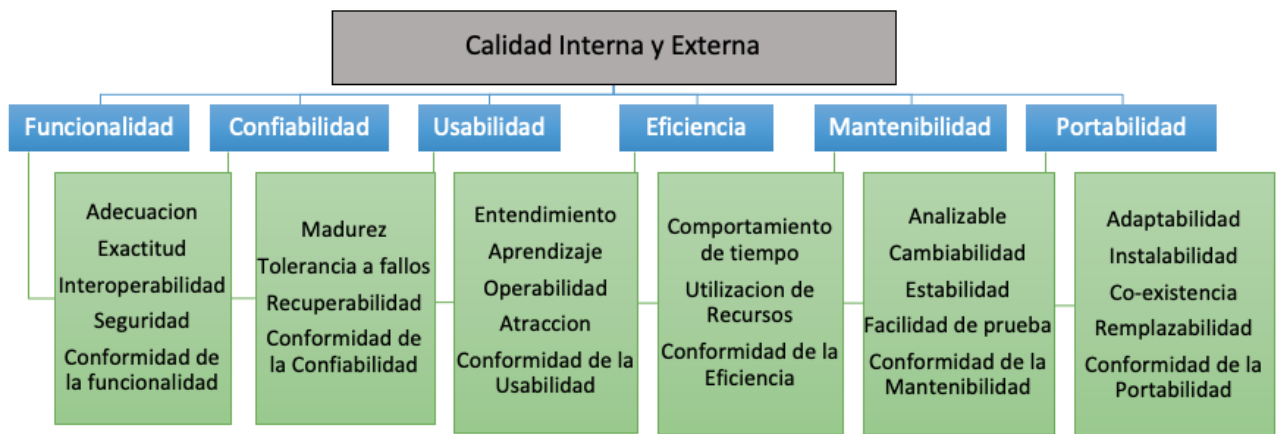


Figura 10. Modelo de dos partes para la calidad del producto de software [ISO/IEC, 2014].

2.6.4 Calidad en Uso

Es la capacidad del producto de software para permitir que usuarios específicos logren objetivos específicos con efectividad, productividad, seguridad y satisfacción en contextos de uso específicos. La Figura 11 muestra el diagrama de la calidad en uso de [ISO/IEC, 2014].



Figura 11. Modelo de calidad para la Calidad en Uso [ISO/IEC, 2014].

2.6.5 Atributos de Calidad

El termino de atributo de calidad es el conjunto de características y sub características de un modelo de calidad, [Zubrow, 2004]. La Figura 12 muestra el modelo de calidad con características y sub características.

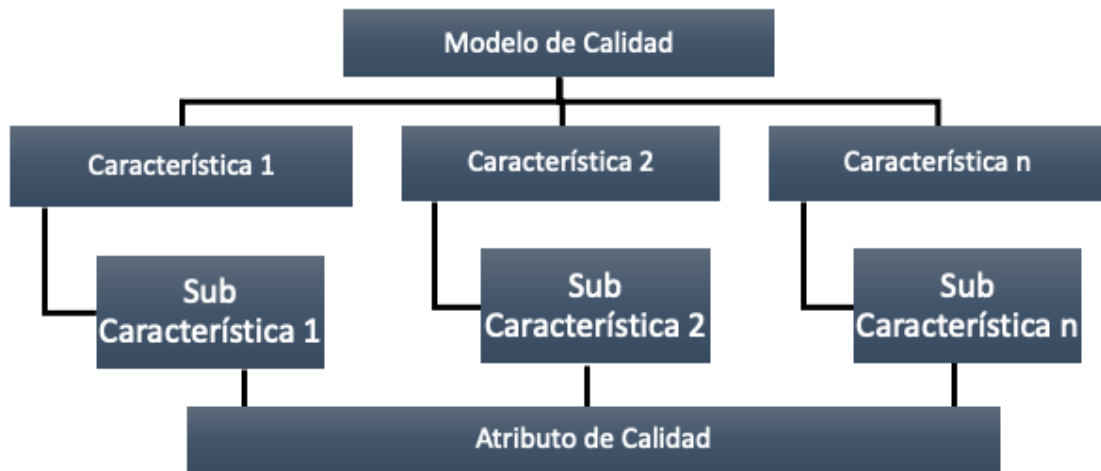


Figura 12. Modelo de Calidad de [Zubrow, 2004]

2.6 Medición

2.6.1 Teoría de la Medición

La necesidad de medir es evidente en la mayoría de las actividades técnicas o científicas. Sin embargo, no interesa sólo contar con medidas sino también saber si dichas medidas son válidas. De acuerdo con [Fenton & Pfllege, 1997], “medir es el proceso por el cual se asignan números o símbolos a atributos de entidades del mundo real de tal forma que los describa de acuerdo a reglas claramente definidas”. De acuerdo con [Baird, 1991], una **magnitud física** es un atributo de un cuerpo, un fenómeno o una sustancia, que puede determinarse cuantitativamente, es decir, es un atributo susceptible de ser medido.

2.6.2 ¿Qué es medir?

De acuerdo con [Baird, 1991], es el proceso de cuantificar nuestra experiencia del mundo exterior. Medir es comparar una cantidad de una magnitud con otra cantidad de la misma magnitud fijada arbitrariamente como unidad.

2.6.3 Protocolo de Medición

De forma general, el protocolo que propone que toda medición experimental debe seguir estos principios:

- ◆ Definir el objeto de la medición.
- ◆ Selección del instrumento de medición.
- ◆ Estimación de la incertidumbre.
- ◆ Identificación de errores
- ◆ Repetición de las mediciones
- ◆ Análisis estadístico de los resultados.

- ◆ Reporte adecuado de los resultados.

De acuerdo con [Fenton & Bieman, 2014], la medición de software es un componente esencial de una buena ingeniería de software.

- ◆ Los desarrolladores de software miden las características de su software para tener una idea de si los requisitos son consistentes y completos, si el diseño es de alta calidad y si el código está listo para ser lanzado.
- ◆ Los gerentes de proyectos efectivos miden los atributos de los procesos y productos para poder decir cuándo el software estará listo para su entrega y si se excederá un presupuesto.
- ◆ Las organizaciones utilizan medidas de evaluación de procesos para seleccionar proveedores de software.
- ◆ Los clientes informados miden los aspectos del producto final para determinar si cumple con los requisitos y es de calidad suficiente.
- ◆ Los encargados de mantenimiento deben poder evaluar el producto actual para ver qué se debe actualizar y mejorar.

2.6.4 Experimentación

El proceso o sistema productivo puede ser visualizado como una combinación de máquinas, métodos, personas, y otros recursos que transforman alguna entrada (materia prima), en una salida que tiene una o más respuestas observables, [Montgomery, 2004]. Todo proceso puede mostrarse mediante el modelo que se muestra en la Figura 13. El diseño experimental busca "medir probabilísticamente la relación causal que se establece entre las variables y estar en posibilidad de aceptar o rechazar las hipótesis sometidas a prueba" [Rojas, 2013].

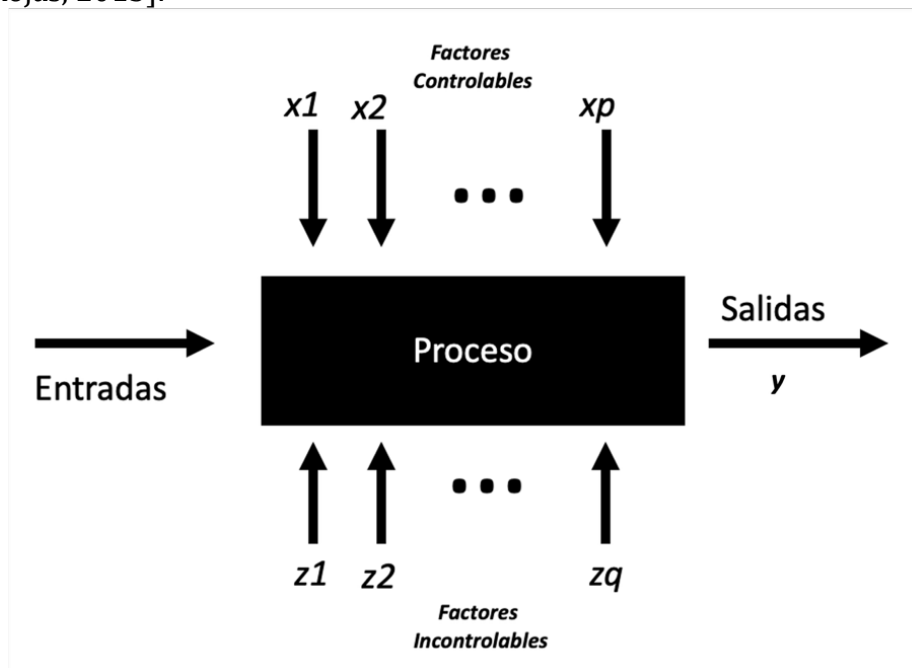


Figura 13. Modelo general de un proceso [Montgomery, 1991].

En general, los experimentos se usan para estudiar el desempeño de procesos y sistemas. El proceso o sistema puede representarse con el modelo ilustrado en la Figura 13, que se visualiza una entrada que se transforma en una salida, y que además tiene una o más respuestas observables. Algunos factores del proceso $x_1, x_2, \dots, x_p$ son controlables, mientras que otros factores $z_1, z_2, \dots, z_q$ son no controlables. Los objetivos del experimento podrían comprender los siguientes:

1. Determinar cuáles son las variables que tienen mayor influencia sobre la salida y .
2. Determinar cuál es el ajuste de las x que tiene mayor influencia para que la salida o y , esté casi siempre cerca del valor nominal deseado.
3. Determinar cuál es el ajuste de las x que tiene mayor influencia para que la variabilidad de y sea reducida.
4. Determinar cuál es el ajuste de las x que tiene mayor influencia para que los efectos de las variables no controlables $z_1, z_2, \dots, z_q$ sean mínimos.

Capítulo III. Trabajos Relacionados

En este capítulo se presentan los trabajos relacionados que más se apegan a este contexto, los cuales se clasificaron en cinco grupos: modelos de QoS, aplicaciones, evaluaciones, evaluación de la calidad de los servicios web SOAP y REST, comparación de arquitecturas de servicios Web.

3.1. Trabajos Relacionados

Se documentaron en 59 trabajos clasificados de acuerdo al propósito que describen, entre estas clasificaciones se encuentran modelos de calidad de servicios, aplicaciones y evaluaciones de atributos de calidad, comparación de la calidad de los servicios Web SOAP y REST, y comparación de arquitecturas de servicios Web SOAP y REST. El objetivo final de la búsqueda de trabajos relacionados es encontrar los atributos de calidad de servidores y de clientes que intervienen en la entrega de recursos a través de servicios web.

Cabe mencionar que, aunque la calidad de servicios (**QoS**) es un tema muy recurrente en trabajos de investigación, existen pocos trabajos que abordan dicho tema en aspectos de infraestructura como el hardware, sistema operativo, software de aplicaciones de servicios, atributos del servidor y atributos del cliente. La mayoría de los trabajos se enfocan en atributos de calidad de servicio que ya son comunes, y que para propósitos de este trabajo denominaremos “Clásicos”, por ejemplo: **performance**, tiempo de respuesta, **throughput**, disponibilidad, latencia, líneas de código, modularidad, número de clases, etc. En el Anexo A. Trabajos Relacionados, se describen más extensamente todos los trabajos relacionados.

3.1.1 Métricas Clásicas

En las tablas 2, 3, y 4 se establece una columna en la que se definen las métricas clásicas, las cuales, para desarrollo y propósito de este trabajo, son aquellas métricas de servicios Web que se encuentran definidas en [ISO/IEC, 2014] o que son métricas evaluadas concurrentemente como, por ejemplo: **performance**, tiempo de respuesta, **throughput**, disponibilidad, latencia, líneas de código, modularidad, número de clases, etc.

3.1.2 Métricas ISO 25000

En las tablas 2, 3, y 4 se establece una columna en la que se definen las métricas ISO 25000, las cuales, para desarrollo y propósito de este trabajo, son aquellas categorías de métricas de SW y que se encuentran definidas como métricas internas, externas y en uso, como se definen en [ISO/IEC, 2014].

Los enfoques encontrados en los trabajos relacionados se describen en las secciones:

- 1) Modelos de QoS.
- 2) Aplicaciones.
- 3) Evaluaciones.
- 4) Evaluación de la calidad de los servicios Web SOAP y REST.
- 5) Comparación de Arquitecturas de Servicios Web SOAP y REST.

3.2. Modelos de QoS

En esta categoría se agrupan trabajos de investigación que proponen modelos de QoS con atributos de calidad. En este apartado solo se listan las características de cada trabajo, pero en el Anexo A. Descripción de los Trabajos Relacionados, específicamente en la sección 7.1 Modelos de QoS se describen cada uno de los trabajos que aquí se listan.

Los trabajos reportados en la Tabla 2 corresponden a la sección de modelos de QoS, estos trabajos indican que los atributos de calidad no solo intervienen en la arquitectura del SW, sino también en la arquitectura de hardware, en la arquitectura de las aplicaciones que intervienen con el SW. Algunos otros trabajos indican que las políticas de negociación de los servicios proveedor-cliente, también influyen en la calidad de servicios.

En relación a este trabajo de tesis, los modelos de QoS permiten identificar los enfoques existentes entre la calidad de los servicios, ζ y la relación existente entre sus atributos. Además, estos modelos apoyan la identificación de atributos y permiten determinar cuáles son los más adecuados para evaluar el desempeño, comportamiento y calidad de los servicios Web.

En esta categoría se presentan trabajos de investigación que proponen modelos de QoS, los cuales presentan atributos de calidad que pretenden complementar una solución a uno o a una gran variedad de problemas, es decir, en ocasiones son soluciones abiertas a complementar distintos objetivos y con diferentes enfoques. La diferencia de estos enfoques con el trabajo que se describe en este documento es que en este último se consideran los RA en la entrega de SW, y el conjunto de atributos del cliente y del servidor.

Trabajos Relacionados

Tabla 2. Trabajos relacionados que proponen modelos de QoS.

Artículo	Proponen	Evaluación	Herramientas	Métricas ISO 25000	Clasificación propia
[Kropp & Torre, 2020]	Descripción de Docker en el despliegue aplicaciones referente a otras aplicaciones de virtualización.	Tecnologías Docker vs Virtualización.	N/A.	N/A.	N/A.
[Begin et al., 2020]	Modelo basado en probabilidad estadística para calcular el performance de servicio de video.	Performance.	Modelos matemáticos.	En Uso.	Clásicas y otras.
[Guerrón et al., 2020]	Modelo basado en una SLR sobre la evaluación de los SW en cloud.	Resultado de SLR.	SLR.	Internas – Externas.	Clásicas.
[Z. Chen et al., 2020]	Modelo de predicción.	Eficiencia del modelo.	Métricas simples.	Externas.	Clásicas.
[Eisa et al., 2020]	Modelo de selección de SW con un enfoque multicriterio.	Evalúa proveedores de servicios.	AHP basado en MCDM y Algoritmo SAW.	Internas – Externas.	Clásicas.
[Abosaif & Hamza, 2020]	SLR de los algoritmos de selección de IoT.	Algoritmos de selección de servicios.	SLR.	Externas.	Clásicas.
[Al-Fraihat et al., 2020]	Un modelo integral de SW basado en una SLR.	Utilidad.	SLR.	En Uso.	Otras.
[P. Kumar & Dezfouli, 2019]	Implementación del protocolo QUIC para aplicaciones basadas en IoT.	Performance de protocolos IoT.	Métricas simples y Modelos Matemáticos.	Externas.	Clásicas.
[Torres Kompen et al., 2019]	Presentación de un conjunto de lineamientos para utilizar los ambientes de aprendizaje personal.	Ambiente de aprendizaje personal.	Instrumentos de investigación.	Externas.	Otras.
[Padhy et al., 2018]	Modelo llamado “Representación de pronóstico de reutilización de software resistente al envejecimiento”.	Reusabilidad.	N/A.	Internas.	Clásicas.
[Abdel-Basset et al., 2018]	Modelo para estimar servicios Cloud.	Rendimiento .	N/A.	Internas – Externas.	Clásicas.
[Bakri et al., 2015]	Modelo de implementación de QUIC en aplicaciones 3D.	Las versiones de protocolo HTTP.	N/A.	N/A.	N/A.
[Zela Ruiz & Rubira, 2016]	Herramienta y modelo de QoS para monitorear SW.	Rendimiento y precisión.	Resultados del monitoreo.	Externas.	Clásicas.
[Ardagna et al., 2014]	Modelo de QoS cloud.	Estados ocultos de la secuencia.	Algoritmos.	Internas – Externas.	Clásicas.
[Farsandaj & Ding, 2012]	Modelo modificado de exploración Scatter/Gather.	Modelo modificado de exploración Scatter/Gather.	Algoritmos y Modelos Matemáticos.	Internas – Externas.	Clásicas.
[Li et al., 2010]	Un grupo de parámetros de QoS desde la perspectiva del usuario.	Servicio de Correo Electrónico.	Métricas simples y Modelos Matemáticos.	Externas.	Clásicas.

3.3. Aplicaciones

El objetivo de los trabajos que proponen aplicaciones que se listan en esta sección, es mejorar la capacidad de los servicios web, los cuales son objetivos muy específicos para lograr sus propósitos. Algunas soluciones de estas aplicaciones están enfocadas en mejorar la infraestructura para que los servicios puedan ser utilizados por sus consumidores. Algunas otras aplicaciones, que se consideran proactivas, tienen el objetivo de monitorear ya sea el procesamiento, los datos, el tráfico en la red, etc., para poder actuar ante una inminente falla o caída del servicio.

En esta sección solo se listan las características de cada trabajo, pero en el Anexo A. Descripción de los Trabajos Relacionados, específicamente en el punto 7.2 Aplicaciones, se describen cada uno de los trabajos que aquí se listan.

En relación con este trabajo de tesis, algunos trabajos que se listan en esta sección se pueden considerar algunos métodos o aplicaciones que se proponen en los trabajos listados como en [Aghdaie & Tamir, 2009], y en [Portillo et al., 2016], que proponen estrategias para atención de solicitudes de servicios. Otros trabajos que presentan herramientas de monitoreo del comportamiento de servicios como en [Buccafurri et al., 2009] y trabajos que proponen la implementación de la tolerancia a fallos como se menciona en el trabajo que propone [Aghdaie & Tamir, 2009].

La principal diferencia del trabajo de este trabajo de tesis con los trabajos relacionados que se encontraron y que se mencionan a continuación, es que en estos últimos no se presentan atributos del servidor o del cliente. Dentro de la arquitectura cliente-servidor, el servidor es el que mayor atención recibe por parte de los investigadores, y en su mayoría proponen mejorar su infraestructura mediante las aplicaciones que se proponen, y aunque en la Tabla 3 solo existe un trabajo que se enfoca en monitorear los datos del cliente, el objetivo es mejorar la infraestructura del servidor.

Finalmente, los trabajos relacionados que presentan o proponen aplicaciones que se enfocan en mejorar la capacidad de los SW se listan en la Tabla 3.

Trabajos Relacionados

Tabla 3. Trabajos relacionados que presentan aplicaciones.

Artículo	Proponen	Evalúan	Herramientas	Métricas ISO 25000	Clasificación propia
[Rossi et al., 2020]	Marco de software elástico y adaptable a atributos de red.	Elasticidad de la aplicación.	Métricas simples con Modelos Matemáticos.	Externas y En uso.	Clásicas y otras.
[Shang et al., 2020]	Modelo de tarificación servicios de computación en cloud.	Propensión a riesgo.	Modelos Matemáticos.	En uso.	Otras.
[Simonsson et al., 2019]	Sistema de inyección de fallas.	Capacidad de inyección de Fallas.	Métricas simples.	Externas.	Clásicas.
[Haroune-Belkacem et al., 2019]	Técnica para compresión de mensajes XML.	Performance.	Algoritmos.	Internas - Externas y En uso.	Clásicas y otras.
[Guha Roy et al., 2019]	Servicio blockchain para servicios basados en IoT.	Blockchain.	Modelos Matemáticos.	Internas – Externas.	Clásicas.
[Stan et al., 2019]	Red elástica de recopilación de contenido multimedia.	Correctitud, escalabilidad, sobrecarga del sistema.	Métricas simples.	Externas y En uso.	Clásicas.
[Portillo-Domínguez et al., 2016]	Balanceo de carga basada en el Garbage Collector .	Throughput y tiempo de respuesta.	Algoritmos.	Internas – Externas.	Clásicas.
[Abawajy et al., 2015]	Marco de gestión de SLA basado en capacidades.	Performance del marco.	Métricas simples.	Internas – Externas.	Clásicas.
[X. Chen et al., 2014]	Marco recomendación de SWA a partir de QoS.	Precisión de la predicción.	Algoritmo.	Externas.	Clásicas y otras.
[Jae et al., 2012]	Administración de Smart Web Services .	Frecuencia de monitoreo en dispositivos móviles.	Cálculo aritmético.	Externas.	Otras.
[Kocabicak & Dural, 2012]	Marco flexible orientada a la estructura de SW.	N/A.	N/A.	Externas.	Clásicas.
[Md Fudzee & Abawajy, 2011]	Método de adaptación de contenido orientada al SW.	Método de adaptación de contenido orientada al servicio.	Métricas simples con Modelos Matemáticos.	Externas.	Clásicas.
[Wu et al., 2011]	Arquitectura para integrar servicios geoespaciales.	Tiempo de respuesta.	Algoritmo.	Externas y En uso.	Clásicas.
[Barros et al., 2011]	Marco para la gestión de servicios Web Semánticos.	Marco propuesto.	Algoritmos.	Externas.	Clásicas.
[Fudzee et al., 2010]	Marco de adaptación de contenido orientado a servicios.	Eficiencia en la selección de servicios.	Métricas simples.	Internas – Externas.	Clásicas.
[Fang Fang & Chien Sing, 2009]	Marco para integrar SOA con procesos de aprendizaje.	N/A.	N/A.	N/A.	N/A.
[Aghdaie & Tamir, 2009]	Herramienta Fault-Tolerant con soporte In-Flight.	Performance.	Métricas simples.	En Uso.	Clásicas.

3.4. Evaluaciones

Los trabajos que se presentan en esta categoría están enfocados en evaluar la calidad interna o externa de los servicios web. En la mayoría de los casos, el objetivo de estos trabajos es mejorar la eficiencia de los SW con respecto a algunos de sus atributos, a partir de los resultados de sus evaluaciones. En esta categoría existe una gran diversidad de propuestas, es decir, algunos de los métodos de evaluación que figuran son algoritmos, técnicas, ecuaciones o modelos matemáticos, métodos de pronóstico de resultados, etc.

En este trabajo de tesis, el propósito es evaluar la capacidad de entrega de recursos a través de servicios Web a partir de un modelo de QoS, basado en atributos de clientes y servidores. El modelo de capacidad de entrega de SWA debe permitir discernir entre atributos positivos, negativos y la relación entre los atributos, para poder hacer una estimación de los atributos y otorgar valores de referencia a los usuarios, a los encargados de las plataformas de SWA. Un caso similar es el estudio de [Calvo et al., 2017], en el que los autores proponen un enfoque de evaluación basado en datos de gran volumen, de acuerdo a la carga de trabajo del servidor y el tráfico de la red.

La mayoría de los trabajos que proponen evaluaciones basadas en atributos como el *performance*, tiempo de respuesta, latencia, disponibilidad, entre otros; que en este trabajo se consideran como clásicos. Es importante mencionar que todas las categorías de los atributos evaluados de esta sección, están consideradas en el modelo de [ISO/IEC, 2014], ya sea de calidad interna, calidad externa o calidad en uso.

En la Tabla 4 se listan los trabajos relacionados que presentan métodos de evaluación de la de calidad de servicios, y cabe mencionar que, la mayoría de estas evaluaciones se realizan del lado del servidor o están enfocadas en atributos internos del servidor, solo el trabajo de [Roquero et al., 2020] considera la entidad del cliente para hacer estimaciones de la evaluación.

En esta sección solo se listan las características principales de cada trabajo, en el Anexo A. Descripción de los Trabajos Relacionados, específicamente en el punto 7.3 Evaluaciones, se describen de manera detallada cada uno de los trabajos que aquí son mencionados.

Trabajos Relacionados

Tabla 4. Trabajos relacionados que proponen evaluaciones de QoS.

Artículo	Propuesta	Evaluación	Herramientas	Métricas ISO 25000	Clasificación
[Song, 2021]	Método predictivo de confiabilidad de SW basado en aprendizaje automático.	Precisión de la predicción.	Algoritmos.	Externas.	Otras.
[Fayos-Jordan et al., 2020]	Comparación exhaustiva, funcional y de performance de tecnologías de cómputo fog en el contexto de la IoT .	Kubernetes y Docker Swarm.	Pruebas homogéneas y heterogéneas.	Externas.	Otras.
[Yaghoubi & Maroosi, 2020]	Método de composición de SW basado en un algoritmo de optimización Multi Universo.	Confiabilidad, disponibilidad, tiempo de respuesta y costo.	Algoritmos.	Externas.	Clásicas.
[Roquero et al., 2020]	Método de optimización de datos y monitoreo distribuido del cliente.	Rendimiento y tráfico de red.	Monitoreo de red.	Externas.	Clásicas y otras.
[Potdar et al., 2020]	Evaluación comparativa entre contenedores Docker y máquinas virtuales.	Performance.	Herramientas y pruebas.	Externas.	Clásicas y Otras.
[Tibermachine et al., 2018]	Método de estimación de reputación de SW.	Estimación de la reputación.	Algoritmos.	Externas.	Clásicas.
[Mavridis & Karatza, 2019]	Presentación de las ventajas de ejecutar contenedores en VM.	Tecnologías de contenedores y MV's.	Herramientas de pruebas.	Externas.	Clásicas.
[Santos et al., 2018]	Impacto de la contenerización en el consumo de energía.	Herramientas de medición con modelos matemáticos.	Herramientas de Medición de Energía.	En Uso.	Otras.
[Plauth et al., 2017]	Evaluación de métricas que sean aplicables a las aplicaciones Cloud .	Throughput y memory print.	Métricas simples.	Externas.	Clásicas.
[Ramadan & Mohamed, 2016]	Implicaciones de implementación de SW en el protocolo HTTP/2.	Compara las diferentes versiones de HTTP.	Métricas simples.	Externas.	Clásicas y otras.
[Palacios et al., 2015]	Enfoque para la evaluación de servicios.	Evaluación del SLA.	Algoritmos y MCDC.	Externas.	Clásicas.
[Calvo et al., 2017]	Modelo de evaluación de QoS .	Composición de SW en base al tiempo de respuesta.	Métodos probabilísticos.	Externas.	Clásicas.
[Kamel et al., 2015]	Evaluación de QoS del lado del cliente.	Demora del tiempo de ida y vuelta de los datos.	Algoritmos.	Externas.	Clásicas y otras.
[Fudzee et al., 2014]	Estrategia de control de solicitudes basada en carga y prioridad.	Tasa de éxito.	Algoritmos.	Externas.	Clásicas y otras.
[Zheng et al., 2013]	Marco de predicción de clasificación de QoS basado experiencias colaborativas.	Marco de predicción	Algoritmos y modelos matemáticos.	Internas Externas.	Clásicas.
[Di Sorte et al., 2009]	Metodología de enrutamiento basada en la topología de red.	Algoritmos enfocados en minimizar el tráfico de red.	Algoritmos.	Externas.	Otras.

3.5. Comparación de la Calidad de los Servicios Web SOAP y REST

Esta categoría de la revisión de los trabajos relacionados está basada en dos partes: la primera es la búsqueda de modelos de calidad, relacionados a la definición de atributos de calidad que apoyan la capacidad de entrega de los SW. En este trabajo, se consideran los tipos de implementación SOAP y REST para SWA, por lo que es conveniente estudiar la arquitectura y el comportamiento de cada uno de los tipos de implementación. Estos trabajos están listados en la Tabla 5, en el enfoque Evaluación.

3.6. Comparación de Arquitecturas de Servicios Web SOAP y REST

En la Tabla 5 se listan los estudios que comparan las arquitecturas de servicios SOAP y REST. Estos estudios permiten identificar los factores, atributos y/o capacidades de dichas arquitecturas que permiten mejorar la calidad de entrega de SWA, e identificar las diferencias tanto técnicas como teóricas en cuanto a su arquitectura, interacción e implementación. En esta tabla se encuentran numeraciones entre corchetes los cuales especifican la siguiente información:

- [1] hace referencia a SOAP.
- [2] hace referencia a REST.
- [3] hace referencia a Tiempo de respuesta.
- [4] hace referencia a **Throughput**.
- [5] hace referencia a Entregabilidad

Tabla 5. Trabajos relacionados de Evaluación y Comparación de servicios Web SOAP y REST.

Enfoque	Artículo	Propuesta	Método	Descripción	SW		Resultados
					REST	SOAP	
Evaluación	[Singh & Tomar, 2016]	Modelo de reusabilidad.	Lógica Difusa.	Evalúa la reusabilidad tomando en cuenta [3]			No muestra resultados.
	[Zheng et al., 2010b]	Resultados.	Estadístico.	Evalúa la reusabilidad tomando en cuenta [3],[4]		X	Prueba estadística.
	[Buccafurri et al., n.d.]	Proyecto Quadrantis.	Métricas.	Entrega de un recurso en tiempo real.	X	X	Modelo de calidad para proveer servicios de calidad en tiempo real.
Evaluación y Comparación	[Kumari & Rath, 2015]	LoanBroker.	Métricas.	Evalúa [1],[2], y [4] tiene mejores resultados	X	X	REST tiene mejores resultados.
Comparación	[Bloebaum & Johnsen, 2015]	Análisis del flujo entre sistemas back-end y dispositivos móviles.	Estadístico.	Compara SOAP y REST en 4 escenarios distintos.	X	X	SOAP tiene mejores resultados de respuesta y REST tiene mejores resultados de performance de la vida útil de la batería.
	[Mulligan & Gracanin, 2009]	Portal Middleware .	Estadístico.	[2] presenta mejores resultados.	X	X	REST presenta mejores resultados.
	[Pautasso et al., 2008]	Modelo de decisión.	Comparación conceptual tecnológica.	[2] se debe implementar en aplicaciones empresariales.	X	X	Modelos de comparación entre SOAP y REST.
	[Malik & Kim, 2017]	Modelo de redes de actuadores.	Estadístico.	Comparación de estilos de interacción de servicios Web para dispositivos inteligentes.	X	X	Concluye que REST tiene mejores resultados de acuerdo al modelo de redes de actuadores.
	[Tihomirovs & Grabis, 2016]	SLR en términos de métricas de evaluación de software.	Revisión Sistemática.	Cada proyecto debe evaluarse individualmente mediante requisitos funcionales y no funcionales.	X	X	Comparación de las métricas de evaluación de software y al enfoque de implementación de servicios Web.

En la revisión de trabajos relacionados podemos mencionar que, para propósitos de este trabajo de tesis, se encontraron muy pocos trabajos que evalúan atributos externos a la arquitectura de los servicios web, la gran mayoría de trabajos se enfocan en atributos de calidad de los servicios web calificados como clásicos, entre ellos el **performance**, tiempo de respuesta, **throughput**, etc. Como referencia a esto podemos mencionar los trabajos de [Aghdaie & Tamir, 2009] y [Portillo et al., 2016] que hacen referencia al manejo de recursos computacionales de hardware o software como la memoria, procesamiento de CPU, etc.

Existen temas muy recurrentes, recientes, estudiados y analizados en la actualidad y que estos son factores que mejoran el rendimiento de las aplicaciones con respecto a trabajos relacionados, son los temas de orquestación, contenerización y virtualización para la implementación de servicios web, de estos podemos mencionar los trabajos relacionados como los de [Simonsson et al., 2019], [Potdar et al., 2020], [Plauth et al., 2017], [Kropp & Torre, 2020], [Mavridis & Karatza, 2019], [Rossi et al., 2020], [Santos et al., 2018], [Fayos-Jordan et al., 2020]. Los trabajos de [Bakri et al., 2015], [Kumar & Dezfouli, 2019], estudian la implementación de servicios web en otras versiones del protocolo HTTP, ya sea HTTP/2 o HTTP/3, que a este último lo nombran protocolo QUIC, y mencionan que hay ciertas ventajas para compartir recursos y/o contenido a través de servicios Web. Además, existen atributos o características que no fueron mencionados y que son importantes para el despliegue de aplicaciones de servicios, por ejemplo, el sistema operativo, y los distintos tipos de servidores de aplicaciones.

Un tema muy recurrente en los estudios que analizan la calidad de entrega de SW, es el tema de las políticas de SLA, estas políticas establecen la relación de QoS entre el cliente de SW y el proveedor de servicio **Cloud**. Los trabajos de [Yaghoubi & Maroosi, 2020], [Abawajy et al., 2015], [Ardagna et al., 2014], [Fudzee et al., 2014], [Palacios et al., 2015] y [Fudzee et al., 2010], están basados en el análisis del SLA y determinan que es un punto importante para mejorar la calidad de entrega de los servicios Web. Los trabajos de [Calvo et al., 2017], [Begin et al., 2020], [Fudzee et al., 2010] y [Buccafurri et al., 2009], se enfocan en temas de transmisión o distribución de recursos a través de plataformas de servicios Web. Existen trabajos que proponen evaluaciones, en los que automatizan el proceso a través de algoritmos, por lo que, en muy pocos casos, las evaluaciones son manuales. Por otra parte, también existen trabajos que proponen métodos para pronóstico de resultados que se pueden implementar para los SW, como el **Machine Learning**. De todo el conjunto de trabajos relacionados, solo los trabajos de [Md Fudzee & Abawajy, 2011] y [Kamel et al., 2015], consideran el lado del cliente como objetivo para monitorear las capacidades de los servicios Web.

Del conjunto de trabajos analizados, se encuentran seis trabajos secundarios, ya sea revisiones sistemáticas de literatura o mapeos sistemáticos. El trabajo de [Guerrón et al., 2020], propone un modelo basado en los atributos de QoS en ambientes **Cloud** y una clasificación taxonómica el cual está enfocado en los atributos de [ISO/IEC, 2014]. El trabajo de [Ardagna et al., 2014], estudia los métodos automatizados de gestión de QoS que pueden aprovechar la alta capacidad de programación de los recursos de hardware y software en **Cloud**, a través del estado del arte de los enfoques de modelado de QoS

aplicables al cómputo **Cloud**, por lo que propone la garantía de un nivel de servicio en dimensiones como desempeño, disponibilidad y confiabilidad. [Al-Fraihat et al., 2020] hace un estudio para identificar los factores de éxito del **E-Learning** para maximizar la eficacia de estos sistemas y a partir de estos, propone un modelo para evaluar sistemas de gestión de aprendizaje. En este estudio determina que la calidad del sistema técnico, la calidad de la información, la calidad del servicio, la calidad del sistema de apoyo, la calidad del alumno, la calidad del instructor y la utilidad percibida son determinantes en la calidad del LMS. El trabajo de [Oriol et al., 2014], presenta un mapeo sistemático de modelos de QoS de servicios web de trabajos que están basados en [ISO/IEC, 2014]. El estudio de [Tihomirovs et al., 2016], presenta una revisión sistemática de la literatura en términos de métricas de evaluación de software. El estudio de [Abosaif & Hamza, 2020] hace un estudio de estado del arte de los algoritmos de selección de servicios en ambientes IoT, basados en la capa de aplicación, capa de sensor y capa de red. En la Figura 14 se muestran los resultados obtenidos de la revisión sistemática de este último trabajo, en la cual se indica que los factores de QoS considerados en la mayoría de los estudios son el tiempo de optimización, el costo, la disponibilidad, la confiabilidad y el consumo de energía. Esta figura no presenta datos que reflejen el porcentaje como tal, solo la gráfica de pastel que podemos estimar los datos de acuerdo a la percepción.

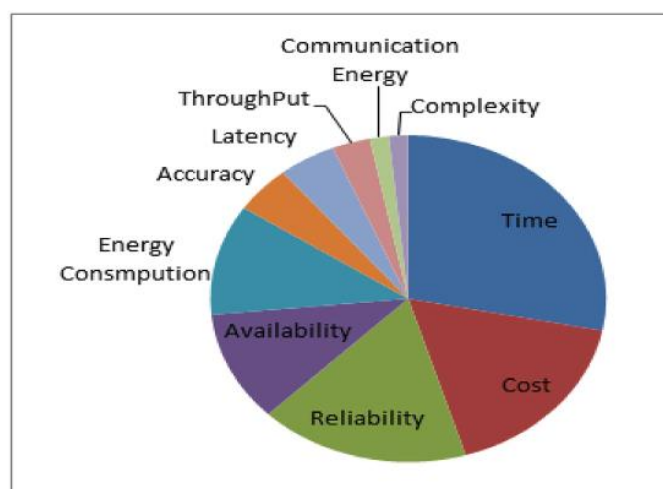


Figura 14. Factores de QoS considerados en [Abosaif & Hamza, 2020].

Con relación a este trabajo de tesis que aborda la capacidad de entrega de SWA, no existen trabajos de investigación enfocados en modelos de calidad de entrega o de evaluación de la capacidad de entrega mediante SW. Tampoco se encontraron atributos de calidad interna de clientes, o enfocados en el lenguaje de programación. En muy pocos trabajos se hace referencia a la comparativa entre SOAP y REST, aunque en este trabajo se considera que estos deberían ser más abordados, por la diferencia arquitectural que existe entre estos. Algunos temas importantes están inmiscuidos dentro de otros temas mayores, por ejemplo, el Server Cache, tolerancia a fallas y balanceo de carga, que están inmersos en temas como las CDN, que son parte de una infraestructura mayor de las arquitecturas cliente-servidor.

Capítulo IV. Metodología de Solución

4.1 Metodología de Solución

En esta sección se describe la metodología de solución y se describen los pasos, los procedimientos, herramientas y configuraciones realizadas para el desarrollo de este trabajo de tesis.

Para atender el problema descrito en la sección I. Introducción, se propusieron ocho actividades que se listan a continuación y se ilustran en la Figura 15:

- ◆ Definir los atributos de los servidores. Esta etapa trata de la búsqueda de atributos que intervienen en la entrega de recursos a través de servicios Web, en dicha búsqueda se incluyen artículos científicos, antecedentes, búsqueda Web tradicional, etc.
- ◆ Definir los atributos de los clientes. Esta etapa trata de la búsqueda de atributos que intervienen en la entrega de recursos a través de servicios Web, en dicha búsqueda se incluyen artículos científicos, antecedentes, búsqueda Web tradicional, etc.
- ◆ Definir el plan experimental a partir de los atributos encontrados.
 - Definición del plan de pruebas.
 - Creación de escenario de pruebas.
- ◆ Generación de SWA con diferentes formatos y medidas.
 - Análisis de la granularidad de los RA.
 - Generación de SWA.
 - Generación de artefactos Java.
- ◆ Ejecución de pruebas.
 - Ejecución de pruebas.
 - Recolección de los resultados.
- ◆ Obtener los resultados
 - Obtención de los archivos CSV.
 - Formateo de los datos para claridad de la información.
- ◆ Análisis de los resultados
 - Para dos muestras independientes
 - Para más de dos muestras independientes
- ◆ Definición de las métricas de entregabilidad.
 - Modelo de entrega de SWA del lado del servidor.
 - Modelo de entrega de SWA del lado del cliente.
- ◆ Evaluación de los resultados

4.2 Diagrama del Método de Solución

Para el desarrollo de este trabajo, se definió el método de solución que se ilustra en la Figura 15.

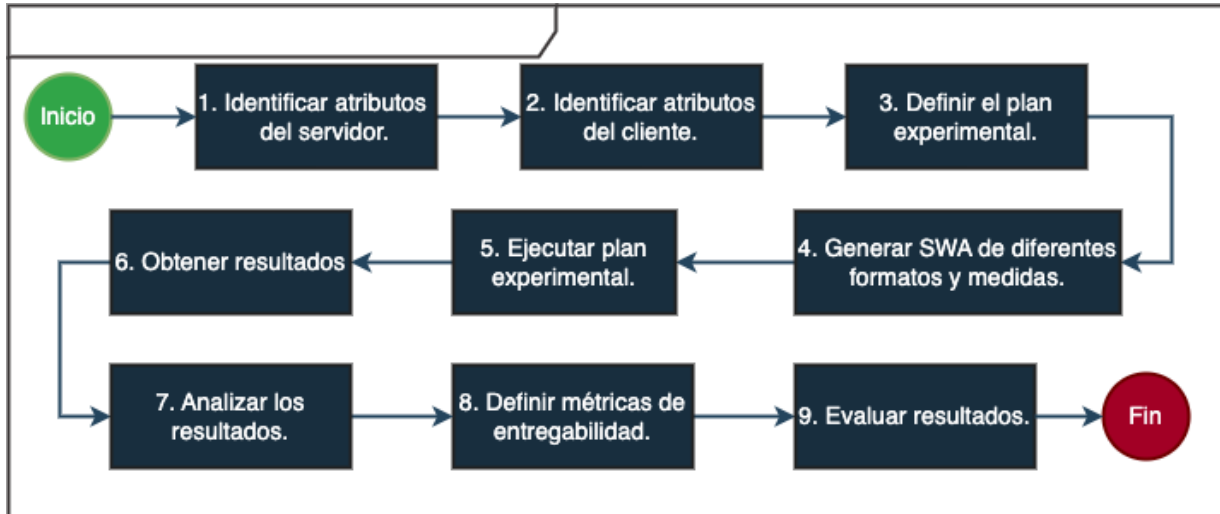


Figura 15. Diagrama de método de solución.

4.3 Identificación de Atributos Del Servidor

Como parte del trabajo realizado en la revisión en la literatura, en documentos técnicos y en internet, se identificaron los atributos de calidad de entrega de SWA del lado del servidor que se proponen para evaluar y comparar el comportamiento de los SWA con los mejores resultados de entrega.

La capacidad de entrega se evalúa de acuerdo a los resultados de las métricas del tiempo de respuesta, la tasa de éxito y el **throughput**. De tal manera que el modelo de calidad para evaluar la capacidad de entrega de SWA queda como se muestran en la Figura 16, para los atributos de calidad de entrega del lado del servidor.

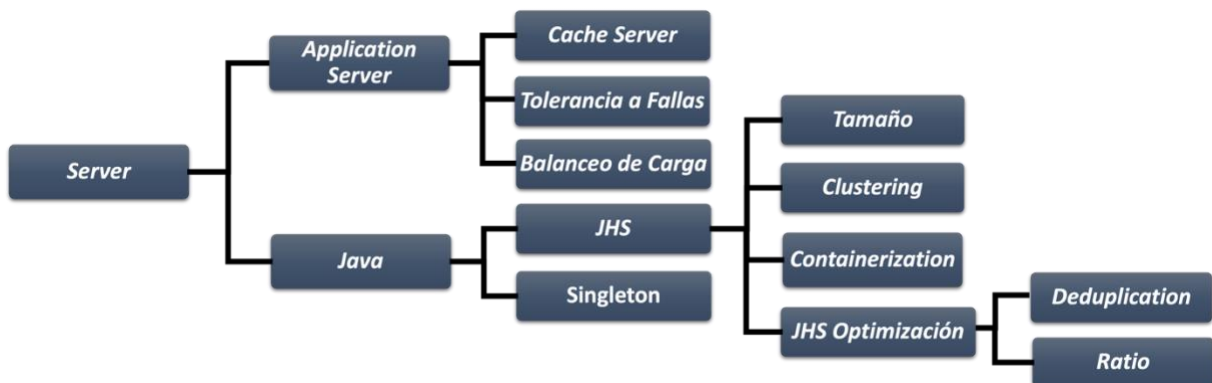


Figura 16. Modelo de QoS de entrega de SWA del servidor.

4.4 Identificación de Atributos Del Cliente

Como parte del trabajo realizado en la revisión en la literatura, en documentos técnicos y en internet, se identificaron los atributos de calidad de entrega de SWA del lado del cliente que se proponen emplear para evaluar y comparar el comportamiento de los SWA con los mejores resultados de entrega.

La capacidad de entrega se evalúa de acuerdo a los resultados de las métricas del tiempo de respuesta, el **throughput** y la tasa de éxito, de tal manera que el modelo de calidad usado para evaluar la capacidad de entrega de SWA queda como se muestran en la Figura 17, que indica los atributos de calidad de entrega de SWA del lado del cliente.

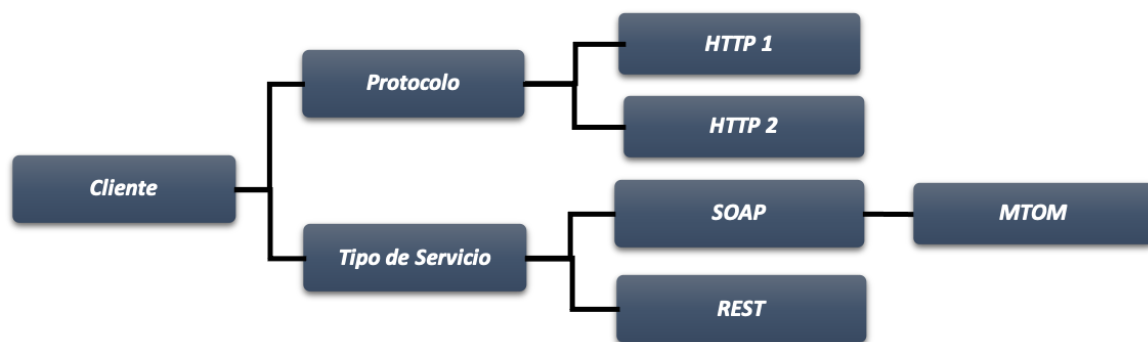


Figura 17. Modelo de QoS de entrega de SWA del cliente.

4.5 Definición del Plan Experimental

Esta sección se compone de dos fases, la primera fase consta de la definición del plan de pruebas en la que se definió la manera en la que se ejecutaron de las pruebas. En la segunda fase se definieron los escenarios de prueba de acuerdo a los atributos encontrados en los puntos 1 y 2 de la metodología de solución.

4.5.1 Plantilla de Plan de Pruebas

Para probar el desempeño de los SWA se diseñó un plan de pruebas, el cuál consta de 25 grupos de solicitudes y 100 solicitudes de SWA por cada grupo de solicitudes. En total 2500 solicitudes por ejecución de plan de pruebas, se realizaron 5 ejecuciones por prueba, 125,000 solicitudes en total por caso de prueba. En la Tabla 6 se muestran los detalles de la plantilla del plan de pruebas, en la que se recolectaron los datos de cada una de las ejecuciones de las pruebas.

Tabla 6. Plantilla de plan de pruebas.

Prueba No.	N	Caso de Prueba:	XXXXXXXXXX			
URL:	http://IP:PORT/context/SWA		Tamaño de SWA		XXXX Bytes	
XMS	XXXX		XMX:		XXXX	
Grupo	Usuarios	Solicitudes	Exitosas	Fallidas	Tiempo	Throughput
Grupo 1	1	100	0	0	00:00:00	0 MB/S
Grupo 2	1	100	0	0	00:00:00	0 MB/S
Grupo 3	1	100	0	0	00:00:00	0 MB/S
Grupo 4	1	100	0	0	00:00:00	0 MB/S
Grupo 5	1	100	0	0	00:00:00	0 MB/S
Grupo 6	1	100	0	0	00:00:00	0 MB/S
Grupo 7	1	100	0	0	00:00:00	0 MB/S
Grupo 8	1	100	0	0	00:00:00	0 MB/S
Grupo 9	1	100	0	0	00:00:00	0 MB/S
Grupo 10	1	100	0	0	00:00:00	0 MB/S
Grupo 11	1	100	0	0	00:00:00	0 MB/S
Grupo 12	1	100	0	0	00:00:00	0 MB/S
Grupo 13	1	100	0	0	00:00:00	0 MB/S
Grupo 14	1	100	0	0	00:00:00	0 MB/S
Grupo 15	1	100	0	0	00:00:00	0 MB/S
Grupo 16	1	100	0	0	00:00:00	0 MB/S
Grupo 17	1	100	0	0	00:00:00	0 MB/S
Grupo 18	1	100	0	0	00:00:00	0 MB/S
Grupo 19	1	100	0	0	00:00:00	0 MB/S
Grupo 20	1	100	0	0	00:00:00	0 MB/S
Grupo 21	1	100	0	0	00:00:00	0 MB/S
Grupo 22	1	100	0	0	00:00:00	0 MB/S
Grupo 23	1	100	0	0	00:00:00	0 MB/S
Grupo 24	1	100	0	0	00:00:00	0 MB/S
Grupo 25	1	100	0	0	00:00:00	0 MB/S

- **No. Prueba.** Indica el número de prueba realizada.
- **Caso de Prueba.** Indica el caso de prueba.
- **Tamaño del SWA.** Tamaño en bytes del SWA.
- **URL.** Dirección URL del SWA de la prueba.
- **Grupo.** Número de grupo de solicitudes, del 1 al 25.
- **Usuarios.** Configuración del número de usuarios de JMeter.
- **Solicitudes.** Número de solicitudes realizadas a SWA
- **Exitosas.** Número de solicitudes respondidas con éxito.
- **Fallidas.** Número de solicitudes no respondidas con éxito.
- **Tiempo.** Duración en segundos de la prueba
- **Throughput.** Cantidad de MB enviados por segundo.

4.5.2 Definición de Escenarios de Prueba

De acuerdo a los atributos de calidad de servidores y clientes que se encontraron en la etapa de búsqueda de atributos, se definieron siete escenarios de pruebas de entregabilidad de SWA. De manera general se utilizó la arquitectura cliente-servidor y la arquitectura proxy, esta última consta de un dispositivo intermedio entre el cliente y el servidor que tiene diferentes funciones, entre ellas el balanceo de carga y Server Cache , en la Figura 18 se ilustran estas arquitecturas. En el Anexo D de este documento se describe el conjunto de escenarios de prueba definidos para las pruebas de capacidad de entrega de SWA.

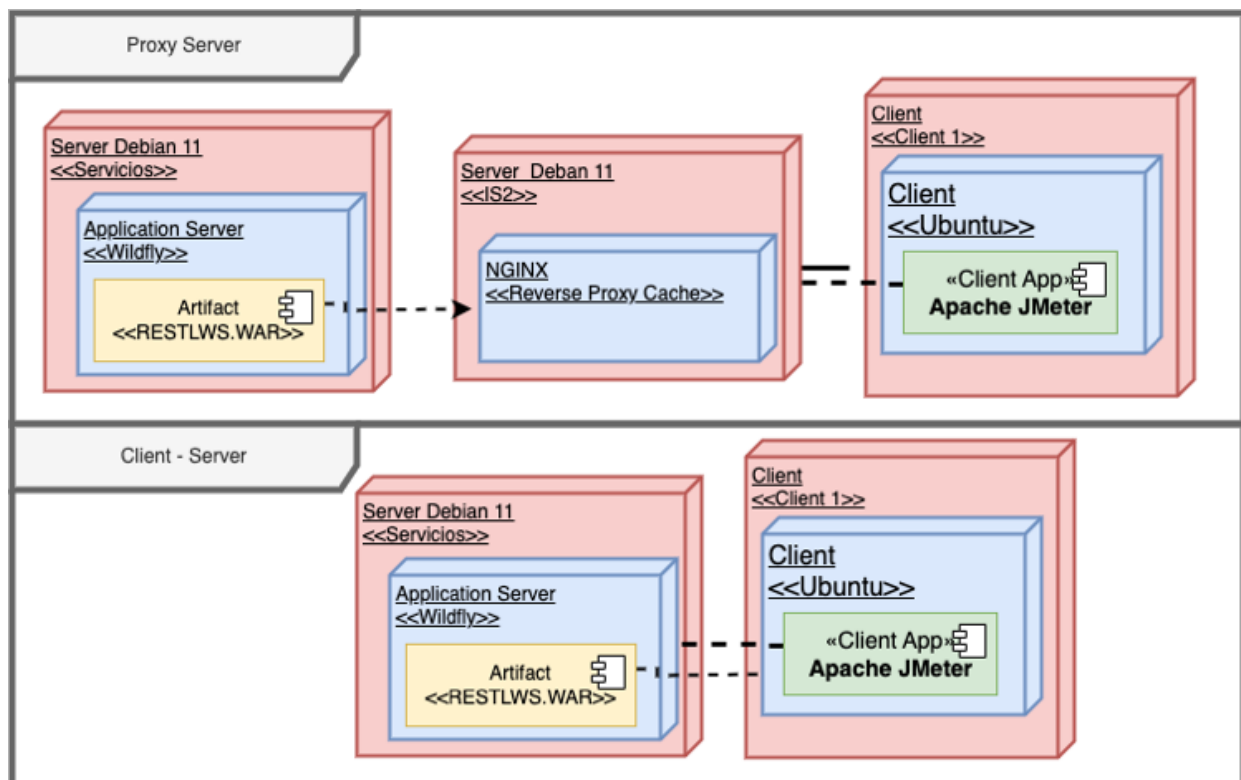


Figura 18. Arquitectura con Servidor Proxy y Cliente Servidor

Debido al conjunto de atributos encontrados que se pueden probar para entregar SWA, se definieron los casos de prueba que se listan en la Tabla 7, en la que se define el ID o identificador del caso de prueba y la descripción de cada uno de estos.

Tabla 7. Casos de prueba para entregar SWA.

ID del Caso	Descripción
D-DPA	Despliegue de artefactos SOAP y REST.
S-TSW1	Entregabilidad de SWA de tipo de implementación SOAP-REST
S-SM1	Entregabilidad de SWA con SOAP-SOAP MTOM
S-TPC1	Entregabilidad de SWA con el protocolo de comunicación HTTP- HTTPS
S-RC1	Entregabilidad de SWA con REST Cache
C-SC1	Entregabilidad de SWA con Server Cache
C-TF2	Entregabilidad de SWA con Tolerancia a Fallas
C-C4	Entregabilidad de SWA con Clustering
C-CD4	Entregabilidad de SWA Contenerizados
S-JHS1	Entregabilidad de SWA con configuraciones de JHS
S-GCA5	Entregabilidad de SWA de algoritmos de <i>Garbage Collector</i>
S-G1R1	Entregabilidad de SWA de G1 First Region
S-G1R2	Entregabilidad de SWA de G1 First Ratio
S-G1P3	Entregabilidad de SWA de G1 First Pauses
S-G1DD1	Entregabilidad de SWA de G1 First <i>Deduplication</i>
S-JPS1	Entregabilidad de SWA con el patrón de diseño Singleton

4.6 Generación de SWA de Diferentes Formatos y Medidas

Esta etapa está compuesta de tres fases, análisis de granularidad, la creación de los SWA con la herramienta generadora de SWA, [Escobar, 2017] y por último, la generación del artefacto WAR. La herramienta generadora de SWA se integraron los RA con los SWA, y posterior, con el IDE Netbeans se generaron los artefactos WAR utilizados para desplegar los SWA. En dichos artefactos se empaquetan o se agrupan los SWA creados, con las librerías y archivos de configuración que se requieren para desplegar los SWA.

4.6.1 Análisis de Granularidad

Para definir los elementos de prueba, se realizó un análisis de la granularidad de los SWA con el objetivo de determinar el tamaño adecuado por tipo de RA, ya sea texto, imagen, audio y video. En el Anexo E se aborda la información del análisis de granularidad de SWA.

El resultado del análisis de granularidad de los RA que están contenidos en los SWA se presenta en la Tabla 8, en donde se listan por nivel de granularidad. Para estas pruebas se presentan los SWA que contienen RA de granularidad fina, media y gruesa y que puede ser de tipo texto (Word, Excel, Power Point, PDF), imagen, audio o video; en los tipos de implementación de SW, SOAP y REST.

Tabla 8. Servicios Web de Aprendizaje generados en SW SOAP y REST.

Tipo de RA	Grano Fino	Grano Medio	Grano Grueso
Word	SOAP_DOCX_1MB	SOAP_DOCX_3MB	SOAP_DOCX_5MB
Excel	SOAP_XLSX_1MB	SOAP_XLSX_3MB	SOAP_XLSX_5MB
Power Point	SOAP_PPTX_1MB	SOAP_PPTX_3MB	SOAP_PPTX_5MB
PDF	SOAP_PDF_1MB	SOAP_PDF_3MB	SOAP_PDF_5MB
Imagen	SOAP_IMG_1MB	SOAP_IMG_3MB	SOAP_IMG_5MB
Audio	SOAP_AUDIO_1MB	SOAP_AUDIO_5MB	SOAP_AUDIO_10MB
Video	SOAP_VIDEO_5MB	SOAP_VIDEO_10MB	SOAP_VIDEO_15MB
Tipo de RA	Grano Fino	Grano Medio	Grano Grueso
Word	REST_DOCX_1MB	REST_DOCX_3MB	REST_DOCX_5MB
Excel	REST_XLSX_1MB	REST_XLSX_3MB	REST_XLSX_5MB
Power Point	REST_PPTX_1MB	REST_PPTX_3MB	REST_PPTX_5MB
PDF	REST_PDF_1MB	REST_PDF_3MB	REST_PDF_5MB
Imagen	REST_IMG_1MB	REST_IMG_3MB	REST_IMG_5MB
Audio	REST_AUDIO_1MB	REST_AUDIO_5MB	REST_AUDIO_10MB
Video	REST_VIDEO_5MB	REST_VIDEO_10MB	REST_VIDEO_15MB

Adicional a los SWA creados con la herramienta generadora automática de SWA, se crearon manualmente SWA para pruebas específicas de entregabilidad, estos SWA se listan en la Tabla 9. Las pruebas específicas de entregabilidad que se mencionan consisten en SOAP MTOM, patrón de diseño Singleton y con REST Cache. En el encabezado de esta tabla se indica el propósito de la prueba para lo que se utilizaron.

Tabla 9. Servicios Web de Aprendizaje para pruebas adicionales de entrega.

SWA MTOM	SWA Singleton	REST Cache
MTOM_AUDIO_1MB	SingletonWS1MB	RESTCACHE_VIDEO_15MB
MTOM_DOCX_3MB	SingletonWS3MB	
MTOM_PDF_5MB	SingletonWS5MB	
MTOM_VIDEO_10MB	SingletonWS10MB	
MTOM_VIDEO_15MB	SingletonWS15MB	

4.6.2 Creación de SWA

Para propósito de las pruebas de entregabilidad de SWA, se definieron 42 SWA de distintos tipos de RA con granularidad fina, mediana y gruesa, estos se encuentran listados en las Tablas 8. Por lo que se crearon dos artefactos WAR los cuales encapsulan los SWA creados, de acuerdo al tipo de implementación de SW, es decir, en el artefacto SOAPLWS se encapsularon los SWA de tipo SOAP y en el artefacto RESTLWS se encapsularon los SWA de tipo REST. Para pruebas adicionales como MTOM, REST Cache y EJB Singleton se crearon para cada uno un artefacto WAR. Todos los artefactos se listan en la Tabla 10.

El artefacto WAR es el que el servidor de aplicaciones en Java despliega como componente Web. La Figura 19 ilustra el proceso de creación hasta el despliegue de SWA, en el que interviene la herramienta generadora para integrar el RA con el SWA, la interfaz Java para compilar las clases Java en un proyecto y para crear el artefacto WAR, y el usuario

que interviene para utilizar la herramienta generadora de SWA, la compilación para obtener el artefacto WAR y el despliegue en el servidor de aplicaciones.

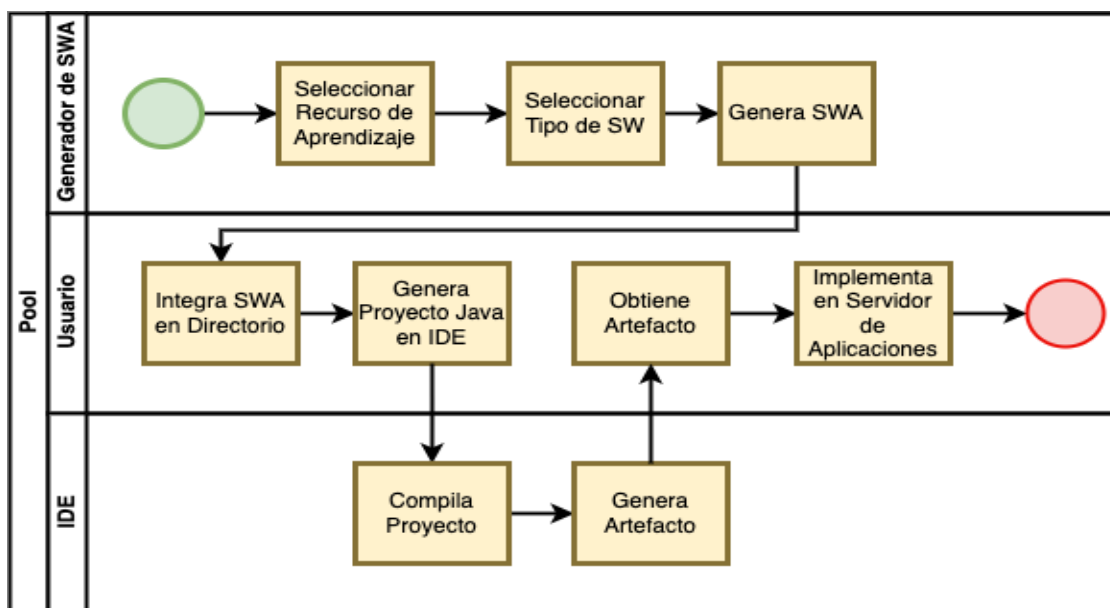


Figura 19. Proceso de Generación de SWA.

4.6.3 Creación de Artefactos WAR

Para propósito de las pruebas de entregabilidad de SWA, se definieron 21 SWA de distintos tipos de RA con granularidad fina, mediana y gruesa, en los tipos de implementación de SW SOAP y REST. Por lo que se crearon dos artefactos WAR los cuales encapsulan los SWA creados, de acuerdo al tipo de implementación de SW, es decir, en el artefacto SOAPLWS se encapsularon los SWA de tipo SOAP y en el artefacto RESTLWS se encapsularon los SWA de tipo REST.

Para pruebas adicionales como MTOM, REST Cache y EJB Singleton se crearon para cada uno un artefacto WAR. Todos los artefactos se listan en la Tabla 10.

Tabla 10. Artefactos Java para pruebas de entregabilidad de SWA.

Artefacto	Tipo de SW	Número de SWA	Tamaño
SOAPLWS-1.WAR	SOAP	21	84 MB
RESTLWS-1.WAR	REST	21	84.1 MB
SOAPLWSMTOM-1.WAR	SOAP	5	35.3 MB
RESCacheVideo10MB-1.WAR	REST	1	9.8 MB
SingletonLWS-1.WAR	REST	5	35.3 MB

4.7 Ejecución del plan de Pruebas

En el punto **5. Ejecución del Plan de Pruebas** se presenta la información de la ejecución del plan de pruebas, esta sección indica cada una de las ejecuciones de los 17 casos de prueba que se ejecutaron, los parámetros de entrada y los resultados.

Los equipos computacionales utilizados, así como la configuración y descripción de los componentes de hardware se encuentran descritos en el Anexo C.

4.8 Obtención de los Resultados

En cada caso de prueba de la sección **5.2 Ejecución del Plan de Pruebas** en el apartado de **Especificaciones de Salida** de cada caso de prueba. En el Anexo B de este documento, explícitamente en la sección con el mismo nombre, se indica la manera de cómo se obtuvieron de los resultados de las pruebas de entrega de SWA, por ejemplo el tiempo de respuesta, el throughput y la tasa de éxito. Además de las especificaciones de salida, en dicho Anexo también se describe la entrega de un recurso completo.

4.9 Análisis de Resultados

La primera parte del análisis de resultados se expone en la sección **5.2 Ejecución del Plan de Pruebas** en el apartado **Resultados de Caso de Prueba**. La segunda parte se indica en la sección **5.3 Análisis de Resultados**, en esta se indica la metodología utilizada para determinar estadísticamente la existencia de diferencias significativas y posterior, en el Anexo F se muestran los resultados de este análisis estadístico.

4.10 Definición de Métricas de Entregabilidad

La definición de métricas de entregabilidad de SWA se basa en los atributos de tiempo de respuesta, throughput y tasa de éxito.

Posterior de la definición de métricas de entregabilidad, se define el modelo de QoS de entrega de SWA, el cual se basa en atributos heterogéneos en el cual convergen el lenguaje de programación, estrategias de manejo de memoria, aplicaciones de servidores, entre otros. En la sección **5.4 Definición de Métricas de Entregabilidad** se establece la manera de evaluar la capacidad de entrega.

4.11 Evaluación de Resultados

La evaluación de resultados se presenta en la sección **5.5 Evaluación de Resultados**, considerando los atributos de calidad del Modelo de Entregabilidad de SWA.

Capítulo V. Pruebas y Análisis de Resultados

En este capítulo se presenta la información correspondiente a la ejecución casos de pruebas, así como los resultados de las pruebas de entregabilidad de SWA y el análisis de dichos los resultados. El objetivo principal es determinar la capacidad de entrega de SWA de cada uno de los atributos de clientes y servidores encontrados y descritos en el capítulo IV.

Los SWA utilizados para las pruebas de entregabilidad de SWA se definieron en la sección 4.6, de tipo texto, imagen, audio y video, con tamaños de 1, 3, 5, 10 y 15MB. En cada caso de prueba se encuentran definidos los SWA que se utilizaron para realizar las pruebas de entregabilidad. Los SWA fueron seleccionados de manera aleatoria para cada caso de prueba, por lo que no existe influencia, interés o alteración de los resultados ni en la validez de estos. Se seleccionó un SWA de cada tamaño para la ejecución de las pruebas, por lo que no se ejecutaron todos los SWA creados para cada caso de prueba, con las ejecuciones realizadas son suficientes para obtener las muestras representativas y a su vez, dar solidez a los resultados del experimento.

En algunos casos de prueba se utilizaron SWA de todo tamaño de granularidad definido previamente, en algunos casos se utilizaron solo SWA de 10 y de 15MB, debido a que en algunas pruebas realizadas se detectó cierto grado de degradación pero continuidad de las condiciones del escenario de pruebas. Además de las pruebas de entregabilidad de SWA con diferentes tamaños de JHS, en algunas pruebas se configuró la JHS con un tamaño de 2GB y en otras pruebas con 4GB, para continuar analizando el tamaño de la JHS con respecto a otros casos de prueba.

Para este trabajo se definieron siete escenarios de prueba definidos en el Anexo D, para los 17 casos de prueba, listados y definidos en la sección 4.5. En el caso de prueba de D-DPA1 se utilizaron los artefactos WAR generados para el despliegue de SWA, estos están descritos en la sección 4.6 de este trabajo.

El conjunto de pruebas se diseñó con el objetivo de comparar la capacidad de entrega de SWA del atributo bajo estudio, manteniendo constantes el resto de las condiciones del escenario de pruebas. Este enfoque permitió analizar de forma controlada el impacto del atributo evaluado sobre la capacidad de entrega de los SWA y determinar que las diferencias observadas en los resultados se atribuyen exclusivamente al atributo considerado en la prueba. Se emplearon las métricas de entregabilidad definidas en [Benitez, 2018], para evaluar la capacidad de entrega de las pruebas. El primer atributo es el tiempo de respuesta en segundos, el *throughput* en MB, el cual representa la cantidad de información procesada en respuesta a una solicitud. Por último, la tasa de éxito, como se mencionó anteriormente, representa el porcentaje de solicitudes a SWA respondidas con éxito.

El nivel de influencia de la capacidad de entrega de los atributos se determinó comparando los conjuntos de resultados mediante el análisis estadístico presentado, para dos muestras independientes y para más de dos muestras independientes, según sea el caso de prueba.

En este mismo capítulo, se realiza una evaluación de distintos escenarios de arquitecturas de SWA, basados en el modelo de calidad de entrega que se propone en este trabajo, el cual considera atributos heterogéneos basados en arquitecturas de SW, técnicas de programación como Singleton, técnicas de optimización de memoria, entre otros elementos. También se definieron los valores utilizados como referencia de los atributos definidos en el modelo de calidad de entrega de SWA.

Por último, se presenta la discusión de los resultados, y los hallazgos de la ejecución de las pruebas, entre otros aspectos relevantes en el desarrollo de este trabajo.

5.1 Ejecución del Plan Experimental

En esta sección se describe la ejecución del plan experimental para validar las hipótesis, detallando los procedimientos, configuraciones y condiciones de las pruebas para garantizar la reproducibilidad de los resultados.

5.1.1 Caso de Prueba D-DPA1

Caso de prueba D-DPA1. Despliegue de artefactos de SWA con distintos tamaños de JHS de acuerdo a la Tabla 11.

Objetivo del caso de prueba D-DPA1. Determinar la configuración mínima requerida de JHS para el despliegue de artefactos WAR.

Elementos del caso de prueba D-DPA1. En estas pruebas se realizaron con los siguientes artefactos WAR:

- ◆ SOAPLWS-1.war
- ◆ RESTLWS-1.war

Especificaciones de entrada del caso de prueba D-DPA1. Para el caso de prueba D-DPA1 se realizaron cuatro despliegues de los artefactos WAR, cada despliegue con diferente configuración de los parámetros de JHS, XMS y XMN. La configuración del parámetro XMS (Tamaño inicial de la JHS), y el parámetro XMN (Tamaño máximo de la JHS), de cada prueba se indican en la Tabla 11, en la que se presenta la prueba realizada. Los elementos a considerar como especificaciones de entrada de la prueba de despliegue de artefactos WAR están listadas a continuación:

- ◆ El parámetro de JHS XMS.
- ◆ El parámetro de JHS XMN.
- ◆ El artefacto SOAPLWS-1.WAR.
- ◆ El artefacto RESTLWS-1.WAR.

Tabla 11. Parámetros de JHS para pruebas de despliegue

Prueba	XMS	XMx
Prueba 1	512 MB	1024 MB
Prueba 2	1024 MB	2048 MB
Prueba 3	2048 MB	3072 MB
Prueba 4	3072 MB	4096 MB

Especificaciones de salida del caso de prueba D-DPA1.

- ◆ Artefacto desplegado en el servidor de aplicaciones, es decir, los SWA están disponibles y listos para ser consumidos.
- ◆ Artefacto no desplegado.

Resultados del caso de prueba D-DPA1. Los resultados del caso de prueba D-DPA1 se presentan en la Tabla 12.

Tabla 12. Pruebas de despliegues de artefactos de SWA.

Caso	Artefacto SOAP	Artefacto REST	Artefactos SOAP y REST
Prueba 1	Se desplegó	Se desplegó	No se desplegaron
Prueba 2	Se desplegó	Se desplegó	Se desplegaron
Prueba 3	Se desplegó	Se desplegó	Se desplegaron
Prueba 4	Se desplegó	Se desplegó	Se desplegaron

Los resultados de las pruebas de despliegue de artefactos indican que se debe definir el tamaño de JHS al menos en 1024MB como un tamaño adecuado para desplegar un artefacto WAR, e incrementar el tamaño del parámetro XMx conforme el número de artefactos que se requieran desplegar. En la Tabla 12 se indica que en la prueba 1 no se desplegaron los dos artefactos WAR de manera simultánea, debido a la falta de espacio asignado en la JHS. Este incidente se resalta en color rojo para facilitar su identificación. En el resto de las pruebas los artefactos WAR se desplegaron satisfactoriamente.

5.1.2 Caso de Prueba S-TSW1

Caso de prueba S-TSW1. Capacidad de entrega de SWA por tipo de implementación de SWA, SOAP y REST.

Objetivo del caso de prueba S-TSW1. Evaluar la capacidad de entrega de SWA de acuerdo al tipo de implementación, SOAP y REST.

Elementos del caso de prueba S-TSW1. Los elementos de prueba para el caso de prueba S-TSW1 se listan en la Tabla 13, de lado izquierdo aparecen los SWA de tipo de implementación SOAP y de lado derecho los de tipo REST.

Tabla 13. Elementos de prueba del caso de prueba S-TSW1.

Tipo de SWA	SWA	Tipo de SWA	SWA
SOAP	SOAP_PPTX_1MB	REST	REST_PPTX_1MB
SOAP	SOAP_PPTX_3MB	REST	REST_PPTX_3MB
SOAP	SOAP_PPTX_5MB	REST	REST_PPTX_5MB
SOAP	SOAP_VIDEO_10MB	REST	REST_VIDEO_10MB
SOAP	SOAP_VIDEO_15MB	REST	REST_VIDEO_15MB

Especificaciones de entrada del caso de prueba S-TSW1. Para el caso de prueba S-TSW1 las especificaciones de entrada se listan a continuación:

- ◆ Los SWA SOAP debidamente desplegados.
- ◆ Los SWA REST debidamente desplegados.
- ◆ Para los SWA de tipo de implementación SOAP, se requiere del Payload XML.
- ◆ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-TSW1. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV como se indica en el Anexo B, del cual se obtienen la siguiente información:

- ◆ Tiempo de respuesta de la entrega de SWA en segundos.
- ◆ **Throughput** de la entrega de SWA en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-TSW1. Los resultados del caso de prueba S-TSW1 se presentan en la Tabla 14, en razón de la media estadística. En la columna 1 se indica el SWA, en las columnas 2 y 3 se presentan los resultados de tiempo de respuesta en la columna 3 y 4 los resultados de **throughput** y en la columna 6 y 7 se presentan los resultados de tasa de éxito de SOAP y de REST, respectivamente.

De acuerdo a la Tabla 14, a excepción en los SWA de 5MB que aparece resaltado en color naranja, la diferencia de los resultados es mínima y estadísticamente los datos indican que no hay diferencia significativa. Por otro lado, los SWA de 1MB tardan 23 segundos en responder el grupo de solicitudes, estos aparecen sombreados en color azul.

En cambio, el tiempo de respuesta de los SWA de 3, 5, 10 y 15MB oscila entre 60 y 80 segundos. En el caso de los SWA de 1 y 3MB, que aparecen sombreados en la columna tasa de éxito, en estos se obtuvo el 100 por ciento de éxito, en el resto de los SWA, conforme el tamaño del SWA es mayor, el resultado de tasa de éxito se degrada.

Tabla 14. Resultados del caso de prueba S-TSW1

SWA	Tiempo de Respuesta (Segundos)		Throughput (MBPS)		Tasa de Éxito (Porcentaje)	
	SOAP	REST	SOAP	REST	SOAP	REST
PPTX_1MB	23.62	23.70	5.55	5.52	100	100
PPTX_3MB	65.63	65.70	5.76	5.75	100	100
PPTX_5MB	78.51	73.48	5.41	5.77	65.94	65.85
VIDEO_10MB	68.90	68.49	5.51	5.43	31.26	29.59
VIDEO_15MB	65.49	69.26	5.41	4.91	18.28	17.67

En la Figura 20 se ilustran los resultados de tiempo de respuesta, **throughput** y tasa de éxito del caso de prueba S-TSW1. La línea en color azul ilustra los resultados de los SWA de tipo de implementación SOAP y la línea en color rojo ilustra los resultados de los SWA de tipo de implementación REST.

En la Figura 20(a) se ilustran los resultados de tiempo de respuesta del caso de prueba S-TSW1. De acuerdo con la gráfica del tiempo de respuesta, el tipo de implementación de SW no influye en los resultados, las dos líneas que representan los resultados de SOAP y REST siguen la misma tendencia.

En la Figura 20(b) se ilustran los resultados de **throughput** del caso de prueba S-TSW1. El tipo de implementación de SWA no influye en los resultados de **throughput**, los resultados son similares en SOAP y en REST. Todos los resultados oscilan en un rango entre 5 y 6MBPS.

En la Figura 20(c) se ilustran los resultados de tasa de éxito del caso de prueba S-TSW1 y se observa que respecto al tipo de implementación de SWA, no existe diferencia significativa entre los dos tipos de implementación de WS, SOAP y REST, los resultados siguen la misma tendencia en función del tamaño del SWA. Por otro lado, se puede mencionar que existe una correlación entre el tamaño del SWA y la tasa de éxito, el tamaño del SWA influye en la tasa de éxito y entre menor sea el tamaño del SWA, es mejor el resultado.

Es importante mencionar que existe diferencia en la implementación de los SWA implementados con SOAP y REST. Específicamente, los SWA REST utiliza un conjunto de librerías adicionales que no son utilizadas en ellos SWA implementados con SOAP. Esto puede impactar en el desempeño de la capacidad de entrega de SWA.

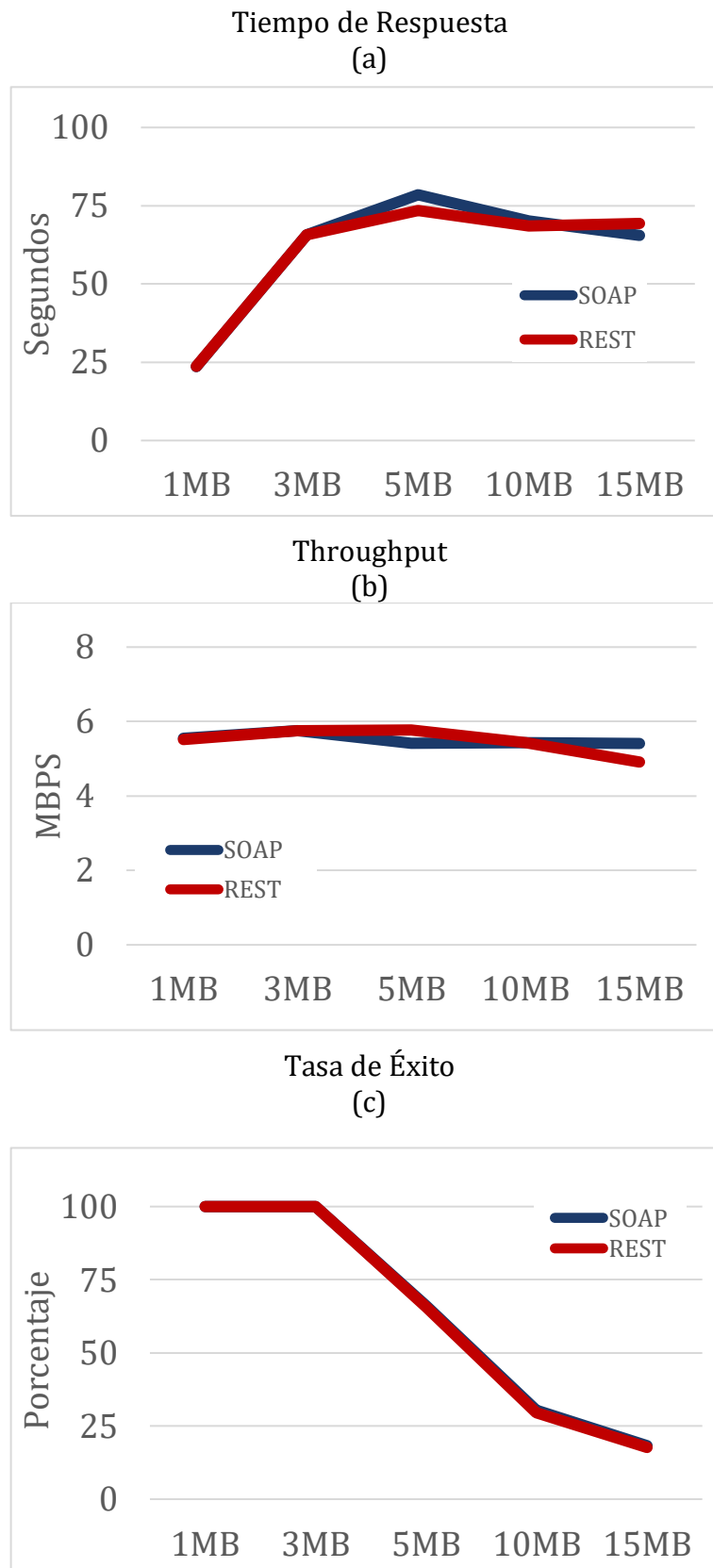


Figura 20. Resultados del caso de prueba S-TSW1.

5.1.3 Caso de Prueba S-TPC1

Caso de prueba S-TPC1. Capacidad de entrega de SWA en función del protocolo de comunicación, HTTP y HTTPS.

Objetivo del caso de prueba S-TPC1. Evaluar la capacidad de entrega de SWA cuando se utilizan los protocolos de comunicación HTTP y HTTPS, e identificar el más conveniente.

Elementos del caso de prueba S-TPC1. Los elementos de prueba para el caso de prueba S-TPC1 se listan en la Tabla 15.

Tabla 15. Elementos de prueba del caso de prueba S-TPC1.

SOAP	REST
SOAP_AUDIO_1MB	REST_AUDIO_1MB
SOAP_DOCX_3MB	REST_DOCX_3MB
SOAP_PDF_5MB	REST_PDF_5MB
SOAP_VIDEO_10MB	REST_VIDEO_10MB
SOAP_VIDEO_15MB	REST_VIDEO_15MB

Especificaciones de entrada del caso de prueba S-TPC1. Para el caso de prueba S-TPC1, las especificaciones se listan a continuación:

- ◆ Los SWA desplegados a través del protocolo HTTPS.
- ◆ Los SWA desplegados a través del protocolo HTTP.
- ◆ Para los SWA de tipo de implementación SOAP, se requiere del Payload XML.
- ◆ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-TPC1. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV en el directorio definido en el plan de Apache JMeter como se indica en el Anexo B, del cual se obtienen la siguiente información:

- ◆ Tiempo de respuesta de la entrega de SWA en segundos.
- ◆ **Throughput** de la entrega de SWA en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-TPC1. Los resultados del caso de prueba S-TPC1 se presentan en la Tabla 16. En la columna 1 se indica el SWA, en el resto de las columnas se presentan los resultados de las pruebas entre el protocolo de comunicación HTTP y HTTPS con los tipos de implementación SOAP y REST, respectivamente.

Es importante mencionar que en los resultados de entrega de SWA de 1 y 3MB de REST con HTTP, con respecto SOAP y a REST con HTTPS, estos son menos favorables, por este motivo se resaltan en color amarillo. El mismo caso se presentó con los SWA de 15MB de tipo SOAP con HTTP. En contraparte, resaltados en color azul se presentan los

resultados de entrega de SWA de 1MB con SOAP, los cuales son más favorables, sin importar el tipo de protocolo de comunicación.

Tabla 16. Resultados del caso de prueba S-TPC1.

SWA	Tiempo de Respuesta (Segundos)				Throughput (MBPS)				Tasa de Éxito (Porcentaje)			
	HTTP		HTTPS		HTTP		HTTPS		HTTP		HTTPS	
	SOAP	REST	SOAP	REST	SOAP	REST	SOAP	REST	SOAP	REST	SOAP	REST
AUDIO_1MB	8.45	13.51	8.45	11.15	17.51	11.18	17.51	13.77	100	100	100	100
DOCX_3MB	22.02	28.99	24.41	23.94	22.02	13.91	24.41	16.44	100	99.54	99.98	98.95
PDF_5MB	26.75	25.62	25.62	26.75	16.39	16.69	16.69	16.39	60.10	59.76	59.42	60.10
VIDEO_10MB	22.53	27.04	26.98	28.59	18.19	14.62	14.02	12.21	32.61	31.00	28.65	27.48
VIDEO_15MB	46.34	33.24	22.46	32.43	7.76	10.77	14.95	10.04	18.14	17.16	18.08	16.82

En la Figura 21 se ilustran los resultados del caso de prueba S-TPC1, la línea en color azul indica los resultados de HTTP/SOAP, en color rojo los resultados de HTTP/REST, en color verde los resultados de HTTPS/SOAP, y en color naranja los resultados de HTTPS/ REST.

La Figura 21(a) ilustra los resultados de tiempo de respuesta del caso de prueba S-TPC1. En esta se visualiza que los resultados no evidencian una mejoría entre utilizar distintas versiones del protocolo de comunicación para entregar SWA, respecto al tiempo de respuesta. En los datos presentados en la columna Tiempo de Respuesta de la Tabla 16, se observa que el menor tiempo se presenta en los resultados de SWA de 1MB, presentados en color azul, y su contraparte con SWA de 15MB, resaltado en color amarillo.

En la Figura 21(b) se ilustran los resultados de **throughput** del caso de prueba S-TPC1. Los mejores resultados de entrega de SWA se presentan en los SWA de 1MB con el protocolo HTTP, en SWA de granularidad gruesa (10MB y 15MB) los resultados son mejores con la versión HTTPS, estos indican una diferencia de 3MBPS entre una versión y otra. En el resto de SWA los resultados de **throughput** tienen la misma tendencia en ambos casos, HTTP o HTTPS, indistintamente el tipo de implementación del SWA, SOAP o REST.

En la Figura 21(c) se ilustran los resultados de tasa de éxito del caso de prueba S-TPC1, en esta se visualiza que indistintamente la versión del protocolo de comunicación o el tipo de implementación de WS, los resultados siguen la misma tendencia. Sin embargo, la tasa de éxito es afectada por el tamaño del SWA. Los resultados de la tasa de éxito que se presentan en la Tabla 16, indican que el tamaño del SWA afecta la capacidad de entrega, los SWA de 1MB son entregados con una tasa de éxito de 100 por ciento; de 99 por ciento de los SWA de 3MB, y conforme el tamaño del SWA es mayor, la tasa de éxito se degrada. Este hecho es afectado por el espacio asignado en memoria de la JHS, donde se generan los objetos Java, al definirse previo a la ejecución de la prueba, provoca un desbordamiento de memoria por generar objetos enormes que contienen los RA de los SWA.

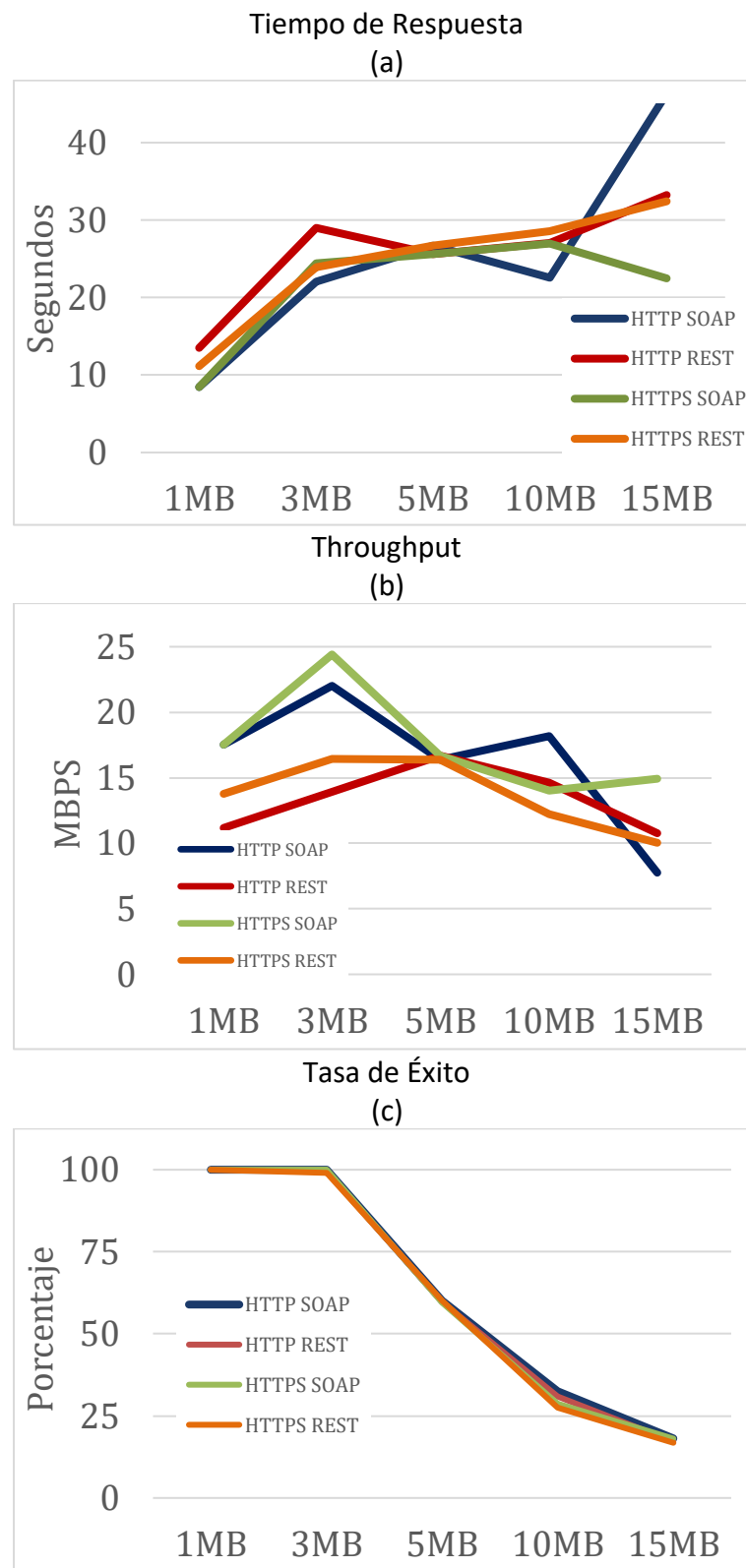


Figura 21. Resultados del caso de prueba S-TPC1.

5.1.4 Caso de Pruebas S-SM1

Caso de prueba S-SM1. Evaluación de la capacidad de entrega de SWA con la implementación de SOAP MTOM.

Objetivo del caso de prueba S-SM1. Evaluar la influencia de SOAP MTOM para entregar SWA.

Elementos del caso de prueba S-SM1. Los elementos de prueba para el caso de prueba S-SM1 se listan en la Tabla 17.

Tabla 17. Elementos de prueba para el caso de prueba S-SM1.

SOAP	MTOM
SOAP_AUDIO_1MB	MTOM_AUDIO_1MB
SOAP_DOCX_3MB	MTOM_DOCX_3MB
SOAP_PDF_5MB	MTOM_PDF_5MB
SOAP_VIDEO_10MB	MTOM_VIDEO_10MB
SOAP_VIDEO_15MB	MTOM_VIDEO_15MB

Especificaciones de entrada del caso de prueba S-SM1. Las especificaciones de entrada para el caso de prueba S-SM1 son las siguientes:

- ♦ Los SWA SOAP con la opción de MTOM.
- ♦ Los SWA SOAP sin la opción de MTOM.
- ♦ El tamaño del SWA.
- ♦ Se requiere el Payload XML para las pruebas de SOAP y SOAP MTOM.
- ♦ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-SM1. Como salida de cada una de la ejecución de las pruebas se genera un archivo con formato CSV, como se indica en el Anexo B. De esta salida se obtiene la siguiente información:

- ♦ Tiempo de respuesta de la entrega de SWA en segundos.
- ♦ **Throughput** de la entrega de SWA en MBPS.
- ♦ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-SM1. Los resultados del caso de prueba S-SM1 se presentan en la Tabla 18 en razón de la media estadística del tiempo de respuesta en segundos, **throughput** en MBPS y tasa de éxito en formato porcentual. En la columna 1 se indica el SWA que fue objeto de las pruebas.

Tabla 18. Resultados del caso de prueba S-SM1.

SWA	Tiempo de Respuesta (Segundos)		Throughput (MBPS)		Tasa de Éxito (Porcentaje)	
	MTOM	SOAP	MTOM	SOAP	MTOM	SOAP
1MB	9.90	8.45	15.62	17.51	100	100
3MB	22.70	24.41	17.55	16.39	100	100
5MB	25.13	27.82	15.92	14.52	54.99	54.95
10MB	30.81	28.59	11.80	13.40	26.95	28.73
15MB	21.27	22.46	15.68	14.95	17.39	17.16

La Figura 22 ilustra los resultados entregabilidad de SWA del caso de prueba S-SM1, el tiempo de respuesta en segundos, el **throughput** en MBPS y la tasa de éxito en formato porcentual. Los resultados de las pruebas de entregabilidad de SWA de SOAP MTOM se visualizan en la línea en color rojo y los resultados de las pruebas de SOAP se visualizan en la línea en color azul.

La Figura 22(a) ilustra los resultados de tiempo de respuesta del caso de prueba S-SM1. En esta figura se observa visualmente que no existe diferencia en los tiempos de respuesta de SOAP MTOM y de SOAP. Los datos de las columnas de MTOM y SOAP de tiempo de respuesta de la Tabla 18 indican que hay una diferencia, esta diferencia es mínima y no afecta o perjudica estadísticamente los resultados. Los datos que aparecen resaltados en color azul de la columna tiempo de respuesta son los mejores resultados, estos se obtuvieron con SWA de 1MB con resultados menores a 10 segundos, ya sea con SOAP o con SOAP MTOM.

La Figura 22(b) ilustra los resultados de **throughput** para el caso de prueba S-SM1. En esta se puede visualizar que los resultados no indican mejoría cuando se utiliza la implementación de SOAP MTOM para entregar SWA. En la Tabla 18 se puede corroborar la información.

La Figura 22(c) ilustra los resultados de tasa de éxito del caso de prueba S-SM1. En esta se observa que no hay algún beneficio o mejora cuando se utiliza la implementación de SOAP MTOM para entregar SWA, comparados con los resultados de SOAP, estos son muy similares. Por otro lado, un factor que debe ser considerado para entregar SWA es el tamaño del SWA, los datos presentados en la Tabla 18 de la columna tasa de éxito aparecen resaltados en color azul, son los casos de SWA de 1 y 3MB ya sea con SOAP o con SOAP MTOM, y en estos, la tasa de éxito es de 100 por ciento. En el resto de los casos, mientras mayor es el tamaño, el resultado de tasa de éxito es menor.

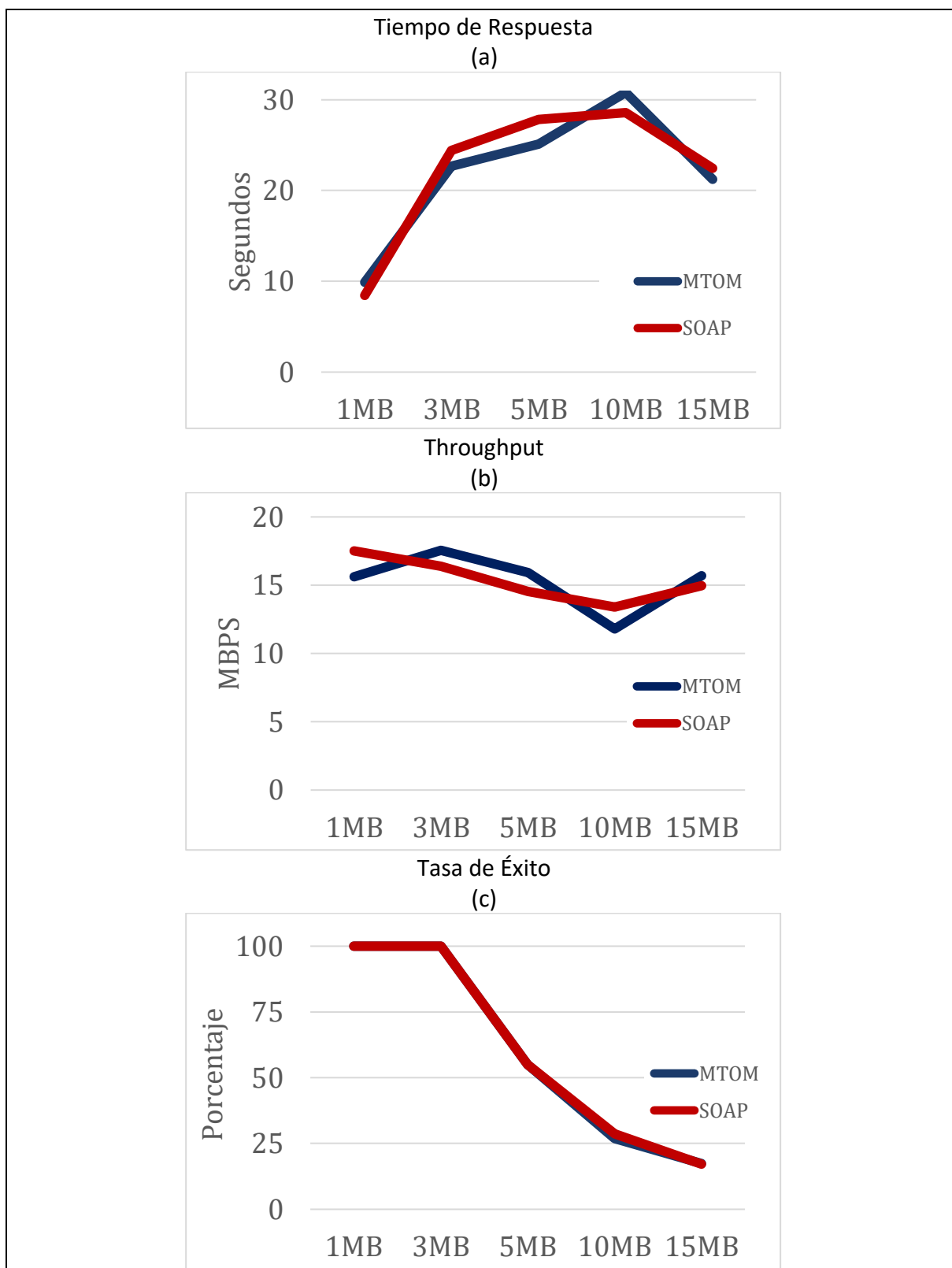


Figura 22. Resultados del caso de prueba S-SM1.

5.1.5 Caso de Prueba C-C4.

Caso de prueba C-C4. Capacidad de entrega de los SWA en un ambiente clusterizado.

Objetivo del caso de prueba C-C4. Evaluar la capacidad de entrega de SWA desplegados en un ambiente clusterizado.

Elementos del caso de prueba C-C4. En la Tabla 19 se listan los elementos utilizados para el caso de prueba C-C4.

Tabla 19. SWA para el caso de pruebas C-C4.

Tamaño	SOAP	REST
1 MB	SOAP_XLSX_1MB	REST_XLSX_1MB
3 MB	SOAP_XLSX_3MB	REST_XLSX_3MB
5 MB	SOAP_XLSX_5MB	REST_XLSX_5MB
10 MB	SOAP_VIDEO_10MB	REST_VIDEO_10MB
15 MB	SOAP_VIDEO_15MB	REST_VIDEO_15MB

Especificaciones de entrada del caso de prueba C-C4. Las especificaciones de salida para el caso de prueba C-C4 se listan a continuación.

- ◆ El servidor de aplicaciones Wildfly configurado en clúster .
- ◆ El servidor de aplicaciones Wildfly configurado en **standalone**.
- ◆ Los SWA debidamente desplegados.
- ◆ El balanceador de carga configurado para la arquitectura de cómputo implementada en clúster.
- ◆ Para los SWA de tipo de implementación SOAP, se requiere del **payload XML**

Especificaciones de salida del caso de prueba C-C4. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV, como se indica en el Anexo B, del cual se obtiene la siguiente información:

- ◆ Tiempo de respuesta en segundos.
- ◆ **Throughput** en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba C-C4. Los resultados del caso de prueba C-C4 se presentan en la Tabla 20, en razón de la media estadística, de acuerdo a la métrica de entregabilidad, el tiempo de respuesta en segundos, **throughput** en MBPS y tasa de éxito en formato porcentual. Estos también se dividen de acuerdo a la arquitectura del servidor de aplicaciones y a su vez divididos por el tipo de implementación de WS.

Tabla 20. Resultados del caso de prueba C-C4.

SWA	Tiempo de Respuesta (Segundos)				Throughput (MBPS)				Tasa de Éxito (Porcentaje)			
	Clúster		Standalone		Clúster		Standalone		Clúster		Standalone	
	SOAP	REST	SOAP	REST	SOAP	REST	SOAP	REST	SOAP	REST	SOAP	REST
1MB	9.58	11.34	10.08	13.55	14.14	12.25	13.47	10.28	100	100	100	100
3MB	20.78	25.99	32.34	49.54	18.83	15.40	12.45	11.01	100	100	100	100
5MB	38.92	36.26	36.26	31.75	17.79	18.17	16.52	17.54	100	100	96.67	84.24
10MB	65.99	107.19	74.50	37.65	19.11	11.90	8.18	5.42	100	99.42	30.78	28.80
15MB	112.92	110.16	41.94	32.43	10.68	10.45	9.02	10.04	58.7	56.20	18.72	16.82

La Figura 23 ilustra los resultados del caso de prueba C-C4 de acuerdo a la métrica de entregabilidad de SWA, el tiempo de respuesta en segundos, el **throughput** en MBPS, y la tasa de éxito en formato porcentual. Las líneas representan los resultados en el siguiente orden, en color azul Clúster/SOAP, en color rojo Clúster/REST, en color verde **standalone**/SOAP y en color naranja **standalone**/REST.

En la Figura 23(a) se ilustran los resultados de tiempo de respuesta del caso de prueba C-C4. En estos resultados se visualiza que los mejores resultados se presentan con SWA de 1 y 3 MB, ya sea con SOAP o con REST. En el caso de los SWA desplegados en arquitecturas clústerizadas, estos toman más tiempo debido a que responden más solicitudes con éxito a diferencia de los que están desplegados en arquitecturas **standalone**.

La Figura 23(b) ilustra los resultados de **throughput** del caso de prueba C-C4. En estos resultados se visualiza que los resultados presentan variaciones, aunque los mejores resultados se presentan en arquitecturas clústerizadas con SWA de 10 y 15MB.

La Figura 23(c) ilustra los resultados de tasa de éxito del caso de prueba C-C4. En esta se visualiza que la tasa de éxito mejora significativamente en un ambiente clusterizado, obteniendo resultados de 99 por ciento con SWA de 10MB y de 56 por ciento de éxito con SWA de 15MB, a diferencia de los resultados obtenidos en pruebas de ambientes **standalone**, en donde se obtuvo 30 por ciento de tasa de éxito con SWA de 10MB y hasta 18 por ciento con SWA de 15MB, aproximadamente. Los resultados de tasa de éxito con SWA de 1 y 3MB son muy similares en los dos ambientes.

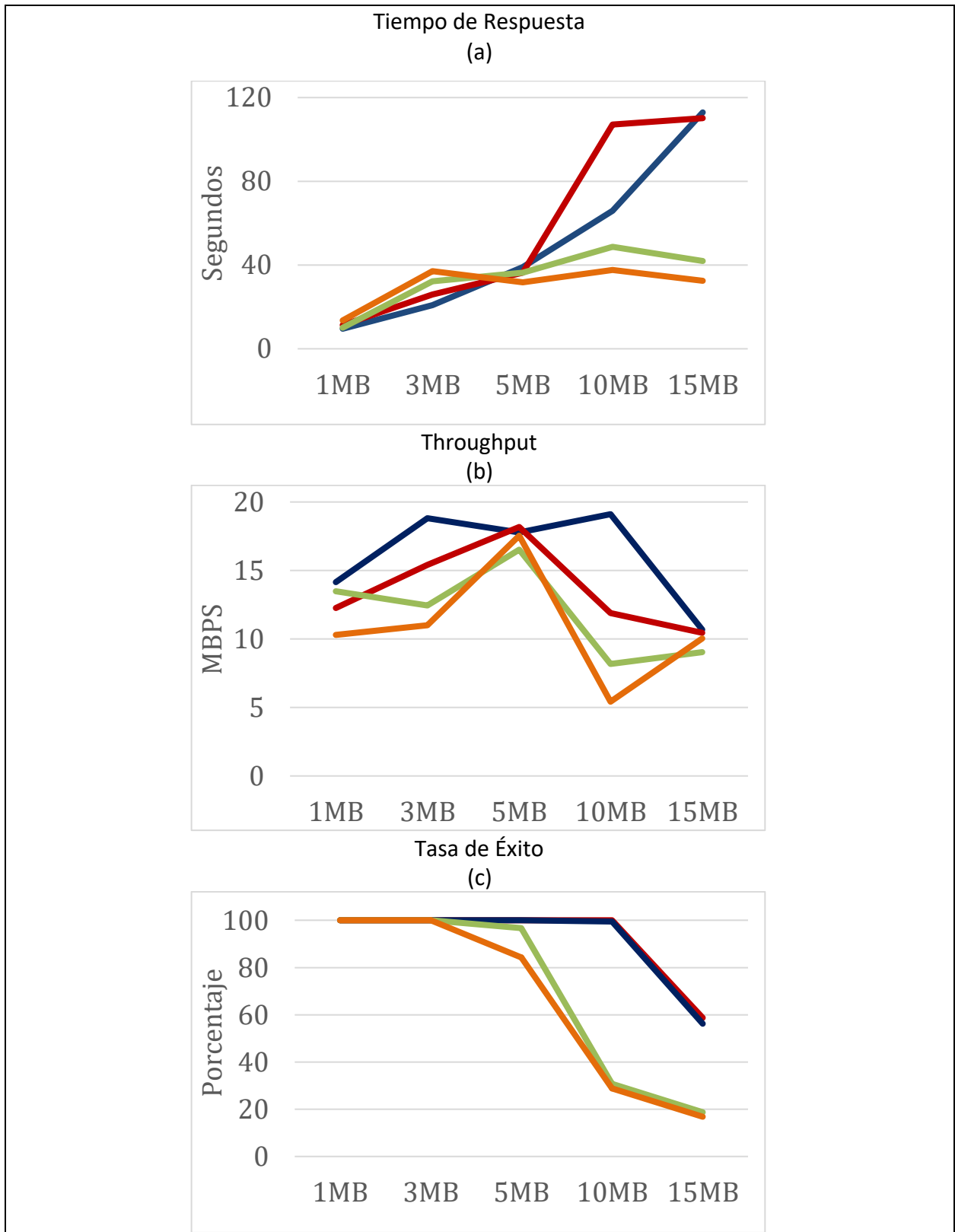


Figura 23. Resultados de pruebas de entrega de SWA de Clustering.

5.1.6 Caso de Prueba C-TF2

Caso de prueba C-TF2. Capacidad de entrega de SWA en una arquitectura tolerante a fallas.

Objetivo del caso de prueba C-TF2. Evaluar la capacidad de entrega de SWA desplegados en una arquitectura tolerante a fallas.

Elementos del caso de prueba C-TF2. Los elementos de prueba del caso de prueba C-TF2 se listan en la Tabla 21.

Tabla 21. SWA utilizados para el caso de prueba C-TF2.

Tamaño	SOAP	REST
10 MB	SOAP_VIDEO_10MB	REST_VIDEO_10MB
15 MB	SOAP_VIDEO_15MB	REST_VIDEO_15MB

Especificaciones de entrada del caso de prueba C-TF2. Las especificaciones de entrada para el caso de prueba C-TF2 son las siguientes:

- ♦ La arquitectura de cómputo implementada con Tolerancia a Fallas con un tiempo de espera de 2 segundos.
- ♦ La arquitectura de cómputo implementada en standalone.
- ♦ Los SWA debidamente desplegados.
- ♦ Para los SWA de tipo de implementación SOAP, se requiere del payload XML.
- ♦ Endpoint del SWA.

Especificaciones de salida del caso de prueba C-TF2. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV en el directorio definido en el plan de Apache JMeter como se indica en el Anexo B, del cual se obtienen la siguiente información:

- ♦ Tiempo de respuesta en segundos.
- ♦ **Throughput** en MBPS.
- ♦ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba C-TF2. Los resultados del caso de prueba C-TF2 se presentan en la Tabla 22, en razón de la media estadística, de acuerdo a la métrica de entregabilidad, el tiempo de respuesta en segundos, el **throughput** en MBPS y tasa de éxito en formato porcentual. En la columna 1 se indica el SWA, y en las continuas se presentan los datos de la arquitectura Tolerante a Fallos o Standalone y el tipo de implementación de WS, SOAP o REST.

Tabla 22. Resultados de entregabilidad de SWA del caso de prueba C-TF2.

SWA	Tiempo de Respuesta				Throughput				Tasa de Éxito			
	Fault Tolerance		standalone		Fault Tolerance		standalone		Fault Tolerance		standalone	
	SOAP	REST	SOAP	REST	SOAP	REST	SOAP	REST	SOAP	REST	SOAP	REST
10MB	41.92	34.59	26.87	28.59	12.98	14.05	13.92	12.21	41.69	38.34	28.64	27.48
15MB	41.67	41.85	22.46	32.43	10.53	14.95	9.96	10.04	22.79	21.75	17.16	16.82

La Figura 24 se muestran las gráficas de los resultados del caso de prueba C-TF2 de acuerdo a la métrica de entregabilidad. Los resultados de entregabilidad de SWA en la arquitectura tolerante a fallas se muestran en color azul y rojo para los SWA SOAP y REST, respectivamente. Los resultados de entrega de SWA en la arquitectura *standalone* se ilustran en color verde y naranja para SOAP y REST, respectivamente.

En la Figura 24(a) ilustra los resultados del tiempo de respuesta en segundos del caso de prueba C-TF2. Los resultados indican que el ambiente *standalone* responde más rápido las solicitudes realizadas a SWA, comparado con el ambiente tolerante a fallas, ya sea con SWA de 10 o de 15MB. En el ambiente *standalone* las solicitudes a SWA no requieren un reenvío a otro servidor que pueda atender las solicitudes que el primer servidor no puede responder, con esto se menciona que es más tardado un ambiente tolerante a fallas.

La Figura 24(b) ilustra los resultados de *throughput* del caso de pruebas C-TF2. Los mejores resultados de *throughput* se presentan con la arquitectura tolerante a fallas, y esta se compone de dos dispositivos, en la cual ambas responden a las solicitudes a SWA, y eso pone en desventaja a los SWA desplegados en la arquitectura *standalone*, debido a que esta arquitectura dispone de un solo dispositivo para atender todas las solicitudes.

La Figura 24(c) ilustra los resultados de tasa de éxito del caso de pruebas C-TF2. Esta figura y la Tabla 22 dan certeza que hay un porcentaje más alto cuando se utiliza una arquitectura tolerante a fallas, que en una arquitectura *standalone*. Los resultados del caso de prueba C-TF2 determinan que la implementación de SWA en arquitecturas tolerantes a fallas no es la opción más rápida, pero el porcentaje de entrega es mejor.

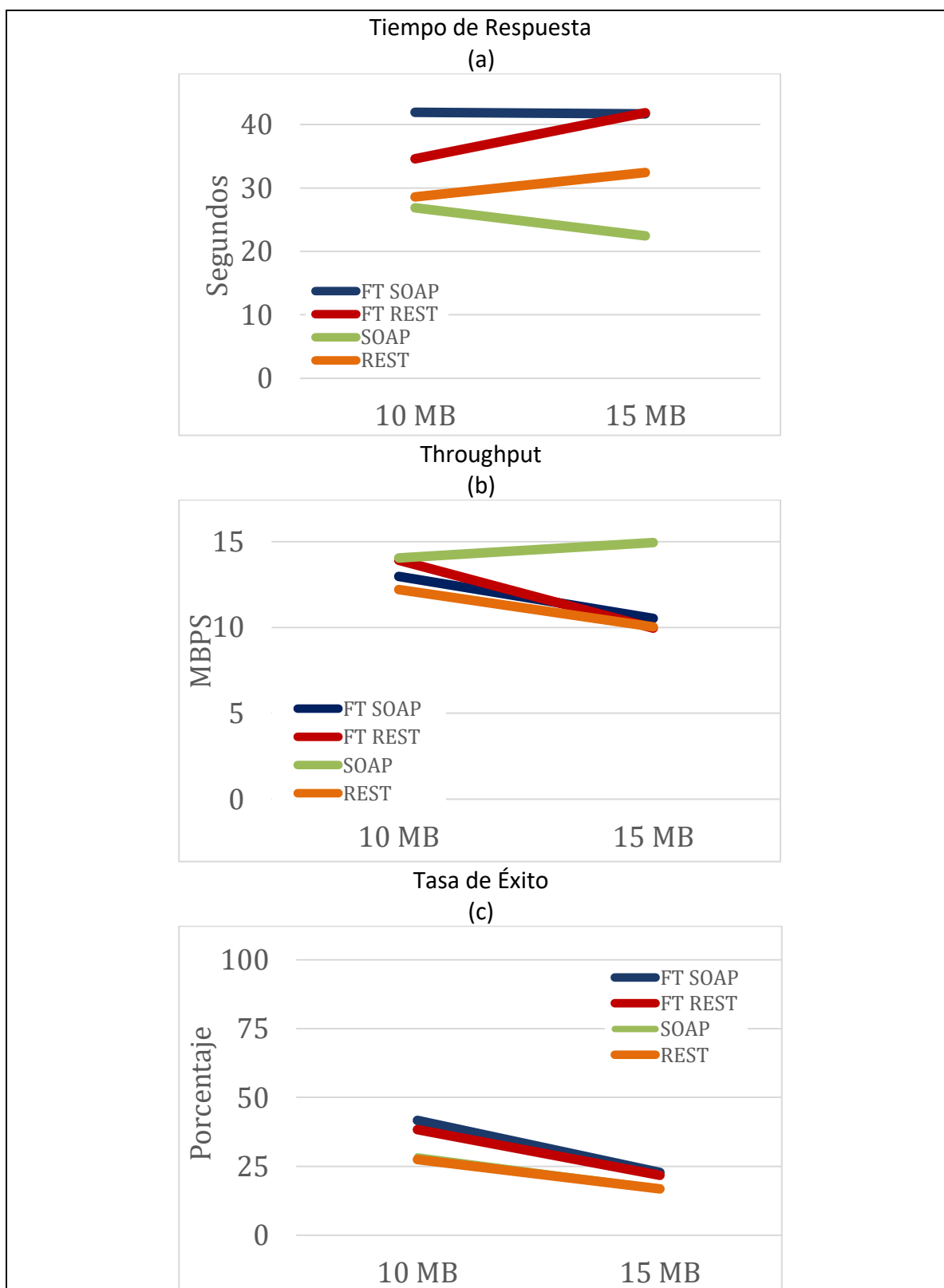


Figura 24. Resultados del caso de prueba C-TF2.

5.1.7 Caso de Prueba S-JHS1

Caso de prueba S-JHS1. Evaluación de la capacidad de entrega de SWA bajo diferentes configuraciones de tamaño de la JHS.

Objetivo del caso de prueba S-JHS1. Evaluar la configuración de diferentes tamaños de JHS para entregar SWA.

Elementos del caso de prueba S-JHS1. Para este caso los elementos de prueba corresponden a los parámetros que definen el tamaño mínimo y máximo de JHS, XMS y XMX respectivamente. Los valores de estos parámetros están listados en la Tabla 23, en donde se especifican dos configuraciones diferenciadas.

Tabla 23. Configuración de parámetros XMS y XMX de JHS.

Configuración 1		Configuración 2	
XMS	XMX	XMS	XMX
512MB	2GB	1GB	2GB
512MB	3GB	2GB	3GB
512MB	4GB	3GB	4GB

Especificaciones de entrada del caso de prueba S-JHS1. Las especificaciones de entrada para el caso de prueba S-JHS1 son las siguientes:

- ♦ La configuración de los parámetros de JHS
- ♦ Los SWA debidamente desplegados.
- ♦ El Payload XML para las solicitudes a SWA del tipo de implementación SOAP.
- ♦ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-JHS1. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV en el directorio definido en el plan de Apache JMeter como se indica en el Anexo B, del cual se obtienen la siguiente información:

- ♦ Tiempo de respuesta en segundos.
- ♦ **Throughput** en MBPS.
- ♦ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-JHS1. En la Tabla 24 se presenta la media estadística de los resultados del caso de prueba S-JHS1, en esta se presentan las columnas del tiempo de respuesta en segundos, **throughput** en MBPS y tasa de éxito en formato porcentual. Las columnas Conf 1 y Conf 2 hacen referencia a las columnas Configuración 1 y Configuración 2 de la Tabla 23, respectivamente.

Tabla 24. Resultados de Entregabilidad de SWA de Java Heap Space.

SWA	Tamaño JHS	Tiempo de Respuesta (Segundos)		Throughput (MBPS)		Tasa de Éxito (Porcentaje)	
		Conf 1	Conf 2	Conf 1	Conf 2	Conf 1	Conf 2
10MB	2GB	24.38	27.04	20.86	13.98	29.87	28.65
10MB	3GB	26.40	26.28	22.25	22.37	47.04	47.04
10MB	4GB	45.33	42.23	16.11	18.16	57.58	60.96

La Figura 25 ilustra los resultados del caso de prueba S-JHS1 de acuerdo a la métrica de entregabilidad de SWA, el tiempo de respuesta en segundos, el **throughput** en MBPS y tasa de éxito en formato porcentual. Los resultados se muestran dos líneas, en color rojo y azul que corresponden a la configuración 1 y configuración 2, respectivamente. El eje de las X corresponde al tamaño de la JHS, de acuerdo al parámetro XMx.

La Figura 25(a) ilustra los resultados del tiempo de respuesta del caso de prueba S-JHS1. En estos se visualiza que no existe una diferencia en cuanto al tiempo de respuesta en el manejo del parámetro XMS. Cabe señalar que el tiempo de respuesta es más alto cuando el parámetro XMx se incrementa, en este caso de 2, 3 o 4 GB debido a que este parámetro define el tamaño total de la JHS, por lo que existe más espacio para generar respuestas exitosas, por lo que se requiere más tiempo de procesamiento de dichas solicitudes.

La Figura 25(b) ilustra los resultados del **throughput** obtenidos del caso de prueba S-JHS1. En esta gráfica se observa que no existe diferencia en el manejo del parámetro XMS, sin embargo, ambas líneas, en color azul y rojo, indican que un valor óptimo es utilizar 3GB como parámetro XMx, por lo que se recomienda encontrar un equilibrio del tamaño asignado de la JHS basado en los resultados de **throughput** para entregar SWA.

En la Figura 25(c) se ilustran los resultados de la tasa de éxito correspondientes al caso de prueba S-JHS1. En dicha figura se observa que el parámetro XMx, correspondiente al tamaño de la JHS, influye de manera directa en el número de solicitudes SWA respondidas con éxito. En particular, a medida que el tamaño de la JHS aumenta, se obtienen mejores valores de tasa de éxito. Estos resultados se encuentran resaltados en color azul en la columna de tasa de éxito de la Tabla 24.

Cuando el tamaño de la JHS es de 2 GB, la tasa de éxito se ubica entre el 28 % y el 29 %. Al incrementar el tamaño de la JHS a 3 GB, la tasa de éxito aumenta al 47 %, mientras que con un tamaño de 4 GB se alcanza un valor de entre el 57 % y el 60 %. Este incremento en la tasa de éxito se explica porque un mayor tamaño de la JHS permite la creación de un mayor número de objetos en memoria, lo que facilita la generación de más respuestas a las solicitudes.

En la Figura 25 se ilustran los resultados de la tasa de éxito obtenidos del caso de prueba S-JHS1. En esta figura se visualiza que en el caso del parámetro XMx o el Tamaño de la JHS permite obtener más solicitudes a SWA exitosas, esto quiere decir que, entre más grande sea el tamaño de la JHS, mejores resultados de tasa de éxito se obtienen, estos datos

están presentes sombreados en color azul en la columna tasa de éxito de la Tabla 24. Cuando el tamaño de la JHS es de 2 GB la tasa de éxito es de 28 y 29 por ciento, cuando el tamaño de la JHS es de 3GB la tasa de éxito es de 47 por ciento y cuando es de 4GB la tasa de éxito incrementó al 57 y 60 por ciento de éxito. El incremento de la tasa de éxito se debe a que se pueden generar más objetos en la JHS y por lo tanto más solicitudes a SWA son respondidas exitosamente.

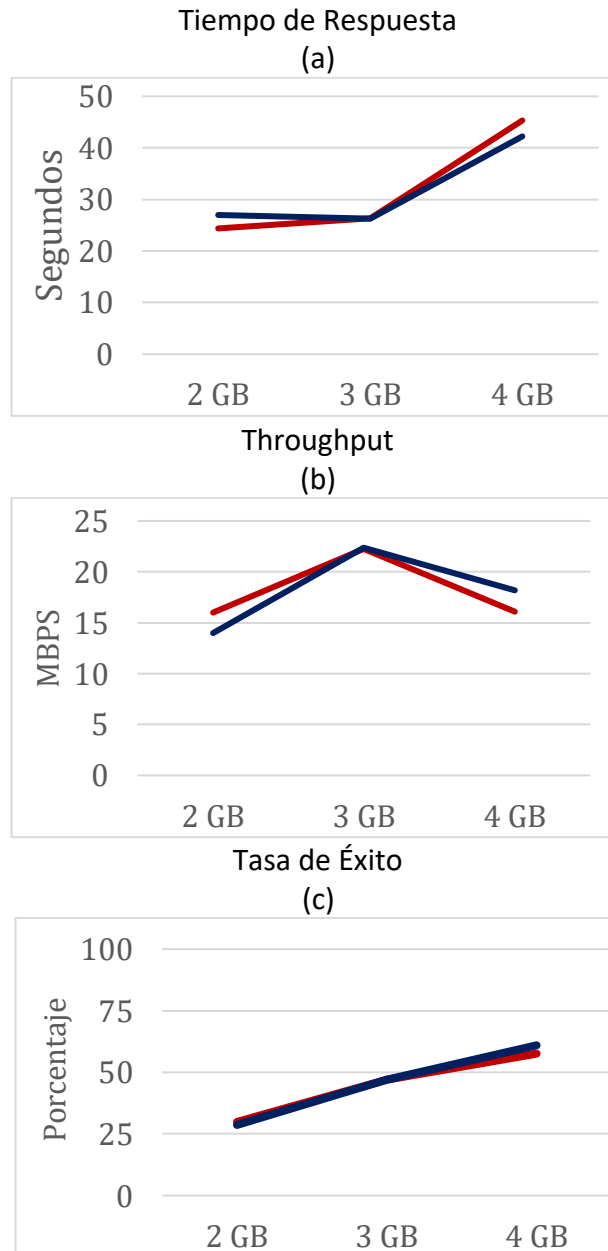


Figura 25. Resultados del caso de Prueba S-JHS1.

5.1.8 Análisis de Manejo de Volumen en JHS en Entregabilidad de SWA

Uno de los motivos por los que las solicitudes no son respondidas con éxito, es debido al tamaño de la JHS. Sin embargo, el objetivo del análisis de esta sección es determinar los límites que se manejan dentro de los ambientes de SWA.

La información que se presenta en esta sección es de los resultados de cada caso de prueba, y en este caso, los SWA mencionados se listan en la respectiva sección de ejecución del respectivo caso de prueba. En la Tabla 25 se lista la media estadística del tamaño de memoria utilizada de acuerdo al tamaño del SWA en BASE64 y el número de solicitudes exitosas en las pruebas de entrega de SWA. La primera columna indica el tamaño del SWA, seguida por el caso de prueba, el tamaño del RA en bytes, tamaño del SWA estimado en BASE64 en megabytes o MB, el resultado de las pruebas de tasa de éxito del SWA de su respectivo caso de prueba y el volumen en MB estimado de utilización de la JHS en este caso de prueba. En esta se puede consultar que los límites con SWA de 5, 10 y 15MB ocupan entre 335 y 440MB cuando se utilizan 2GB en el tamaño de la JHS, estos registros no vienen señalados o resaltados en color, a diferencia del resto.

Por otro lado, cuando se utilizan 3 GB de espacio en la JHS, la tasa de éxito incrementa entre 625 y 640MB de uso de la JHS, lo que permite incrementar el porcentaje o la tasa de éxito, véanse los registros resaltados en color verde. Al igual incrementa el resultado cuando la JHS se define con un espacio de 4 GB, el volumen incrementa en un 100 por ciento aproximadamente, comparado cuando se define en 2 GB, como se puede observar en los registros de la Tabla 25 resaltados en color azul y en color amarillo, en donde el resultado es de entre 691 MB y 810 MB, aproximadamente. En color amarillo se resaltan los resultados de los casos de prueba relacionados con el algoritmo G1 First.

Tabla 25. Análisis de manejo de volumen en JHS en entregabilidad de SWA.

SWA	Caso de Prueba	Tipo	Tamaño Estimado del RA	Tamaño Estimado del SWA	Tasa de Éxito	JHS en Uso Estimada
5MB	S-TSW1	SOAP	5120	6.65	65.94	438.50
5MB	S-TSW1	REST	5120	6.65	65.86	437.97
5MB	S-TPC1	HTTP SOAP	5120	6.65	60.10	399.67
5MB	S-TPC1	HTTP REST	5120	6.65	59.76	397.40
5MB	S-TPC1	HTTPS SOAP	5120	6.65	59.42	395.14
5MB	S-TPC1	HTTPS REST	5120	6.65	60.10	399.67
10MB	S-TSW1	SOAP	10240	13.30	30.34	403.52
10MB	S-TSW1	REST	10240	13.30	29.59	393.55
10MB	S-TPC1	HTTP SOAP	10240	13.30	32.61	433.71
10MB	S-TPC1	HTTPS SOAP	10240	13.30	28.65	381.05
10MB	S-TPC1	HTTP REST	10240	13.30	31.00	412.30
10MB	S-TPC1	HTTPS REST	10240	13.30	27.48	365.48
10MB	S-G1P3	100 Millis	10240	13.30	59.52	791.62
10MB	S-G1P3	200 Millis	10240	13.30	60.43	803.72
10MB	S-G1P3	300 Millis	10240	13.30	58.21	774.19
10MB	S-G1DD	G1 DD	10240	13.30	98.70	1312.71

Pruebas y Análisis de Resultados

10MB	S-G1R2	G1GC Ratio	10240	13.30	62.01	824.73
10MB	S-JHS1	JHS 2GB	10240	13.30	29.87	397.27
10MB	S-JHS1	JHS 2GB	10240	13.30	28.65	381.05
10MB	S-JHS1	JHS 2GB	10240	13.30	30.11	400.46
10MB	S-JHS1	JHS 2GB	10240	13.30	28.65	381.05
10MB	S-JHS1	JHS 3GB	10240	13.30	47.04	625.63
10MB	S-JHS1	JHS 3GB	10240	13.30	47.04	625.63
10MB	S-JHS1	JHS 3GB	10240	13.30	47.40	630.42
10MB	S-JHS1	JHS 3GB	10240	13.30	47.80	635.74
10MB	S-JHS1	JHS 4GB	10240	13.30	57.58	765.81
10MB	S-JHS1	JHS 4GB	10240	13.30	60.96	810.77
10MB	S-JHS1	JHS 4GB	10240	13.30	57.58	765.81
10MB	S-JHS1	JHS 4GB	10240	13.30	60.96	810.77
10MB	S-GC5	PGC	10240	13.30	51.99	691.47
10MB	S-GC5	SGC	10240	13.30	71.96	957.07
10MB	S-GC5	STD	10240	13.30	50.49	671.52
10MB	S-GC5	CMS	10240	13.30	82.80	1101.24
10MB	S-GC5	G1GC	10240	13.30	52.99	704.77
10MB	S-G1DD	G1GC	10240	13.30	52.48	697.98
10MB	S-G1R2	G1GC	10240	13.30	53.25	708.23
15MB	S-TPC1	HTTP SOAP	15360	19.95	18.14	361.89
15MB	S-TPC1	HTTP REST	15360	19.95	17.16	342.34
15MB	S-TPC1	HTTPS SOAP	15360	19.95	18.08	360.70
15MB	S-TPC1	HTTPS REST	15360	19.95	16.82	335.56
15MB	S-TSW1	SOAP	15360	19.95	18.28	364.69
15MB	S-TSW1	REST	15360	19.95	17.67	352.52

5.1.9 Caso de Prueba S-GCA5

Caso de prueba S-GCA5. Entregabilidad de SWA bajo diferentes algoritmos de *Garbage Collector*.

Objetivo del caso de prueba S-GCA5. Evaluar la capacidad de entrega de SWA de los algoritmos de GC.

Elementos del caso de prueba S-GCA5. Los elementos de prueba para el caso de pruebas S-GCA5 se listan en la Tabla 26.

Tabla 26. Algoritmos de GC utilizados para el caso de prueba S-GCA5.

Algoritmos GC	Abreviatura
Standard	STD
Concurrent Mark Sweep	CMS
SerialGC	SGC
ParallelGC	PGC
G1 First	G1GC

Especificaciones de entrada del caso de prueba. Las especificaciones de entrada para el caso de prueba S-GCA5 son las siguientes:

- ◆ El algoritmo de *Garbage Collector* configurado.
- ◆ El tamaño de JHS en 4 GB.
- ◆ Los SWA debidamente desplegados.
- ◆ E el Payload XML para las solicitudes a SWA del tipo de implementación SOAP.
- ◆ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-GCA5. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV en el directorio definido en el plan de Apache JMeter como se indica en el Anexo B, del cual se obtienen la siguiente información:

- ◆ Tiempo de respuesta en segundos.
- ◆ **Throughput** en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-GCA5. En la Tabla 27 se presenta la media estadística de los resultados del caso de prueba S-GCA5, el tiempo de respuesta en segundos, **throughput** en MBPS y tasa de éxito en formato porcentual. La columna 1 indica el algoritmo utilizado en el caso de prueba S-GCA5.

Tabla 27. Resultados del caso de prueba S-GCA5.

Algoritmo	Tiempo de Respuesta (Segundos)	Throughput (MBPS)	Tasa de Éxito (Porcentaje)
STD	330.77	1.92	50.49
CMS	317.56	3.29	82.80
SGC	304.40	2.99	71.96
PGC	328.23	1.98	51.99
G1GC	138.22	4.75	52.99

La Figura 26 ilustra los resultados del caso de prueba S-GCA5 de acuerdo a la métrica de entregabilidad, el tiempo de respuesta en segundos, **throughput** en MBPS y tasa de éxito en formato porcentual. Al costado de la figura se presenta la leyenda que hace referencia a los resultados de cada algoritmo.

En la Figura 26(a) se ilustran los resultados del tiempo de respuesta del caso de prueba S-GCA5. En esta se visualiza que el algoritmo G1 First presenta mejores resultados de tiempo de respuesta, a diferencia de los cuatro algoritmos restantes, que de acuerdo a los datos de tiempo de respuesta presentados en la Tabla 27, tardan más de 300 segundos en responder el grupo de solicitudes, mientras que el algoritmo G1 First demora 138 segundos.

En la Figura 26(b) se ilustran los resultados del **throughput** del caso de prueba S-GCA5. En esta figura se visualiza que el algoritmo G1 First entrega mejores valores en términos de MBPS con 4.75 MBPS, comparado con el resto de los algoritmos, los cuales oscilan entre 1.98 y 3.29 MBPS, de acuerdo a la columna correspondiente de la Tabla 27.

En la Figura 26(c) se ilustran los resultados de la tasa de éxito del caso de prueba S-GCA5. En esta figura se visualiza que el resultado más alto con un porcentaje de 82.80, se obtuvo con el algoritmo CMS, por el contrario, el resultado más bajo se presentó con algoritmo STD de JHS, con un porcentaje de 50.49.

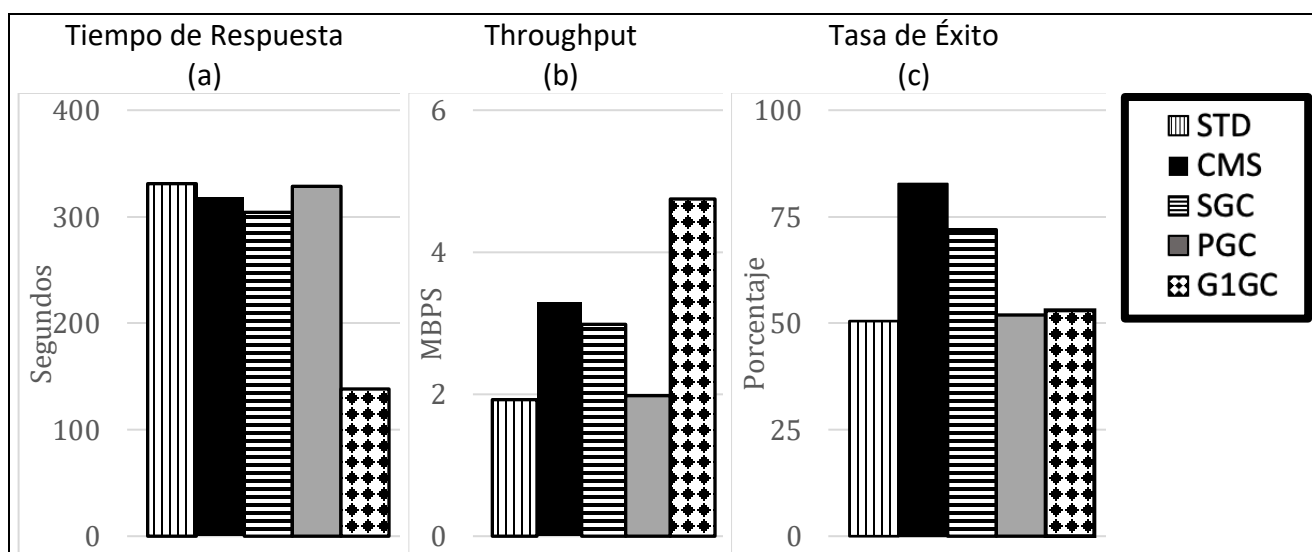


Figura 26. Resultados del caso de prueba S-GCA5.

5.1.10 Caso de Pruebas S-G1R1

Caso de prueba S-G1R1. Capacidad de entrega de SWA utilizando la técnica **G1 First Region**.

Objetivo del caso de prueba S-G1R1. Evaluar la capacidad de entrega de SWA utilizando la técnica **G1 First Region**.

Elementos del caso de prueba S-G1R1. Para las pruebas de entregabilidad de SWA utilizando la técnica **G1 First Region**, se utilizó un SWA de 10MB.

Especificaciones de entrada del caso de prueba S-G1R1. En este caso se realizaron las pruebas con un SWA de 10MB con el objetivo de obtener resultados más representativos, se detectó cierto grado de degradación de las pruebas pero continuidad de las condiciones del escenario de pruebas. Además, se eligió un tamaño de JHS de 4GB para continuar analizando el tamaño de la JHS con respecto a otros casos de prueba. El resto de las especificaciones de entrada para el caso de prueba S-G1R1 se listan a continuación:

- ◆ El algoritmo de **Garbage Collector** G1 First configurado.
- ◆ El parámetro Region del algoritmo G1 First configurado.
- ◆ Los SWA debidamente desplegados.
- ◆ El Payload XML para las solicitudes a SWA del tipo de implementación SOAP.
- ◆ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-G1R1. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV como se indica en el Anexo B, del cual se obtienen la siguiente información:

- ◆ Tiempo de respuesta en segundos.
- ◆ **Throughput** en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-G1R1. Los resultados del caso de prueba S-G1R1 se presentan en la Tabla 28 en razón de la media estadística, de acuerdo a la métrica de entregabilidad, el tiempo de respuesta en segundos, el **throughput** en MBPS y la tasa de éxito en formato porcentual.

Tabla 28. Resultados del caso de prueba S-G1R1.

Algoritmo	Tiempo de Respuesta (Segundos)	Throughput (MBPS)	Tasa de Éxito (Porcentaje)
G1 Region	145.30	4.30	49.52
G1GC	138.22	4.75	52.48

La Figura 27 ilustra los resultados del caso de prueba S-G1R1 a partir de las métricas definidas para la entregabilidad de SWA, el tiempo de respuesta en segundos,

throughput en MBPS y tasa de éxito en formato porcentual. En las barras color negro en color negro se ilustran los resultados de **G1 First Region** y en color gris se ilustran los resultados de **G1GC First**.

En la Figura 27(a) se ilustran los resultados el tiempo de respuesta del caso de prueba S-G1R1 que indica que existe una mínima diferencia en los resultados del tiempo de respuesta. Esta diferencia favorece a G1 First con una diferencia de 7 segundos, sin embargo, es aproximadamente el cinco por ciento de diferencia.

En la Figura 27(b) se ilustran los resultados de **throughput** del caso de prueba S-G1R1, en esta se visualiza que si existe una diferencia mínima a favor de **G1 First Region**, en la Tabla 28 se puede corroborar la diferencia de 4.3 MBPS de G1 First Region y 4.75 de **G1 First**, y se puede indicar que no hay evidencia que la opción de G1 First Region mejora de manera significativa los resultados de **throughput**.

En la Figura 27(c) se ilustran los resultados de la tasa de éxito del caso de prueba S-G1R1. En esta se puede visualizar una diferencia mínima a favor de **G1 First**, en la Tabla 28 se observan resultados con una diferencia de 3 por ciento a favor del caso **G1 First**. Sin embargo, estadísticamente no se puede mencionar que hay una diferencia significativa entre **G1 First** y **G1 First Region** para entregar SWA, tampoco podemos mencionar que la opción G1 First Region tenga que ser considerada en arquitecturas de SWA.

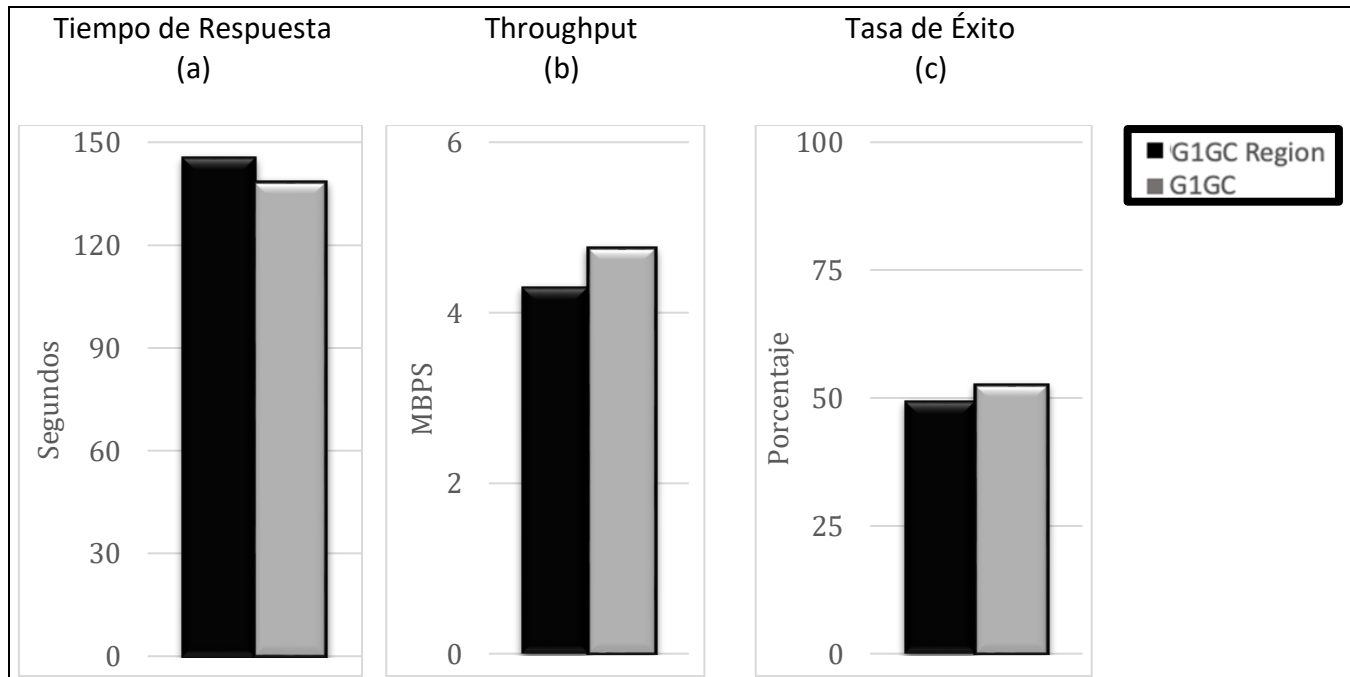


Figura 27. Resultados del caso de prueba S-G1R1.

5.1.11 Caso de Prueba S-G1R2

Caso de prueba S-G1R2. Capacidad de entrega de SWA utilizando la técnica **G1 First Ratio**.

Objetivo del caso de prueba S-G1R2. Evaluar la capacidad de entrega utilizando la técnica **G1 First Ratio**.

Elementos del caso de prueba S-G1R2. Para las pruebas de entregabilidad de SWA con la opción de **G1 First Ratio**, se utilizó un SWA de 10MB.

Especificaciones de entrada del caso de prueba S-G1R2. Las especificaciones de entrada para el caso de prueba S-G1R2 se listan a continuación:

- ◆ El algoritmo de **Garbage Collector** utilizado es G1 First.
- ◆ El parámetro adicional al algoritmo G1 First es Ratio.
- ◆ El tamaño de JHS en 4 GB.
- ◆ Los SWA debidamente desplegados.
- ◆ El payload XML para las solicitudes a SWA del tipo de implementación SOAP.
- ◆ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-G1R2. La salida de cada una de la ejecución de las pruebas es un archivo con formato CSV en el directorio definido en el plan de Apache JMeter como se indica en el Anexo B, del cual se obtienen la siguiente información:

- ◆ Tiempo de respuesta en segundos.
- ◆ **Throughput** en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-G1R2. La Tabla 29 presenta la media estadística de los resultados del caso de prueba S-G1R2 de acuerdo a la métrica de entregabilidad, el tiempo de respuesta en segundos, el **throughput** en MBPS y la tasa de éxito en formato porcentual.

Tabla 29. Resultados del caso de prueba S-G1R2.

Algoritmo	Tiempo de Respuesta (Segundos)	Throughput (MBPS)	Tasa de Éxito (Porcentaje)
G1 First Ratio	110.14	7.08	62.01
G1GC	134.90	4.98	53.25

La Figura 28 ilustra los resultados del caso de prueba S-G1R2 a partir de las métricas definidas para la entregabilidad de SWA, el tiempo de respuesta en segundos, el **throughput** en MBPS y la tasa de éxito en formato porcentual. La leyenda ubicada al lado derecho de la figura indica que en color gris presenta los resultados de **G1 First Ratio** y en color negro los resultados de **G1 First**.

En la Figura 28(a) ilustra los resultados de tiempo de respuesta del caso de prueba S-G1R2. En esta gráfica se visualiza que existe una diferencia en los resultados del tiempo de respuesta, en la columna Tiempo de Respuesta de la Tabla 29 se presentan los resultados con una diferencia de 24 segundos a favor de **G1 First Ratio**, es decir, este último es más rápido.

La Figura 28(b) ilustra los resultados de **throughput** del caso de prueba S-G1R2. En esta se visualiza que existe una diferencia a favor de **G1 First Ratio**, que se confirma en la columna **throughput** de la Tabla 29, con una diferencia mayor de 2MBPS, lo que significa una mejoría de 40 por ciento con respecto a los resultados de **G1 First**.

La Figura 28(c) ilustra los resultados de la tasa de éxito del caso de prueba S-G1R2. En esta se visualiza que la opción de **G1 First Ratio** presenta mejores resultados de entrega de SWA con respecto a G1 First. En la columna tasa de éxito de la Tabla 29 se presenta el resultado de entrega de SWA de **G1 First Ratio**, este es de 62 por ciento, lo que representa 9 por ciento mayor que los resultados de entrega de **G1 First**.

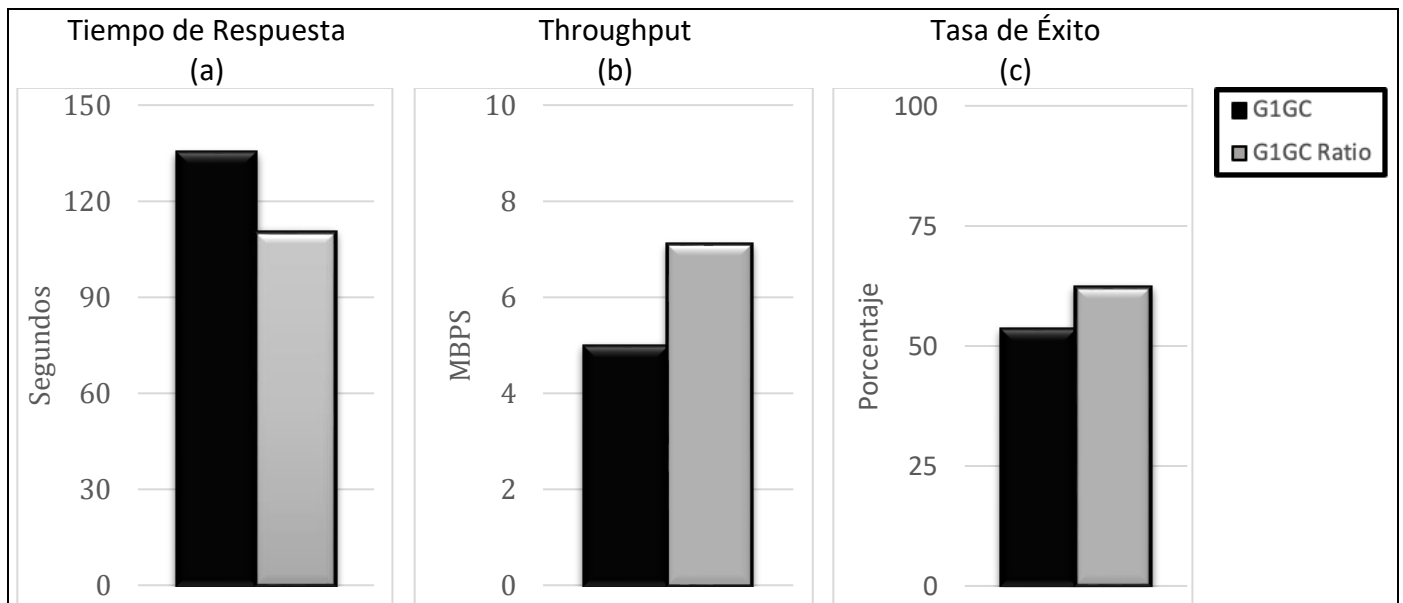


Figura 28. Resultados del caso de prueba S-G1R2.

5.1.12 Caso de Prueba S-G1P3

Caso de prueba S-G1P3. Capacidad de entrega de SWA utilizando la estrategia **G1 First Pauses**.

Objetivo del caso de prueba S-G1P3. Evaluar la capacidad de entrega de SWA con diferentes longitudes de pausas de Garbage Collector con **G1 First Pauses**.

Elementos del caso de prueba S-G1P3. En la Tabla 30 se presentan los elementos de prueba utilizados en el caso de prueba S-G1P3.

Tabla 30. Elementos de prueba del caso de prueba S-G1P3.

SWA	Pausa
SOAP_VIDEO_10MB	100 Millis
SOAP_VIDEO_10MB	200 Millis
SOAP_VIDEO_10MB	300 Millis

Especificaciones de entrada del caso de prueba S-G1P3. Las especificaciones de entrada para el caso de prueba S-G1P3 se listan a continuación:

- ◆ El algoritmo de **Garbage Collector** G1 First configurado.
- ◆ El parámetro Pause del algoritmo G1 First configurado.
- ◆ Los SWA debidamente desplegados.
- ◆ El Payload XML para las solicitudes a SWA del tipo de implementación SOAP.
- ◆ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-G1P3. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV en el directorio definido en el plan de Apache JMeter como se indica en el Anexo B, del cual se obtienen la siguiente información:

- ◆ Tiempo de respuesta en segundos.
- ◆ **Throughput** en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-G1P3. En la Tabla 31 se presenta la media estadística de los resultados de entregabilidad de SWA del caso de prueba S-G1P3. La columna 1 indica la longitud de las pausas del algoritmo **G1 First**.

Tabla 31. Resultados del caso de prueba S-G1P3.

Pausa	Tiempo de Respuesta (Segundos)	Throughput (MBPS)	Tasa de Éxito (Porcentaje)
100 Millis	124.16	6.03	59.52
200 Millis	110.23	6.92	60.43
300 Millis	115.13	6.41	58.21

En la Figura 29 se visualizan los resultados del caso de prueba S-G1P3, tiempo de respuesta, **throughput** y tasa de éxito.

Los resultados de tiempo de respuesta del caso de prueba S-G1P3 se ilustran en la Figura 29(b), lo que se observa que existe una diferencia entre las tres muestras, entre la primera y la segunda muestra existe una diferencia de 14 segundos a favor de la última, esto representa un 12 por ciento de rapidez. Con respecto a la prueba de 200 Millis con la prueba de 300 Millis, la diferencia es de 9 segundos a favor de la prueba de 200 Millis.

En la Figura 29(b) se visualizan los resultados de **throughput** del caso de prueba S-G1P3, estos también se presentan numéricamente en la columna de **throughput** de la Tabla 31, en la se indican una diferencia entre 100 y 200 Millis a favor de la segunda opción, un 15 por ciento de mejoría aproximadamente.

En la Figura 29(c) se visualizan los resultados de tasa de éxito del caso de prueba S-G1P3. Los datos de estos resultados se presentan en la columna de tasa de éxito en la Tabla 31. Estos datos en las tres pausas configuradas son similares, es decir, la configuración o administración de las pausas del algoritmo **G1 First** no indican que se pueden mejorar los resultados de entrega de SWA.

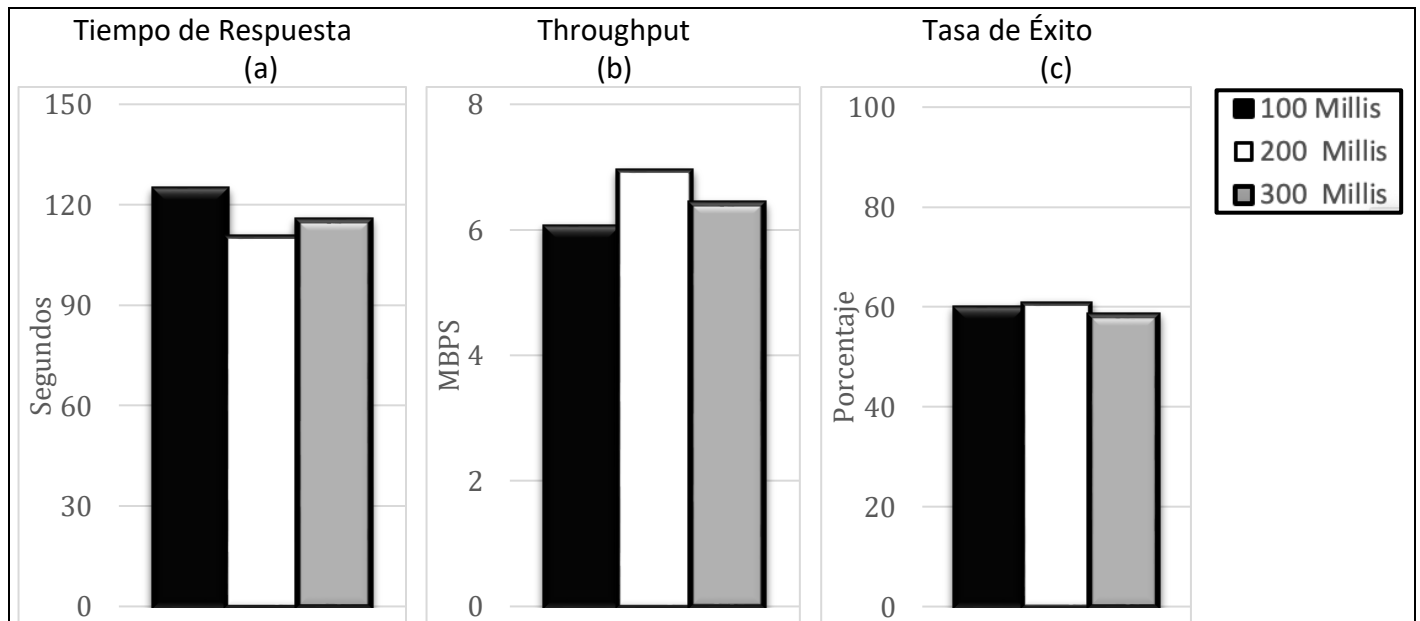


Figura 29. Resultados del caso de prueba S-G1P3.

5.1.13 Caso de Prueba S-G1DD1

Caso de prueba S-G1DD1. Capacidad de entrega de SWA utilizando la técnica **G1 First Deduplication**.

Objetivo del caso de prueba S-G1DD1. Evaluar la capacidad de entrega de SWA al emplear la técnica **G1 First Deduplication**.

Elementos del caso de prueba S-G1DD1. El caso de prueba S-G1DD1 utiliza un SWA de 10MB de tipo de implementación SOAP.

Especificaciones de entrada del caso de prueba S-G1DD1. Las especificaciones de entrada para el caso de prueba S-G1DD1 son las siguientes:

- ◆ El algoritmo de **Garbage Collector** G1 First configurado.
- ◆ El parámetro **deduplication** del algoritmo G1 First configurado.
- ◆ Los SWA debidamente desplegados.
- ◆ El Payload XML para las solicitudes a SWA del tipo de implementación SOAP.
- ◆ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-G1DD1. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV en el directorio definido en el plan de Apache JMeter como se indica en el Anexo B, del cual se obtiene la siguiente información:

- ◆ Tiempo de respuesta en segundos.
- ◆ **Throughput** en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-G1DD1. En la Tabla 32 se presenta la media estadística de los resultados del caso de prueba S-G1DD1.

Tabla 32. Resultados del caso de prueba G1DD1.

Parámetro	Tiempo de Respuesta (Segundos)	Throughput (MBPS)	Tasa de Éxito (Porcentaje)
G1GC	138.22	4.75	52.48
Deduplicación	81.16	15.65	98.7

En la Figura 30 se ilustran los resultados del caso de prueba S-G1DD1, mismos que están representados en la Tabla 32.

La Figura 30(a) ilustra los resultados de tiempo de respuesta del caso de prueba S-G1DD1, esta información esta presentada en la columna Tiempo de Respuesta de la Tabla 32. Los resultados indican que cuando se utiliza la técnica de **deduplication** se reduce en un 40 por ciento los tiempos de respuesta, G1GC tarda 138 segundos mientras que la **deduplication** hace el trabajo en 82 segundos, aproximadamente. Por lo que es evidente

utilizar la técnica de G1 First **deduplication** para mejorar los tiempos de respuesta de aplicaciones que implementen SWA.

En la Figura 30(b) se ilustran los resultados de **throughput** del caso de prueba S-G1DD1, esta información esta presentada en la columna **throughput** de la Tabla 32. Los datos de **throughput** indican que cuando se utiliza la técnica de **deduplication** se mejora significativamente estos resultados, debido a que solo se genera una instancia de la cadena **String** del SWA, y esta se reutiliza para todas las instancias que se generan en memoria JHS para las solicitudes realizadas al mismo SWA.

En la Figura 30(c) se ilustran los resultados de tasa de éxito del caso de prueba S-G1DD1, la información también se presenta en la columna tasa de éxito de la Tabla 32. Estos datos indican que la utilización de **G1 First Deduplication** obtiene 98.7 por ciento, mientras que los resultados de **G1 First** se obtiene el 52 por ciento de tasa de éxito. La opción de **G1 First Deduplication** es un parámetro que se debe considerar para entregar SWA. Esta característica reduce los tiempos de respuesta, incrementa la capacidad de **throughput** y de tasa de éxito. En la técnica de **deduplication** del algoritmo G1 First solo se genera una instancia de la cadena **String** del SWA, y esta se reutiliza para todas las solicitudes realizadas al mismo SWA, permitiendo obtener más solicitudes exitosas, permitiendo al lenguaje de programación Java solo encargarse de administrar las solicitudes, y evitar otra instancia del SWA y además de administrar la memoria Java Heap Space.

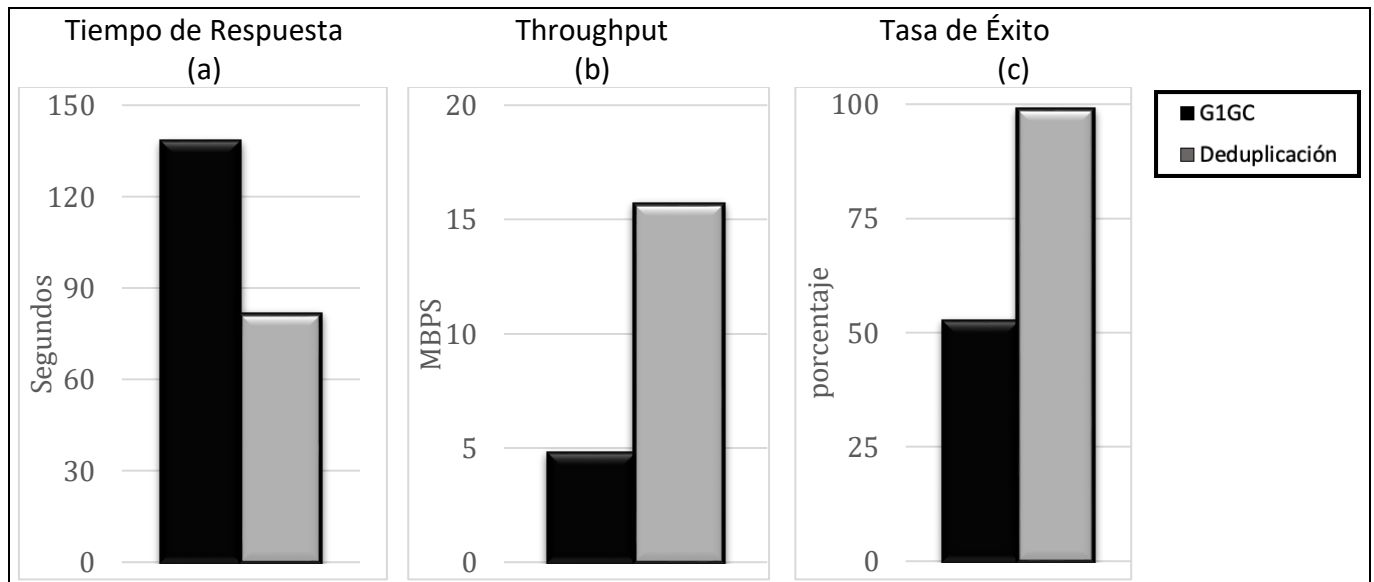


Figura 30. Resultados del caso de prueba S-G1DD1.

5.1.14 Caso de Prueba C-SC1

Caso de prueba C-SC1. Capacidad de entrega de SWA con la arquitectura Server Cache.

Objetivo del caso de prueba. Evaluar la capacidad de entrega de SWA con la implementación de la arquitectura Server Cache en SWA.

Elementos del caso de prueba. Los elementos de prueba para el caso de prueba C-SC1 se listan en la Tabla 33.

Tabla 33. Elementos del caso de prueba C-SC1.

Tamaño	SOAP	REST
10MB	SOAP_VIDEO_10MB	REST_VIDEO_10MB
15 MB	SOAP_VIDEO_15MB	REST_VIDEO_15MB

Especificaciones de entrada del caso de prueba C-SC1. Las especificaciones de entrada para el caso de prueba S-SC1 se listan a continuación:

- ♦ La arquitectura de cómputo implementada con Server Cache.
- ♦ La arquitectura de cómputo implementada en standalone.
- ♦ Los SWA debidamente desplegados.
- ♦ Para los SWA de tipo de implementación SOAP, se requiere del payload XML.
- ♦ Endpoint del SWA.

Especificaciones de salida del caso de prueba C-SC1. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV, en el directorio definido en el plan de Apache JMeter como se indica en el Anexo B, del cual se obtiene la siguiente información:

- ♦ Tiempo de respuesta en segundos.
- ♦ **Throughput** en MBPS.
- ♦ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba C-SC1. En la Tabla 34 se presenta la media estadística de los resultados del caso de prueba C-SC1, el tiempo de respuesta en segundos, el **throughput** en MBPS y el porcentaje de tasa de éxito. La columna 1 se hace referencia al SWA objeto de las pruebas de entregabilidad de SWA, en el resto se hace referencia a la arquitectura con implementación de Server Cache y a la arquitectura Standalone.

Tabla 34. Resultados de Entregabilidad de SWA con Server Cache.

SWA	Tiempo de Respuesta (Segundos)		Throughput (MBPS)		Tasa de Éxito (Porcentaje)	
	Server Cache	Standalone	Server Cache	Standalone	Server Cache	Standalone
10MB	13.52	76.53	89.17	5.27	96.84	30.78
15MB	163.43	32.19	12.62	11.98	85.74	17.94

En la Figura 31 se ilustran los resultados de entregabilidad de SWA del caso de prueba C-SC1. Las dos líneas en color azul y en color rojo indican los resultados de la implementación de la arquitectura Server Cache y de la arquitectura **standalone**, respectivamente.

En la Figura 31(a) se ilustran los resultados de entregabilidad de SWA con respecto al tiempo de respuesta de respuesta del caso de prueba C-SC1. Como se puede. Observar, los mejores resultados se presentan cuando se implementa la arquitectura Server Cache para entregar SWA de 10MB, no es el caso para los SWA de 15MB. El volumen de información es excesivo cuando se utilizan SWA de 15MB, esto provoca la saturación del canal de comunicación por lo que se genera un cuello de botella.

En la Figura 31(b) se ilustran los resultados de entregabilidad de SWA con respecto al **throughput** del caso de prueba C-SC1, y se presentan en la Tabla 34. En estos se observa que el mejor resultado se presenta con la implementación de **Server Cache**, en este se obtuvo 89MBPS con un SWA de 10MB, a diferencia con la arquitectura **standalone**, en esta última se obtuvo un **throughput** de 5.3MBPS. En la arquitectura Server Cache no se procesa la solicitud, esta se mantiene en memoria y solo se re envía la información. En contraste, en la arquitectura **standalone**, se tiene que procesar la respuesta de cada solicitud de manera individual.

En la Figura 31(b) se ilustran los resultados de entregabilidad de SWA con respecto a la tasa de éxito del caso de prueba C-SC1, estos están presentados en la Tabla 34. Como se observan estos resultados, la tasa de éxito es bastante favorable cuando se implementa la arquitectura de Server Cache, la diferencia con SWA de 10 y 15MB, con 96 y 85 por ciento respectivamente, a 30 y 17 por ciento de los resultados de la arquitectura **standalone**.

Los resultados de las pruebas de Server Cache permiten determinar que esta opción es una de las más viables para utilizar SWA, indistintamente el tipo de implementación de SWA. Estadísticamente los resultados de las pruebas con Server Cache son superiores que cuando se utiliza una arquitectura **standalone**.

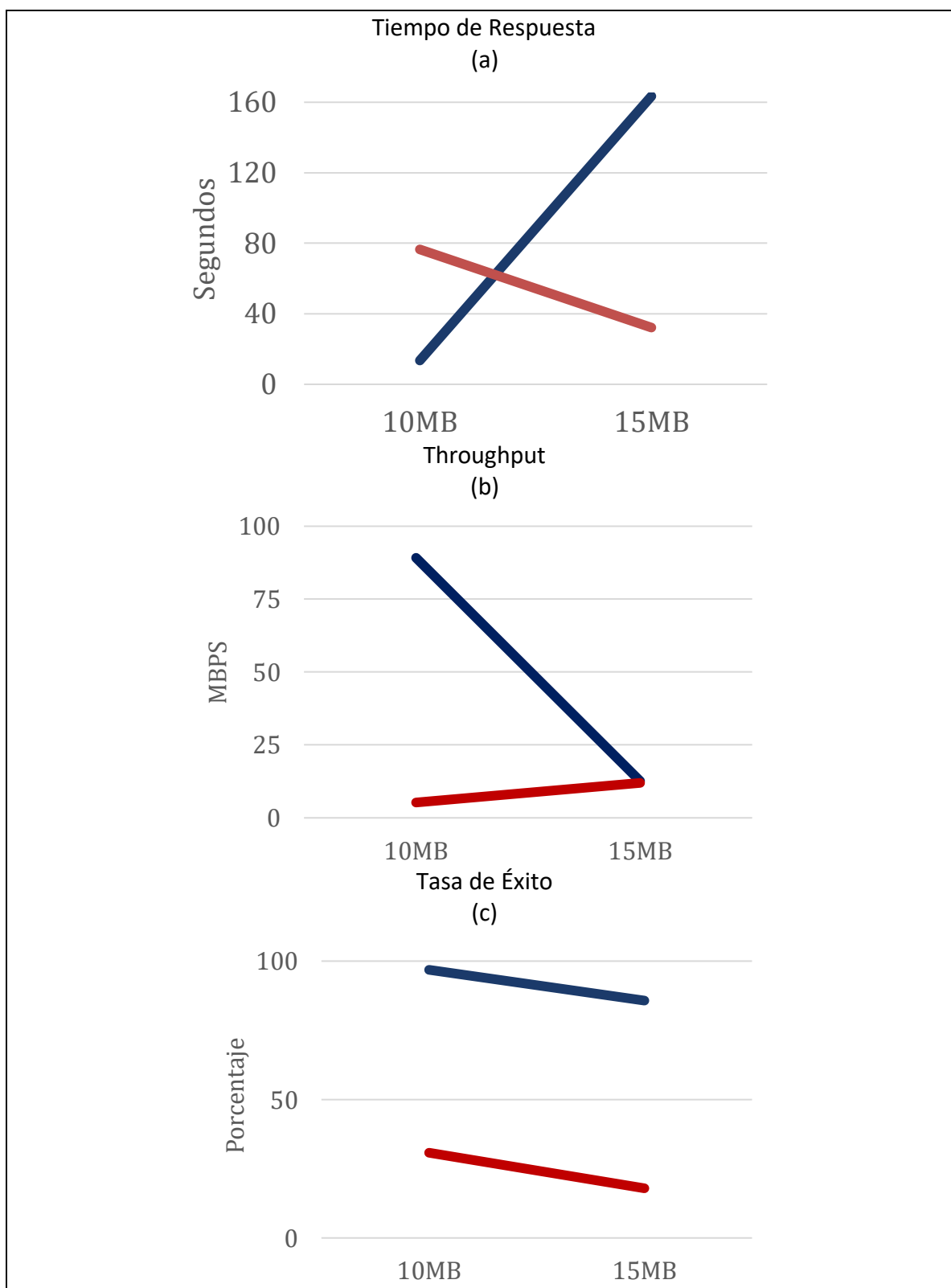


Figura 31. Resultados del caso de pruebas C-SC1.

5.1.15 Caso de Prueba C-CD4

Caso de prueba C-CD4. Capacidad de entrega de SWA con aplicaciones desplegadas en arquitecturas contenerizadas con Docker.

Objetivo del caso de prueba C-CD4. Evaluar la capacidad de entrega de SWA con aplicaciones desplegadas en arquitecturas contenerizadas.

Elementos del caso de prueba C-CD4. Los elementos de prueba para el caso de prueba C-CD4 se listan en la Tabla 35.

Tabla 35. Elementos del caso de prueba C-CD4.

SWA
SOAP_IMAGEN_1MB
SOAP_IMAGEN_3MB
SOAP_IMAGEN_5MB
SOAP_VIDEO_10MB
SOAP_VIDEO_15MB

Especificaciones de entrada del caso de prueba C-CD4. Para este caso de prueba dichas especificaciones son las siguientes:

- ◆ La arquitectura de cómputo contenerizada debidamente implementada.
- ◆ El balanceador de carga configurado para el caso de la arquitectura de cómputo contenerizada.
- ◆ La arquitectura de cómputo implementada en standalone.
- ◆ Los SWA debidamente desplegados.
- ◆ El **payload XML** para los SWA de tipo de implementación SOAP.
- ◆ Endpoint del SWA.

Especificaciones de salida del caso de prueba C-CD4. Como salida de cada una de la ejecución de las pruebas es un archivo con formato CSV en el directorio definido en el plan de Apache JMeter como se indica en el Anexo B, del cual se obtiene la siguiente información:

- ◆ Tiempo de respuesta en segundos.
- ◆ **Throughput** en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba C-CD4. En la Tabla 36 se presenta la media estadística de los resultados del caso de pruebas C-CD4. En la primera columna se indica el SWA objeto de las pruebas, en el resto de las columnas se presentan los resultados de las arquitecturas contenerizadas o **standalone**, según indique el encabezado.

Tabla 36. Resultados del caso de pruebas C-CD4.

SWA	Tiempo de Respuesta (Segundos)		Throughput (MBPS)		Tasa de Éxito (Porcentaje)	
	Docker	Standalone	Docker	Standalone	Docker	Standalone
1MB	9.60	23.62	14.37	5.55	100	100
3MB	22.70	65.63	17.09	5.76	100	100
5MB	51.00	80.77	13.46	5.27	100	65.94
10MB	80.35	68.90	11.17	5.51	77.10	30.24
15MB	84.26	65.49	7.85	5.41	39.46	18.28

La Figura 32 ilustra los resultados de entregabilidad de SWA del caso de prueba C-CD4. Las líneas en color azul indican los resultados de entrega de SWA desplegados en arquitecturas contenerizadas, en color rojo se indican los resultados de los SWA desplegados en arquitecturas *standalone*.

La Figura 32(a) ilustra los resultados de tiempo de respuesta del caso de prueba C-CD4, estos están presentados en la Tabla 36. De acuerdo con esta información, la utilización de los contenedores permite mejorar las arquitecturas de SWA. En SWA de granularidad fina y media, la arquitectura contenerizada entrega mejores resultados de tiempo de respuesta, la diferencia con los SWA de granularidad gruesa radica en que hay más solicitudes respondidas exitosamente con arquitecturas contenerizadas, y esas solicitudes tienen que procesarse, por lo que se considera más tiempo para ser respondidas.

La Figura 32(b) ilustra los resultados del *throughput* del caso de prueba C-CD4, los datos de los resultados están presentados en la Tabla 36. En esta gráfica se observa que los SWA desplegados en arquitecturas contenerizadas obtiene mejores resultados. Los datos de *throughput*, oscilan entre 11 y 17 MBPS, a excepción del SWA de 15MB en el que se obtuvo una media de 7.85 MBPS. En el caso de las aplicaciones desplegadas en arquitecturas *standalone*, los resultados oscilan entre 5 y 6 MBPS, lo que es inferior a los resultados obtenidos en arquitecturas contenerizadas.

La Figura 32(c) ilustra los resultados de los resultados de tasa de éxito del caso de prueba C-CD4. Los resultados de las pruebas de entregabilidad de SWA de tasa de éxito más alta se presentan con aplicaciones desplegadas en arquitecturas contenerizadas. Los datos presentados en la Tabla 36 indican mejores resultados en SWA de 5, 10 y 15 MB, que aparecen sombreados en color azul. En el caso de los SWA de 1 y 3 MB son 100 por ciento en los casos de aplicaciones desplegadas en arquitecturas contenerizadas.

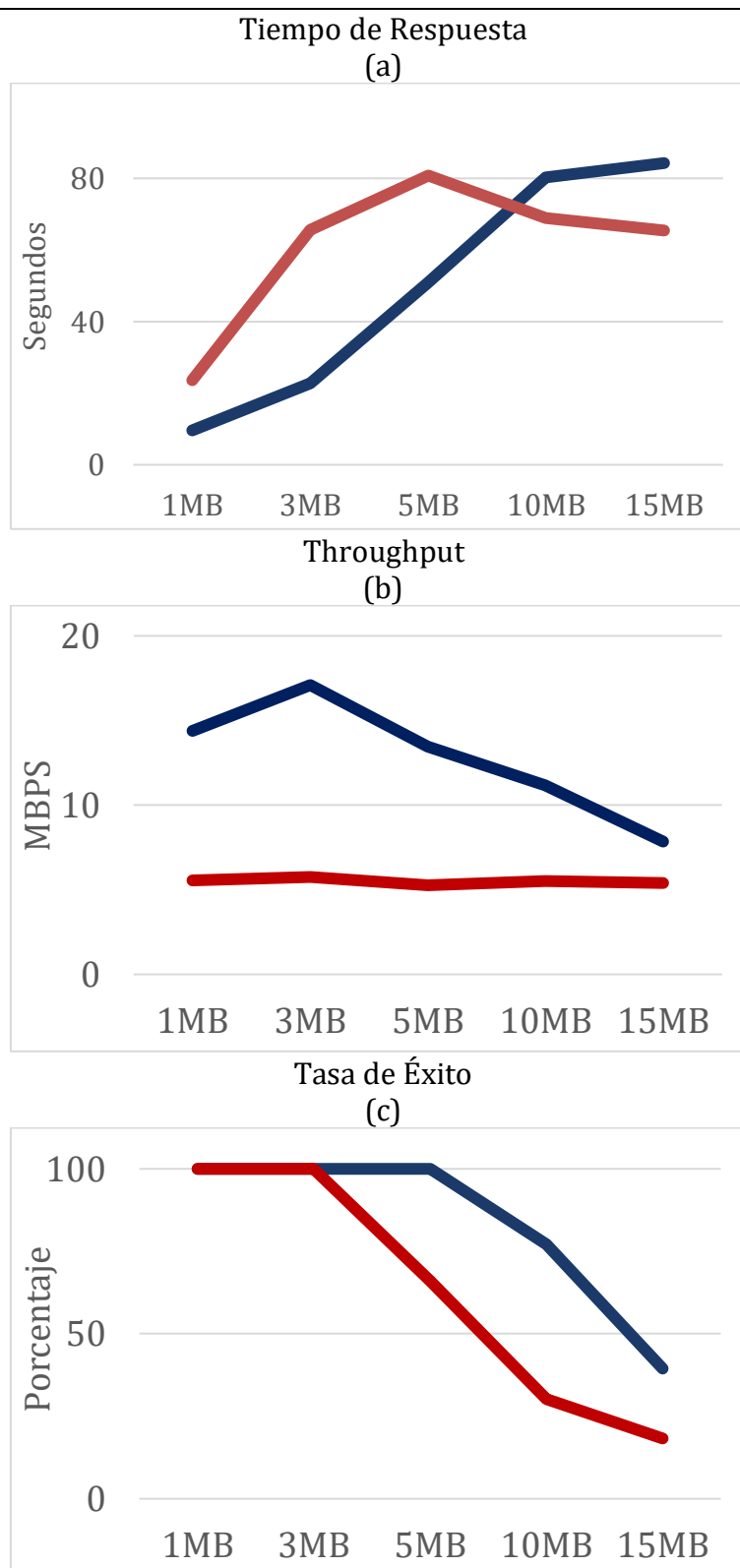


Figura 32. Resultados del caso de pruebas C-CD4.

5.1.16 Caso de Prueba S-JPS1

Caso de prueba S-JPS1. Capacidad de entrega de SWA diseñados con *EJB Singleton*.

Objetivo del caso de prueba S-JPS1. Determinar la capacidad de entrega de SWA implementados con el diseño *EJB Singleton* en SWA.

Elementos del caso de prueba S-JPS1. Para las pruebas del caso de prueba S-JPS1 se listan en la Tabla 37.

Tabla 37. Elementos del caso de prueba S-JPS1.

Singleton	Estándar
SingletonWS1MB	SOAP_IMAGEN_1MB
SingletonWS3MB	SOAP_IMAGEN_3MB
SingletonWS5MB	SOAP_IMAGEN_5MB
SingletonWS10MB	SOAP_VIDEO_10MB
SingletonWS15MB	SOAP_VIDEO_15MB

Especificaciones de entrada del caso de prueba S-JPS1. A continuación de listan las especificaciones de entrada para el caso de prueba S-JPS1:

- ◆ Los SWA diseñados con EJB Singleton debidamente desplegados.
- ◆ Los SWA estándar debidamente desplegados.
- ◆ El payload XML.
- ◆ Endpoint del SWA.

Especificaciones de salida del caso de prueba S-JPS1. La salida de cada una de la ejecución de las pruebas es un archivo con formato CSV como se indica en el Anexo B, del cual se obtiene la siguiente información:

- ◆ Tiempo de respuesta en segundos.
- ◆ **Throughput** en MBPS.
- ◆ El porcentaje de respuestas exitosas de las solicitudes realizadas.

Resultados del caso de prueba S-JPS1. En la Tabla 38 se presentan los resultados del caso de prueba S-JPS1, en razón de la media estadística. En estos se destacan en tono azul, los resultados de entregabilidad de SWA con el Singleton, y se determina que el diseño estándar del SW presenta más sobrecarga de procesamiento por parte del servidor, comparado con el diseño Singleton de SWA.

Tabla 38. Resultados del caso de pruebas S-JPS1.

SWA	Tiempo de Respuesta (Segundos)		Throughput (MBPS)		Tasa de Éxito (Porcentaje)	
	Singleton	Estándar	Singleton	Estándar	Singleton	Estándar
1MB	11.22	11.15	13.10	13.77	100.00	100.00
3MB	32.12	23.94	12.12	16.44	100.00	100.00
5MB	50.54	26.86	13.61	16.31	100.00	100.00
10MB	98.29	28.59	12.72	12.21	99.99	27.48
15MB	69.94	32.43	13.75	10.04	43.03	16.82

En la Figura 33 se visualizan los resultados del caso de prueba S-JPS1, los resultados del diseño de SWA. Singleton se destacan en color azul y en color rojo los resultados de entregabilidad de SWA del tipo estándar.

En la Figura 33(a) se ilustran los resultados de tiempo de respuesta del caso de prueba S-JPS1. En esta gráfica se observa que los SWA con el diseño Singleton obtiene mejores resultados de entrega, a diferencia de los SWA con el diseño estándar. En algunos resultados es hasta 50 por ciento menor tiempo de respuesta, como es el caso de los SWA de 1, 3 y 15MB.

La Figura 33(b) ilustra los resultados del **throughput** del caso de prueba S-JPS1, en la que se puede observar que los SWA con el diseño Singleton presentan mejores resultados de **throughput**, que los SWA con el diseño estándar. Los datos en la columna **throughput** de la Tabla 38 son muy parejos, en algunos hay diferencia pero son afectados por la columna tasa de éxito.

La Figura 33(c) ilustra los resultados de tasa de éxito del caso de prueba S-JPS1. En esta se observa que los SWA implementados con Singleton en SWA de 5, 10 y 15MB es mayor el resultado de tasa éxito, que el de los SWA implementados de manera estándar, con cada SWA correspondiente. En la Tabla 38 aparecen sombreados en azul los resultados de tasa de éxito de los SWA implementados con la técnica de programación Singleton y en naranja aparecen las celdas con los resultados de entrega de los SWA implementados de manera estándar. Es evidente que la implementación de SWA con el método Singleton permite mejorar los resultados de entrega.

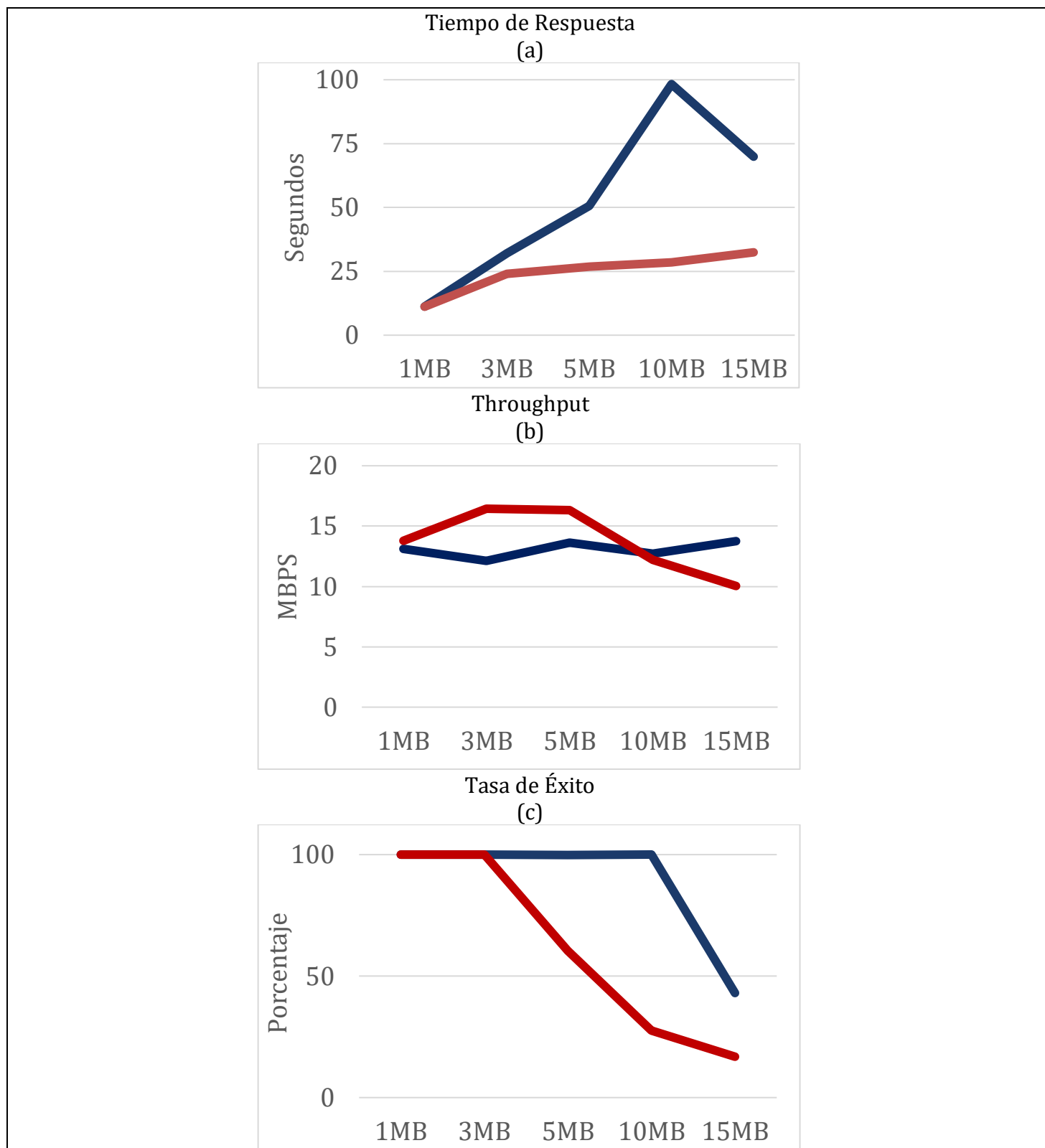


Figura 33. Resultados del caso de pruebas S-JPS1.

5.1.17 Caso de Prueba S-RC1

Caso de prueba S-RC1. Capacidad de entregade SWA implementados con el diseño **REST Cache**.

Objetivo del caso de prueba S-RC1. Evaluar la capacidad de entrega de SWA implementados con el diseño **REST Cache**.

Elementos del caso de prueba S-RC1. Para las pruebas de entregabilidad de SWA implementados con la técnica de programación **REST Cache** se utilizó un SW REST de tipo video de 10MB, con el objetivo de validar el comportamiento de la funcionalidad de REST Cache.

Especificaciones de salida del caso de prueba S-RC1.

- Código BASE64 del recurso contenido en el SWA.

Resultados del caso de prueba S-RC1. Despliegue del código BASE64 del recurso contenido en el SWA.

Después de las pruebas realizadas, se logró identificar que la respuesta se guarda satisfactoriamente en la caché del navegador del cliente, por lo que se deduce que este método de implementación no ayuda a entregar los SWA, sin embargo, ayuda a guardar el recurso posterior a la entrega de acuerdo al parámetro de tiempo de vida configurado en el servicio. En la Figura 34 se muestra el código en BASE64 que se obtiene de respuesta a una solicitud de un SWA de 10MB de tipo video y en la Figura 35 se pueden visualizar los parámetros de respuesta de la Cache del cliente.

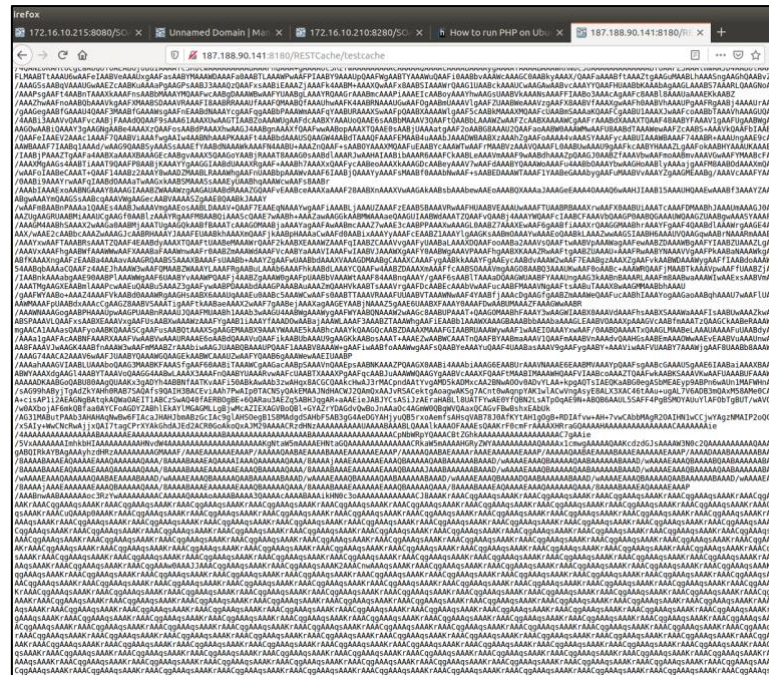


Figura 34. Resultado del caso de pruebas S-RC1.

Pruebas y Análisis de Resultados

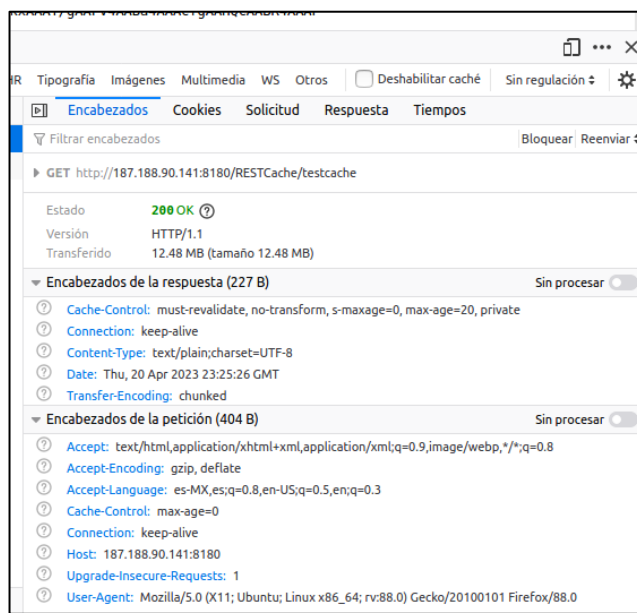


Figura 35. Parámetros de respuesta de la prueba S-RC1.

5.2 Análisis de Resultados

5.2.1 Análisis Estadístico de los Resultados de Entregabilidad de SWA de Dos Muestras

En el análisis estadístico de las pruebas independientes, en las que se busca determinar la diferencia entre dos muestras mediante métodos estadísticos; se consideraron las características de los datos obtenidos y los supuestos asociados a cada prueba. En este contexto, se aplicaron a los siguientes casos de prueba:

- Caso de Prueba S-TSW1
- Caso de Prueba S-SM1
- Caso de Prueba S-TPC1
- Caso de Prueba C-SC1
- Caso de Prueba C-TF2
- Caso de Prueba C-C4
- Caso de Prueba C-CD4
- Caso de Prueba S-G1R1
- Caso de Prueba S-G1R2
- Caso de Prueba S-G1DD1
- Caso de Prueba S-JPS1

5.2.2 Pruebas Estadísticas para Muestras Independientes

- **Prueba de normalidad (Shapiro-Wilk y Kolmogorov-Smirnov).** Estas pruebas se aplican para validar la normalidad de una distribución de las diferencias entre las dos muestras.

- **Prueba T de Welch.** Es una alternativa robusta a la T de Student cuando no se cumple el supuesto de homogeneidad de varianzas.

- **Prueba de Wilcoxon.** Es una prueba que se emplea como alternativa no paramétrica a la prueba T, en los casos en que no se cumple el supuesto de normalidad, permitiendo evaluar si la mediana de las diferencias difiere de cero.

- **Medidas de asimetría y curtosis (Skewness y Kurtosis).** Este cálculo se realiza para caracterizar el comportamiento de la distribución de los datos y apoyar la validación de los supuestos estadísticos.

- **Prueba robusta de Yuen.** Esta prueba se aplica para confirmar la existencia de diferencias significativas considerando la influencia de valores atípicos y la robustez cuando no existe normalidad ni homocedasticidad.

En el análisis estadístico se utilizó el método de Shapiro-Wilk y Kolmogorov-Smirnov para evaluar la normalidad de los datos y determinar si éstos siguen una distribución gaussiana, además de la homogeneidad de varianzas. Al no cumplirse los supuestos, se procedió a calcular los valores de asimetría (*Skewness*) y curtosis (*Kurtosis*) con el fin de caracterizar la distribución de los datos.

Posteriormente, se empleó la prueba robusta de Yuen, la cual permite validar la existencia de diferencias estadísticas a partir del tamaño del efecto. Este procedimiento se aplicó a las métricas de tiempo de respuesta, *throughput* y tasa de éxito. La Figura 36 ilustra el flujo del análisis estadístico para la comparación de dos muestras en las pruebas de entregabilidad de los SWA.

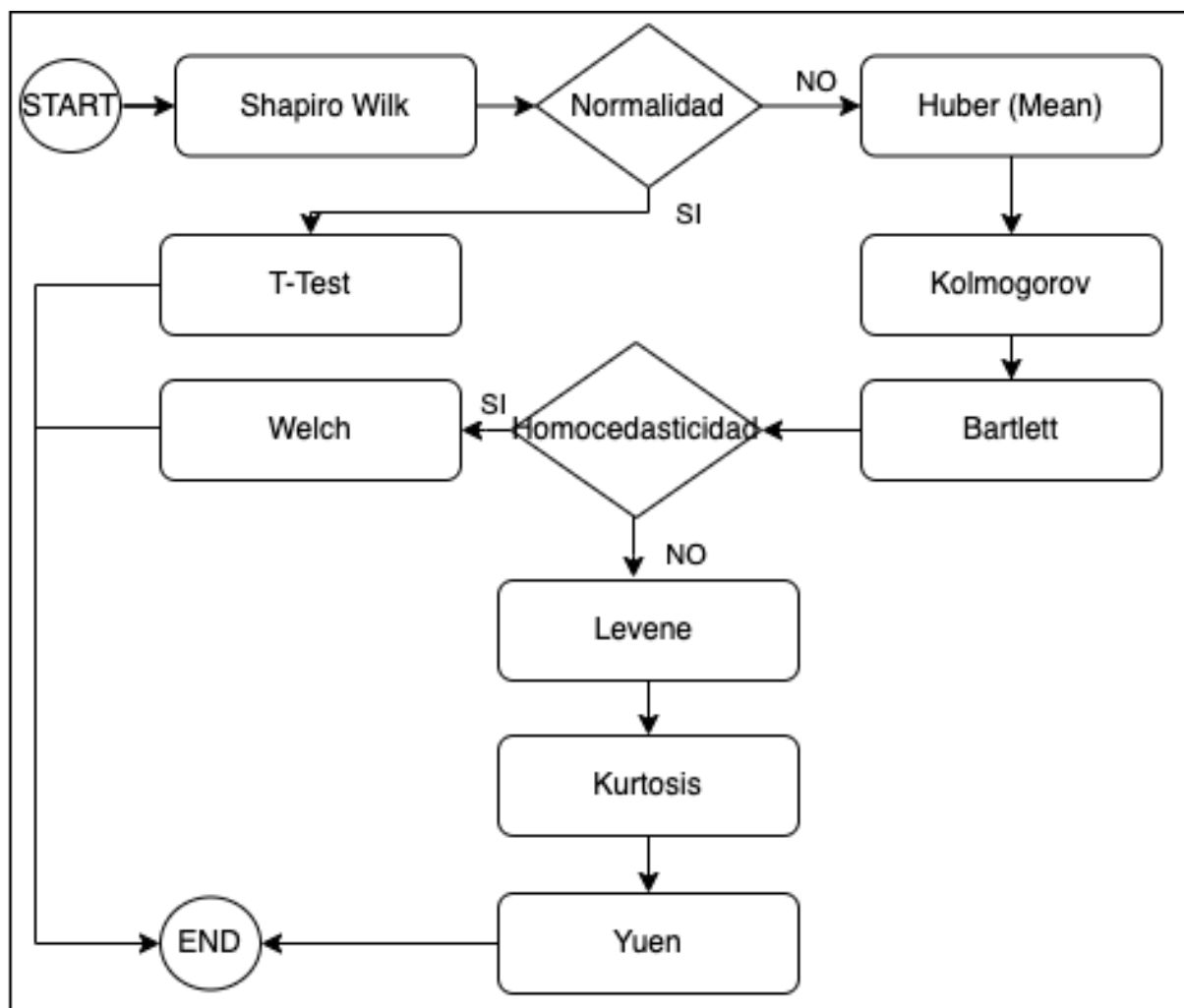


Figura 36. Proceso del análisis estadístico para muestras independientes.

5.2.3 Análisis Estadístico de los Resultados de Entregabilidad de SWA de Más de Dos Muestras

El análisis estadístico es una etapa en la evaluación de los resultados obtenidos en pruebas de entregabilidad de SWA, permite transformar los datos crudos en información significativa para sustentar la experimentación y la interpretación de los resultados. Para realizar la comparación de más de dos muestras, se deben seleccionar los métodos que se ajusten a las características de los datos y al cumplimiento de los supuestos estadísticos, como normalidad, homocedasticidad e independencia. Cuando existen más de dos muestras independientes no es adecuado si no se cumplen los supuestos de normalidad y homocedasticidad. En estos casos es necesario recurrir a métodos no paramétricos robustos para determinar conclusiones válidas. El análisis estadístico para evaluar las diferencias entre más de más de dos muestras para los siguientes casos de prueba:

- ◆ Caso de Prueba S-GCA5
- ◆ Caso de Prueba S-G1P3
- ◆ Caso de Prueba S-JHS1

En este análisis se aplicaron tanto pruebas paramétricas como técnicas robustas y no paramétricas, acorde a las propiedades de los datos.

ANOVA de un Factor. Esta prueba permite comparar las medias de más de dos grupos para determinar si existen diferencias significativas, bajo los supuestos de normalidad e igualdad de varianzas.

Prueba Kruskal-Wallis. Esta prueba permite identificar las influencia de un valor categórico sobre una variable cuantitativa.

Prueba de Dunn. Esta prueba permite realizar comparaciones múltiples *post hoc* entre pares de grupos después de haber encontrado diferencias estadísticamente significativas.

La Figura 37 ilustra el flujo del análisis estadístico llevado a cabo para la comparación de más de dos muestras en las pruebas de entregabilidad de SWA.

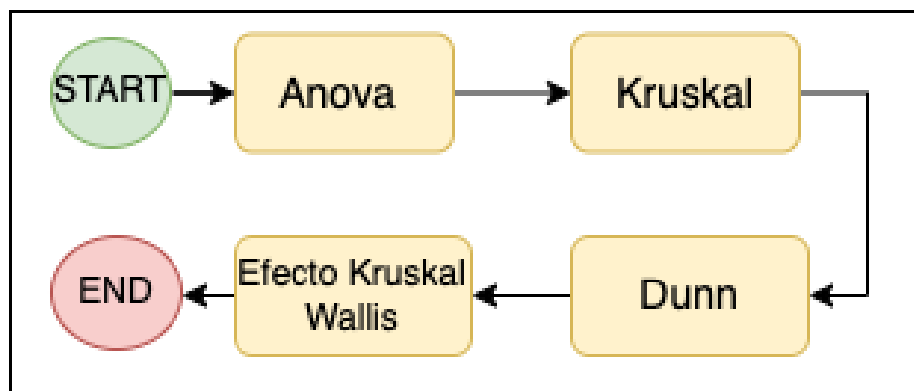


Figura 37. Proceso del análisis estadístico para más de 2 muestras.

5.2.4 Prueba Global

Para determinar si existen diferencias estadísticamente significativas entre las distribuciones de las muestras analizadas, se empleó la prueba de **Kruskal-Wallis**, esta prueba evalúa la hipótesis nula de que todas las muestras provienen de la misma distribución.

H_0 : Las muestras provienen de la misma distribución.

H_1 : Al menos una muestra difiere significativamente.

5.2.5 Comparaciones Post-hoc

En caso de obtener un resultado significativo en la prueba de **Kruskal-Wallis**, se realizaron comparaciones múltiples **post-hoc** mediante la prueba de **Dunn**. Este análisis permite identificar entre qué pares de muestras existen diferencias estadísticamente significativas.

5.2.6 Tamaño del Efecto

Con el objetivo de cuantificar la magnitud de las diferencias observadas, se calculó el tamaño del efecto mediante el estadístico **épsilon cuadrado (ϵ^2)**, el cual indica la proporción de variabilidad explicada por la variable de agrupación.

La interpretación de ϵ^2 se realizó conforme a los siguientes criterios:

- ♦ $\epsilon^2 \approx 0.01$: efecto pequeño
- ♦ $\epsilon^2 \approx 0.06$: efecto mediano
- ♦ $\epsilon^2 \geq 0.14$: efecto grande

5.3 Definición de Métricas de Entregabilidad

En esta sección se definen las fórmulas propuestas para evaluar la capacidad de entrega de SWA. En el trabajo de [Benítez, 2018] se propone la evaluación de entregabilidad de SWA, pero en esta solo se consideran los atributos dinámicos como el tiempo de respuesta, throughput y la tasa de éxito, en esta no se consideran atributos internos de clientes o servidores los cuales son los que se analizan en este trabajo.

5.3.1 Entregabilidad

Para propósitos de este trabajo, la fórmula de entregabilidad se definió tomando en cuenta tres atributos: (Th) **throughput**, (Tr) tiempo de respuesta y (Te) tasa de éxito. Los pesos asignados a los atributos th , tr y te listan en la Tabla 39 de acuerdo al nivel de importancia de este trabajo, la fórmula que aquí se propone considera que un atributo puede tener mayor importancia que los otros y por lo tanto se le puede asignar mayor peso. En este caso se le proporciona un valor de 1 para no alterar el resultado, como lo muestra la ecuación 1.

Tabla 39. Pesos asignados para atributos de entregabilidad del trabajo de [Benítez, 2018]

Atributo	Peso
Tr	0.2
Th	0.3
Te	0.5

$$E = \sum_{i=1}^n A_i(\text{peso}) \quad (1)$$

Donde:

- ♦ $\sum_{i=1}^n A_i$ Es la sumatoria de los atributos de E .
- ♦ A : Es el valor de cada atributo que pertenece a E , en este caso Th , Tr y Te .
- ♦ n : Es la sumatoria de los atributos tomados en cuenta para la entregabilidad, en este caso $n = 3$.

5.3.2 Tiempo de Respuesta

El tiempo de respuesta está definido como el tiempo que tarda en llegar la respuesta exitosa después de que fue recibida la petición, es un valor que proporciona la herramienta Apache JMeter. El cálculo se realiza considerando que más de una petición pueda hacerse a un SWA al mismo tiempo, como se describe en la ecuación (2).

$$Tr = 1 \left(\frac{t}{x} \right) \quad (2)$$

Dónde:

- ♦ t : Es el tiempo que tarda en responder el grupo de solicitudes, obtenido de JMeter.
- ♦ x : Es el número total de solicitudes exitosas del grupo de solicitudes, para el mismo SWA.
- ♦ Donde Tr es un atributo con comportamiento opuesto a T_h y Te , es decir, mientras el tiempo de respuesta debe tender al mínimo, los otros dos deben tender al máximo. Para solventar este comportamiento, el valor obtenido para Tr se calcula como muestra la ecuación (2) dividiendo 1 entre el tiempo de respuesta.

5.3.3 Throughput

El *throughput* se define en función del tamaño del recurso de aprendizaje en MB. La forma de calcularlo se define en la ecuación 3.

$$Th = \frac{\sum_{i=1}^n MB * x}{t} \quad (3)$$

Dónde:

- ♦ MB : Es el tamaño del recurso que entrega el servicio Web de aprendizaje a una o múltiples solicitudes.
- ♦ t : Es el tiempo que tarda en responder el grupo de solicitudes, obtenido de JMeter.
- ♦ x : Es el número total de solicitudes exitosas del grupo de solicitudes para el mismo SWA.

5.3.4 Tasa de Éxito

La tasa de éxito es el porcentaje de las solicitudes respondidas satisfactoriamente del total de las solicitudes realizadas. Para definir que la solicitud es respondida con éxito, HTTP responde con el código de respuesta 200. Esta métrica es evaluada utilizando la ecuación 4.

$$Te = \frac{x}{q} \quad (4)$$

Donde:

- x : Es el número de solicitudes exitosas.
- q : Es el total de solicitudes realizadas.

5.4 Modelo De Calidad de Entrega de SWA

El modelo de QoS que se propone para evaluar la capacidad de entrega de SWA es un modelo basado en el modelo aditivo, esto se debe a que se presentan un conjunto de atributos de calidad heterogéneos que requieren crear un balance entre los atributos. Por ejemplo, el tamaño de la memoria JHS y la optimización de esta, la arquitectura de Server Caché, la estrategia de programación Java EJB Singleton.

5.4.1 Atributos del Servidor

De acuerdo a los resultados obtenidos de la capacidad de entrega de SWA, la ponderación de atributos del servidor se presenta en la Tabla 40, en el cual el Server Cache, Java EJB Singleton y G1 First Deduplication se ponderaron con los valores más altos. En cambio, G1 First Ratio, y Fault Tolerance se ponderaron con los valores más bajos. Por otro lado, el atributo JHS se ponderó con un valor importante, el cual se compone de otros tres valores como es Contenerización, Clusterización y el tamaño asignado a JHS. Es importante mencionar que este último se le debe asignar un valor razonable que permita desplegar los SWA. Sin embargo, existen técnicas de optimización de memoria como Deduplication, Ratio y la técnica de programación Singleton mejoran la capacidad de entrega y evitan que exista el problema de desbordamiento de memoria.

Tabla 40. Peso asignado por atributo del servidor de acuerdo al modelo de entregabilidad de SWA.

Atributo	Abreviatura	Peso
Fault Tolerance	FT	0.05
Server Cache	SC	0.20
Java EJB Singleton	JSP	0.15
G1 First Ratio	G1FR	0.05
G1 First Deduplication	G1DD	0.20
Contenerización	CD	0.10
Clusterización	CL	0.05
Java Heap Space	JHS	0.20

El atributo JHS del modelo de entrega de SWA, es un atributo compuesto que consta de 3 sub atributos, Tamaño de JHS, Contenerización y Clusterización. Estos tres sub atributos están agrupados debido a que amplían el espacio asignado a la JHS disponible, por lo que es posible generar más objetos en memoria y por lo tanto, incrementar la tasa de éxito. A continuación se presentan las métricas propuestas para obtener el valor adecuado para el atributo JHS, ya que estos sub atributos son valores dinámicos que pueden modificar la capacidad de entrega de SWA.

5.4.1.1 Contenerización

La métrica de la contenerización es una relación entre el factor de pérdida de memoria, el número de contenedores, el tamaño de la JHS asignado a cada contenedor y el tamaño de la JHS. El valor se realiza calculando el producto de la ponderación de la JHS y el cálculo del tamaño de la JHS de los contenedores existentes que contienen el SWA

desplegado, por el factor de tolerancia de memoria, esto último dividido entre el tamaño del SWA. Esta métrica se presenta en la ecuación 5.

$$Contenerización = JHS * Min \left(1, \left[\frac{K * \sum_{i=1}^n Tamano JHS}{Tamano SWA} \right] \right) \quad (5)$$

Donde:

- JHS*: Corresponde a la ponderación asignada por entregabilidad de SWA definida en la Tabla 40, en este caso el valor es de 0.20.
- $\sum_{i=1}^n Tamano JHS$: Corresponde a la sumatoria del tamaño de JHS asignado a cada contenedor en donde se encuentran desplegados los SWA.
- n*: Indica cada contenedor que despliega los SWA.
- Tamano SWA*: Indica el tamaño del SWA en MB.
- K*: Es el factor de tolerancia de memoria, para nuestro caso se define en 1.5.

5.4.1.2 Clusterización

La métrica de la clusterización es una relación entre el factor de pérdida de memoria, el tamaño de la JHS y el tamaño del SWA. Este resultado se obtiene calculando el producto de la ponderación de la JHS y el producto del tamaño de la JHS por dos, por el factor de tolerancia de memoria, esto último dividido entre el tamaño del SWA. En este caso se multiplica por dos debido a que la clusterización solo replica los artefactos desplegados en el host en el resto de los nodos. Esta métrica se presenta en la ecuación 6.

$$Clusterización = JHS * Min \left(1, \left[\frac{K * 2 * Tamano JHS}{Tamano SWA} \right] \right) \quad (6)$$

Donde:

- JHS*: Corresponde a la ponderación asignada por entregabilidad de SWA definida en la Tabla 40, en este caso es 0.20.
- $\sum_{i=1}^n Tamano JHS$: Corresponde a la sumatoria del tamaño de JHS asignado a cada nodo del cluster en donde se encuentran desplegados los SWA.
- n*: Indica cada nodo que despliega los SWA.
- Tamano SWA*: Indica el tamaño del SWA en MB.
- K*: Es el factor de tolerancia de memoria, para nuestro caso se define en 1.5.

5.4.1.3 Tamaño JHS

La métrica del tamaño de la JHS es una relación entre el factor de pérdida de memoria, el tamaño de la JHS y el tamaño del SWA. Este resultado se obtiene calculando el producto de la ponderación de la JHS y el producto del tamaño de la JHS, por el factor de tolerancia de memoria, esto último dividido entre el tamaño del SWA. Esta métrica se presenta en la ecuación 7.

$$JHS\ Size = JHS * Min \left[1, \left(\frac{k * Tamano\ JHS}{Tamano\ SWA} \right) \right] \quad (7)$$

Donde:

JHS: Corresponde a la ponderación asignada por entregabilidad de SWA definida en la Tabla 40, en este caso es 0.2.

Tamano JHS: Corresponde a la sumatoria del tamaño de JHS asignado a cada nodo del cluster, contenedor o dispositivo en donde se encuentran desplegados los SWA.

Tamano SWA: Indica el tamaño del SWA en MB.

K: Es el factor de tolerancia de memoria, para nuestro caso está definida en 1.5.

5.4.1.4 Entregabilidad Del Servidor

La capacidad de entrega del servidor se estima a partir de la suma de los atributos definidos en la Tabla 40. Este valor esta expresado como se presenta en la ecuación 8.

$$ES = \sum_{i=1}^n Ai(peso) \quad (8)$$

Donde:

$\sum_{i=1}^n Ai$ Es la sumatoria de los atributos de *E*.

Ai Es el valor de cada atributo que pertenece a *E*, en este caso *FT*, *SC*, *G1FR*, *G1DD*, *JSP*, *JHS*.

n Es la sumatoria de los atributos tomados en cuenta para la entregabilidad, en este caso *n* = 1.

FT: Corresponde a la ponderación asignada al atributo Tolerancia a Fallas, definida en la Tabla 40, en este caso es 0.05.

SC: Corresponde a la ponderación asignada al atributo Server Cache, definida en la Tabla 40, en este caso es 0.20.

G1FR Corresponde a la ponderación asignada al atributo G1 First Ratio, definida en la Tabla 40, en este caso es 0.05.

G1DD Corresponde a la ponderación asignada al atributo G1 First Deduplication, definida en la Tabla 40, en este caso es 0.20.

- JSP* Corresponde a la ponderación asignada al atributo Java EJB Singleton, definida en la Tabla 40, en este caso es 0.15.
- JHS:* Corresponde a la ponderación asignada a la métrica Tamaño JHS, definida en el punto 5.1.3 en este caso se suman los valores obtenidos del cálculo de las métricas de Contenerización, Clusterización y el tamaño de JHS, posteriormente, el resultado de este por el producto de ponderación de JHS de la Tabla 40, que corresponde el valor 0.2.
- CD* Corresponde a la ponderación asignada a la métrica Clusterización, este se calcula de acuerdo a la definición del punto 5.5.1.1.
- CL* Corresponde a la ponderación asignada a la métrica Contenerización, este se calcula de acuerdo a la definición del punto 5.5.1.2.

5.4.2 Atributos del Cliente

La ponderación de los atributos del cliente para entregar SWA se presenta en la Tabla 41. En este se consideran atributos como el Tipo de SW y el protocolo de comunicación, debido a que el cliente es el que indica cual método utilizar para solicitar un SWA y cuál sería el protocolo de comunicación que mejor le conviene. El tipo de SW se pondera con un valor más alto debido a que la arquitectura de SW es más amplia y se pueden encontrar otros métodos de entrega de SWA, por ejemplo, SOAP MTOM.

Tabla 41. Peso asignado a los atributos del cliente de acuerdo al modelo de entregabilidad de SWA.

Atributo	Peso
Tipo de SWA	0.60
Protocolo de Comunicación	0.40

Los tipos de SW, SOAP y REST, utilizados para las pruebas de entregabilidad de SWA están ponderados en la Tabla 42.

Tabla 42. Peso asignado al atributo Tipo de SWA.

Atributo	Peso
SOAP	1.00
REST	0.90

Los tipos protocolos de comunicación HTTP y HTTPS en las pruebas de entregabilidad de SWA están ponderados en la Tabla 43.

Tabla 43. Peso asignado al atributo Protocolo de Comunicación.

Atributo	Peso
HTTP	1.00
HTTPS	0.80

5.4.2.1 Entregabilidad del Cliente

La capacidad de entrega de los atributos del cliente que se propone en este trabajo, se evalúa como se presenta en la ecuación 9:

$$EC = (TSWA * pt) + (PCSW * pp) \quad (9)$$

Donde:

EC: Indica la entregabilidad del cliente

TSWA: Indica la ponderación asignada por tipo de SW.

PCSWA: Indica la ponderación del atributo de comunicación de los SWA.

pp: Indica la ponderación asignada al protocolo de comunicación, definida en la tabla 41.

pt: Indica la ponderación asignada al tipo de SWA, definida en la tabla 41.

5.4.3 Entregabilidad

En la Tabla 44 se presentan las ponderaciones del cliente y servidor del modelo de capacidad de entrega de SWA. Para este caso, el servidor se pondera con el valor más alto debido a que incluye los atributos que permiten recibir las solicitudes, procesar y enviar la información de respuesta de los SWA. Por otro lado, el cliente mantiene un valor mínimo, considerando que entre más opciones tenga para solicitar un SWA, ya sea de diferente tipo de SW o de diferente protocolo de comunicación, estos atributos puedan ser mejor evaluados o ponderados en trabajos posteriores. En la ecuación 10 se presenta la fórmula de la entregabilidad en la que se incluye el resultado de la entregabilidad del servidor y la entregabilidad del cliente.

Tabla 44. Peso asignado a los atributos del cliente de acuerdo al modelo de entregabilidad de SWA.

Atributo	Peso
Servidor	0.95
Cliente	0.05

$$E = (ES * ps) + (EC * pc) \quad (10)$$

Donde:

E: Indica la entregabilidad del cliente

EC: Indica el resultado de la capacidad de entrega de los atributos del cliente.

ES: Indica el resultado de la capacidad de entrega de los atributos del servidor.

ps: Indica la ponderación del servidor, definida en la tabla 44.

pc: Indica la ponderación del cliente, definida en la tabla 44.

5.5 Evaluación de Resultados

En esta sección se evalúa el modelo de entregabilidad de SWA propuesto en este trabajo de tesis.

5.5.1 Evaluación 1

La evaluación 1 se presenta con un SWA De 15MB en el que se incluyen atributos como deduplicación y JSP Singleton. En la Tabla 45 se presentan los datos de la evaluación en donde se incluye el nombre del SWA, el conjunto de atributos del modelo de QoS de servidores y de clientes, la respectiva ponderación y el resultado.

Tabla 45. Evaluación de un SWA de 15MB.

Atributo	Descripción	Ponderación	Resultado
SWA	REST_SINGLETON_15MB	0.90	0.54
Protocolo	HTTP	0.10	0.40
Total Cliente		0.94	0.047
Deduplication	SI	0.20	0.20
Singleton	SI	0.15	0.15
JHS	2 GB	0.20	0.04
Total Servidor:		0.95	0.39
		Total Entrega:	0.4175

Aplicando la fórmula de JHS:

$$Tamano JHS = JHS * Min \left[1, \left(\frac{k * (Tamano JHS)}{Tamano SWA} \right) \right]$$

$$Tamano JHS = 0.2 * Min \left[1, \left(\frac{1.5 * 2}{15} \right) \right]$$

$$Tamano JHS = 0.2 * Min[1, 0.2]$$

$$Tamano JHS = 0.2 * 0.2$$

$$Tamano JHS = 0.04$$

Aplicando la fórmula de Entregabilidad del Servidor:

$$ES = \sum_{i=1}^n Ai(peso)$$

$$ES = 0 + 0 + 0 + 0.20 + 0.15 + 0.04$$

$$ES = 0.39$$

Calculando la entregabilidad del Cliente:

$$EC = (TSWA * pt) + (PCSW * pp)$$

Sustituyendo los valores de la fórmula:

$$\begin{aligned} EC &= (0.9 * 0.6) + (1 * 0.4) \\ EC &= 0.54 + 0.40 \\ EC &= 0.94 \end{aligned}$$

Calculando la entregabilidad global:

$$E = (ES * ps) + (EC * pc)$$

Sustituyendo los valores de la fórmula:

$$\begin{aligned} E &= (0.39 * 0.95) + (0.94 * 0.05) \\ E &= 0.39 + 0.046 \\ E &= 0.4175 \end{aligned}$$

5.5.2 Evaluación 2

La evaluación 2 se realiza con un SWA De 1MB incluyendo atributos como deduplicación. En la Tabla 46 se presentan los datos de la evaluación en donde se incluye el nombre del SWA, el conjunto de atributos del modelo de QoS de servidores y de clientes, la respectiva ponderación y el resultado de la entregabilidad.

Tabla 46. Evaluación de un SWA de 1MB.

Atributo	Descripción	Ponderación	Resultado
SWA	REST_AUDIO_1MB	0.90	0.54
Protocolo	HTTP	0.10	0.40
Total Cliente		0.94	0.047
Deduplication	SI	0.20	0.20
JHS	2 GB	0.20	0.20
Total Servidor		0.95	0.38
		Total:	0.4270

Aplicando la fórmula de JHS:

$$Tamano JHS = JHS * Min \left[1, \left(\frac{k * (Tamano JHS)}{Tamano SWA} \right) \right]$$

$$Tamano JHS = 0.2 * Min \left[1, \left(\frac{1.5 * 2}{1} \right) \right]$$

$$Tamano JHS = 0.2 * Min[1,3]$$

$$Tamano JHS = 0.2 * 1$$

$$Tamano JHS = 0.2$$

Calculando la entregabilidad del Servidor:

$$E = \sum_{i=1}^n Ai(peso)$$

Aplicando la fórmula de Entregabilidad:

$$E = 0.0 + 0.0 + 0.0 + 0.20 + 0.20 + 0.0$$

$$E = 0.40$$

Calculando la entregabilidad del Cliente:

$$EC = (TSWA * pt) + (PCSW * pp)$$

Sustituyendo los valores de la fórmula:

$$EC = (0.9 * 0.6) + (1 * 0.4)$$

$$EC = 0.54 + 0.40$$

$$EC = 0.94$$

Calculando la entregabilidad global:

$$E = (ES * ps) + (EC * pc)$$

Sustituyendo los valores de la fórmula:

$$E = (0.40 * 0.95) + (0.94 * 0.05)$$

$$E = 0.38 + 0.047$$

$$E = 0.4270$$

5.5.3 Evaluación 3

La evaluación 3 se realiza con un SWA De 5MB incluyendo atributos como Tolerancia a Fallas, Singleton y contenerización. Cada contenedor tiene asignado 3GB de espacio en JHS. En la Tabla 47 se presentan los datos de la evaluación en donde se incluye el nombre del SWA, el conjunto de atributos del modelo de QoS de servidores y de clientes, la respectiva ponderación y el resultado.

Tabla 47. Evaluación de un SWA de 5MB.

Atributo	Descripción	Ponderación	Resultado
SWA	SOAP_IMAGEN_5MB	1.00	0.60
Protocolo	HTTPS	0.80	0.32
Total Cliente		0.05	0.046
Fault Tolerance	SI	0.05	0.05
Singleton	SI	0.15	0.15
Contenerización	2	0.10	0.10
JHS	2 GB	0.20	0.12
Total Servidor		0.95	0.39
Total:			0.445

Aplicando la fórmula de Contenerización:

$$\text{Contenerización} = JHS * \text{Min} \left(1, \left[\frac{K * \sum_{i=1}^n \text{Tamano JHS}}{\text{Tamano SWA}} \right] \right)$$

Sustituyendo los valores:

$$\text{Contenerización} = 0.1 * \text{Min} \left(1, \left[\frac{1.5 * 3 * 2}{5} \right] \right)$$

$$\text{Contenerización} = 0.1 * \text{Min} \left(1, \left[\frac{9}{5} \right] \right)$$

$$\text{Contenerización} = 0.1 * \text{Min} (1, 1.8)$$

$$\text{Contenerización} = 0.1 * 1$$

$$\text{Contenerización} = 0.1$$

Aplicando la fórmula de JHS:

$$\text{Tamano JHS} = JHS * \text{Min} \left[1, \left(\frac{k * (\text{Tamano JHS})}{\text{Tamano SWA}} \right) \right]$$

$$\text{Tamano JHS} = 0.2 * \text{Min} \left[1, \left(\frac{1.5 * 2}{5} \right) \right]$$

$$\text{Tamano JHS} = 0.2 * \text{Min} [1, 0.6]$$

$$Tamano\ JHS = 0.2 * 0.6$$

$$Tamano\ JHS = 0.12$$

Calculando la entregabilidad del Servidor:

$$ES = FT + SC + G1FR + G1DD + JSP + JHS$$

Sustituyendo los valores de la fórmula:

$$ES = 0.05 + 0.0 + 0.0 + 0.0 + 0.15 + 0.12 + 0.1$$

$$ES = 0.42$$

Calculando la entregabilidad del Cliente:

$$EC = (TSPA * pt) + (PCSW * pp)$$

Sustituyendo los valores de la fórmula:

$$EC = (1 * 0.6) + (0.8 * 0.4)$$

$$EC = 0.6 + 0.32$$

$$EC = 0.92$$

Calculando la entregabilidad global:

$$E = (ES * ps) + (EC * pc)$$

Sustituyendo los valores de la fórmula:

$$E = (0.42 * 0.95) + (0.92 * 0.05)$$

$$E = 0.39 + 0.046$$

$$E = 0.445$$

5.6 Discusión de Resultados

Los resultados obtenidos a partir de la métrica de entregabilidad permiten identificar los atributos de servidores y clientes que influyen en la capacidad de entrega de SWA. En particular, la configuración del algoritmo G1 First del **Garbage Collector** de Java entregó uno de los mejores resultados en las pruebas realizadas con una arquitectura standalone y en una arquitectura compuesta, la incorporación de Server Cache mostró mejores resultados de entregabilidad de SWA.

La inclusión de aplicaciones clústerizadas y contenerizadas permiten mantener resultados óptimos de entregabilidad de SWA. Estas permiten distribuir la carga de las solicitudes entre diferentes aplicaciones desplegadas concurrentemente. Incluir mecanismos como la tolerancia a fallas beneficia considerablemente la capacidad de entrega de SWA.

Asimismo, los ajustes realizados en los parámetros de configuración de JHS, particularmente mediante el uso del algoritmo G1 First, influyeron positivamente en la entregabilidad de SWA. Específicamente, los parámetros Ratio y **deduplication** reducen la degradación del espacio en JHS, lo que permite el incremento de la tasa de éxito. Por otro lado, los resultados de las pruebas de entrega de SWA con los parámetros habilitados G1 First Pauses y G1 First Region habilitados no indican que haya evidencia suficiente que mejore la calidad de entrega.

Los resultados muestran que la implementación del patrón de diseño **Singleton** en SWA contribuye a un comportamiento más estable del sistema durante las pruebas de entregabilidad. La creación de una única instancia del SWA, se reduce el consumo de recursos y evita la creación redundante de objetos bajo escenarios de alta concurrencia.

En conjunto, estos resultados indican que la entregabilidad no depende de un único elemento, sino de la combinación de atributos tanto del servidor como del cliente. Entre los que se incluyen los protocolos, las aplicaciones, los métodos de comunicación, los parámetros de configuración, así como el tamaño, la arquitectura y el diseño de los SWA. La correcta selección y configuración de estos atributos es importante para garantizar la calidad de la entrega de SWA en cualquier tipo de circunstancias.

5.7 Hallazgos

Los resultados obtenidos indican que el tamaño de los RA contenidos en el SWA es un factor crítico que debe ser considerado por los creadores de los SWA, ya que representa el elemento que mayor degradación presenta en la capacidad de entrega de los SWA. Los resultados muestran que, conforme es mayor tamaño del recurso, se presenta una degradación más significativa en las métricas de entregabilidad, lo que afecta directamente el desempeño de la capacidad de entrega bajo condiciones de alta demanda. A partir de SWA de 5MB, la relación entre el tamaño y la tasa de éxito indica que entre 5 y 7 solicitudes no son respondidas con éxito por cada MB que incrementa el SWA en arquitecturas standalone y sin optimización de la memoria de la JHS, por cada 100 solicitudes realizadas. Por ejemplo con SOAP, REST, HTTP-HTTPS, SOAP MTOM, Singleton, etcétera.

Respecto al uso del atributo **REST Cache**, los resultados muestran que la respuesta es enviada inicialmente al cliente y posteriormente almacenada en la memoria Caché. En este proceso, el servidor continúa procesando tanto la solicitud como la respuesta del SWA, por lo que este mecanismo no aporta beneficios directos en términos de entregabilidad y, bajo las condiciones evaluadas, no representa una característica determinante para la entrega de los SWA. Es importante mencionar que en escenarios de alta concurrencia se puede provocar una saturación en el canal de comunicación o como se llama comúnmente a estos, cuellos de botella.

Por otro lado, las arquitecturas computacionales de replicación de datos, incluyendo estrategias de tolerancia a fallos, la contenerización y la clusterización, contribuyen de manera significativa a mejorar la capacidad de entrega de los SWA. Estas configuraciones permiten una distribución más eficiente de la carga de trabajo en respuestas a solicitudes, así como un comportamiento más estable en aplicaciones de SWA ante escenarios de alta demanda. No obstante, la clusterización del servidor de aplicaciones requiere un monitoreo continuo, ya que la caída de un nodo demanda la necesidad de reiniciar el clúster completo para restablecer su correcto funcionamiento. En contraste, la contenerización de aplicaciones se presenta como una alternativa más flexible y eficiente para el despliegue de SWA, aunque bajo ciertas configuraciones la clusterización puede implicar un menor consumo de recursos computacionales.

En cuanto a los mecanismos de entrega de contenido de SW SOAP, el uso de SOAP MTOM requiere de un cliente capaz de procesar el flujo completo de la respuesta del SWA, para aprovechar adecuadamente los beneficios durante la entrega del servicio que este método ofrece.

La elección de un algoritmo de GC de la JHS que permita mejorar la calidad de entrega de SWA. En el caso del algoritmo G1 First con la configuración de parámetros como G1 First deduplication y G1 First Ratio, estos permiten manejar de manera más eficiente los objetos dentro de la JHS. El primero evita que las cadenas String no se dupliquen dentro de la JHS y el segundo parámetro genera un espacio más amplio para crear más objetos dentro de la JHS.

El objetivo de la implementación de SWA con el diseño *Singleton* es el de generar una única instancia dentro de las aplicaciones Java. Sin embargo, en pruebas realizadas con SWA de 15MB se encontraron problemas en el manejo de una única instancia y provocó el desbordamiento de memoria de JHS, por lo que es recomendable incluir otras estrategias para alcanzar el cien por ciento en la entrega de SWA.

Finalmente, la implementación de Server Cache incrementa la capacidad de entrega de los SWA; sin embargo, la eficiencia de este depende del tamaño del SWA. En las pruebas realizadas con SWA de 10 MB se observaron resultados favorables, mientras que en los SWA de 15 MB el impacto fue más limitado, lo que evidencia la necesidad de incluir otras alternativas que se presentan en este modelo de calidad de entrega de SWA.

Capitulo VI.

Conclusiones y Trabajos Futuros

6.1 Conclusiones y aportaciones

A partir del objetivo planteado se puede definir las siguientes conclusiones:

Existen atributos de los servidores y de los clientes que permiten mejorar las arquitecturas de servicios y mejorar la calidad de entrega de los SWA.

Los atributos del modelo de entregabilidad de SWA que aquí se mencionan permiten dar valores de referencia de la capacidad de entrega de SWA a los usuarios, a los creadores de contenido educativo y a los encargados de las plataformas de *E-Learning*.

A pesar de que las pruebas se realizaron en un ambiente local, se puede garantizar que los resultados son representativos y consistentes, además de que estas pruebas se pueden replicar sin ningún inconveniente. Las condiciones de las pruebas en un ambiente semi-controlado permitió observar el comportamiento y las variaciones del desempeño para analizar los resultados de los atributos considerados.

Utilizar los atributos del modelo de calidad de entrega de SWA que se presenta, permite que se beneficien otras áreas o dominios, como por ejemplo, geografía, geología, medicina o salud, astronomía, multimedia y plataformas de distribución digital, páginas web, los sitios web y las aplicaciones donde el contenido estático está separado del contenido dinámico, o donde se necesita gestionar contenido como documentos.

Los atributos del modelo de calidad de entrega de SWA son un conjunto de atributos heterogéneos, por lo que se probaron múltiples casos de prueba y en cada uno de estos se obtuvieron resultados distintos en comparación con otros casos de prueba. Además, se utilizaron RA de distinto tipo y de distinto tamaño, cada SWA se evaluó con su correspondiente.

El estudio de granularidad da referencia de los valores adecuados para el tamaño de los RA que deben definirse como SWA. Entre estos se encuentran para SWA de tipo texto, audio, imagen y video.

El modelo de QoS que se propondrá permite identificar los mejores atributos de calidad y da respuesta al usuario y a los responsables de las plataformas de *E-Learning*, información de la capacidad de entrega de los SWA. Este modelo consiste en dos partes: el modelo de atributos del lado del servidor y el modelo de atributos del lado del cliente.

A partir de los atributos encontrados, se definió un plan experimental y el conjunto de escenarios de pruebas.

Se realizó el análisis estadístico de los resultados, para dos muestras independientes y para más de dos muestras independientes. A partir de este análisis se determinó que no todos los atributos que se probaron mejoran la calidad de entrega de SWA.

Conclusiones y Trabajos Futuros

Posterior al análisis estadístico, se definió el modelo de QoS de entregabilidad de SWA. Este modelo consta de dos partes, del lado del servidor y del lado del cliente, como se mencionó anteriormente.

Las pruebas se llevaron a cabo en servidores locales, por lo que no se consideraron atributos como la distancia, latencia o disponibilidad de los SWA.

Los SWA se implementaron en el lenguaje de programación Java, en las pruebas de SWA no se implementaron estrategias de SW, como composición, orquestación o coreografía de los SW.

El estudio de granularidad de los RA permite identificar datos importantes como el tamaño o volumen computacional. A partir de esta información, el RA contenido en el SWA se convierte en un componente de software más reutilizable. Por otro lado, no se consideraron los elementos internos ni los pedagógicos de los recursos de aprendizaje para la evaluación de la entrega de SWA.

6.2 Trabajos Futuros

Algunos de los trabajos que quedan por abordar en el futuro para el modelo de entregabilidad de SWA son los siguientes:

- ◆ Métodos de pronóstico de entrega con algoritmos e inteligencia artificial.
- ◆ Crear una estrategia para el enrutamiento de paquetes de red.
- ◆ Implementación de SWA en la arquitectura de microservicios.
- ◆ Definir una arquitectura Cloud, Fog o Edge, que permita acercar los SWA a los usuarios finales.
- ◆ Definir un modelo de JHS correcto para la implementación de SWA.
- ◆ Definir un modelo adecuado de Server Caché para entregar SWA.
- ◆ Implementación de un cliente con SWA basados en SOAP que implemente MTOM.
- ◆ Perfilamiento de los usuarios de las plataformas de ***E-Learning*** que sea analizado mediante información de acuerdo a la edad, sexo, estado civil, grado académico, área de estudio, etc.
- ◆ Definir una métrica con factor de penalización de acuerdo al tamaño del SWA.
- ◆ Definir un cliente para la utilización de SOAP MTOM para entregar SWA.

Referencias

- Abawajy, J., Fudzee, M. F., Hassan, M. M., & Alrubaian, M. (2015). Service level agreement management framework for utility-oriented computing platforms. *Journal of Supercomputing*, 71(11), 4287–4303. <https://doi.org/10.1007/s11227-015-1526-5>
- Abdaldhem, A., Fuhrer, P., & Pasquier, J. (2009). *Web Services Technologies: State of the Art Definitions, Standards, Case Study*.
- Abdel-Basset, M., Mohamed, M., & Chang, V. (2018). NMCD: A framework for evaluating cloud computing services. *Future Generation Computer Systems*, 86, 12–29. <https://doi.org/10.1016/j.future.2018.03.014>
- Abosaif, A. N., & Hamza, H. S. (2020). Quality of service-aware service selection algorithms for the internet of things environment: A review paper. *Array*, 8, 100041. <https://doi.org/10.1016/j.array.2020.100041>
- Aghdaie, N., & Tamir, Y. (2009). CoRAL: A transparent fault-tolerant web service. *Journal of Systems and Software*, 82(1), 131–143. <https://doi.org/10.1016/j.jss.2008.06.036>
- Al-Fraihat, D., Joy, M., Masa'deh, R., & Sinclair, J. (2020). Evaluating E-learning systems success: An empirical study. *Computers in Human Behavior*, 102, 67–86. <https://doi.org/10.1016/j.chb.2019.08.004>
- Al-Humaimedy, A. S., & Fernández, M. (2015). Enabling Synchronous and Asynchronous Communications in CSP for SOC. *Electronic Notes in Theoretical Computer Science*, 312, 69–88. <https://doi.org/10.1016/j.entcs.2015.04.005>
- Ardagna, D., Casale, G., Ciavotta, M., Pérez, J. F., & Wang, W. (2014). Quality-of-service in cloud computing: modeling techniques and their applications. In *Journal of Internet Services and Applications* (Vol. 5). <http://www.jisajournal.com/content/5/1/11>
- Baird, D. (1991). *Experimentación. Una introducción a la teoría de mediciones y al diseño de experimentos* (Prentice Hall, Ed.; 2nd ed.). Prentice Hall.
- Bakri, H., Allison, C., Miller, A., & Oliver, I. (2015). HTTP/2 and QUIC for virtual worlds and the 3D web? *Procedia Computer Science*, 56(1), 242–251. <https://doi.org/10.1016/j.procs.2015.07.204>
- Barros, H., Silva, A., Costa, E., Bittencourt, I. I., Holanda, O., & Sales, L. (2011). Steps, techniques, and technologies for the development of intelligent applications based on Semantic Web Services: A case study in e-learning systems. *Engineering Applications of Artificial Intelligence*, 24(8), 1355–1367. <https://doi.org/10.1016/j.engappai.2011.05.007>
- Barton, J. J., Thatte, S., & Frystyk Nielsen, H. (2000, December 11). *SOAP Messages with Attachments*. <https://www.w3.org/TR/SOAP-attachments/#MultipartRelated>
- Begin, T., Busson, A., Lassous, I. G., & Boukerche, A. (2020). Delivering Video-on-Demand services with IEEE 802.11p to major non-urban roads: A stochastic performance analysis. *Computer Networks*, 182. <https://doi.org/10.1016/j.comnet.2020.107440>
- Belshe, M., & Peon, R. (2015). *Hypertext Transfer Protocol Version 2 (HTTP/2)*. <https://doi.org/10.17487/RFC7540>

- Benítez Domínguez, S. (2018). *Evaluación de Capacidades SOAP y REST para Entregar Servicios Web de Aprendizaje* [Tesis de Maestría]. Tecnológico Nacional de México/CENIDET.
- Bloebaum, T. H., & Johnsen, F. T. (2015). Exploring SOAP and REST communication on the Android platform. *MILCOM 2015 - 2015 IEEE Military Communications Conference*, 599–604. <https://doi.org/10.1109/MILCOM.2015.7357509>
- Buccafurri, F., De Meo, P., Lax, G., Ursino, D., Fugini, M., Modafferi, S., Pernici, B., Di Milano, P., Furnari, R., Goy, A., Lops, P., Redavid, D., & Semeraro, G. (n.d.). *A Framework for Using Web Services to Enhance QoS for Content Delivery*. <http://www>.
- Buccafurri, F., De Meo, P., Lax, G., Ursino, D., Fugini, M., Modafferi, S., Pernici, B., Furnari, R., Goy, A., Lops, P., Redavid, D., & Semeraro, G. (2009). A framework for using web services to enhance QoS for content delivery. *IEEE Multimedia*, 16(1), 26–35. <https://doi.org/10.1109/MMUL.2009.1>
- Calvo, J., Gracia, J., & Bayo, E. (2017). Robust design to optimize client–server bi-directional communication for structural analysis web applications or services. *Advances in Engineering Software*, 112, 136–146. <https://doi.org/10.1016/j.advengsoft.2017.04.010>
- Castiglione, A., Pizzolante, R., De Santis, A., Carpentieri, B., Castiglione, A., & Palmieri, F. (2015). Cloud-based adaptive compression and secure management services for 3D healthcare data. *Future Generation Computer Systems*, 43–44, 120–134. <https://doi.org/10.1016/j.future.2014.07.001>
- Chen, X., Zheng, Z., Yu, Q., & Lyu, M. R. (2014). Web Service Recommendation via Exploiting Location and QoS Information. *IEEE Transactions on Parallel and Distributed Systems*, 25(7), 1913–1924. <https://doi.org/10.1109/TPDS.2013.308>
- Chen, Z., Sun, Y., You, D., Li, F., & Shen, L. (2020). An accurate and efficient web service QoS prediction model with wide-range awareness. *Future Generation Computer Systems*, 109, 275–292. <https://doi.org/10.1016/j.future.2020.03.062>
- Chinnici, R., Moreau, J.-J., Ryman, A., & Sanjiva, W. (2007). *Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language*. <https://www.w3.org/TR/wsdl/>
- Clement, L., Hately, A., von Riegen, C., & Rogers, T. (2004, October 19). *UDDI Spec TC*. <http://www.uddi.org/pubs/uddi-v3.0.2-20041019.htm>
- Debu, P., Reza, R., & Lane, D. (2007). EJB3 in Action. In *EJB3 in Action* (1st ed., Vol. 1).
- Delgado Fernández, P. (2012). *Servicios Web de Aprendizaje* (Tecnológico Nacional de México/Cenidet, Ed.) [Tesis de Maestría]. Tecnológico Nacional de México/CENIDET.
- Di Sorte, D., Femminella, M., & Reali, G. (2009). QoS-enabled multicast for delivering live events in a Digital Cinema scenario. *Journal of Network and Computer Applications*, 32(1), 314–344. <https://doi.org/10.1016/j.jnca.2008.02.004>
- Eisa, M., Younas, M., Basu, K., & Awan, I. (2020). Modelling and Simulation of QoS-Aware Service Selection in Cloud Computing. *Simulation Modelling Practice and Theory*, 103. <https://doi.org/10.1016/j.simpat.2020.102108>
- Erl, T. (2008). *SOA: Principles of Service Design* (Prentice Hall, Ed.; First ed.).
- Escobar Megchún, J. A. (2017). *Generador de Servicios Web de Aprendizaje Compuestos a partir de Recursos Educativos*. TecNM/Cenidet.
- Ewing, J. M., & Menascé, D. A. (2009). *Business-Oriented Autonomic Load Balancing for Multitiered Web Sites*.

- Fang Fang, C., & Chien Sing, L. (2009). Collaborative learning using service-oriented architecture: A framework design. *Knowledge-Based Systems*, 22(4), 271–274. <https://doi.org/10.1016/j.knosys.2009.01.003>
- Farsandaj, K., & Ding, C. (2012). Scatter/Gather browsing of web service QoS data. *Future Generation Computer Systems*, 28(7), 1145–1154. <https://doi.org/10.1016/j.future.2011.08.020>
- Fayos-Jordan, R., Felici-Castell, S., Segura-Garcia, J., Lopez-Ballester, J., & Cobos, M. (2020). Performance comparison of container orchestration platforms with low cost devices in the fog, assisting Internet of Things applications. *Journal of Network and Computer Applications*, 169. <https://doi.org/10.1016/j.jnca.2020.102788>
- Feiler, P. H., Gluch, D. P., & Hudak, J. J. (2006). *The Architecture Analysis & Design Language (AADL): An Introduction*. February, CMU/SEI-2006-TN-011. <http://www.sei.cmu.edu/library/abstracts/reports/06tn011.cfm>
- Fenton, N., & Bieman, J. (2014). *Software Metrics: A Rigorous and Practical Approach* (CRC Press, Ed.; 3rd ed.). CRC Press. <https://doi.org/10.1201/b17461>
- Fenton, N., & Pfleeger, S. L. (1997). *Software Metrics: A Rigorous and Practical Approach*. (2nd ed.). PWS Publishing Co.
- Fielding, R. T. (2000). *Architectural styles and the design of network based software architecture*.
- Fielding, R. T., & Reschke, J. (2014). *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing*. <https://tools.ietf.org/html/rfc7230>
- Fudzee, M. F. M., Abawajy, J., & Deris, M. M. (2010). Multi-criteria Content Adaptation Service Selection Broker. *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, 721–726. <https://doi.org/10.1109/CCGRID.2010.128>
- Fudzee, M. F. M., Mohamed, J., Abawajy, J., Kasim, S., & Ismail, M. N. (2014). An SLA Evaluator for Multimedia Content Adaptation Services. *2014 International Conference on Information Science & Applications (ICISA)*, 1–4. <https://doi.org/10.1109/ICISA.2014.6847361>
- Fuentes Chab, I. H. (2019). *Gestión de Recursos de Aprendizaje para Generar Servicios Web de Aprendizaje*. Tecnológico Nacional de México/CENIDET.
- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns Addison-Wesley Professional* (A.-W. Professional, Ed.). Addison-Wesley Professional.
- Gorski, T., & Wozniak, A. P. (2021). Optimization of Business Process Execution in Services Architecture: A Systematic Literature Review. In *IEEE Access* (Vol. 9, pp. 111833–111852). Institute of Electrical and Electronics Engineers Inc. <https://doi.org/10.1109/ACCESS.2021.3102668>
- Guerron, X., Abrahao, S., Insfran, E., Fernandez-Diego, M., & Gonzalez-Ladron-De-Guevara, F. (2020). A Taxonomy of Quality Metrics for Cloud Services. *IEEE Access*, 8, 131461–131498. <https://doi.org/10.1109/ACCESS.2020.3009079>
- Guha Roy, D., Das, P., De, D., & Buyya, R. (2019). QoS-aware secure transaction framework for internet of things using blockchain mechanism. *Journal of Network and Computer Applications*, 144, 59–78. <https://doi.org/10.1016/j.jnca.2019.06.014>
- Hall, A. (2009). *Thanks for the memory, Linux Understanding how the JVM uses native memory on Windows and Linux*.
- Haroune-Belkacem, N., Semchedine, F., Al-Shammari, A., & Aissani, D. (2019). SMCA: An efficient SOAP messages compression and aggregation technique for improving web

- services performance. *Journal of Parallel and Distributed Computing*, 133, 149–158. <https://doi.org/10.1016/j.jpdc.2019.07.001>
- He, H., Cui, L., Zhou, F., & Wang, D. (2017). Distributed proxy cache technology based on autonomic computing in smart cities. *Future Generation Computer Systems*, 76, 370–383. <https://doi.org/10.1016/j.future.2016.03.015>
- Hunga, M. H., Wu, S. W., Wang, T. L., Cheng, F. T., & Feng, Y. Y. (2010). An efficient data exchange scheme for semiconductor engineering chain management system. *Robotics and Computer-Integrated Manufacturing*, 26(5), 507–516. <https://doi.org/10.1016/j.rcim.2010.03.014>
- Ibrahim, A., & Surya, R. A. (2019). The Implementation of Simple Additive Weighting (SAW) Method in Decision Support System for the Best School Selection in Jambi. *Journal of Physics: Conference Series*, 1338(1). <https://doi.org/10.1088/1742-6596/1338/1/012054>
- ISO/IEC. (2000). *INTERNATIONAL STANDARD ISO/IEC FDIS 9126-1: Quality model*. www.iso.ch
- ISO/IEC. (2014). *ISO/IEC 2501n-División de Modelo de Calidad*.
- Jae, Y. L., Moon, K. K., La, H. J., & Kim, S. D. (2012). A software framework for enabling smart services. *2012 Fifth IEEE International Conference on Service-Oriented Computing and Applications (SOCA)*, 1–8. <https://doi.org/10.1109/SOCA.2012.6449443>
- Kamel, A., Al-Fuqaha, A., & Guizani, M. (2015). Exploiting Client-Side Collected Measurements to Perform QoS Assessment of IaaS. *IEEE Transactions on Mobile Computing*, 14(9), 1876–1887. <https://doi.org/10.1109/TMC.2014.2370648>
- Kocabicak, U., & Dural, D. (2012). Secure and Interoperable e-Learning Platforms Based on Web Services. *Procedia - Social and Behavioral Sciences*, 55, 1265–1271. <https://doi.org/10.1016/j.sbspro.2012.09.624>
- Kropp, A., & Torre, R. (2020). Docker: containerize your application. In *Computing in Communication Networks* (pp. 231–244). Elsevier. <https://doi.org/10.1016/B978-0-12-820488-7.00026-8>
- Kumar, N., Malik, P., Bhardwaj, S., & Jain, S. C. (2016). Comparative analysis of deduplication techniques for enhancing storage space. *2016 4th International Conference on Parallel, Distributed and Grid Computing, PDGC 2016*, 480–487. <https://doi.org/10.1109/PDGC.2016.7913243>
- Kumar, P., & Dezfouli, B. (2019). Implementation and analysis of QUIC for MQTT. *Computer Networks*, 150, 28–45. <https://doi.org/10.1016/j.comnet.2018.12.012>
- Kumari, S., & Rath, S. K. (2015). Performance comparison of SOAP and REST based Web Services for Enterprise Application Integration. *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 1656–1660. <https://doi.org/10.1109/ICACCI.2015.7275851>
- Li, Y. Z., Liu, F., Lei, Z. M., & Chen, L. (2010). User-perceived e-mail service QoS parameters and measurement. *Journal of China Universities of Posts and Telecommunications*, 17(3), 85–90. [https://doi.org/10.1016/S1005-8885\(09\)60469-6](https://doi.org/10.1016/S1005-8885(09)60469-6)
- Liyanaarachchi, A., & Weerawarana, S. (2012, December). An end-to-end caching protocol for web services. *International Conference on Advances in ICT for Emerging Regions (ICTer2012)*. <https://doi.org/10.1109/ICTer.2012.6421410>

- López, P., & Baydal, E. (2017). On a course on computer cluster configuration and administration. *Journal of Parallel and Distributed Computing*, 105, 127–137. <https://doi.org/10.1016/j.jpdc.2017.01.009>
- Malik, S., & Kim, D.-H. (2017). A comparison of RESTful vs. SOAP web services in actuator networks. *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, 753–755. <https://doi.org/10.1109/ICUFN.2017.7993893>
- Marulanda López, J. E. (2014). *Aseguramiento de la calidad en el diseño del software*.
- Mavridis, I., & Karatza, H. (2019). Combining containers and virtual machines to enhance isolation and extend functionality on cloud computing. *Future Generation Computer Systems*, 94, 674–696. <https://doi.org/10.1016/j.future.2018.12.035>
- Md Fudzee, M. F., & Abawajy, J. H. (2011). QoS-based adaptation service selection broker. *Future Generation Computer Systems*, 27(3), 256–264. <https://doi.org/10.1016/j.future.2010.09.005>
- Montgomery, D. C. (2004). *Diseño y análisis de experimentos* (2da ed.). Limusa Wiley.
- Mulligan, G., & Gracanin, D. (2009). A comparison of SOAP and REST implementations of a service based interaction independence middleware framework. *Proceedings of the 2009 Winter Simulation Conference (WSC)*, 1423–1432. <https://doi.org/10.1109/WSC.2009.5429290>
- Muñoz Arteaga, J., Álvarez R., F. J., & Chan Núñez, M. E. (2007). *Tecnología de objetos de Aprendizaje* (First, Vol. 1).
- Muñoz Arteaga, J., Rodríguez Álvarez, J. F., Osorio Urrutia, B., & Cardona Salas, J. P. (2011). *Objetos de aprendizaje integrados a un sistema de gestión de aprendizaje*.
- Newcomer, E. (2002). *Understanding Web Services: XML, WSDL, SOAP, and UDDI* (D. Chappell, Ed.; 1st ed.). Addison-Wesley Professional.
- Oracle Corp. (2016). *Java Platform, Standard Edition Troubleshooting Guide*. Java SE 10 Documentation. <https://docs.oracle.com/javase/8/docs/technotes/guides/troubleshoot/toc.html>
- Oracle Corp. (2018). *Java Platform, Standard Edition HotSpot Virtual Machine Garbage Collection Tuning Guide*. <https://docs.oracle.com/javase/10/gctuning/garbage-first-garbage-collector.htm#JSGCT-GUID-ED3AB6D3-FD9B-4447-9EDF-983ED2F7A573>
- Oracle Corp. (2024). *HotSpot Virtual Machine Garbage Collection Tuning Guide*. <https://docs.oracle.com/en/java/javase/17/gctuning/index.html>
- Padhy, N., Singh, R. P., & Satapathy, S. C. (2018). Software reusability metrics estimation: Algorithms, models and optimization techniques. *Computers and Electrical Engineering*, 69, 653–668. <https://doi.org/10.1016/j.compeleceng.2017.11.022>
- Palacios, M., García-Fanjul, J., Tuya, J., & Spanoudakis, G. (2015). Coverage-Based Testing for Service Level Agreements. *IEEE Transactions on Services Computing*, 8(2), 299–313. <https://doi.org/10.1109/TSC.2014.2300486>
- Pautasso, C., Zimmermann, O., & Leymann, F. (2008). Restful web services vs. “big” web services. *Proceedings of the 17th International Conference on World Wide Web*, 805–814. <https://doi.org/10.1145/1367497.1367606>
- Plauth, M., Feinbube, L., & Polze, A. (2017). *A Performance Evaluation of Lightweight Approaches to Virtualization* (pp. 34–48). https://doi.org/10.1007/978-3-319-67262-5_3
- Polsani, P. R. (2003). Use and Abuse of Reusable Learning Objects. *Journal of Digital Information*, 3.

- Portillo-Dominguez, A. O., Perry, P., Magoni, D., Wang, M., & Murphy, J. (2016). TRINI: an adaptive load balancing strategy based on garbage collection for clustered Java systems. *Software - Practice and Experience*, 46(12), 1705–1733. <https://doi.org/10.1002/spe.2391>
- Potdar, A. M., Narayan, D. G., Kengond, S., & Mulla, M. M. (2020). Performance Evaluation of Docker Container and Virtual Machine. *Procedia Computer Science*, 171, 1419–1428. <https://doi.org/10.1016/j.procs.2020.04.152>
- Prieto Mendez, M. E., Pech Campos, S. J., & Pérez de la Cruz, A. (2013). *Tecnologías y Aprendizaje, Avances en Iberoamerica* (Vol. 2).
- Ramadan, N., & Mohamed, I. (2016). Impact of Implementing HTTP/2 in Web Services. *International Journal of Computer Applications*, 147(9), 27–32. <https://doi.org/10.5120/ijca2016911182>
- Rojas Soriano, R. (2013). *GUÍA PARA REALIZAR INVESTIGACIONES SOCIALES* (Plaza y Valdes, Ed.; 13th ed.).
- Roquero, P., Magaña, E., Leira, R., & Aracil, J. (2020). Performance evaluation of client-based traffic sniffing for very large populations. *Computer Networks*, 166. <https://doi.org/10.1016/j.comnet.2019.106985>
- Rossi, F., Cardellini, V., Lo Presti, F., & Nardelli, M. (2020). Geo-distributed efficient deployment of containers with Kubernetes. *Computer Communications*, 159, 161–174. <https://doi.org/10.1016/j.comcom.2020.04.061>
- Santos, E. A., McLean, C., Solinas, C., & Hindle, A. (2018). How does docker affect energy consumption? Evaluating workloads in and out of Docker containers. *Journal of Systems and Software*, 146, 14–25. <https://doi.org/10.1016/j.jss.2018.07.077>
- Shang, R., Kauffman, R. J., Huang, J., & Yang, Y. (2020). Client risk informedness in brokered cloud services: An experimental pricing study. *Electronic Commerce Research and Applications*, 39, 100893. <https://doi.org/10.1016/j.elerap.2019.100893>
- Sicilia, M.-A. (2005). *Reusability and reuse of learning objects: myths, realities and possibilities*. <http://www.prolearn-project.org>
- Simonsson, J., Zhang, L., Morin, B., Baudry, B., & Monperrus, M. (2019). *Observability and Chaos Engineering on System Calls for Containerized Applications in Docker*. <https://doi.org/10.1016/j.future.2021.04.001>
- Singh, A. P., & Tomar, P. (2016). Web Service Component Reusability Evaluation: A Fuzzy Multi-Criteria Approach. *International Journal of Information Technology and Computer Science*, 8(1), 40–47. <https://doi.org/10.5815/ijitcs.2016.01.05>
- Song, Y. (2021). Web service reliability prediction based on machine learning. *Computer Standards & Interfaces*, 73, 103466. <https://doi.org/10.1016/j.csi.2020.103466>
- Stan, R. G., Negru, C., & Pop, F. (2019). CloudWave: Content gathering network with flying clouds. *Future Generation Computer Systems*, 98, 474–486. <https://doi.org/10.1016/j.future.2019.03.033>
- The Apache Software Foundation. (2017, April 5). *JAX-RS: Failover*. <https://cxf.apache.org/docs/jax-rs-failover.html>
- The Kubernetes Authors. (2022). *Arquitectura de Kubernetes - Nodos*. <https://kubernetes.io/es/docs/concepts/architecture/nodes/>
- Tibermachine, O., Tibermachine, C., & Cherif, F. (2018). Estimating the reputation of newcomer web services using a regression-Based method. *Journal of Systems and Software*, 145, 112–124. <https://doi.org/10.1016/j.jss.2018.08.026>

- Tihomirovs, J., & Grabis, J. (2016a). Comparison of SOAP and REST Based Web Services Using Software Evaluation Metrics. *Information Technology and Management Science*, 19(1). <https://doi.org/10.1515/itms-2016-0017>
- Tihomirovs, J., & Grabis, J. (2016b). Comparison of SOAP and REST Based Web Services Using Software Evaluation Metrics. *Information Technology and Management Science*, 19(1). <https://doi.org/10.1515/itms-2016-0017>
- Torres Kompen, R., Edirisingha, P., Canaleta, X., Alsina, M., & Monguet, J. M. (2019). Personal learning Environments based on Web 2.0 services in higher education. *Telematics and Informatics*, 38, 194–206. <https://doi.org/10.1016/j.tele.2018.10.003>
- Vossen, G., & Westerkamp, P. (2003). *E-Learning as a Web Service*.
- W3C. (2004). *Web Service Glossary*. <https://www.w3.org/TR/ws-gloss/>
- Whatis.com. (2005). *Footprint*. <https://whatis.techtarget.com/definition/footprint>
- Wu, H., Li, Z., Zhang, H., Yang, C., & Shen, S. (2011). Monitoring and evaluating the quality of Web Map Service resources for optimizing map composition over the internet to support decision making. *Computers and Geosciences*, 37(4), 485–494. <https://doi.org/10.1016/j.cageo.2010.05.026>
- Xu, Z., Zhang, H., & Huang, H. (2019). A Performance-Aware Selection Strategy for Cloud-based Video Services with Micro-Service Architecture. *Proceedings of the ACM Multimedia Asia*, 1–6. <https://doi.org/10.1145/3338533.3366609>
- Yaghoubi, M., & Maroosi, A. (2020). Simulation and modeling of an improved multi-verse optimization algorithm for QoS-aware web service composition with service level agreements in the cloud environments. *Simulation Modelling Practice and Theory*, 103. <https://doi.org/10.1016/j.simpat.2020.102090>
- Zagarese, Q., Canfora, G., Zimeo, E., Alshabani, I., Pellegrino, L., Alshabani, A., & Baude, F. (2015). Improving data-intensive EDA performance with annotation-driven laziness. *Science of Computer Programming*, 97(P2), 266–279. <https://doi.org/10.1016/j.scico.2014.03.007>
- Zela Ruiz, J., & Rubira, C. M. (2016). Quality of Service Conflict during Web Service Monitoring: A Case Study. *Electronic Notes in Theoretical Computer Science*, 321, 113–127. <https://doi.org/10.1016/j.entcs.2016.02.007>
- Zhang, Z., Jiang, X., & Xi, H. (2016). Joint resource allocation and traffic management for cloud video distribution over software-defined networks. *2016 8th IEEE International Conference on Communication Software and Networks (ICCSN)*, 1, 407–411. <https://doi.org/10.1109/ICCSN.2016.7586692>
- Zheng, Z., Wu, X., Zhang, Y., Lyu, M. R., & Wang, J. (2013). QoS ranking prediction for cloud services. *IEEE Transactions on Parallel and Distributed Systems*, 24(6), 1213–1222. <https://doi.org/10.1109/TPDS.2012.285>
- Zheng, Z., Zhang, Y., & Lyu, M. R. (2010a). Distributed QoS evaluation for real-world Web services. *ICWS 2010 - 2010 IEEE 8th International Conference on Web Services*, 83–90. <https://doi.org/10.1109/ICWS.2010.10>
- Zheng, Z., Zhang, Y., & Lyu, M. R. (2010b). Distributed QoS evaluation for real-world Web services. *ICWS 2010 - 2010 IEEE 8th International Conference on Web Services*, 83–90. <https://doi.org/10.1109/ICWS.2010.10>
- Zubrow, D. (2004). Software Quality Requirements and Evaluation , the ISO 25000 Series PSM Technical Working Group. *Defense, February*.

Anexo A.

Descripción de los Trabajos Relacionados

7.1 Modelos de QoS

En esta sección se describen los trabajos relacionados en la categoría de Modelos de Calidad de Servicios.

7.1.1 Docker: containerize your application

El trabajo de [Kropp & Torre, 2020], hace referencia a las ventajas de utilizar las herramientas de Docker para desplegar aplicaciones virtualizadas. Docker es una plataforma para desarrolladores con el objetivo principal de desarrollar, implementar y ejecutar aplicaciones virtualizadas en entornos *sandbox*, llamados contenedores. En este se especifica que un contenedor es una instancia de una imagen, un paquete ejecutable independiente que ejecuta una aplicación, emparejada con su estado. Una imagen incluye los componentes necesarios para ejecutar la aplicación: i) la aplicación en sí, ii) las bibliotecas necesarias, iii) las variables de entorno y iv) los archivos de configuración. Por otro lado, se menciona que Kubernetes es un software de código abierto diseñado para gestionar y organizar aplicaciones contenidas en un clúster.

La Tabla 48 presenta las diferencias entre la implementación de contenedores y máquinas virtuales en ambientes de servicios web.

Tabla 48. Comparación entre Container y virtual machine [Kropp & Torre, 2020].

Características	Máquinas Virtuales	Contenedores
Seguridad	Cada MV ejecuta el OS propio.	Kernel compartido.
Diversidad	Múltiples OS .	El sistema operativo host e invitado deben ser iguales.
Tiempo de arranque	Lento.	Rápido, en ms.
Tamaño	Grande, con dificultades para hacer copy y push .	Ligero.
Migración	Tiempo de migración prolongado, copias pesadas, poco tiempo de inactividad.	Tiempo de migración corto, copia ligera, tiempo de inactividad elevado (la copia previa no es compatible con Docker).
Recursos	Diferentes sabores, recursos compartidos. Pesado.	Posibilidad de proporcionar el número de núcleos al arrancar. Normalmente, pueden coexistir más contenedores que MV en el host .
Aislamiento	Los procesos de VM no pueden verse entre sí.	Los procesos de contenedor tampoco pueden.

7.1.2 Delivering Video-on-Demand services with IEEE 802.11p to major non-urban roads: A stochastic performance analysis

En el trabajo de [Begin et al., 2020], los autores proponen un modelo simple y versátil para calcular el **performance** de un servicio de video a demanda (VoD por sus siglas en inglés) para vehículos conectados a lo largo de una carretera o autopista. El objetivo es proporcionar la ayuda y la información suficiente para dimensionar los recursos de red de una plataforma VoD basada en IEEE 802.11p, y calcular para cualquier vehículo dada su ubicación en la carretera la cantidad esperada de datos descargados, tomando en cuenta las interrupciones por el camino. Se utiliza un enfoque probabilístico para tener en cuenta la dinámica móvil de los vehículos y proporciona estimaciones del **performance** alcanzado de cada vehículo suscriptor, así como de los tiempos de interrupción en la reproducción de su video.

7.1.3 A Taxonomy of Quality Metrics for Cloud Services

En el trabajo de [Guerrón et al., 2020], los autores proponen un modelo taxonómico basado en una revisión sistemática de la literatura, para aumentar el cuerpo de conocimiento sobre la evaluación de los servicios **cloud**. Este modelo proporciona un conjunto de métricas de calidad para los servicios **cloud**, los atributos de calidad que miden las métricas seleccionadas, están alineados con las características de calidad propuestas por la [ISO/IEC, 2014]. El objetivo es identificar sistemáticamente, clasificar taxonómicamente y comparar las métricas de los atributos de calidad existentes en el dominio del cómputo **cloud**.

7.1.4 An accurate and efficient web service QoS prediction model with wide-range awareness

En el trabajo de [Z. Chen et al., 2020], los autores proponen un modelo de predicción llamado Factorización de Matriz Consciente de Rango Amplio (WRAMF por sus siglas en inglés), que aborda la influencia de rango amplio a través de una combinación de información de sesgo y un mapeo de función activa explícitamente. El modelo WRAMF es ventajoso para los modelos existentes basados en Matriz de Factorización (MF), que optimiza el modelo mediante una estrategia de tasa de aprendizaje adaptativa que guía a WRAMF para abordar la solución óptima con precisión y entrena el modelo mediante un algoritmo de descenso de gradiente estocástico paralelo. Se encontró un fenómeno en el que los valores de QoS entre los usuarios del servicio varían en un rango diferente, lo que demuestra la característica de rango amplio de los datos de QoS, y esta característica también ocurre entre los servicios Web. Puede incurrir en un alto sesgo y reducir drásticamente la precisión de los métodos tradicionales de predicción de filtrado colaborativo.

Los valores de QoS no solo pueden ayudar a los usuarios del servicio a realizar una mejor selección de servicios web, sino que también permiten a los proveedores de

servicios mejorar el *performance* de sus servicios web y proporcionar servicios de alta calidad a los usuarios.

7.1.5 Modelling and Simulation of QoS-Aware Service Selection in Cloud Computing

Los autores [Eisa et al., 2020], proponen un modelo de selección de servicios de acuerdo a la evaluación de servicios, utilizando un proceso de análisis jerárquico (AHP por sus siglas en inglés), que es un modelo de toma de decisiones de múltiples criterios estructurado donde el problema se define dentro de una jerarquía, que ayuda a definir la mejor estructura para representar los atributos de QoS. Los enfoques son proveedor de servicio, la perspectiva del usuario y las herramientas de los servicios *cloud*. El modelo de selección es sistemático y representa los atributos de calidad de servicios, los cuales los consumidores pueden utilizar fácilmente y comprender cuando estos sean seleccionados.

El cálculo del método peso aditivo simple (SAW, por sus siglas en inglés), que se ejecuta con AHP, se centra en el peso de cada atributo para cada proveedor. El peso se calcula. El método AHP calcula en detalle el peso de cada atributo y sus atributos secundarios.

La categorización de los atributos de calidad la realizaron de acuerdo a la relación de las características y la entrega de servicios web. La selección de atributos está relacionada con aspectos técnicos, estratégicos y organizacionales, económicos y políticos y legislativos.

7.1.6 Quality of service-aware service selection algorithms for the internet of things environment: A review paper

El trabajo de [Abosaif & Hamza, 2020], presenta un sistema de clasificación para revisar y analizar el estado del arte de los algoritmos de selección (SSA) de IoT existentes entre 2012 y 2020. El objetivo es proporcionar orientación y conocimiento a la comunidad investigadora, e identificar las tendencias con respecto a los diferentes algoritmos utilizados para resolver el problema de selección de servicios. También se identificaron los métodos utilizados para diseñar un entorno de IoT adecuado y la metodología utilizada para implementar la solución propuesta. Por último, se realizó una evaluación de la eficiencia y eficacia de la solución propuesta. En la investigación se determinaron tres puntos:

- ◆ Métodos utilizados para diseñar SSA
- ◆ Implementaciones propuestas para SSA
- ◆ Técnicas utilizadas de evaluación de desempeño de SSA

Para resolver los tres puntos, se clasificaron los SSA propuestos para los sistemas de IoT de acuerdo con tres niveles principales: nivel de diseño de algoritmo, nivel de implementación del algoritmo y nivel de evaluación del *performance* del algoritmo.

También se revisaron y analizaron los SSA actuales en IoT, con los métodos de vanguardia para SSA para el sistema IoT.

7.1.7 Evaluating E-learning systems success: An empirical study.

El trabajo de [Al-Fraihat et al., 2020], se divide en tres partes: un modelo, una revisión sistemática y la evaluación del modelo propuesto. El objetivo del trabajo es realizar una evaluación de los sistemas de aprendizaje para asegurar una entrega exitosa, un uso efectivo y un impacto positivo en los estudiantes. Los autores proponen un modelo integral, que proporciona una imagen holística e identifica diferentes niveles de éxito relacionados con una amplia gama de determinantes del éxito, estos son satisfacción percibida del *E-Learning*, la calidad del sistema técnico, la calidad de la información, la calidad del servicio, la calidad del sistema de apoyo, la calidad del alumno, la calidad del instructor y la utilidad percibida. El modelo del estudio espera que la utilidad percibida del *E-Learning* pueda influir positivamente en tres constructos: satisfacción percibida, uso y beneficios para los estudiantes.

7.1.8 Implementation and analysis of QUIC for MQTT

El trabajo de [Kumar & Dezfouli, 2019], se propone la implementación y los estudios de la integración del protocolo QUIC con la capa de aplicación en donde se ejecutan las aplicaciones basadas en IoT. QUIC es un protocolo de transporte multiplexado, basado en UDP y en espacio de usuario desarrollado por Google. De acuerdo con, alrededor del 7% del tráfico mundial de Internet emplea QUIC. Además, QUIC resuelve los numerosos problemas que enfrentan otros protocolos orientados a la conexión como TCP y SCTP. Específicamente, los problemas abordados son:

- ◆ Reducir la sobrecarga de configuración de la conexión
- ◆ Permitir la multiplexación
- ◆ Eliminar el bloqueo del encabezado de línea
- ◆ Admitir la migración de la conexión y eliminar las conexiones TCP semi abiertas.

La comparación de los resultados muestra que MQTT w/QUIC reduce la sobrecarga de conexión en términos de la cantidad de paquetes intercambiados con el corredor hasta en un 56%. Además, al eliminar las conexiones semi abiertas, MQTT w/QUIC reduce el proceso o el uso de la memoria hasta en un 83% y un 50%, respectivamente. Además, al eliminar el problema de bloqueo de cabecera, la latencia de entrega se reduce hasta en un 55%.

7.1.9 Personal learning Environments based on Web 2.0 services in higher education

En el estudio de [Torres Kompen et al.,2019], se proponen un conjunto de lineamientos para utilizar los ambientes de aprendizaje personal (PLE, por sus siglas en inglés) como herramientas de apoyo al aprendizaje formal, ya sea por parte de los profesores o por los propios alumnos. El objetivo es desarrollar pautas generales para la implementación y uso de ambientes de aprendizaje personal.

Un PLE se refiere a un grupo de tecnologías Web, con varios grados de integración e interacción, que ayudan a los usuarios y estudiantes a gestionar el flujo de información que se relaciona con el proceso de aprendizaje, la creación de conocimiento y el desarrollo de habilidades. Los objetivos del PLE son:

- Ayudar a los estudiantes a incorporar herramientas y servicios Web 2.0 para estudios formales.
- Guiar a los estudiantes en su aprendizaje para utilizar la Web 2.0 para el aprendizaje permanente.
- Preparar a los estudiantes de educación superior para futuros empleos.

7.1.10 Software reusability metrics estimation: Algorithms, models and optimization techniques

En el trabajo de [Padhy et al., 2018], los autores proponen un modelo llamado “Representación de pronóstico de reutilización de software resistente al envejecimiento” en el que se emplea un nuevo esquema de aprendizaje automático o basado en la computación evolutiva. El objetivo es reutilizar los componentes de software utilizando atributos como la confiabilidad, supervivencia, predicción del envejecimiento y estabilidad, y para la excelencia del software como garantía, propósitos y costos computacional y recursos humanos para la reutilización de software.

Para la estimación de la reutilización emplean métricas de software predominantes orientadas a objetos, como las métricas de Chidamber y Kemerer (CK). Dicho modelo de predicción de reutilización eficiente y robusto para productos de servicios web utilizando métricas orientadas a objetos, como la complejidad, la cohesión y el acoplamiento, y que pueden ser útiles para predecir la reutilización en productos de software de servicios web.

7.1.11 A framework for evaluating cloud computing services

En el trabajo de [Abdel-Basset et al., 2018], los autores proponen un modelo para ayudar a estimar la calidad de diferentes servicios cloud y proporcionar un enfoque de

análisis de decisiones multicriterio neutrosófico. El objetivo es resolver el problema de la estimación del performance de servicios y mejorar la calidad de los servicios creando una fuerte competencia entre los proveedores de servicios cloud.

7.1.12 Quality of Service Conflict during Web Service Monitoring: A Case Study

En el trabajo de [Zela Ruiz & Rubira, 2016], se propone la herramienta FlexMonitorWS para identificar el conflicto de calidad entre el performance y la precisión durante el monitoreo de un SW, es decir, cómo la herramienta de monitoreo puede generar conflictos entre atributos de calidad. El monitoreo se puede llevar a cabo en: un servicio web, una aplicación de servidor, un servidor y/o una red. Los atributos de calidad indican lo que se debe monitorear para confirmar lo que se busca, como disponibilidad, performance, confiabilidad, precisión, solidez, estado del hardware, fallas en el archivo de registro y/o QoS de la red.

7.1.13 HTTP/2 and QUIC for Virtual Worlds and the 3D Web?

El trabajo de investigación [Bakri et al., 2015a], los autores explican la funcionalidad proporcionada por los nuevos protocolos web y proporciona un mapeo que propone cómo se pueden explotar las características en beneficio de Mundos Virtuales Multiusuario (MUVW) como parte de convergencia con la Web 3D. HTTP/2 es una mejora importante de HTTP, principalmente motivada por el deseo de reducir el tiempo de carga de la página (PLT) de las páginas web modernas, grandes y complejas.

En la Tabla 49 se indican las evoluciones de HTTP de la versión 1 a la versión 2. En la Tabla 50 se indica un resumen de la comparación del protocolo HTTP con el protocolo QUIC.

Tabla 49. Descripción general de las características principales de HTTP 1.0, 1.1 y HTTP/2.

HTTP 1.0	HTTP 1.1	HTTP/2
Solicitud secuencial ASCII: mensajes de respuesta a través de TCPS	Mensajes secuenciales de solicitud-respuesta ASCII a través de TCPS	Tramas binarias full dúplex sobre TLS/TCP o TCP
	PDU: Mensaje HTTP	PDU: Tramas HTTP/2 (10 Tipos)
Nueva conexión TCP para cada solicitud/respuesta	Conexión TCP persistente	Una conexión TCP persistente por dominio
Los navegadores abren múltiples conexiones TCP paralelas dentro del mismo dominio	Los navegadores abren múltiples conexiones TCP paralelas dentro del mismo dominio	Canalización obligatoria
		Transmisión de multiplexación
	Canalización especificado pero no obligatorio y no implementado.	Dependencias de secuencias dinámicas
		(Re) Priorización de secuencias

Anexo A. Trabajos Relacionados

Almacenamiento en caché, opción de compresión de contenido.	Almacenamiento en caché, opción de compresión de contenido	Almacenamiento en caché, compresión de contenido
		Compresión de encabezado
		Empuje del servidor
		Control de flujo

Tabla 50. Diferencias entre el protocolo QUIC con el protocolo HTTP/2.

Protocolo QUIC	Protocolo HTTP/2
Corre sobre UDP	Corre sobre TCP (puertos 80, 443)
Multiplexación de múltiples solicitudes/respuestas a través de una pseudo conexión UDP por dominio.	Multiplexación de múltiples solicitudes/respuestas a través de una conexión TCP por dominio
Promete resolver el bloqueo del jefe de línea en la capa de transporte (causado por el comportamiento de TCP)	Promete resolver el bloqueo del jefe de línea en la capa de la aplicación (causado por la canalización HTTP 1.1)
En el mejor de los casos (en conexiones repetidas, el cliente puede enviar datos inmediatamente (viajes de ida y vuelta cero)	El mejor escenario posible (1 a 3 viajes de ida y vuelta para el establecimiento de la conexión TCP y/o la conexión TLS)
Reducción de la RT obtenida por las características del protocolo, como la multiplexación en una conexión, etc.	Reducción de RT en comparación con HTTP 1.X obtenida por características como multiplexación en una conexión y Server Push
HTTP/2 o SPDY pueden superponerse a QUIC (es decir, todas las funciones de SPDY son compatibles con QUIC)	HTTP/2 o SPDY pueden superponerse a QUIC o TCP
Corrección de errores de reenvío a nivel de paquete	ARQ de rechazo selectivo de TCP utilizado para la corrección de errores
Característica de migración de conexión	N/A
La seguridad en QUIC es similar a TLS pero con un protocolo de enlace más eficiente	Seguridad proporcionada por TLS subyacente
Control de congestión basado en TCP Cubic	Control de congestión proporcionado por TCP subyacente

7.1.14 Quality-of-service in cloud computing: modeling techniques and their applications

En el estudio de [Ardagna et al., 2014], los autores realizaron una revisión sistemática de la literatura, con el objetivo de proporcionar una visión general de los trabajos de investigación en el espacio de modelado de QoS cloud, categorizando las contribuciones de acuerdo con las áreas relevantes y los métodos utilizados para la gestión de servicios cloud como: modelado de cargas de trabajo, modelado de sistemas y aplicaciones. El propósito es encontrar el balance para la asignación de recursos para mantener los niveles de QoS de acuerdo al SLA, para garantizar un nivel de servicio en dimensiones como desempeño, disponibilidad y confiabilidad.

7.1.15 Quality models for web services: A systematic mapping

En el trabajo de [Oriol et al., 2014], los autores proponen un modelo de calidad para servicios Web basados en algunos estándares como el [ISO/IEC, 2014] y en algunos trabajos de investigación desde el año 2001 al 2012. El objetivo es evaluar el estado actual del estado del arte de los modelos de calidad propuestos para servicios Web.

En este trabajo se hizo una revisión sistemática de la literatura y un mapeo sistemático, en la que se revisaron 47 diferentes modelos de calidad de 65 trabajos de investigación que cumplieron con los criterios de selección. Las búsquedas fueron realizadas en bases de datos *ISI Web of Science (WoS)*, *IEEE Xplore* y *ACM Digital Library*. Las palabras clave utilizadas fueron “Web service” y “*Quality models*”. Los filtros aplicados fueron seleccionados por título, resumen, lectura rápida (si presenta un modelo o define algún modelo), literatura de organizaciones (consorcios y organizaciones), si fueron base para otros trabajos.

El primer criterio de búsqueda fue el orden cronológico del 2001 al 2010 obteniendo más en el año 2009-2010 con 19 artículos seleccionados, el segundo fue la lista de las características de los modelos propuestos para este caso se tomó en cuenta el tamaño de los nodos y la completitud de la definición de los términos, el tercer criterio fue discernir entre los factores más comunes de los modelos de calidad de los que se encontraron idoneidad funcional, *performance*, compatibilidad, usabilidad, confiabilidad, mantenibilidad, seguridad, portabilidad y características no técnicas, y el cuarto criterio fue definir los factores de calidad más consolidados, en esta etapa el autor selecciono por características y sub características definiéndolas de acuerdo al ISO 25000 de la siguiente manera: idoneidad funcional: precisión; *performance*: accesibilidad, capacidad, tiempo de respuesta y *throughput*; compatibilidad: interoperabilidad; confiabilidad: disponibilidad, tiempo de mantenimiento y robustez; seguridad: autenticación y confidencialidad.

Como conclusiones dejan la información de su trabajo en un proyecto TIN2010-19130-C02-01, además de que se obtuvieron 19 atributos de calidad que aparecen en al menos el 30 % de los modelos de calidad clasificados.

7.1.16 Scatter/Gather browsing of web service QoS data. Future Generation Computer Systems

En el estudio de [Farsandaj & Ding, 2012], los autores proponen utilizar un modelo modificado de exploración *Scatter/Gather*, para implementar la exploración de QoS, ajustado a los datos de QoS y con algunas variaciones del modelo. Este proyecto propone resolver los problemas de selección de servicios con algoritmos de agrupamiento con datos de intervalo QoS, representados como un vector de valores simbólicos, para lo que se hicieron cuatro implementaciones diferentes del modelo de exploración.

Este modelo ayuda a los solicitantes de servicios a obtener algún conocimiento sobre la distribución del valor de QoS de los servicios disponibles, de modo que puedan formular requisitos de QoS con valores realistas. En segundo lugar, la exploración ayuda a los proveedores de servicios a comprender la demanda real de los solicitantes.

7.1.17 Steps, techniques, and technologies for the development of intelligent applications based on Semantic Web Services: A case study in e-learning systems

En el trabajo de [Barros et al., 2011], los autores proponen un marco para la gestión e integración de los procesos involucrados en el uso de Servicios Web Semánticos (SWS) en aplicaciones inteligentes. La investigación sobre la construcción de sistemas basados en SWS indica que los problemas relacionados con el mantenimiento y el performance de sus herramientas y servicios dificultan la adopción de este tipo de aplicación. Los servicios web semánticos prometen la combinación de tecnología de la Web semántica y servicios web. Hereda características de ambos enfoques: interoperabilidad semántica de la Web semántica y dinámica de la disponibilidad de recursos de la tecnología de servicios Web.

La aplicación propuesta Grinv es un Middleware que integra los mecanismos de invocación y descubrimiento de servicios, y permite a los desarrolladores la adaptación de la aplicación a sus necesidades, para evitar algunos de los problemas en la implementación de SWS en sistemas inteligentes.

7.1.18 User-perceived e-mail service QoS parameters and measurement

En el trabajo de investigación de [Li et al., 2010], se propone un grupo de parámetros de calidad de servicio (QoS) del tráfico de correo electrónico (**email**) desde la perspectiva de los usuarios. En la metodología del e-mail participan 4 entidades, el cliente **email**, el servidor **email delivery** (SMTP), el servidor **email retrieval** (POP3) y el cliente **email receiver**. El servidor SMTP contiene una gran cantidad de cuentas de usuarios. La remitente entrega el email editado al servidor SMTP a través del cliente. El servidor SMTP identifica la dirección y la contraseña del destinatario y luego envía el correo electrónico al servidor POP3. El servidor POP3 almacena el correo electrónico en la bandeja de entrada de mensajes del destinatario y luego informa al destinatario de la llegada de nuevos mensajes. El destinatario se conecta al servidor POP3 por parte del cliente y luego recibe el nuevo correo.

Se realizó una encuesta a docenas de usuarios de **email** sobre los factores que afectan la calidad del servicio de **email**, de la cual se obtuvieron los siguientes parámetros: Tiempo de inicio de sesión percibido por el usuario, tiempo de transferencia de correo percibido por el usuario, tiempo total percibido por el usuario y tasa de éxito del servicio.

7.2 Aplicaciones

Los trabajos de esta categoría presentan los trabajos relacionados que presentan aplicaciones enfocadas en mejorar la QoS de los servicios web.

7.2.1 Geo-distributed efficient deployment of containers with Kubernetes

El trabajo de [Rossi et al., 2020], los autores proponen una herramienta llamada Geo-kubes que se basa en la elasticidad de kubernetes y extiende su adaptación con capacidades de red como la ubicación. Esta propuesta está basada en una solución de aprendizaje por refuerzo (RL) basada en modelos, que aprende una política de escala adecuada a partir de la experiencia.

Para implementar la solución, se formularon problemas de optimización y una heurística basada en la red, que tiene en cuenta explícitamente el retardo de red entre los nodos. La solución propuesta se enfoca en la implementación de la elasticidad. Una aplicación elástica puede asignar y desasignar dinámicamente recursos informáticos para manejar en consecuencia su carga de trabajo. La elasticidad puede mejorar la disponibilidad de la aplicación mientras limita el costo de ejecución. El problema de elasticidad determina cómo escalar el número de contenedores (es decir, escala horizontal) y/o la cantidad de recursos informáticos asignados a cada uno de ellos (es decir, escala vertical) frente a variaciones de la carga de trabajo. Para determinar las acciones que se deben de tomar, se necesita una política de implementación, esta política solo maneja las aplicaciones y sus instancias pero no cambia el conjunto de recursos computacionales. Para la implementación de políticas, se implementó el enfoque NP-Hard que es una idea de solución de forma ascendente.

Kubernetes puede replicar contenedores para mejorar el uso de recursos, la distribución de carga y la tolerancia a fallas. Kubernetes tiene una arquitectura de maestro-trabajador. El maestro está a cargo de preservar en tiempo de ejecución el estado deseado del clúster mediante la administración de recursos (usando trabajadores) y orquestando aplicaciones (usando pods). Un nodo trabajador es una máquina física o virtual que ofrece su capacidad computacional para ejecutar pods de manera distribuida. Un Pod es la unidad de implementación más pequeña de Kubernetes, consiste en un solo contenedor o en un número reducido de contenedores fuertemente acoplados. Kubernetes permite configurar cada contenedor de un Pod con solicitudes y límites de recursos específicos. Una solicitud de recursos es la cantidad mínima de recursos (CPU y/o memoria) que necesita el contenedor, un límite de recursos es la cantidad máxima de recursos que se pueden asignar al contenedor. Kubernetes puede ejecutar varias instancias (o réplicas) de un Pod usando un ReplicaSet. Un ReplicaSet asegura que una determinada cantidad de pods estén activos y en ejecución para proporcionar un servicio específico.

El modelo formulado (ILP) está basado en NP-Hard, el cual minimiza el tiempo de adaptación y el costo de virtualización. Para su solución se propuso el algoritmo enfocado en red que resuelve una variante del problema de empaque en contenedores, teniendo en cuenta el tiempo de implementación de los pods, los recursos computacionales y el tiempo de demora de la activación de los nodos trabajadores.

7.2.2 Client risk informedness in brokered cloud services: An experimental pricing study

En la investigación de [Shang et al., 2020], propone un mecanismo para fijar el precio económico de los servicios de computación **cloud** con intermediación que incluye la finalización del trabajo de computación garantizada y precios basados en el riesgo. Los resultados sugieren que tal información sobre el riesgo juega un papel importante al influir en la disposición a pagar de los clientes por los servicios de intermediación.

Para el desarrollo del modelo de este trabajo, se definieron tres variables: la información sobre los riesgos del cliente, la propensión al riesgo del cliente y la duración del trabajo, así como su interacción, sobre la disposición de los clientes a pagar por servicios **cloud** con intermediarios.

Se desarrolló un mercado simplificado de servicios **cloud** que involucra a un corredor **cloud** y sus clientes. El agente **cloud** aprovecha los recursos reservados y al contado, para brindar servicios personalizados con finalización del trabajo garantizada. Implementaron un mecanismo de precios en un banco de pruebas experimental en línea. El corredor **cloud** descubre la disposición de los clientes a pagar por sus servicios a través de un algoritmo de mejor oferta. Mientras tanto, los clientes necesitan adquirir servicios de cómputo **cloud** para diferentes trabajos. Los precios de los servicios a pedido y los servicios al contado de los proveedores de **cloud** pública se ponen a disposición de los clientes para su referencia para que puedan hacer ofertas al corredor **cloud**. Los servicios a pedido garantizan la finalización del trabajo informático a un precio más alto, y los servicios de precio al contado se ofrecen a un precio mucho más bajo, pero están sujetos a interrupciones.

7.2.3 SMCA: An efficient SOAP messages compression and aggregation technique for improving web services performance

En el estudio de [Haroune-Belkacem et al., 2019], se propone una técnica para realizar procesos de agregación y compresión en todos los mensajes XML, reagrupando los datos XML de la misma ruta en un contenedor. El objetivo es resolver el problema del tráfico de red en los servicios web, reducir el número y el tamaño de los mensajes XML en poco tiempo. Esta técnica procesa todos los mensajes que comparten el mismo destino como un solo mensaje, estos mensajes comparten el diccionario de etiquetas, el

diccionario de ruta, el contenedor de estructura y los contenedores de datos. Los diccionarios y contenedores son comprimidos con un compresor de texto. Además, los formatos comprimidos son reagrupados en un solo mensaje agregado y comprimido.

7.2.4 QoS-aware secure transaction framework for internet of things using blockchain mechanism

En el trabajo de [Guha Roy et al., 2019], los autores proponen un servicio **blockchain** basado en seguridad junto con algunos enfoques de cifrado para una comunicación segura para mitigar el problema de la carencia de seguridad, privacidad y la fiabilidad de los sistemas centralizados de la **IoT**, utilizando el cifrado y descifrado mediante una clave secreta basada en hash, además de un modelado de riesgos para prevenir ataques. A través de **blockchain**, los autores proponen el desarrollo de una red paralela segura para redes IoT. Esta se divide en tres secciones principales de UC denominadas red pública, una red **cloud** enriquecida con **blockchain** y una red empresarial o paralela con la que interactúa la aplicación de **blockchain**. La velocidad de transferencia de datos es lenta y este es el mayor problema porque detiene los demás procesos se toma de decisiones por parte del administrador de red. Para resolver esta dificultad, los autores proponen una red paralela para empresas u organizaciones basadas en la clasificación de datos, la cual selecciona la ruta más pequeña con concesión de red nominal para acelerar los procesos.

7.2.5 TRINI: an adaptive load balancing strategy based on garbage collection for clustered Java systems

En el trabajo de [Portillo-Domínguez et al., 2016], los autores proponen una estrategia de balanceo de carga en **clustering** de sistemas en Java. Esta estrategia mejora el performance del sistema evitando los impactos de en el sistema de **Major Garbage Collector** (MaGC por sus siglas en inglés), la cual es una importante causa de degradación en el **performance** de Java.

El objetivo de este trabajo es validar la herramienta TRINI, haciendo una evaluación experimental en términos de generalidad, escalabilidad y confiabilidad. Esta herramienta es un balanceador de carga basada en el **Garbage Collector** (GC por sus siglas en inglés), implementado en un **clúster** Java. Esta herramienta recupera periódicamente información de los nodos de la aplicación, luego identifica la política más adecuada en función de las características de GC de la aplicación que se ejecuta en cada nodo. Finalmente, la política elegida se utiliza para pronosticar los eventos de MaGC y equilibra la carga de trabajo entrante entre los nodos de aplicación disponibles. Está compuesto por un elemento de supervisión para obtener información; un elemento de análisis para evaluar si se requiere alguna adaptación; un elemento para planificar la adaptación, y un elemento para ejecutarla. El quinto se encarga de apoyar a los demás elementos en sus respectivas tareas.

7.2.6 Service level agreement management framework for utility-oriented computing platforms

En el trabajo de investigación de [Abawajy et al., 2015], los autores proponen un esquema de niveles de aceptación de servicio (SLA) que permita adaptar el contenido en una plataforma orientada a servicios, esto permite al proveedor de servicios determinar la calidad del servicio entregable antes de establecer el SLA con los consumidores potenciales del servicio para optimizar las garantías de la calidad del servicio.

Los objetivos del marco de gestión de SLA son (a) formalizar los requisitos, prioridades y especificaciones de Calidad de Servicio esperados; (b) permitir las negociaciones bilaterales entre los proveedores de servicios y los consumidores; (c) implementar el aprovisionamiento de servicios basado en SLA, realizar evaluaciones y monitorear los parámetros de QoS.

7.2.7 Secure and Interoperable e-Learning Platforms Based on Web Services

En el trabajo de [Kocabicak & Dural, 2012], proponen un marco flexible, extensible, configurable, más segura y orientada a la estructura de servicio.

Uno de los problemas al compartir el contenido de aprendizaje para entornos *E-Learning* es la seguridad de los usuarios, en donde se encuentra información confidencial, así como las credenciales de identificación y autorización. Lo que se propone la arquitectura es con un simple acceso poder acceder a diferentes ambientes *E-Learning*, a través de un árbol de bases de datos y un árbol de servicios web de seguridad de autenticación y autorización separados con la finalidad de controlar la información.

El modelo sigue una secuencia, en el que el estudiante es registrado por una universidad, el alumno ingresa a través de usuario y contraseña, esta solicitud transmitida con identificación, tiempo, y validaciones a un *Mediator Service*, el cual agrega un certificado SOAP y re envía a un *Authentication Service*, el cual valida las autorizaciones y es reenviado a *Reporting Service* el cual determina a cuáles bases de datos se puede acceder.

7.2.8 Web Service Recommendation via Exploiting Location and QoS Information

En el trabajo de [X. Chen et al., 2014], se propone una aplicación de recomendación de servicios basada en el filtrado colaborativo (CF), para ayudar a los usuarios a seleccionar servicios con un *performance* óptimo de calidad de servicio (QoS), empleando la información de la ubicación del usuario y los valores de QoS para agrupar a los usuarios y servicios y, hacer recomendaciones de servicio personalizadas para los usuarios en función de los resultados de la agrupación. El objetivo del CF es

predecir y recomendar elementos favoritos potenciales para un usuario en particular empleando datos de clasificación recopilados de otros usuarios.

7.2.9 A software framework for enabling smart services

En el trabajo de [Jae et al., 2012], se propone un marco de software para servicios de vida inteligente conscientes del contexto, *Smart Service Framework* (SSF), el objetivo es integrar la informática móvil y los servicios inteligentes. *Smart Service* es un servicio con características de "inteligencia". A través de los dispositivos móviles se infiere la situación de los usuarios y se suministran servicios específicos de la situación. El objetivo de los servicios inteligentes es proporcionar sus funcionalidades a los usuarios de una manera conveniente para apoyar la vida diaria de los usuarios. Estos proporcionan sus funcionalidades personalizadas al comprender las intenciones de los usuarios. Los servicios inteligentes están disponibles hoy en día en teléfonos móviles en las carreteras, televisores inteligentes en los hogares y dispositivos de atención médica.

7.2.10 Monitoring and evaluating the quality of Web Map Service resources for optimizing map composition over the internet to support decision making

En el estudio de [Wu et al., 2011], los autores proponen un sistema arquitectural para integrar servicios de información geográfica que respalda la selección de calidad. Este modelo incluye un módulo de operación continua para monitorear y evaluar la calidad de los servicios de información geográfica registrados. El resultado de este seguimiento, evaluación y la información histórica asociada se almacenan en un registro. En dicha interfaz el cliente puede consultar la calidad de un servicio en línea, el objetivo es que cualquier cliente pueda vincular el mejor servicio en tiempo de ejecución para obtener el mejor *performance*. Utilizan la estructura de árbol jerárquico de *Quality Factor Model* que es una propuesta de [Hanwu, 2008], el cual puede utilizar sub arboles ya que se pueden recortar y se pueden agregar algunas características ad-hoc para un propósito especial, además de que cada hoja representa una de las características de granularidad más finas de los requisitos del usuario. Se desarrollaron tres sistemas prototipo para: (a) monitorear y evaluar las capas de *Web Map Services* (WMS), (b) recopilar la evaluación subjetiva de la calidad de WMS por parte del usuario, y (c) desarrollar un método factible para fusionar los recursos de WMS adecuado para tomar de decisiones.

7.2.11 An SLA Evaluator for Multimedia Content Adaptation Services

En el trabajo de [Fudzee et al., 2010], los autores proponen un marco de adaptación de contenido orientado a servicios (SOCA, por sus siglas en inglés), es un intermediario de selección de servicios de adaptación de criterios múltiples, que permite a los usuarios finales, seleccionar el mejor servicio entre los candidatos a

servicios de adaptación de contenido disponibles. El objetivo de SOCA es mejorar la experiencia de usuario ofreciendo contenido de valor agregado y también proporcionar un mecanismo flexible y escalable de entrega de contenido basado en servicios. Este marco consta de componentes que brindan acceso a los servidores de contenido, formulan la solicitud del usuario al formato de origen, administran y proporcionan una descripción del contenido.

El problema de la selección de servicio de adaptación de contenido multicriterio (CASS por sus siglas en inglés), es definir cómo se deben asignar las tareas de adaptación (por ejemplo, transcodificación, resumen de video, etc.) junto con los segmentos de contenido a los servicios de adaptación apropiados con el objetivo de lograr la QoS especificada por el usuario, basándose en criterios como el tiempo, costo, disponibilidad y calificación, de acuerdo a la especificación del acuerdo de nivel de servicio (SLAS) y al acuerdo de nivel de servicio. La solución se realiza a través de un algoritmo determinista de pila que consiste en una función de asignación de objetivo único y ávido que se construye sobre un árbol de trayectoria de adaptación. Este algoritmo utiliza dos relaciones (valores máximo y mínimo) para calcular las puntuaciones de los servicios para obtener una única ruta óptima que mejora la ejecución de la selección de servicios.

7.2.12 Collaborative learning using service-oriented architecture: A framework design

El trabajo de [Fang Fang & Chien Sing, 2009], propone un marco para integrar la arquitectura orientada a servicios (SOA) con procesos de aprendizaje colaborativos significativos e interactivos. El objetivo es mejorar la interoperabilidad, flexibilidad y reutilización del contenido de aprendizaje electrónico en un entorno colaborativo. Los alcances se limitan al registro de los servicios.

7.2.13 CoRAL: A transparent fault-tolerant web service

En el trabajo de [Aghdaie & Tamir, 2009], los autores proponen una herramienta llamada **CoRAL** que proporciona alta confiabilidad y disponibilidad a los servicios web. Esta herramienta implementa la tolerancia a fallos incluso para las solicitudes que se procesan en el momento de la falla del servidor.

CoRAL es un esquema de servicio web tolerante a fallas basado en la replicación de conexión y el registro a nivel de aplicación. CoRAL recupera las solicitudes en curso y no requiere servidores deterministas ni cambios en los clientes. El estado de la conexión TCP del lado del servidor se replica activamente en una copia de seguridad en espera, lo que permite una rápida conmutación por error. Este esquema utiliza la replicación activa de los servidores, donde cada solicitud del cliente es procesada por dos o más réplicas de servidor, la respuesta de solo uno de los servidores se envía al cliente. Este enfoque tiene un alto costo de procesamiento porque cada solicitud de cliente se procesa dos veces.

7.3 Evaluaciones

En esta categoría se presentan los trabajos enfocados en evaluar la calidad interna o externa de los servicios web.

7.3.1 Web service reliability prediction based on machine learning

En el trabajo de [Song, 2021], los autores proponen un método de predicción de la confiabilidad del servicio web basado en el aprendizaje automático, considerando el usuario, el servicio web y la condición de la web. La investigación se enfoca en evaluar la confiabilidad del servicio, que es una propiedad no funcional.

La confiabilidad del servicio se define como la probabilidad de que una invocación de servicio se complete con éxito. El valor de confiabilidad se calcula a partir de muestras de invocaciones pasadas como la relación entre el número de invocaciones exitosas y el total de invocaciones realizadas. Para evaluar la confiabilidad desde la perspectiva del usuario se debe tomar en cuenta las condiciones de la web, por ejemplo, la ubicación del usuario, las variaciones de carga del servicio y las condiciones dinámicas de la red. El método de evaluación de la confiabilidad que proponen, considera diferentes parámetros (usuario, servicio web y condición web). Primero, los parámetros del usuario incluyen algunos factores que afectan la confiabilidad del servicio web, como el usuario, la red y el dispositivo, capacidades de límite y perfiles de uso. En segundo lugar, los parámetros del servicio incluyen la ubicación del servicio, la configuración del sistema (por ejemplo, CPU, RAM, disco y operaciones de Entrada y Salida). En tercer lugar, los parámetros de la condición de la red, condiciones del entorno, como el *performance* de la carga de la red.

7.3.2 Simulation and modeling of an improved multi-verse optimization algorithm for QoS-aware web service composition with service level agreements in the cloud environments

En el trabajo de [Yaghoubi & Maroosi, 2020], los autores proponen un algoritmo mejorado de optimización de múltiples universos (IMMVO por sus siglas en inglés), para resolver problemas de optimización numérica. El objetivo es resolver el problema de composición del servicio con respecto a las restricciones del acuerdo de nivel de servicio (SLA) con el proveedor. La composición de servicios permite agregar servicios de un nivel de abstracción menor, o integrar servicios existentes, para el desarrollo de nuevos procesos de negocio. El problema de composición del servicio se considera un problema de optimización NP-Hard. Los atributos que se consideran en la selección de servicios son: confiabilidad, disponibilidad, tiempo de respuesta y costo, para la composición de servicios de acuerdo a los patrones de composición: secuenciales, paralelos, probabilísticos y circulares.

7.3.3 Performance comparison of container orchestration platforms with low-cost devices in the fog, assisting Internet of Things applications

El trabajo de [Fayos-Jordan et al., 2020], realiza una comparación exhaustiva, funcional y de **performance** con redes cableadas e inalámbricas y utilizando clústeres tanto homogéneos como heterogéneos de dispositivos de tablero pequeñas, de tecnologías de cómputo fog en el contexto de la IoT. Para realizar las comparaciones de Plataformas Orquestadoras de Contenedores (COP por sus siglas en inglés), se utilizaron Kubernetes y Docker Swarm, en contenedores Linux con sistema **host** aislado, accediendo a través de una API REST. Las pruebas se realizaron de manera homogénea y heterogénea, en escenarios inalámbricos y cableados, se evaluó el porcentaje de RAM, el porcentaje de CPU y la temperatura de CPU. Kubernetes incluye un conjunto de funcionalidades que podrían ser interesantes para servidores de mayor **performance** que no están incluidos de forma nativa en Docker Swarm, como el auto escalado y la prueba inicial para ponderar los diferentes esclavos con el fin de realizar un equilibrio de carga predeterminado pero a un costo adicional de uso de la CPU.

Los resultados indican que Kubernetes utiliza 14% de RAM más que Docker Swarm, Kubernetes muestra un uso de CPU de 4% más que de Docker Swarm en la ejecución de las mismas tareas, los resultados de temperatura indican que Swarm también se calienta un poco más que Kubernetes por atender más solicitudes que Kubernetes, este último no respondió cerca del 25% de las solicitudes del cliente. Los autores indican que Docker Swarm siempre supera a Kubernetes, tanto en uso de recursos como en tiempo de procesamiento en estos dispositivos SBC y con diferentes topologías.

7.3.4 Performance evaluation of client-based traffic sniffing for very large populations

En el trabajo de [Roquero et al., 2020], los autores proponen realizar la implementación de **microSniffers** para medir la carga de datos de gran volumen pero del lado del cliente. Indican que el monitoreo del flujo de datos del lado del servidor o en la red es un proceso complicado porque genera carga en los procesos de registro de datos. Para este trabajo, las mediciones se realizan implementando sondas de red ligeras que realizan mediciones pasivas en la terminal del cliente, manejando grandes volúmenes de datos y alternativas de reducción de datos que reducen la carga de trabajo del tráfico en un 80 % con una mínima pérdida de información. El objetivo es ofrecer alternativas, proyecciones, ayuda de problemas de responsabilidad y análisis a los proveedores de servicios de internet, analizando la información de los teléfonos inteligentes que se conectan a una red WIFI, móvil o fijo y que cambian a otra red cuando la señal es más fuerte.

El microSniffers está compuesto por dos módulos de software. La primera captura el tráfico y ofrece registros de flujo que se escriben en el disco. El segundo se comunica con el orquestador y envía los registros de flujo a los hundimientos de monitoreo.

7.3.5 Estimating the reputation of newcomer web services using a regression-Based method

En el trabajo de [Tibermachine et al., 2018], los autores proponen un método para estimar la reputación de los servicios web, que es uno de los criterios utilizados para los sistemas de recomendación en ambientes orientados a servicios (WSRS). Básicamente, emplea modelos de regresión, incluido **Support Vector Regression**, en el proceso de estimación de los valores de reputación desconocidos de los recién llegados a partir de sus valores de QoS conocidos.

El objetivo es demostrar la eficiencia del método propuesto estimando la reputación de los servicios nuevos a través de evidencias estadísticas, obtenidas de experimentación por medio de un conjunto de servicios web del mundo real, recolectados a través de WSDream con 5825 con 339 usuarios en diferentes ubicaciones geográficas y de QWS con 365 servicios. El modelo hace un refinamiento y completa la propuesta inicial para estimar los valores de reputación iniciales de los servicios de recién llegados en los WSRS. La estimación correcta de los valores permite mejores recomendaciones y, por lo tanto, una mejor ayuda en la selección de servicios web que satisfagan los requisitos de los clientes

7.3.6 Performance Evaluation of Docker Container and Virtual Machine

El trabajo de [Potdar et al., 2020], realiza una evaluación comparativa entre contenedores Docker y máquinas virtuales, en la cual se utilizan herramientas de medición estándar como Sysbench, Phoronix y Apache, en la que se compara el **performance** de la CPU, el de la memoria, y el de lectura/escritura del almacenamiento, la prueba de carga y la medición de la velocidad de operaciones. En la Tabla 51 se analiza las diferencias de la tecnología de virtualización, aunque docker no es una máquina virtual, muchas veces es referenciada como tal.

Tabla 51. Tabla comparativa entre Docker y Máquinas Virtuales [Potdar et al., 2020].

	Máquinas Virtuales	Contenedores Docker
Proceso de aislamiento	A nivel de Hardware	Sistema Operativo
Sistema operativo	Separado	Compartido
Tiempo de arranque	Largo	Corto
Uso de recursos	Más recursos	Menos recursos
Imágenes prediseñadas	Difícil de encontrar y administrar	Ya disponible para servidor doméstico
Imágenes pre configuradas personalizadas	Difícil de construir	Fácil de construir
Tamaño	Más grande porque contienen todo el sistema operativo debajo	Más pequeño con solo motores de ventana acoplable sobre el sistema operativo host
Movilidad	Fácil de cambiar a un nuevo sistema operativo host	Destruído y recreado en lugar de
Tiempo de creación	Largo	En segundos

En las pruebas realizadas se hacen comparaciones de CPU, **performance** de memoria; lecturas y escrituras de disco, pruebas de velocidad de carga y velocidad de transacciones. Las pruebas de CPU se realizaron con operación de número primo máximo y compresión de archivos de 1 MB hasta 4 GB, en todos los resultados Docker tiene mejores tiempos. Se hizo una prueba de **performance** de memoria con Multiprocesamiento simétrico en las que Docker tiene tres veces mejores resultados que VM. Para las pruebas de lectura y escritura de disco los resultados siguen siendo mejor en Docker, donde se reduce en un 54% las lecturas de disco. En las pruebas de carga, se midió el número de solicitudes por segundo que un sistema determinado puede tolerar. Docker sigue triplicando la mejoría de resultados a comparación con VM. En la medición de velocidad de transacciones, a Docker le toma menor tiempo resolver los problemas.

7.3.7 Combining Containers and Virtual Machines to Enhance Isolation and Extend Functionality on Cloud Computing

El trabajo de [Mavridis & Karatza, 2019], estudia la combinación las máquinas virtuales y de los contenedores, ejecutando contenedores sobre máquinas virtuales (VM). El objetivo es presentar las ventajas de ejecutar contenedores en VM y explorar cómo varias técnicas de visualización y configuraciones afectan el de **performance** de memoria de este método. El propósito es estudiar y proporcionar información sobre el aislamiento, el de **performance** de memoria y el consumo de energía de los contenedores que se ejecutan en la parte superior de las máquinas virtuales. En la Figura 38 se muestran los tres tipos de virtualización: Virtualización completa, Paravirtualización y Virtualización asistida por hardware.

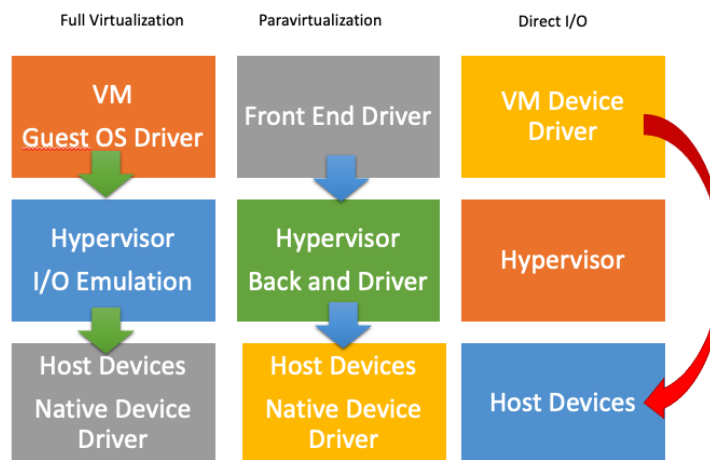


Figura 38. Métodos de virtualización de E/S [Mavridis & Karatza, 2019].

Las pruebas se realizaron con MV KVM, XEN e Hyper-V y para el **performance** se calculó la media y la desviación estándar. Primero se ejecutaron en contenedores montados en sistema operativo (SO) y después en contenedores montados en máquinas virtuales.

Los resultados de las evaluaciones de **CPU** indican que los contenedores montados en SO tienen mejores resultados, en un 5% sobre los contenedores que están en KVM. No hay una gran diferencia en el **performance** de la memoria para los contenedores Docker que se ejecutan en

SO o los que están montados en VM. En las pruebas de **performance** de I/O de disco, existe una diferencia de 5% en el **performance** para los contenedores Docker que se ejecutan en SO o los que están montados en VM, seguido por los SO basados en **hypervisor** KVM y después los de **hypervisor** XEN. El **performance** de red, los contenedores Docker que se ejecutan en SO presentan mejores resultados junto con los montados en **hypervisor** basados en KVM, seguidos por **Hypervisor** XEN y finalmente los de **Hyper-V**.

Por lo que indican los autores, los contenedores montados en SO o Bare Metal tienen los mejores resultados que los que están montados en máquinas virtuales. De acuerdo a los tipos de VM existentes, aquellos que se montan en **hypervisor KVM** presentan mejores resultados que los que se montan en **hypervisors** XEN o Hyper-V.

7.3.8 CloudWave: Content gathering network with flying clouds

El trabajo de [Santos et al., 2018] investiga el impacto de la contenerización en el consumo de energía, comparando estadísticamente el consumo de energía de tres cargas de trabajo de aplicaciones en Docker y en Linux **bare-metal**. El objetivo es determinar el impacto de la contenerización en el consumo energético utilizando numerosas cargas de trabajo medidas, ejecutado con y sin contenerización.

Para las pruebas realizadas se utilizaron scripts de **Python** y **MySQL**, **WordPress** con **MySQL** y **Redis** con **PostgreSQL**. Cada una de las aplicaciones hace hincapié en diferentes recursos de hardware y, en conjunto, proporciona información sobre el **performance** y la energía sobre qué tipos de aplicaciones son más adecuadas para Docker.

Se determinó si las muestras de energía en Linux y en Docker se distribuían normalmente utilizando la prueba de normalidad de **Shapiro-Wilk**. Después, se aplicaron varias pruebas para determinar si las muestras provenían de la misma distribución. Para los datos distribuidos normalmente, se utilizó una prueba T de Student emparejada. Para el resto, se aplicaron pruebas no paramétricas: una prueba de suma de Kruskal-Wallis Rank y una prueba de suma de rangos de **Wilcoxon** por pares. Para cuantificar la diferencia, se calculó el tamaño del efecto. Para todas las pruebas, se utilizó el delta de **Cliff**, que simplemente compara la frecuencia con la que las muestras de una distribución son mayores que las muestras de la otra distribución.

En la prueba realizada en la plataforma de **WordPress**, cuando se ejecutó mediante SSL/TLS el consumo de energía fue bastante mayor y se considera que se debe a la encriptación de la información. Los resultados de la prueba entre Linux **bare-metal** y Docker, la correlación lineal de energía con el tiempo arrojó 0.996 y 0.995, respectivamente. Dada la muy alta correlación de energía con el tiempo, también se comparó la cantidad de tiempo que tomó completar cada prueba y, como en Docker el tiempo transcurrido es mayor, el consumo de energía es mayor.

7.3.9 A Performance Evaluation of Lightweight Approaches to Virtualization

En el trabajo de [Plauth et al., 2017], se realizó una evaluación de **performance** general que cubre contenedores, **unikernels**, virtualización de todo el sistema, hardware nativo y combinaciones de los mismos. El objetivo es evaluar métricas que sean aplicables a las

aplicaciones **cloud**. Para este propósito, se evaluó el **performance** de la aplicación utilizando servidores HTTP y un almacén de valores clave.

Container o Virtualización a nivel de sistema operativo

Los contenedores se introdujeron como un enfoque de oposición a la virtualización de todo el sistema. Está basado en la observación de que todo el kernel induce una sobrecarga de recursos excesiva simplemente para aislar y empaquetar aplicaciones pequeñas. Existen dos clases de enfoques de virtualización de contenedores: contenedores orientados a aplicaciones y al sistema operativo. Para los contenedores orientados a aplicaciones, las aplicaciones individuales constituyen las unidades de implementación como Docker. Para los contenedores orientados al sistema operativo, se reproduce todo el espacio de usuario del sistema operativo, por ejemplo Linux LXD.

Unikernel (Virtualización de Hypervisor)

El Unikernel es un concepto de librería de sistema operativo, que solo proporciona una capa delgada de protección y facilidades de multiplexación para los recursos de hardware, mientras que el soporte de hardware se deja a las bibliotecas empleadas y a la propia aplicación.

1) Rumprun: El Unikernel Rumprun se basa en el proyecto del kernel rump, que es una versión fuertemente modularizada del kernel NetBSD que fue construida para demostrar cualquier concepto de kernel.

2) OSv: está diseñado específicamente para reemplazar sistemas operativos de propósito general como Linux en máquinas virtuales basadas en **cloud**, OSv admite la ejecución de software POSIX existente y sin modificar, siempre que se consideren ciertas limitaciones.

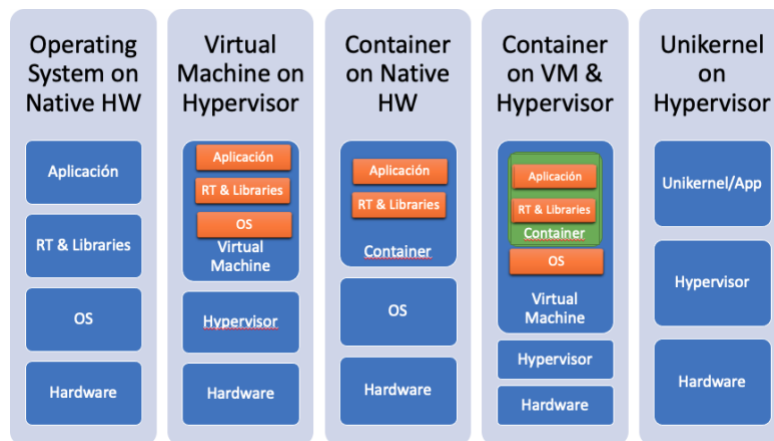


Figura 39. Comparación ilustrada de la complejidad de la pila de software de varias estrategias de implementación, incluidas configuraciones nativas, máquinas virtuales, contenedores, contenedores dentro de máquinas virtuales y unikernels de [Plauth et al., 2017].

Los resultados de las pruebas de **throughput** y **footprint memory**, determinan que las aplicaciones Unikernel se comportaron igual o incluso mejor que las aplicaciones contenerizadas. Las aplicaciones sin proceso se comportan mejor en aplicaciones virtualizadas o en **unikernels**. La principal ventaja de los contenedores sigue siendo que la memoria se puede

asignar dinámicamente, mientras que las máquinas virtuales están restringidas a predefinir la cantidad de memoria asignada en el momento de la instanciación

7.3.10 Robust design to optimize client-server bi-directional communication for structural analysis web applications or services

El objetivo del trabajo de [Calvo et al., 2017], es proporcionar suficiente información para elegir la mejor tecnología de comunicación para el análisis estructural como una aplicación en el entorno de Internet. En esta investigación está enfocada en identificar los factores que afectan el tiempo de comunicación y la tecnología (o combinación de tecnologías) que es más adecuada para intercambiar datos estructurales para el software de análisis estructural como servicio. El tamaño de la estructura es un factor relevante; las estructuras más grandes llevan cadenas más grandes en serie que requieren más tiempo para ser transmitidas, además, el servidor toma más tiempo para procesar y el tiempo de serialización/deserialización también aumenta en consecuencia.

7.3.11 Web Service Component Reusability Evaluation: A Fuzzy Multi-Criteria Approach

Los autores de [Singh & Tomar, 2016], establecen un proyecto de definición de métricas para servicios Web en un enfoque de criterios de lógica difusa para evaluar la reusabilidad de los componentes de los servicios web, basado en un estudio de trabajos relacionados en donde determinan los atributos de reusabilidad de los servicios Web.

Para realizar la evaluación, adoptan la función de lógica difusa y se estiman variables lingüísticas como “Muy Alto”, “Alto”, “Medio”, “Bajo” y muy “Bajo”, en la que la función evalúa mediante un conjunto de variables trianguladas en escala del 0 al 1. El componente de reusabilidad cuantificado es dividido en perspectiva del desarrollador que considera los atributos de Acoplamiento y Complejidad de la Interface, y en la Perspectiva del Desarrollador que considera los atributos de Seguridad, Tiempo de Respuesta y Estado. Para el caso de prueba, se utilizó un componente de software basado en servicios Web llamado TempConvert, el cual permite hacer cálculos de conversión de temperatura, con dos operaciones CelsiusToFahrenheit y FahrenheitToCelsius. La evaluación utiliza lógica borrosa para obtener la media ponderada de los pesos asignados de un conjunto atributos, después se realiza la evaluación de la perspectiva del usuario, también se evalúa la perspectiva del desarrollador.

7.3.12 Evolutionary composition of QoS-aware web services: A many-objective perspective

En el trabajo de [Ramírez et al., 2017], los autores enfocan su trabajo a través de un sistema inteligente de la evaluación de los aspectos de calidad de los servicios Web. El objetivo es utilizar algoritmos multiobjetivo evolutivos (MOEA, por sus siglas en inglés) para determinar los modelos de composición de los servicios web. Los MOEA son meta heurísticas basadas en la población dedicadas a resolver problemas de optimización de objetivos múltiples, es decir,

definen un conjunto de soluciones candidatas, la población de individuos, que se modifican a través de algunas iteraciones en busca de generar mejores soluciones.

En este trabajo se propone que el resultado de la evaluación determine el modelo de composición del servicio Web y que los algoritmos evolutivos permitan lograr el equilibrio del costo computacional razonable y la integración de servicios de terceros.

Algoritmos Evolutivos evaluados: SPEA2, NSGA-II, MOEA/D, ϵ -MOEA, GrEA, IBEA, HypE y NSGA-III.

Atributos evaluados: tiempo de respuesta, disponibilidad, **throughput**, confiabilidad, latencia, porcentaje de éxito, cumplimiento, mejores prácticas y documentación.

Para los diferentes experimentos, cada algoritmo se ejecutó 30 veces, con 10, 20, 30, 40 y 50 tareas, donde cada tarea tiene un conjunto diferente compuesto de 1 a 11 servicios como candidatos. Los resultados obtenidos de acuerdo al frente de Pareto el algoritmo **HypE** ofrece modelos óptimos para la composición, de acuerdo a las capacidades de los servicios. Además los autores mencionan que es posible evaluar las capacidades de calidad de los servicios Web a través de algoritmos evolutivos para obtener modelos de composición.

7.3.13 Coverage-based testing for Service Level Agreements

En el trabajo de [Palacios et al., 2015], los autores proponen un enfoque para probar servicios basados en aplicaciones (SBA por sus siglas en inglés) recolectando datos de la ejecución y tomando decisiones de acuerdo al SLA bajo lineamientos del criterio Cobertura de decisión de condición modificada (MCDC por sus siglas en inglés). El objetivo es garantizar que el SBA cumpla con las condiciones especificadas en el SLA y asegurar que el SBA sea capaz de comportarse apropiadamente incluso cuando otros servicios no cumplan con las especificaciones del SLA.

7.3.14 Response Time based Optimal Web Service Selection

En este trabajo de investigación [Ahmed et al., 2015], los autores proponen un modelo de evaluación de calidad de servicios basado en Modelos Ocultos de Márkov (HMM por sus siglas en inglés), que puede ser utilizada para medir y predecir el comportamiento de los servicios web enfocado al tiempo de respuesta del servicio. El objetivo es proporcionar una forma estándar de medir y predecir el comportamiento de los servicios web en términos de tiempo de respuesta utilizando HMM.

7.3.15 Exploiting Client-Side Collected Measurements to Perform QoS Assessment of IaaS

La propuesta de [Kamel et al., 2015], consiste en un enfoque de evaluación de servicios del lado del cliente que se basa principalmente en la evaluación de los clientes de las ofertas de servicios de infraestructura. El objetivo es construir modelos de Calidad de Servicio (QoS por sus siglas en inglés) para predecir el desempeño de clientes que exhiben comportamientos similares. Por lo que los autores desarrollaron un algoritmo de agrupamiento basado en SNA (Análisis de Redes Sociales por sus siglas en inglés) que analiza los vínculos de interconexión

entre clientes y clientes relacionados con el clúster en comunidades de comportamientos similares, que se utilizó con el Teorema del Valor Extremo Generalizado (EVT) para converger en un modelo preciso de QoS.

7.3.16 Multi-criteria Content Adaptation Service Selection Broker

En el trabajo de [Fudzee et al., 2014], los autores proponen una estrategia para evaluar las solicitudes entrantes y sean capaces de ofrecer la nueva QoS a las solicitudes potencialmente rechazadas inicialmente, tomando en cuenta la carga actual del servidor y la prioridad de las solicitudes. El aspecto innovador es que la estrategia propuesta es capaz de liquidar más SLA's, si los **brokers** aceptan la nueva QoS.

El algoritmo evaluador de SLA permite que las entradas de información (solicitudes), en el cual se incluye la prioridad, la carga actual del servidor y la QoS anunciada. Con estas entradas, el algoritmo estima el número de solicitudes que se pueden realizar dentro de la QoS, en función de la carga actual del servidor. La función de prioridad calcula la prioridad de cada solicitud. El evaluador de SLA verifica la carga actual y estima cuántas solicitudes puede atender dentro del tiempo de espera ofrecido.

7.3.17 QoS Ranking Prediction for Cloud Services

En el trabajo de [Zheng et al., 2013], los autores proponen un marco de predicción de clasificación de QoS para los servicios **cloud**, aprovechando las experiencias de uso de servicios anteriores de otros consumidores, con dos enfoques personalizados de predicción de clasificación de QoS para predecir las clasificaciones de QoS directamente.

Primero se calculan los valores de similitud entre el usuario activo actual con otros usuarios, para identificar los usuarios similares. Posteriormente, dos algoritmos CloudRank1 y CloudRank2 hacen una clasificación de servicios personalizada, al aprovechar las experiencias de uso de servicios anteriores de usuarios similares. Por último, los resultados de la predicción de clasificación se proporcionan al usuario activo. Los cálculos de similitud de clasificación comparan las clasificaciones de QoS de los usuarios en los servicios comúnmente invocados. CloudRank1 evalúa la preferencia del usuario, sean activos o potencial, es decir, la preferencia puede obtenerse implícita o explícitamente a partir de la similitud de usuarios. CloudRank2 evalúa la preferencia a partir de la información de QoS de usuarios similares

7.3.18 QoS-enabled multicast for delivering live events in a Digital Cinema scenario

El propósito de [di Sorte et al., 2009], es presentar un sistema el cual es capaz de soportar la entrega en red de contenidos de cine digital enfocados en eventos en vivo. El servicio consume grandes cantidades de ancho de banda de red, no solo por el volumen de datos transmitidos, también por el número de receptores. DigitalCinema (DC) es una integración de tecnologías (escáner de filmes de alto **performance**, algoritmos de compresión de imágenes digitales, y proyección digital avanzada) que sustituye la distribución de películas basadas en celuloideos.

7.4 Comparación de la calidad de los SW SOAP y REST

Esta categoría de la revisión de los trabajos relacionados está basada en la búsqueda de modelos de calidad, relacionados a la definición de atributos de calidad que apoyan la entregabilidad de los servicios Web.

7.4.1 Distributed QoS evaluation for real-world Web services

En el trabajo de [Zheng et al., 2010], los autores clasifican a los atributos de calidad de los servicios web de acuerdo a las características no funcionales en dependientes del usuario (precio, popularidad) y no dependientes del usuario (**throughput**, tolerancia a fallas, tiempo de respuesta), basándose en características de **throughput** de los servicios Web para utilizar nuevas técnicas y modelos de servicios Web.

Para realizar las pruebas se realizaron tres tipos de pruebas diferentes:

Información de Servicios Web.

Se localizaron 28,529 servicios Web, de los que se obtuvieron 21,358 documentos WSDL de 89 países, se concretaron 16,514 (77.32%) mientras que fallaron 4,844. De las fallas obtenidas por **Empty File** encontraron 249, por **Invalid File Format** encontraron 1232, por **Error Parsing WSDL** encontraron 1135, por **Invocation Target Exception** encontraron 764, por **Null QName** sólo encontraron 22 y el último **Databinding Unmatched Type Exception** encontraron 4.

Información de Fallas a solicitudes a servicios Web.

Para obtener un resultado de probabilidades de falla, los autores mencionan que realizaron 1,542,884 solicitudes de servicios por parte de 150 usuarios de 24 países distintos a 100 servicios Web de 22 países diferentes. Los resultados de las pruebas de error arrojaron la media de 4.05 por ciento y la desviación estándar fue de 17.32 por ciento, de los cuales obtuvieron 58,184 errores en total. La información de los resultados está clasificada por errores del lado del cliente, errores de red, o errores del lado del servidor.

Información de Tiempo de Respuesta y Throughput.

Los autores indicaron que para esta prueba utilizaron 5825 servicios Web localizados en 73 países diferentes y fueron ejecutados 1,974,675 ocasiones por 339 usuarios de 30 países diferentes. De los que se obtuvo el resultado del tiempo de respuesta con una media de 1.43 segundos y la desviación estándar fue de 31.9. Los resultados del **throughput** de la media fue de 102.86 KBPS y la desviación estándar arrojó un resultado de 531.85 KBPS. Los resultados del tiempo de respuesta de un servicio Web puede deberse al tiempo de transferencia de datos o al tiempo de procesamiento de solicitudes en el lado del servidor.

7.4.2 Performance comparison of SOAP and REST based Web Services for Enterprise Application Integration

En [Kumari & Rath, 2015], los autores comparan el *throughput* y el tiempo de respuesta de las pruebas de servicios Web en SOAP y REST. Para realizar las pruebas utilizaron servicios Web basados en SOAP y una herramienta basada en *OpenESB* (integración de varios componentes para direccionar cada servicio con el bus) llamada *LOANBROKER*. Se simuló 28 usuarios que hicieron las solicitudes de manera concurrente. Para los servicios Web basados en REST se utilizó una página JSP.

En este trabajo, el *throughput* es el cálculo del número de solicitudes dividido por el tiempo total de procesamiento de todas las solicitudes. El tiempo de respuesta es el tiempo que transcurre desde que la aplicación o el servidor recibe una solicitud y el cliente recibe una respuesta de esa solicitud. En las evaluaciones de *throughput* y el tiempo de respuesta, se tomaron las muestras de las lecturas de archivos que desde un directorio del disco, de diferentes tamaños que varían de 1000 kb a 7000kb. De acuerdo a los resultados, REST tiene mejores resultados de *performance*, y se observó que a medida que el tamaño del archivo incrementa, el *throughput* incrementa, pero cuando el número de solicitudes aumenta, el *throughput* disminuye para ambos casos, SOAP y REST.

7.4.3 A Framework for Using Web Services to Enhance QoS for Content Delivery

En el trabajo de [Buccafurri et al., 2009], los autores presentan un enfoque y un marco, diseñados para permitir el análisis de QoS de los procesos de los servicios Web, para proporcionar (RTSP), en función de la composición de servicios. La arquitectura incluye un módulo para predecir posibles fallas de calidad de servicios, a través de un método de *machine-learning* y un módulo para monitorear parámetros de calidad de servicios y soportar posibles acciones de adaptación y recuperación en caso de falla.

El modelo de calidad de servicios web se basa en los siguientes atributos:

Tiempo de respuesta, Precio, Disponibilidad, Reputación, Puntualidad, Exactitud e integridad de la calidad de los datos; y otro para la capa RTSP que adopta los parámetros basados en el flujo de la transmisión de multimedia (*Streaming*) como: los relacionados con el usuario: resolución, luminosidad, canal de audio, frecuencia, etc.; y los relacionados con la red de cómputo: ancho de banda para video y ancho de banda para audio. Para anticipar las fallas se utilizó un modelo de predicción, basado en el monitoreo y en el *machine-learning*. El objetivo es adaptarse a los cambios de variables de red y a la recuperación en casos de fallo. Cuando el sistema completa la fase de arranque, puede obtener una predicción de QoS global sobre la operación objetivo, dando los valores actuales de la instancia de entrada para la operación actual al sistema de aprendizaje automático. Debido a que cada operación proporcionada por el servidor al usuario debe cumplir con el contrato correspondiente, si la calidad de los servicios proporcionados no se cumple, se inicia la función de recuperación mediante acciones de reparación, como deshacer, rehacer, sustituir o compensar una operación y en caso de anomalías, la aplicación reporta al administrador de la aplicación.

7.5 Comparación de Arquitecturas de Servicios Web

Para obtener los factores que permitan realizar la comparación de arquitecturas de servicios entre SOAP y REST, se buscó en la literatura los diferentes conceptos de los servicios Web SOAP y los conceptos de servicios Web con REST, basados en encontrar diferencias tanto técnicas como teóricas en cuanto a su arquitectura, interacción e implementación.

7.5.1 Comparison of SOAP and REST Based Web Services Using Software Evaluation Metrics

En el trabajo de [Tihomirovs & Grabis, 2016], los autores comparan los servicios web basados en SOAP y REST utilizando métricas de evaluación de software. El objetivo es resumir las ventajas y desventajas principales del protocolo SOAP y REST. Para lograr este objetivo, se establecieron las siguientes tareas:

- 1 Definir las métricas mediante las cuales es posible comparar mutuamente los protocolos SOAP y REST.
- 2 Comparar SOAP y REST mediante las métricas encontradas.
- 3 Evaluar qué tipo de protocolo es mejor sobre la base de las métricas de evaluación de software

Los autores proponen métricas directas e indirectas. Las métricas directas son: costo, esfuerzo, líneas de código, ejecución, velocidad, memoria, errores. las métricas indirectas son: funcionalidad, calidad, complejidad, eficiencia, confiabilidad, mantenibilidad.

La búsqueda de servicios fue realizada en las bibliotecas SCOPUS, *Web of Science* y en *IEEE Xplore Digital*. En la búsqueda se encontraron 190, 93 y 352 respectivamente. Los criterios de inclusión fueron: artículos científicos revisados por otros para garantizar alta fiabilidad y nivel de calidad y que el enfoque del estudio este enfocado en comparar SOAP con REST. Los criterios de inclusión tomados en cuenta son el idioma inglés, y que sea una publicación científica y que los protocolos no sean distintos a SOAP o REST.

En la Tabla 52 se presentan los enfoques que se deben de implementar de acuerdo a las categorías de las métricas. Por ejemplo, para obtener mejores resultados respecto al costo se recomienda utilizar los servicios Web basados en REST, se identifica la columna marcada con una “X” que indica cual es el enfoque de implementación favorable. El caso de mantenibilidad del lado del servidor es favorable que se implementen servicios Web basados en REST. El estudio se centró en 635 documentos encontrados, la selección basada en el resumen y títulos se limitó a 122 y cuando se tomaron en cuenta los criterios de inclusión y exclusión, dejó un total de 14 artículos científicos.

Anexo A. Trabajos Relacionados

Tabla 52. Enfoque de implementación de servicios Web [Tihomirovs & Grabis, 2016].

Métrica		SOAP	REST
Esfuerzo	Costo (más bajo)		x
	Servidor		x
	Cliente	x	
	LOC	x	
	Velocidad		x
	Memoria		x
	Error	x	
Funcionalidad	Independencia de Protocolo	x	
	Complejidad		x
Eficiencia	<i>Throughput</i>		x
	Confiabilidad	x	
Mantenibilidad	Servidor		x
	Cliente	x	

De acuerdo a los tipos de métricas encontradas relacionadas a los tipos de servicio Web SOAP y REST, no es posible identificar claramente el mejor protocolo para integrar los sistemas de información y las aplicaciones con otros sistemas. Esto significa que cada proyecto debe evaluarse individualmente mediante requisitos funcionales y no funcionales. En la Tabla 53 se describen los criterios a considerar y se observan los ejemplos de proyectos y el enfoque de servicios Web que se debe implementar.

Tabla 53. Selección de enfoque de acuerdo al tipo de proyecto [Tihomirovs & Grabis, 2016].

Implementación	Criterios	Ejemplo de Proyecto
REST	Compatibilidad Rendimiento Simplicidad Escalabilidad Modelo de comunicación punto a punto Ancho de banda limitado	Integración de aplicaciones móviles con sistemas de información. Soluciones simples de intercambio de datos cliente-servidor
SOAP	Alto grado de seguridad y confiabilidad Bajo número de errores Solicitudes asíncronas Computo distribuido Soporte para otros estándares (WSDL, WS).	Sistemas de información de negocios (B2B) Sistemas de información bancarios Sistemas de pago

7.5.2 Comparing Performance of Web Service Interaction Styles : SOAP vs . REST

En el trabajo de investigación de [Pavan et al., 2012], los autores hacen una comparativa desde un enfoque académico de los tipos de interacción basada en transferencia de información utilizando métricas como: balance de carga, **throughput** y tiempo de respuesta de la ejecución de servicios Web en los dos estilos de interacción en diferentes escenarios, con red cableada y con red inalámbrica.

Para las pruebas realizadas se crearon dos servicios Web, uno basado en SOAP y otro basado en REST, los cuales ejecutan las mismas operaciones CRUD sobre una misma base de datos, y una función que permite obtener un archivo almacenado en el servidor, esta función es utilizada para hacer cálculos de **throughput**.

La metodología utilizada para realizar los cálculos del tiempo de respuesta fue la misma para los dos escenarios, mediante una aplicación cliente que invoca a un servicio y envía datos relevantes en un momento A (medido en milisegundos). El servicio recibe la solicitud, la procesa, se conecta a la base de datos, realiza la operación solicitada en la base de datos y envía una respuesta adecuada al cliente. El cliente recibe la respuesta completa en algún momento B. El tiempo de respuesta fue calculado como la diferencia entre los tiempos A y B.

El **throughput** se evaluó utilizando la función **getTheFile**, que recupera un archivo especificado almacenado en algún directorio del servidor. Se usaron diez archivos de diferente tamaño desde 76 KB a 5 MB. Los resultados de los experimentos indicaron que REST tiene mejores tiempos de respuesta y **performance** que SOAP. Sin embargo, para la prueba de **throughput**, la diferencia fue mínima de acuerdo a los datos estadísticos mostrados. La Tabla 54 muestra las características más importantes por enfoque de SW, tomando en cuenta el modelo de implementación de servicios Web y los recursos del enfoque de SW.

Tabla 54. Características de los enfoques de servicios Web [Pavan et al., 2012].

	SOAP	REST
Protocolo	SMTP, FTP, HTTP	HTTP + URI
Modelo	Modelo orientado a comunicación	Modelo orientado a recursos.
Recursos	Utiliza estándares para exponer los recursos. WSDL, WS-BPEL, WS-Choreography, WS-Transaction, WS-Security, WS-Addressing, etc.	Representando y exponiendo los recursos del sistema, y transfiriendo datos a través de HTTP

7.5.3 RESTful web services vs. "Big" web services: Making the right architectural decision

En el trabajo de investigación de [Pautasso et al., 2008], los autores utilizaron un enfoque cuantitativo para comparar las dos tecnologías de estilos de integración. El método de comparación se basó en un modelo de decisión separado en cuatro niveles de abstracción:

- ◆ Comparación de principios arquitectónicos.
En este método se considera el protocolo de capa para entregar servicios de aplicación a clientes, en la forma en que HTTP se considera como aplicación o protocolo de transporte. La heterogeneidad de los componentes puede utilizarse para hablar entre clientes y servidores, por ejemplo, los diferentes navegadores, representaciones, bibliotecas y la variedad de herramientas pueden procesar y admitir el mismo protocolo HTTP.
- ◆ Comparación de decisiones conceptuales.
Este método compara la forma en la que se estipula el contrato del servicio. El nivel de contrato, el uso de la base de datos, el método de transferencia, etc., Tomando en cuenta las similitudes y las diferencias de las características que permiten la interacción.
- ◆ Comparación de decisiones de tecnología.
Esta comparación se centra en revisar las herramientas tecnológicas implementadas por el estilo del servicio Web, desde el formato de salida, el protocolo que utiliza hasta la composición del servicio con *mashups* y la implementación de tecnología para el descubrimiento de servicios como UDDI.
- ◆ Comparación de nivel de activos del proveedor.

Como conclusión, los autores mencionan que en la comparación también se puede ver que los dos estilos son bastante similares, siempre que se compare el mismo subconjunto de decisiones tecnológicas. REST es la opción cuando las características de nivel de empresa de WS-*¹ (transacciones, confiabilidad, seguridad a nivel de mensaje) no son necesarios.

7.5.4 A Comparison of SOAP And REST Implementations of a Service Based Interaction Independence Middleware Framework

En [Mulligan et al., 2009], se describe el diseño conceptual de un marco middleware que es independiente de interacción y describe el rol de la participación de los servicios Web dentro del middleware. Se realizaron pruebas con los paradigmas de implementación de servicios Web SOAP y REST, con el objetivo de medir la transmisión de datos, proporcionar

¹ "WS-" es un prefijo que se usa para indicar las especificaciones asociadas con los servicios web, existen muchos estándares WS-* que incluyen WS-Addressing, WS-Discovery, WS-Federation, WS-Policy, WS-Security y WS-Trust.

implementaciones para la arquitectura orientada a servicios y evaluar el desempeño con respecto a la eficiencia y escalabilidad.

Como plataforma intermedia de aplicaciones entre el modelo cliente servidor, presentan la aplicación Portal como modelo para resolver problemas de heterogeneidad y para soportar una amplia gama de aplicaciones periféricas que se utilizan.

Portal es un middleware que presenta una serie de componentes divididas en dos partes: La primera que se utiliza para interactuar con aplicaciones clientes que se encarguen de traducir los mensajes de entrada a la aplicación y la segunda que consta del servidor portal, encargado de manejar la información, las técnicas de interacción, la aplicación, dispositivos y perfiles de usuario.

El conjunto de pruebas realizadas para hacer la comparación entre SOAP y REST consiste en la implementación de CRUD. La primera prueba consistió en obtener resultados de la latencia por tiempo y por tamaño del paquete en bytes, de los cuales REST obtiene mejores resultados argumentando que el procesamiento de datos por el parseo (Análisis sintáctico) en XML de SOAP es más pesado. La segunda prueba consiste en enviar solicitudes síncronas de las cuales REST tiene los mejores resultados a partir de 100 solicitudes. Como resultado de las pruebas, la implementación REST demostró ser más eficiente en términos del ancho de banda de red utilizado cuando se transmiten solicitudes de servicio.

7.5.5 A Comparison of RESTful vs. SOAP Web Services in Actuator Networks

En [Malik & Kim, 2017], los autores mencionan que los servicios basados en REST y SOAP son los dos servicios de aprovisionamiento Web principales en la actualidad. Los servicios **RESTful** se están volviendo muy populares con el paso del tiempo y están reemplazando a los servicios Web basados en SOAP en muchos campos debido a su facilidad de uso, implementación y ligereza. Por lo tanto, surge una cuestión con respecto a un mejor **performance** entre los servicios **RESTful** y basados en SOAP. Para las pruebas realizadas compara los protocolos SOAP y REST mediante el diseño de redes de actuadores interiores.

La arquitectura básica del sistema de control de actuador tiene seis capas: capa de hardware, capa de control, capa de procesamiento, capa de gestión de datos y capa de federación y cliente de aplicación. Aunque tanto SOAP como la arquitectura **RESTful** tienen las mismas capas, las interacciones entre las capas y los sub módulos son diferentes. Uno sigue los principios de REST, mientras que otro sigue los principios de SOAP. La lógica del sistema y los componentes implementados siguen siendo los mismos para ambos en un nivel básico.

Como conclusión de los resultados, REST proporciona resultados razonablemente beneficiosos en comparación con SOAP. No sólo proporciona acceso de consulta rápido y tiempos de respuesta, sino que también reduce el consumo de CPU y la carga de los servicios a los que se accede. Como REST es liviano, tiene un tamaño de mensaje más

pequeño, lo que se traduce en menos tiempo de análisis requerido y menos latencia, por lo que puede ser un mejor enfoque para el desarrollo de dichas aplicaciones.

7.5.6 Exploring SOAP and REST communication on the Android platform

En el trabajo de [Bloebaum & Johnsen, 2015], se comparan dos posibles enfoques para lograr la interoperabilidad entre los servicios Web SOAP y las aplicaciones que se ejecutan en Android. El primer enfoque es agregar soporte SOAP a la plataforma Android, mientras que el segundo enfoque es utilizar un mecanismo nativo con dispositivos Android.

Los experimentos se establecieron en un entorno controlado, con un servidor que proporciona servicios SOAP y REST, conectados de forma alámbrica a un punto de acceso (conexión por cable) y un teléfono móvil Android conectado al mismo punto de acceso de forma inalámbrica (conexión **Wi-Fi**). Para las pruebas realizadas se utilizaron 4 escenarios, el mensaje se basó en SOAP con GZIP, SOAP en texto plano, REST con GZIP y REST en texto plano. Se monitoreo el tiempo de respuesta y el consumo de batería del dispositivo Android.

Las primeras pruebas realizadas consisten en el envío de un mensaje desde un dispositivo Android al servidor, repetido 20 ocasiones en 4 escenarios, con SOAP con GZIP, SOAP en texto plano, REST con GZIP y REST en texto plano. En este caso SOAP fue ampliamente más rápido que REST. El segundo conjunto de pruebas realizadas consiste en la extracción de un mensaje desde un dispositivo Android. El tiempo de respuesta de SOAP es más bajo que REST.

En el tercer conjunto de pruebas consiste en el envío de un mensaje largo o grande a través de la red, los resultados de las pruebas de los servicios Web basados en SOAP demuestran que consume menos tiempo utilizando el texto plano debido a que no tiene que realizar la compresión del mensaje. En estas mismas pruebas se evaluó el consumo de la batería en donde se demostró que REST tiene un menor consumo debido a que SOAP tiene una capa extra que es la que realiza la serialización y deserialización del mensaje en XML.

Las pruebas mostraron que SOAP en general fue más receptivo que REST, mientras que REST produjo el mejor **performance** con respecto a la duración de la batería. Los autores mencionan que, aunque los servicios REST consumen menos energía o batería del dispositivo, los servicios Web SOAP ofrecen una mejor capacidad de respuesta. Por lo tanto, se pueden resumir las recomendaciones como se muestra en la Tabla 55.

Tabla 55. Recomendaciones de SOAP vs REST [Bloebaum & Johnsen, 2015].

Objetivo General	Recomendación
Interoperabilidad de la OTAN	SOAP
Servicios de infraestructura máquina a máquina	SOAP (o REST, tal vez envolviendo el servicio SOAP)
Servicios de área funcional	SOAP (o REST, tal vez envolviendo el servicio SOAP)

Anexo A. Trabajos Relacionados

Cientes de dispositivos Inteligentes	REST
Cientes de dispositivos No Inteligentes	SOAP (o REST, si el cliente está escrito en JavaScript)

Anexo B.

Especificaciones de Entrada Y Salida

8.1 Especificaciones de Entrada y de Salida

En este apartado se definen las especificaciones de entrada y salida para el plan de pruebas de entrega de SWA, definidos en los tipos de implementación SOAP y REST. También se incluye la definición del Payload requerido para las solicitudes a SWA de tipo de implementación SOAP. Por último, se define la completez de los SWA, que para estas pruebas no solo es indicativo del código de respuesta HTTP 200, se incluye un ejemplo del indicativo de una respuesta completa para este trabajo, debido a que puede responder una solicitud con código de respuesta HTTP 200 y no entregar el RA completo.

8.1.1 Especificaciones de Entrada

Las especificaciones de entrada para pruebas de entregabilidad SWA son las siguientes:

1. La URL o **Endpoint** del SWA.
 - ◆ El puerto en donde se realiza la conexión de red.
 - ◆ Protocolo de red que permite la comunicación entre el cliente y el servidor.
 - ◆ Contexto, nombre del proyecto.
 - ◆ Nombre del SWA.
 - ◆ Para el caso de los SWA de tipo de implementación SOAP se requiere el nombre de la operación, para las pruebas de entregabilidad de SWA el nombre de la operación es **sendData**, como se indica en la Figura 40.
 - ◆ Para el caso de los SWA de tipo de implementación REST se requiere el método HTTP utilizado, para las pruebas de entregabilidad de SWA, el método es GET. La Figura 41 ilustra la configuración del método HTTP GET utilizado en Apache JMeter para realizar solicitudes a SWA de tipo REST.
2. El tamaño del SWA, cada SWA entrega un RA el cual se indica en el nombre del SWA.
3. Para los SWA de tipo de implementación SOAP, se requiere del Payload XML que se describe a continuación.

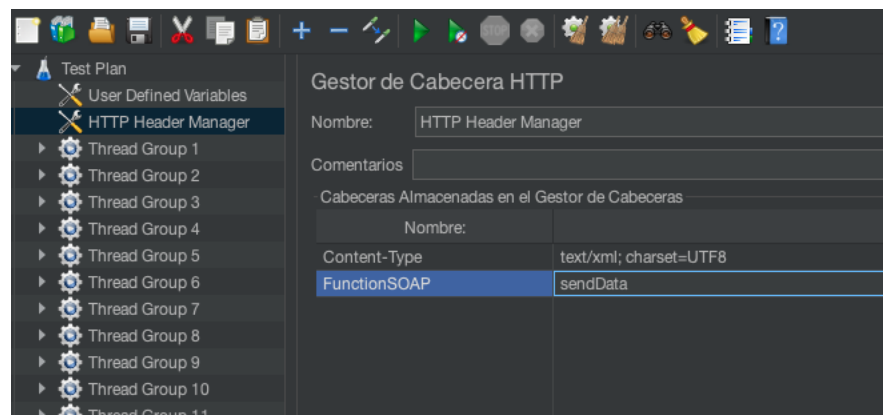


Figura 40. Operación SOAP definida para SWA SOAP.

Figura 41. Método HTTP GET para solicitudes a SWA REST.

8.1.2 Payload

Para realizar la interacción con los SWA basados en SOAP, es requerido un elemento llamado Payload, en este indica como se debe de realizar la comunicación y la información a intercambiar. En la Tabla 56 se presenta el código en formato XML del Payload para los SWA y en la

Figura 42 se presenta el XML Payload en la herramienta Apache JMeter. Para el caso de los SWA de tipo de implementación REST, esto no es requerido.

Tabla 56. XML Payload para SWA de tipo SOAP.

Código XML Payload
<pre><?xml version="1.0" encoding="utf-8"?> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Body> <sendData xmlns="http://Services/" /> </soap:Body> </soap:Envelope></pre>

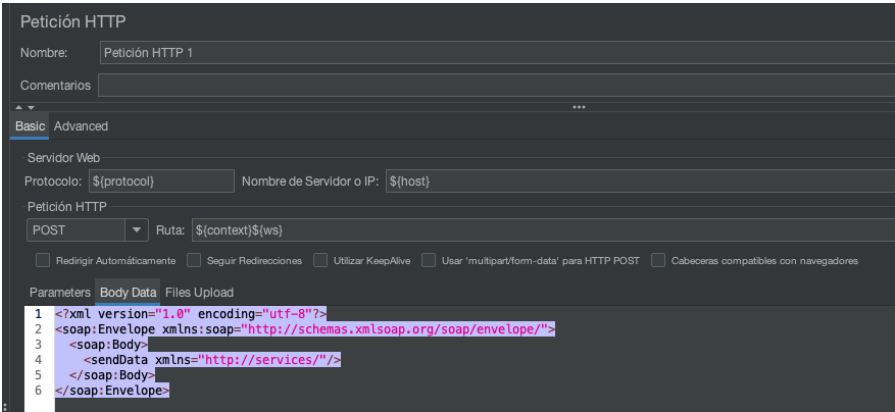


Figura 42. XML Payload en SWA SOAP.

8.1.3 Especificaciones de Salida

El plan de pruebas de entrega de SWA definido en la aplicación Apache JMeter se configuró para que en cada ejecución se genere un archivo de salida con formato CSV (Comma Separated Values) o separado por comas. Este archivo CSV presenta la información de los resultados de las 2500 solicitudes, agrupadas en 100 solicitudes, en 25 grupos de solicitudes. La ruta donde el archivo de salida es generado se configura en las variables del Plan de pruebas en Apache JMeter, en este caso, se definieron la variables **filePath**, **ws** y **test**, como se visualiza en la

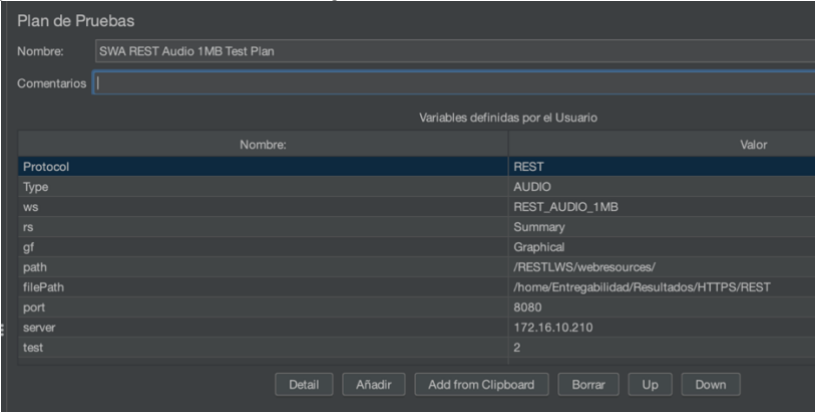
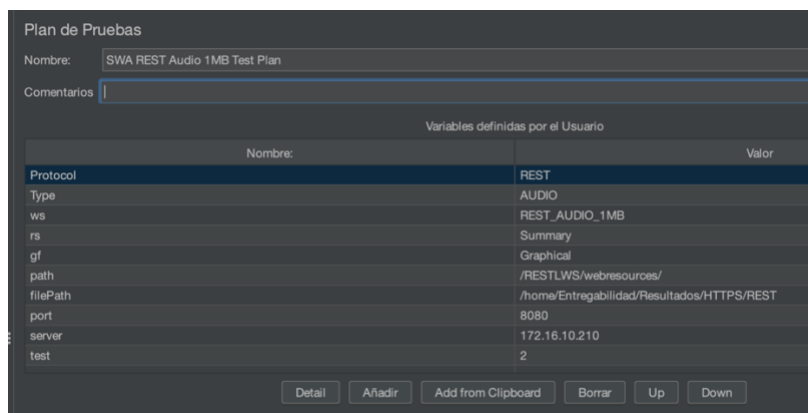


Figura 43. La variable filePath hace referencia al directorio de salida definido en este caso: “/home/Entregabilidad/Resultados/HTTPS/REST/”, para cada caso de prueba, la ruta es diferente. La variable WS que indica el nombre del SWA, para el caso de la imagen el SWA que se muestra es: “REST_AUDIO_1MB” y el número de prueba se muestra en la variable test.

Anexo B. Especificaciones de Entrada y Salida



Nombre:	Valor
Protocol	REST
Type	AUDIO
ws	REST_AUDIO_1MB
rs	Summary
gf	Graphical
path	/RESTLWS/webresources/
filePath	/home/Entregabilidad/Resultados/HTTPS/REST
port	8080
server	172.16.10.210
test	2

Buttons: Detail, Añadir, Add from Clipboard, Borrar, Up, Down

Figura 43. Configuración de variables del plan en Apache JMeter.

En el componente **Reporte resumen** del plan en Apache JMeter, se define el nombre del archivo de salida, el cual debe incluir la ruta en la cual el archivo es generado. Para las pruebas de entregabilidad se definió a través de variables “***\${filePath}\${ws}\${test}***” y al final el nombre del archivo definido como “***summary.csv***”.



Reporte resumen

Nombre: Reporte resumen

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo: Navegar... Log/Mostrar sólo: ☐ Escribir en Log Sólo Errores

Figura 44. Nombre del archivo generado de las pruebas de entregabilidad.

La aplicación cliente Apache JMeter se configuró para que en cada ejecución de las pruebas esta genere un resumen de salida con la etiqueta del grupo de solicitudes. Este resumen está compuesto por el número de solicitudes, la media del tiempo de respuesta, el tiempo de respuesta mínimo, el tiempo de respuesta máximo, la desviación estándar del tiempo de respuesta, el número de errores, el **throughput** en tiempo, el **throughput** en KB por segundo, entre otros datos. En la Figura 45 ilustra el resumen de salida de Apache JMeter para el plan de pruebas de entregabilidad de SWA.

Anexo B. Especificaciones de Entrada y Salida

Reporte resumen

Nombre:

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo Log/Mostrar sólo: ☐ Escribir en Log Sólo Errores ☐ Éxitos

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Est...	% Error	Rendimi...	Kb/sec	Sent KB...	Media d...
Petición HTTP 1	100	15842	3684	22107	4668.12	0.00%	4.4/sec	5976.34	1.88	137939...
Petición HTTP 2	100	15915	4769	22939	4619.63	0.00%	4.3/sec	5797.38	1.84	138012...
Petición HTTP 3	100	15899	1923	22959	4842.92	0.00%	4.3/sec	5783.69	1.84	138012...
Petición HTTP 4	100	33767	3183	58687	15086.95	0.00%	1.7/sec	2280.81	0.73	138012...
Petición HTTP 5	100	21674	12911	28048	4413.84	0.00%	3.5/sec	4771.56	1.52	138012...
Petición HTTP 6	100	17496	4350	23521	4154.21	0.00%	4.2/sec	5617.37	1.79	138012...
Petición HTTP 7	100	16184	772	22812	4996.94	0.00%	4.3/sec	5819.91	1.85	138012...
Petición HTTP 8	100	19938	6052	36701	9078.87	0.00%	2.7/sec	3630.66	1.15	138012...
Petición HTTP 9	100	31222	6225	45494	9938.16	0.00%	2.2/sec	2951.31	0.94	138012...
Petición HTTP 10	100	28115	5948	42163	10114.21	0.00%	2.4/sec	3183.07	1.01	138012...
Petición HTTP 11	100	28682	3972	42435	9850.30	0.00%	2.3/sec	3149.01	1.00	138012...
Petición HTTP 12	100	25371	3664	35182	7269.90	0.00%	2.8/sec	3718.41	1.18	138012...
Petición HTTP 13	100	18557	9597	25052	4460.58	0.00%	3.9/sec	5268.04	1.68	138012...
Petición HTTP 14	100	17185	6815	23481	4082.55	0.00%	4.2/sec	5694.74	1.81	138012...
Petición HTTP 15	100	15764	3150	22460	4975.01	0.00%	4.4/sec	5889.08	1.87	138012...
Petición HTTP 16	100	15592	942	22638	5234.95	0.00%	4.2/sec	5710.67	1.82	138012...
Petición HTTP 17	100	16311	5209	22356	4224.07	0.00%	4.3/sec	5837.05	1.86	138012...
Petición HTTP 18	100	15250	5527	22227	4879.32	0.00%	4.4/sec	5889.59	1.87	138012...
Petición HTTP 19	100	16961	5491	23451	4751.24	0.00%	4.2/sec	5710.42	1.82	138012...
Petición HTTP 20	100	17883	6573	23824	4075.98	0.00%	4.1/sec	5518.92	1.76	138012...
Petición HTTP 21	100	17345	5071	23122	4387.00	0.00%	4.2/sec	5722.79	1.82	138012...
Petición HTTP 22	100	16231	540	23333	4638.24	0.00%	4.2/sec	5699.56	1.81	138012...

Figura 45. Reporte de salida de Apache JMeter.

El archivo CSV que es generado componente Reporte Resumen de la aplicación Apache JMeter de los resultados de las pruebas de entregabilidad, está estructurado por las siguientes columnas: **timestamp**, **elapsed**, **label**, **responseCode**, **responseMessage**, **threadName**, **dataType**, **success**, **failureMessage**, **bytes**, **sentBytes**, **grpThreads**, **allThreads**, **Latency**, **IdleTime**, y **Connect**. La Figura 46 ilustra un archivo de salida CSV en una aplicación de texto plano y el mismo archivo en la aplicación Excel, formateado para mejorar la visibilidad, la manipulación y su presentación.

Para propósito del análisis de los resultados de la entregabilidad de SWA, las columnas del archivo CSV generado como resultado de las pruebas de entrega de SWA que se consideran importantes son las que se describen a continuación:

- **TimeStamp.** Indica el inicio del lanzamiento de la solicitud
- **Elapsed.** El tiempo entre el lanzamiento de la solicitud y en el que se obtiene la respuesta.
- **ResponseCode.** Código de respuesta del protocolo HTTP de la solicitud realizada al SWA.
- **Bytes.** Indica el número de bytes recibido como respuesta.

Anexo B. Especificaciones de Entrada y Salida

The image shows a CSV file named 'SM_4.csv' and its corresponding Excel spreadsheet. The CSV file contains log data with columns like timeStamp, elapsed, label, responseCode, responseMessage, threadName, dataType, success, failureMessage, bytes, sentBytes, grpThreads, allThreads, Latency, IdleTime, and Connect. The Excel spreadsheet shows the same data with columns A through Q. The data includes various HTTP requests and responses, some successful (200 OK) and some failing (Internal Server Error).

timeStamp	elapsed	label	responseCode	responseMessage	threadName	dataType	success	failureMessage	bytes	sentBytes	grpThreads	allThreads	Latency	IdleTime	Connect
2024/08/27 13:53:53.832	46967	Petición HTTP	200	OK	Thread Group	1-14, text, true	,438,441,100,100,46967,0,2								
2024/08/27 13:53:54.103	53335	Petición HTTP	200	OK	Thread Group	1-38, text, true	,438,441,99,99,53335,0,0								
2024/08/27 13:53:53.747	53691	Petición HTTP	200	OK	Thread Group	1-6, text, true	,438,441,99,99,53691,0,1								
2024/08/27 13:53:54.084	53359	Petición HTTP	200	OK	Thread Group	1-36, text, true	,438,441,97,97,53359,0,1								
2024/08/27 13:53:54.064	73329	Petición HTTP	500	Internal Server Error	Thread Group	1-34, text, false	,233,441,96,96,73329,0,1								
2024/08/27 13:53:54.131	76337	Petición HTTP	500	Internal Server Error	Thread Group	1-41, text, false	,233,441,95,95,76337,0,1								
2024/08/27 13:53:54.095	76430	Petición HTTP	200	OK	Thread Group	1-37, text, true	,438,441,94,94,76430,0,1								
2024/08/27 13:53:53.821	76709	Petición HTTP	500	Internal Server Error	Thread Group	1-13, text, false	,233,441,93,93,76709,0,1								
2024/08/27 13:53:54.122	93396	Petición HTTP	500	Internal Server Error	Thread Group	1-40, text, false	,233,441,92,92,93396,0,0								
2024/08/27 13:53:54.075	98926	Petición HTTP	500	Internal Server Error	Thread Group	1-35, text, false	,233,441,91,91,98926,0,1								
2024/08/27 13:53:54.112	111901	Petición HTTP	200	OK	Thread Group	1-39, text, true	,438,441,90,90,111901,0,1								
2024/08/27 13:53:54.217	118148	Petición HTTP	500	Internal Server Error	Thread Group	1-50, text, false	,233,441,89,89,118148,0,1								
2024/08/27 13:53:53.979	146185	Petición HTTP	500	Internal Server Error	Thread Group	1-26, text, false	,233,441,88,88,146185,0,1								
2024/08/27 13:53:54.437	145673	Petición HTTP	500	Internal Server Error	Thread Group	1-73, text, false	,363,441,87,87,145673,0,1								
2024/08/27 13:53:54.158	145973	Petición HTTP	500	Internal Server Error	Thread Group	1-43, text, false	,233,441,86,86,145973,0,1								
2024/08/27 13:53:53.885	178138	Petición HTTP	500	Internal Server Error	Thread Group	1-17, text, false	,233,441,85,85,178138,0,1								
2024/08/27 13:53:53.692	178331	Petición HTTP	500	Internal Server Error	Thread Group	1-1, text, false	,233,441,84,84,178331,0,3								
2024/08/27 13:53:53.954	178878	Petición HTTP	500	Internal Server Error	Thread Group	1-23, text, false	,233,441,83,83,178878,0,1								
2024/08/27 13:53:54.475	177551	Petición HTTP	500	Internal Server Error	Thread Group	1-77, text, false	,233,441,82,82,177550,0,2								
2024/08/27 13:53:54.554	194483	Petición HTTP	500	Internal Server Error	Thread Group	1-85, text, false	,233,441,81,81,194483,0,1								
2024/08/27 13:53:54.293	206552	Petición HTTP	500	Internal Server Error	Thread Group	1-58, text, false	,363,441,80,80,206552,0,2								
2024/08/27 13:53:54.667	211765	Petición HTTP	500	Internal Server Error	Thread Group	1-97, text, false	,233,441,79,79,211765,0,1								
2024/08/27 13:53:54.053	213398	Petición HTTP	200	OK	Thread Group	1-33, text, true	,13090337,441,78,78,98034,0,2								
2024/08/27 13:53:54.544	228789	Petición HTTP	500	Internal Server Error	Thread Group	1-84, text, false	,233,441,77,77,228789,0,2								

Figura 46. Archivo CSV generado de la ejecución del plan de pruebas.

A partir de la información obtenida en el archivo CSV se obtienen las métricas que dan respuesta a la calidad de entrega de SWA.

1. **Tiempo de Respuesta.** Es el tiempo en segundos que tarda en responder el grupo de solicitudes.
2. **Throughput.** Cantidad de bytes recibidos de solicitudes respondidas exitosamente
3. **Tasa de Éxito.** Número de solicitudes exitosas con RA entregado completo.

8.1.4 Completez de una Respuesta de SWA.

Se determina a una solicitud exitosa a aquella solicitud que es respondida con el código de respuesta HTTP 200 y que entrega el recurso completo. En la Figura 47 se muestra el ejemplo del archivo CSV generado por el cliente Apache JMeter como resultado de la ejecución de la prueba de entregabilidad de SWA. En el recuadro rojo se ilustra que el código de respuesta HTTP es 200, mientras que la columna bytes, que indica la cantidad de bytes recibidas como respuesta a una solicitud de SWA en el recuadro amarillo, el resultado es de 438 bytes. Este ejemplo es de una solicitud de SWA no exitosa, aunque el código HTTP de respuesta es 200, se considera que el recurso del SWA no es entregado completo. Por otro lado, las solicitudes que se consideran como exitosas en la evaluación de la entregabilidad de SWA vienen señaladas en el recuadro azul, con el código de respuesta HTTP es 200 y en el recuadro de color verde se observa que la cantidad de bytes recibidos es de 13,090,337 bytes.

Anexo B. Especificaciones de Entrada y Salida

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
	timeStamp	elapsed	label	responseCod	responseMet	threadName	dataType	success	failureMe	res bytes	sen bytes	grpThreads	allThreads	Latency	IdleTime	Connect
1	53:53.8	46967	Peticiv2n	200	OK	Thread Group	text	true		438	441	100	100	46967	0	2
2	53:54.1	53335	Peticiv2n	200	OK	Thread Group	text	true		438	441	99	99	53335	0	0
3	53:53.7	53691	Peticiv2n	200	OK	Thread Group	text	true		438	441	99	99	53691	0	1
4	53:54.1	53359	Peticiv2n	200	OK	Thread Group	text	true		438	441	97	97	53359	0	1
5	53:54.1	73329	Peticiv2n	500	Internal Serv	Thread Group	text	false		233	441	96	96	73329	0	1
6	53:54.0	178070	Peticiv2n	500	Internal Serv	Thread Group	text	false		233	441	83	83	178070	0	1
7	53:54.5	177551	Peticiv2n	500	Internal Serv	Thread Group	text	false		233	441	82	82	177550	0	2
8	53:54.6	194483	Peticiv2n	500	Internal Serv	Thread Group	text	false		233	441	81	81	194483	0	1
9	53:54.3	206552	Peticiv2n	500	Internal Serv	Thread Group	text	false		363	441	80	80	206552	0	2
10	53:54.7	211765	Peticiv2n	500	Internal Serv	Thread Group	text	false		233	441	79	79	211765	0	1
11	53:54.1	213398	Peticiv2n	200	OK	Thread Group	text	true		13090337	441	78	78	79834	0	2
12	53:53.9	228790	Peticiv2n	500	Internal Serv	Thread Group	text	false		233	441	77	77	228789	0	2
13	53:53.9	234490	Peticiv2n	200	OK	Thread Group	text	true		13090337	441	76	76	79983	0	2
14	53:54.5	239955	Peticiv2n	500	Internal Serv	Thread Group	text	false		363	441	75	75	239955	0	1
15	53:54.2	244160	Peticiv2n	200	OK	Thread Group	text	true		13090337	441	74	74	79715	0	1
16	53:54.3	244134	Peticiv2n	200	OK	Thread Group	text	true		13090337	441	73	73	111658	0	2

Figura 47. Ejemplo de respuestas a solicitudes de SWA exitosas, completas y no completas.

Anexo C. Configuración del equipo de cómputo

9.1 Configuración de Equipo de Cómputo

En la ejecución de las pruebas se utilizaron tres equipos configurados como servidores, en las tablas 38, 39 y 40 se muestran los datos de cada una de estos, estos fueron definidos como servicios, Ingeniería de Software 1 (IS1) e Ingeniería de Software 2 (IS2), tablas 41 y 42. A su vez, como dispositivos clientes se utilizaron dos equipos, el primero como OS Linux Ubuntu y otro con OS Windows 10, en los cuales se desplegaron las aplicaciones cliente para realizar las pruebas.

Tabla 57. Descripción del Servidor definido como Servicios.

Dispositivo	Descripción
Nombre	Servicios
Sistema	Debian GNU/Linux 11
CPU	Server Dell PowerEdge 2900
Procesador	Intel Xeon 5110 1596 MHz 2x Dual Core 64 bits 8MB cache L2
Tarjeta Gráfica	AMD ES1000
Tarjeta de Red	Dell Gigabit Ethernet Broadcom NetXtreme II BCM5708 Gigabit Ethernet Gigabit Ethernet Realtek RTL8111/8168/8411 PCI Express
Disco Duro	Kingston SA400S3 1TB

Tabla 58. Descripción del Servidor definido como IS1.

Dispositivo	Descripción
Nombre	Ingeniería de Software 1
Sistema	Debian GNU/Linux 11
CPU	HP ProLiant DL160 Gen9
Procesador	Intel Xeon E5-2603 v3 1.60GHz 15 MB cache L3
Tarjeta Gráfica	X
Tarjeta de Red	
Disco Duro	120 GB
Virtual Machine 1	IS1
RAM	6 GB
Núcleos	6 vCPUS
Disco Duro	100 GB

Tabla 59. Descripción del Servidor definido como IS2.

Dispositivo	Descripción
Nombre	Ingeniería de Software 1
Sistema	Debian GNU/Linux 11
CPU	HP ProLiant DL160 Gen9
Procesador	Intel Xeon E5-2603 v3 1.60GHz 15 MB cache L3
Tarjeta Gráfica	X
Tarjeta de Red	
Disco Duro	
Virtual Machine 1	IS1
RAM	6 GB
Núcleos	6 vCPUS

Anexo C. Configuración del Equipo de Cómputo

Disco Duro	120 GB
------------	--------

Tabla 60. Descripción del Cliente definido como Linux.

Dispositivo	Descripción
Nombre	Ingeniería de Software 2
Sistema	Sistema Operativo Ubuntu/Linux 20
CPU	HP Touch Smart 600
Procesador	Intel Core 2 Duo a 2.0 GHZ
Tarjeta Gráfica	X
Tarjeta de Red	Ethernet Gigabit
Disco Duro	HDD 120 GB
RAM	4 GB

Tabla 61. Descripción del Cliente definido como Windows.

Dispositivo	Descripción
Nombre	Ingeniería de Software 2
Sistema	Sistema Operativo Windows 10
CPU	HP Touch Smart 610
Procesador	Core i5 a 2.8 GHZ
Tarjeta Gráfica	X
Tarjeta de Red	Ethernet Gigabit
Disco Duro	HDD 240 GB
RAM	8 GB

Anexo D.

Escenarios de Prueba

En esta sección se presentan los escenarios de prueba para entrega de SWA, estos se definieron de acuerdo a los atributos de SWA encontrados en la búsqueda. En esta se definieron dos categorías de escenarios de prueba, Standalone y Compuestos. En la primer categoría se utiliza una arquitectura cliente servidor, sin interacción con un tercer dispositivo. En la categoría de escenarios compuestos, interviene un tercer dispositivo.

10.1 Escenarios Standalone para Caso de Pruebas de Entregabilidad de SWA

En el escenario definido en este trabajo como standalone, se implementaron y se desplegaron los artefactos respectivos para cada caso de pruebas. El diagrama de despliegue que se ilustra en la Figura 48 con dos dispositivos que se comunican a través del protocolo HTTP, en donde los SWA se exponen con el servidor de aplicaciones Wildfly, a través del puerto 8081. En el dispositivo Cliente 1 se desplegó la aplicación cliente Apache JMeter y el dispositivo Servicios funcionó como servidor en el que se desplegaron los SWA. Este escenario de prueba aplica para los siguientes casos de prueba:

- ◆ S-SM1
- ◆ S-TSW1
- ◆ S-JHS1
- ◆ S-GCA5
- ◆ S-G1R1
- ◆ S-G1R2
- ◆ S-G1P3
- ◆ S-G1DD1
- ◆ S-G1CS1
- ◆ S-JPS1

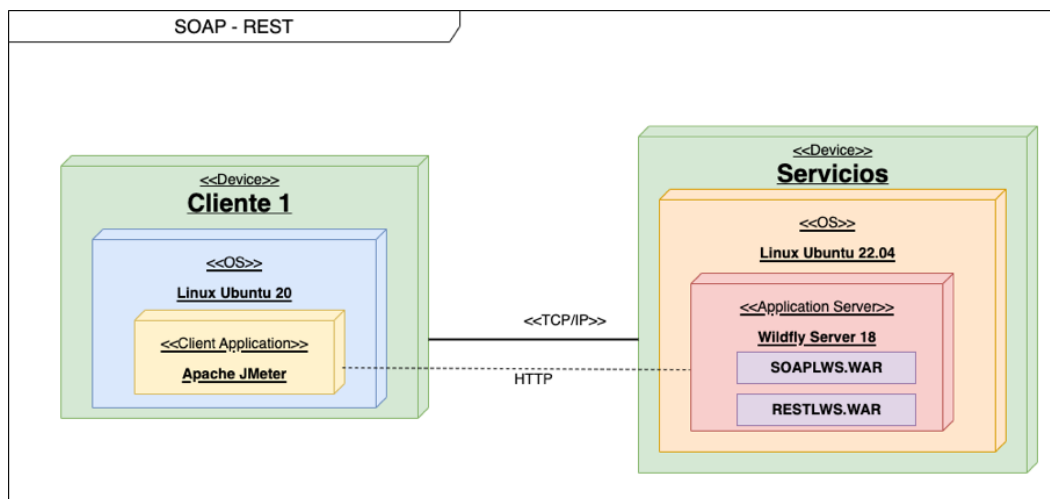


Figura 48. Diagrama de despliegue para los casos de prueba standalone.

10.1.1 Escenario del Caso de Prueba S-TPC1

Para realizar las pruebas comparativas entre HTTP y HTTPS, se definió un escenario de pruebas en donde se implementaron y desplegaron los artefactos que contienen los SWA. El diagrama de despliegue que se ilustra en la Figura 49 se observa dos dispositivos que se comunican a través del protocolo de comunicación, en donde los SWA se exponen con el Application Server Wildfly, a través del puerto 8081. El dispositivo Cliente 1 se desplego la aplicación cliente Apache JMeter y en el dispositivo Servicios se implementó el servidor de aplicaciones Wildfly en el cual se desplegaron los SWA, que están contenidos en los artefactos SOAPLWS-1.WAR y RESTLWS-1.WAR. Con la finalidad de habilitar la comunicación mediante el protocolo HTTPS y se generó un certificado mediante Java Keytools en el servidor Wildfly y se importó en la aplicación cliente Apache JMeter. En contraste, con el protocolo HTTP, dicho certificado no es necesario.

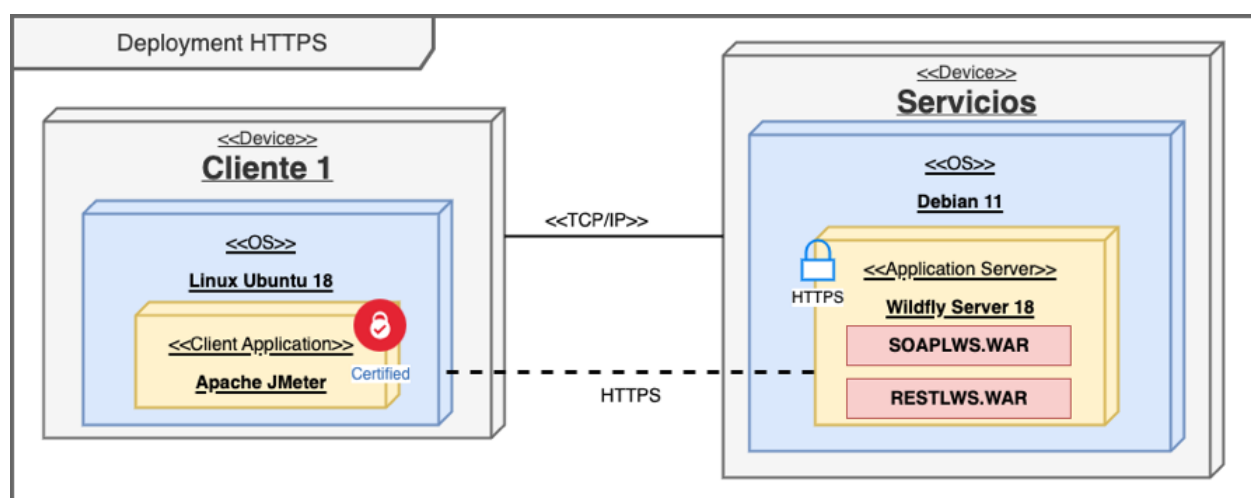


Figura 49. Diagrama de despliegue para pruebas entre HTTP y HTTPS.

10.1.2 Escenario del Caso de Pruebas S-JSP1

El escenario del caso de pruebas S-JSP1 se presenta una arquitectura cliente-servidor, en el que el servidor Servicios es el encargado de responder las solicitudes al SWA. El artefacto WAR desplegado despliega un SWA en el que se implementa el patrón de diseño Singleton con EJB. Este patrón de diseño mantiene una instancia de una clase lista accesible para responder a una solicitud de servicio para múltiples clientes. La Figura 50 ilustra el esquema de clases de esta implementación, en la que la clase **Producer** genera una instancia de la clase **LearningResource** y esta clase es instanciada en la clase **LWSService**.

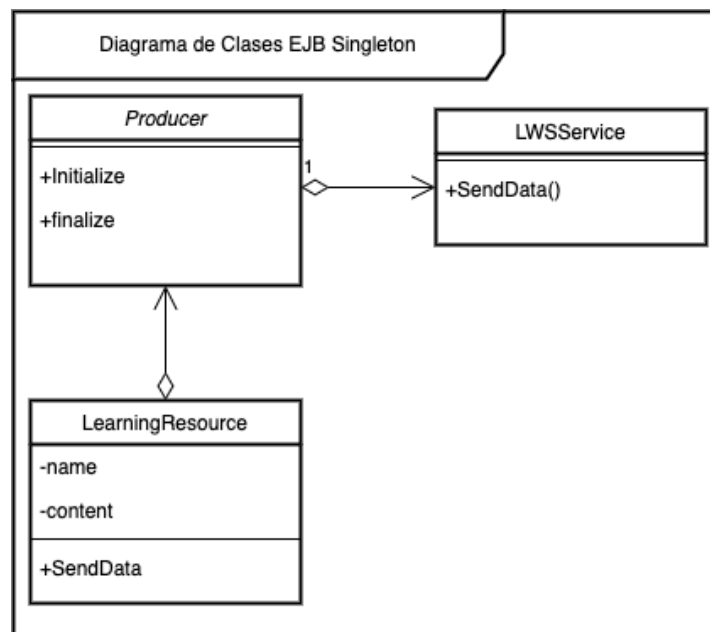


Figura 50. Diagrama de clases del SWA con la implementación del patrón Singleton.

El funcionamiento del patrón **Singleton** se ilustra en la Figura 51, este se inicia en el despliegue del artefacto WAR que contiene el SWA, se inicia el EJB, se inyecta la dependencia del **Producer** al servicio, en el **Producer** posterior a la construcción, se genera una instancia de la clase **LearningResource** (LR Creation) en el proceso **PostConstruct**, y se mantiene listo para ser llamado. Cuando el artefacto es removido (Remove) del servidor de aplicaciones en el proceso **PreDestroy**, el EJB es eliminado.

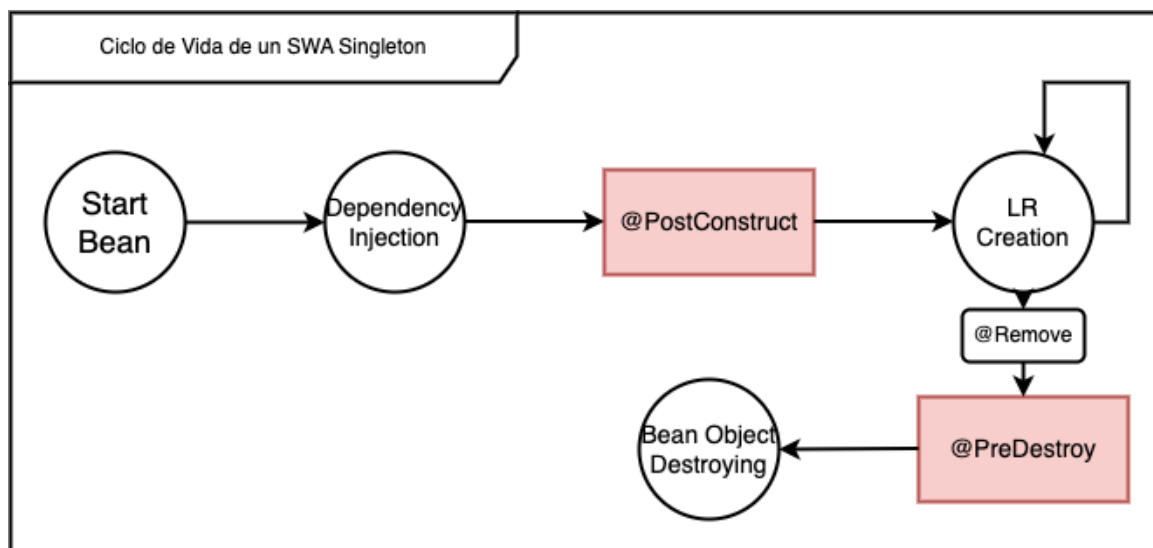


Figura 51. Funcionamiento del Patrón de Diseño Singleton con EJB en SWA.

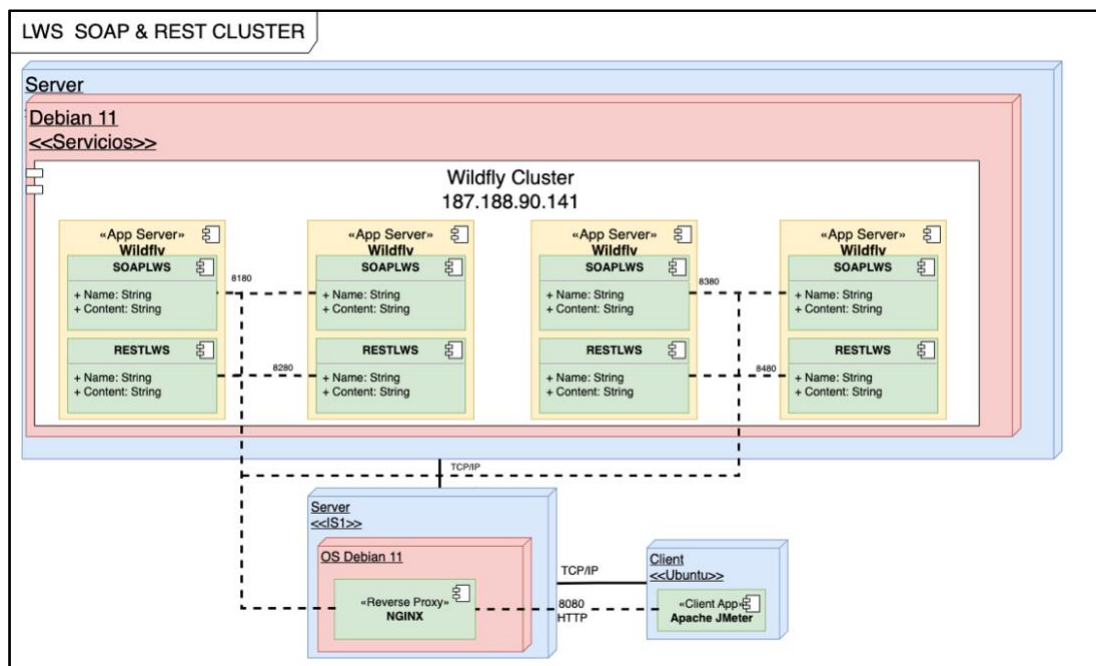
10.2 Escenarios Compuestos para Casos de Pruebas de Entregabilidad de SWA

A continuación se describen los escenarios definidos como compuestos, en el cual se implementaron plataformas y herramientas que requiere una arquitectura más compleja que solo la arquitectura cliente servidor, en el que es indispensable muchas veces un balanceador de carga o un dispositivo intermedio entre el cliente y el servidor.

10.2.1 Escenario del Caso de Prueba C-C4

El escenario definido para pruebas de entregabilidad con un servidor de aplicaciones clusterizado se implementó en el servidor Servicios, y el servidor de aplicaciones Wildfly se configuró con cuatro nodos, es importante mencionar que cada nodo replica todos los SWA desplegados en el nodo Host. En este escenario todos los servicios son accedidos a través de una misma dirección IP, lo que cambia es el cada nodo del servidor de aplicaciones Wildfly responde por puerto el puerto configurado para cada nodo, por lo que se configuro un balanceador de carga que distribuye las solicitudes a cada uno de los nodos. Los puertos habilitados se listan en la Tabla 62, en donde se listan los puertos utilizados para cada instancia.

Para el funcionamiento de esta arquitectura, se habilitó un balanceador de carga en el servidor IS1 con NGINX, el cual responde en el puerto 8080 y distribuye la carga en los nodos o instancias del servidor Servicios. En el cliente se configuró el plan de pruebas en Apache JMeter 5.5 La Figura 52 ilustra la configuración de la arquitectura de clusterización del servidor de aplicaciones desplegado en el servidor Servicios, el servidor IS1 configurado como balanceador de carga y el cliente que envía las solicitudes a los SWA a través de Apache JMeter.



Anexo D. Escenarios de Prueba

Figura 52. Diagrama de despliegue para pruebas de clusterización.

Tabla 62. Puertos habilitados para el clúster de Wildfly en IS2.

Instancia	Puerto
Instancia 1	8180
Instancia 2	8280
Instancia 3	8380
Instancia 4	8480

10.2.2 Escenario del Caso de Pruebas C-CD4

En las pruebas de contenerización de aplicaciones de servicios web se utilizó la aplicación Docker para contenerizar el servidor de aplicaciones Wildfly. Se configuró un Docker Pod con 4 nodos en el servidor Servicios, un balanceador de carga en el servidor IS1 y el cliente. En este caso el servidor responde por la misma dirección IP y por diferente puerto cada nodo del grupo de contenedores Docker o Pod, la lista de puertos se presenta en la Tabla 63.

El balanceador de carga configurado en el servidor IS1 con NGINX, recibe las solicitudes a través del puerto 8080 y distribuye la carga en los nodos configurados en el Pod de Docker en el servidor Servicios. En el cliente se configuró el plan de pruebas en Apache JMeter. La Figura 53 ilustra la configuración implementada para la contenerización del servidor de aplicaciones con cuatro nodos, un servidor que actúa como balanceador de carga y el cliente que despliega la aplicación cliente Apache JMeter.

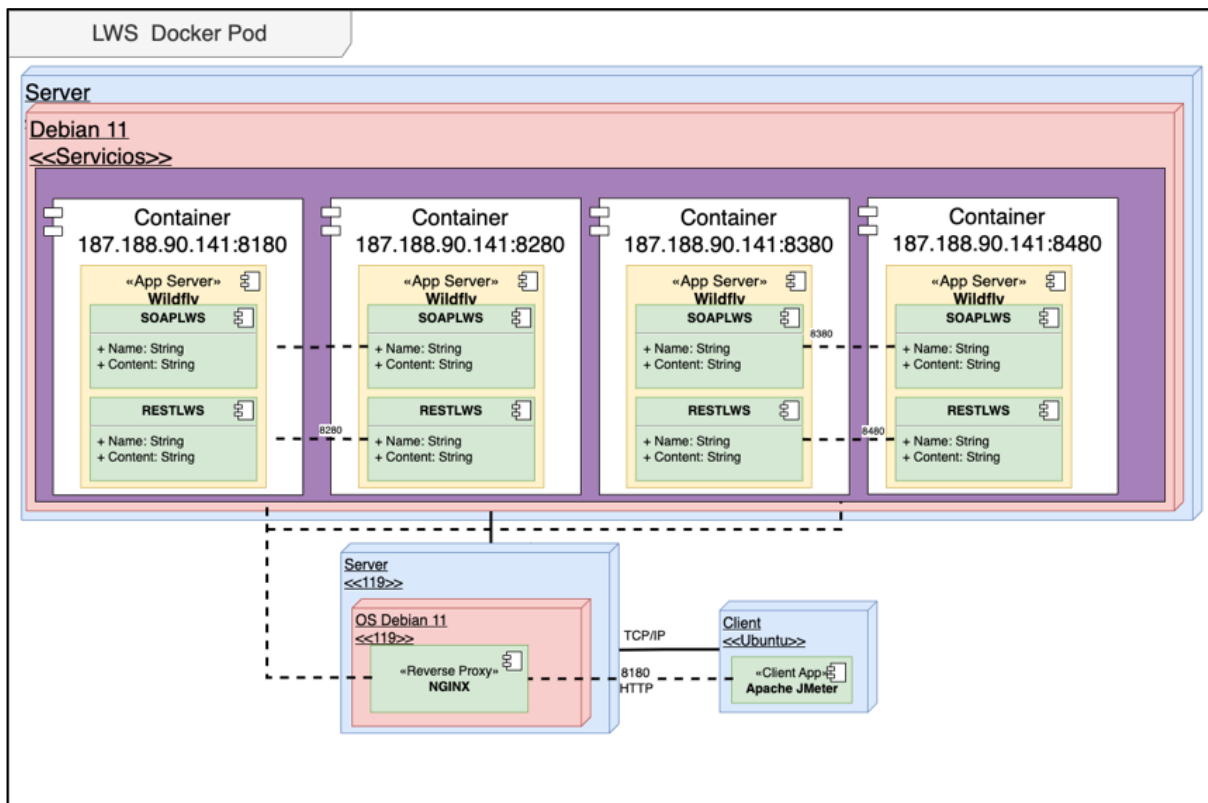


Figura 53. Diagrama de despliegue para pruebas de Contenerización.

Tabla 63. Puertos habilitados para la arquitectura contenerizada de Wildfly en IS2.

Nodo	Puerto
Nodo 1	8180
Nodo 2	8280
Nodo 3	8380
Nodo 4	8480

10.2.3 Escenario del Caso de Pruebas C-SC1

El escenario para el caso de pruebas C-SC1, se configuró una arquitectura computacional que se compone de dos servidores, el primer servidor es un proxy cache de 10 GB con NGINX desplegado en el servidor IS2. El segundo servidor se utilizó el dispositivo definido como Servicios , en el que se desplegaron los artefactos que contienen a los SWA en SOAP y en REST. El tiempo configurado para la caché se estimó en 5 minutos, es decir, el tiempo de vida de la respuesta en la cache del servidor, posteriormente se elimina dicha respuesta y a la siguiente solicitud se vuelve a crear la respuesta en cache por otros 5 minutos. La Figura 54 se ilustra la arquitectura para pruebas de Server Cache, el servidor Servicios con dos artefactos WAR, además del el servidor IS2 con el componente Cache 1 y la Figura 55 ilustra la interacción entre el cliente y el servidor cuando se implementa la arquitectura *Server Cache* .

La interacción se inicia cuando el cliente envía la solicitud a un SWA, el servidor *proxy caché* verifica la caché, en caso de que la respuesta esta almacenada, envía dicha respuesta; si no está cacheada, re envía la solicitud al servidor en donde se ubica el SWA, obtiene la respuesta y cachea los datos de respuesta, por último, envía la respuesta al cliente. La respuesta continua en memoria cache, de acuerdo al tiempo que se haya configurado en el servidor *proxy caché*.

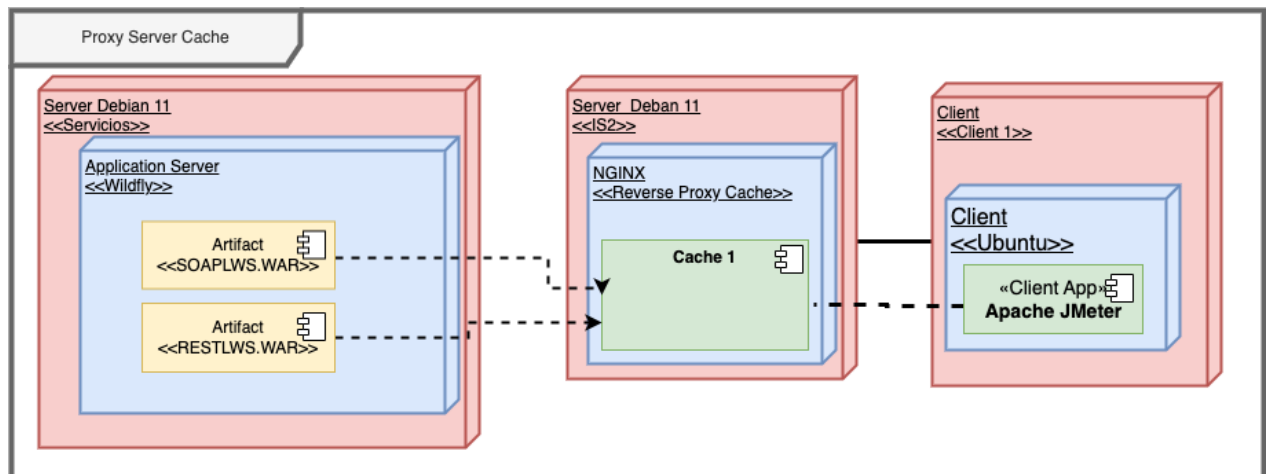


Figura 54. Diagrama de despliegue para pruebas de Server Cache .

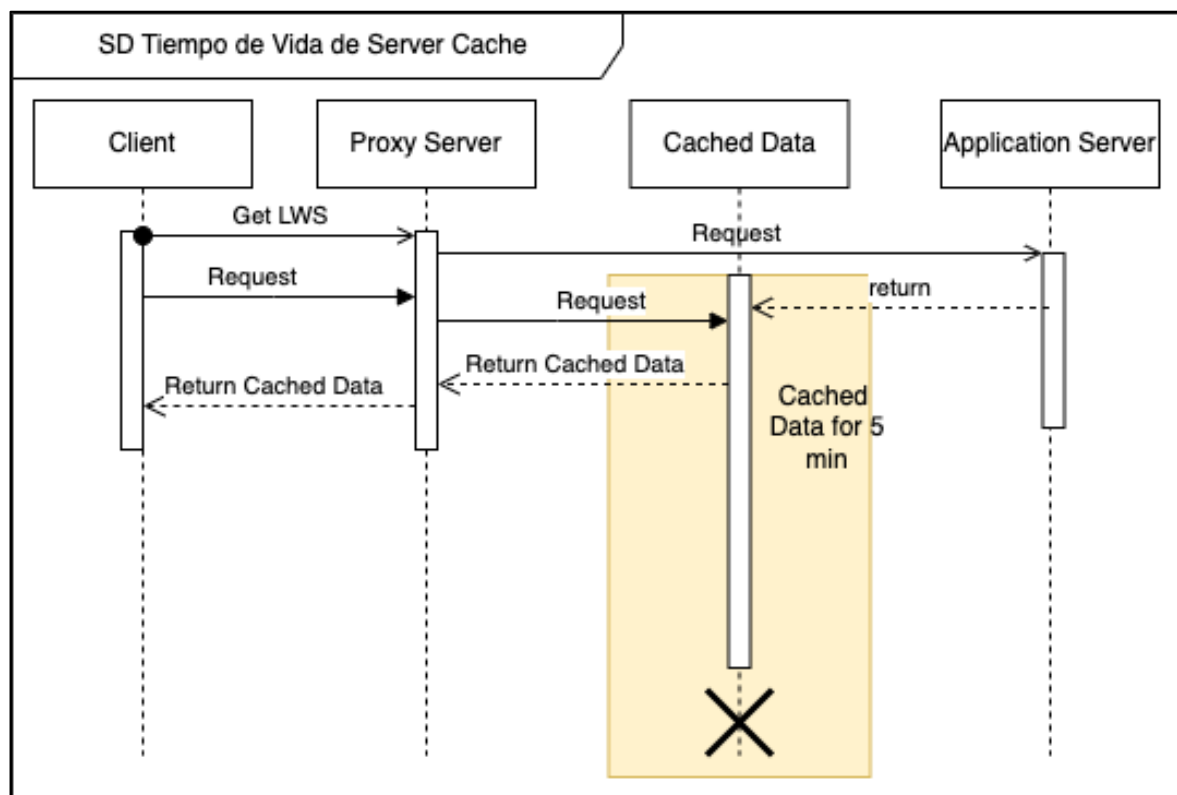


Figura 55. Diagrama de secuencia del tiempo de vida de Server Cache.

10.2.4 Escenario del Caso de Pruebas C-TF2

El escenario del caso de pruebas C-TF2 o un escenario tolerante a fallas para entregar SWA, se definió una arquitectura computacional compuesto de dos servidores, un servidor proxy con NGINX desplegado en el servidor IS2, encargado de controlar las solicitudes y redirigir dichas solicitudes si el servidor principal no puede atender estas solicitudes en un tiempo de 2 segundos. El servidor definido como Servicios se configuró como servidor principal, encargado de responder las solicitudes a SWA y el servidor IS1, en el que se replicaron los SWA del servidor principal, atiende las solicitudes que el servidor Servicios no puede responder. La Figura 56 ilustra la arquitectura computacional tolerante a fallas, con dos servidores que despliegan los servicios y uno que redirige las solicitudes a SWA y controla el tiempo de respuesta. El diagrama de secuencia de la Figura 57 muestra la interacción entre la arquitectura tolerante a fallas y el cliente, en el que en caso de que el servidor Servicios no pueda atender la solicitud en un plazo determinado de 2 segundos, la solicitud es redirigida por parte del Proxy Server al servidor IS1.

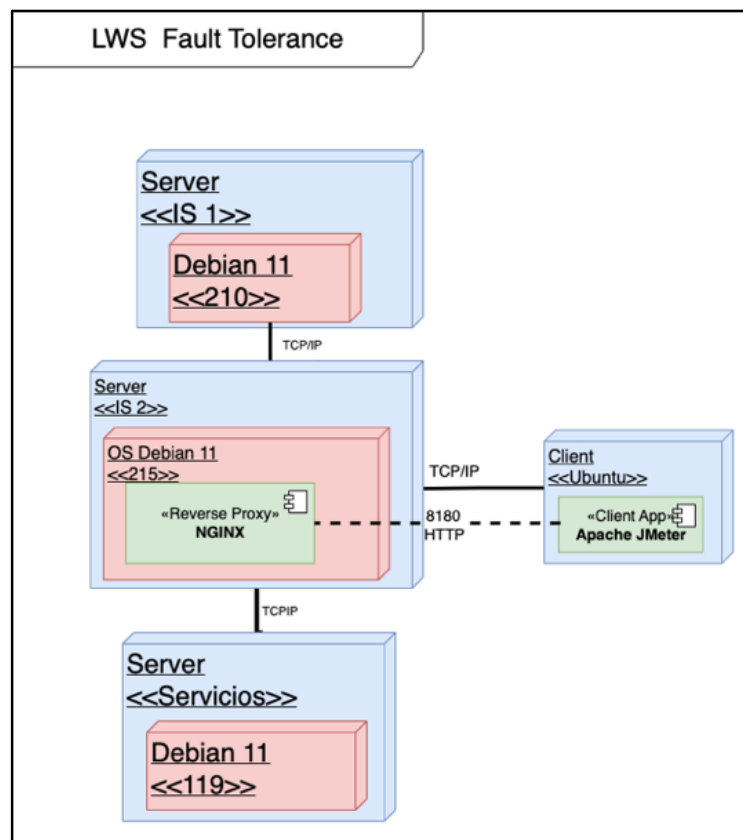


Figura 56. Diagrama de despliegue para pruebas de Entregabilidad de SWA con Tolerancia a Fallas.

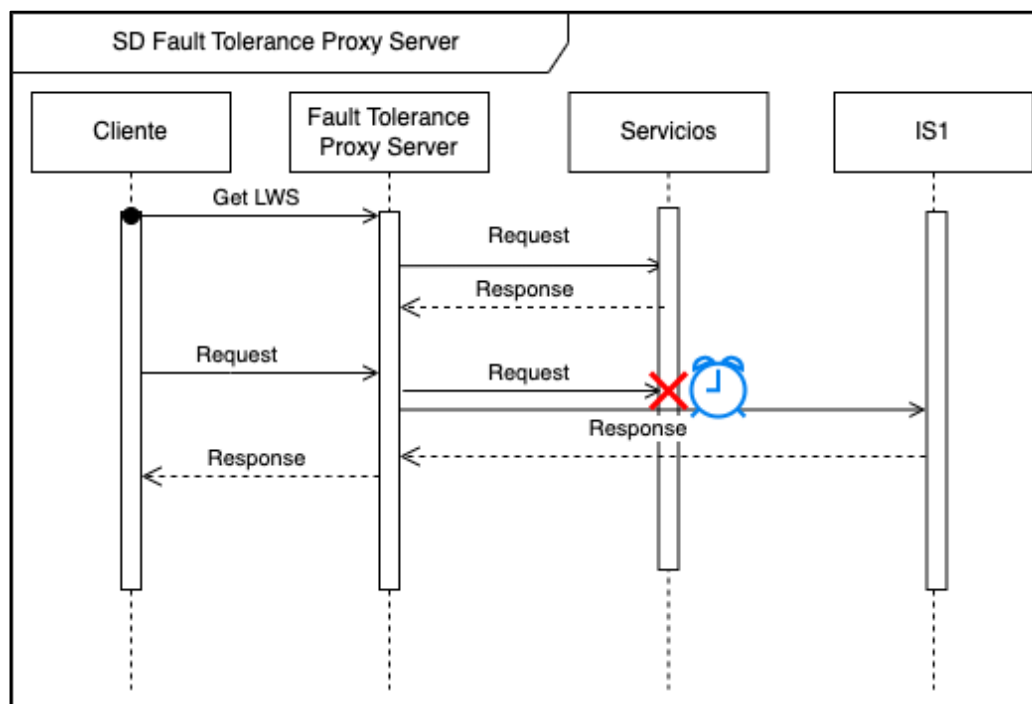


Figura 57. Secuencia de interacción del servidor de Tolerancia a Fallas.

Anexo E. Granularidad de Servicios Web de Aprendizaje

11.1 Granularidad de los SWA

La granularidad de los RA para propósito de este trabajo de investigación se establece en dos perspectivas:

- ♦ **Tamaño del Recurso.** En el trabajo antecedente de [Fuentes, 2019], se propuso la segmentación o división de RA de manera que permitan manipular el recurso en términos computacionales (memoria, procesamiento, latencia, ancho de banda).
- ♦ **Segmentación del RA.** Un RA puede ser dividido en segmentos, partiendo de la perspectiva de la usabilidad del recurso para manipular y visualizar de una manera más fácil y sencilla de acuerdo con los temas, capítulos, objetivos, etc. La segmentación debe ser considerada por el creador de contenido para que pueda ser dividida a partir de los diferentes temas, capítulos, objetivos, etc., de esta manera ayuda a la usabilidad del usuario a desplegar el contenido, a visualizarlo y a manipular el recurso de una mejor manera. De esta forma, el servidor puede procesar la respuesta más rápida, utilizando menos recursos computacionales.

11.1.1 Formato de los Recursos de Aprendizaje

Es importante establecer los tipos de RA que se deben utilizar en un marco cliente para que puedan ser visualizadas y utilizadas para el aprendizaje. Por ejemplo en un RA de tipo audio es muy común la utilización del formato MP3, sin embargo, también existe el formato WMA. Este último formato no se podría utilizar en un equipo cliente con un OS diferente a Windows, a menos que previamente se instalen los códecs adecuados que permitan su reproducción. En WORC, se definen algunos tipos de RA, por ejemplo evaluaciones, animaciones, simulaciones, casos de estudio, ejercicios y prácticas, plantillas. Por otro lado, en el diseño instruccional proponen diferentes tipos de RA como: texto, imágenes, diapositivas, video, audio, páginas Web, entre otros. Por lo que es importante unificar los formatos de los RA para que los clientes cuenten con los recursos computacionales para poder visualizarlos y reproducirlos.

11.1.2 Resolución de Recursos de Aprendizaje

Otra perspectiva de la granularidad de los RA, es la resolución del formato de video, es decir, definir los formatos de salida adecuados para los diferentes dispositivos que se pueden utilizar para esta finalidad. Por ejemplo, en el caso del formato para dispositivos celulares o móviles, que sin problema se puede visualizar un formato de 720p, a diferencia de una computadora que puede reproducir formatos más altos, como el 1080p o 1920p. Otro claro ejemplo es el objetivo del RA, tal es el caso de una clase de un idioma extranjero, la calidad de un RA de tipo audio puede interferir en el aprendizaje, sin embargo, queda a responsabilidad del creador de contenido de aprendizaje definir estos detalles.

11.1.3 Granularidad de Videos

De acuerdo con [Google Inc., 2020], la plataforma de videos YouTube transcodifica automáticamente su video para crear muchos formatos de salida diferentes para que todos sus espectadores en muchos dispositivos y redes puedan visualizar. Los formatos y la resolución definen el tamaño del RA, por lo que para los formatos de tipo video quedan definidos como granularidad por tamaño como se plantea en la Tabla 64. En color gris se muestran las recomendaciones para archivos de video que puedan ser utilizados en ambientes de aprendizaje.

Tabla 64. Resoluciones de archivos de video de acuerdo a Google Inc. (2020).

Resolución	Nombre	Calidad
3840 x 2160	2160p	4K
2560 x 1440	1440p	2K
1920 x 1080	1080p	Resolución máxima Full HD
1280 x 720	720p	Resolución mínima HD
854 x 480	480p	Definición estándar
640 x 360	360p	Resolución normal de website
426 x 240	240p	Tamaño mínimo de vídeo en YouTube

El tamaño o volumen de un video es calculado a partir de sus características como la resolución, número de canales de grabación, número de frames por segundo y el tiempo de duración del video, también se puede calcular a partir del Bitrate del video y el tiempo de duración del video. La plataforma de video streaming YouTube utiliza una tasa de compresión de 8MBPS, por lo que un objeto de aprendizaje de tipo video de un minuto de duración con las mínimas especificaciones tiene un volumen de 3.5 MB. En el portal de [Queensland University, n.d.], se hace un análisis con las recomendaciones para crear videos con diferentes tipos de formato de acuerdo al volumen. En el portal de [CircleHD, 2020] se presenta un análisis del tamaño de archivos de video de acuerdo a la resolución, presenta un conjunto de resultados de archivos de video de 1 minuto de duración que se muestran en la Tabla 65.

Tabla 65. Volumen de archivos de video de 1 minuto de acuerdo a la resolución CircleHD (2020).

Nombre	Bit Rate	Tamaño de 1 min
4K(UHD)	20 MBPS	84 MB
1080p	5 MBPS	20 MB
720p	5 MBPS	5 MB
480p	500 KBPS	2 MB

El análisis que de la utilización de videos en medios educativos que presenta el estudio de [Guo et al., 2014], se recomienda utilizar videos de solo 6 minutos de duración debido a que el tiempo de participación promedio es de 6 minutos como máximo, independientemente de la duración total del video. Además, se indica que 6 minutos de duración es el tamaño ideal porque es mayor el porcentaje de estudiantes que se involucran y participan mucho más con videos de 3 minutos y 6 minutos. Los resultados

son parte del experimento de 4 cursos a más de 127 mil estudiantes con 862 videos y más de 6,902,358 sesiones.

11.1.4 Diferencia entre Video y Película

De acuerdo con [Stiller, 2018], una película narra una historia y puede narrativa, documental o experimental. Una historia en una película está formada por varios elementos de imágenes en movimiento, fotografías, gráficos en movimiento, texto, música, diálogo y sonidos de ubicación. Un video es una grabación de un evento, es decir, es simplemente un medio económico de preservar una memoria por ejemplo podría ser una reunión familiar, un partido de voleibol, un momento de orgullo para los padres o un tiempo con amigos.

11.1.5 Granularidad de Audios

De acuerdo con [Guzmán & Mata, 2008], el MP3 es el formato que se ajusta a los requerimientos de la enseñanza, y la calidad de bitrate de 64 y 128 kbps es suficiente para que los mensajes lleguen claramente al estudiante, además, ayudan a mantener una buena relación peso-calidad y puede ser distribuido con más facilidad en medios ópticos y por Internet. En cuanto a la duración del audio como elemento educativo, [Cebeci & Tekdal, 2006], mencionan que los audios deben tener una duración como máximo de 15 minutos para que los estudiantes no pierdan el interés.

11.1.6 Granularidad de Recursos de Aprendizaje de Tipo Texto

Existen recursos de aprendizaje en formato de texto digital que facilitan y apoyan el proceso de enseñanza-aprendizaje. Entre estos podemos encontrar diferentes tipos de archivos como textos planos, archivos PDF, textos enriquecidos como Power Point, Excel, Word y que pueden ser compartidos entre diferentes plataformas. Entre estos podemos encontrar los siguientes ejemplos:

- ◆ Manuales o guías didácticas
- ◆ Enciclopedias digitales
- ◆ Libros electrónicos (e-books)
- ◆ Informes técnicos
- ◆ Tesis
- ◆ Blogs educativos
- ◆ Publicaciones en línea (boletines, newsletters)
- ◆ Documentos PDF especializados
- ◆ Transcripciones de clases o conferencias
- ◆ Presentaciones con contenido textual (diapositivas)

En [Muñoz Arteaga et al., 2006], los autores plantean que los Objetos de Aprendizaje (OA) debe estar encaminado en cuatro ejes principales: Actividad, Objetivo, Evaluación y Contenido. En este sentido, un recurso de aprendizaje debe estar enfocado en una atender un solo eje de los 4 que se plantean, es decir, un recurso debe realizar una de los 4

elementos, por ejemplo contener solo una actividad escolar, una evaluación, definir un objetivo o contener el programa del OA.

Lo que se define en este trabajo es que un libro extenso no puede atender una actividad para propósitos de enseñanza, más bien se plantea que, el libro puede dividirse en uno o más capítulos para poder atender sesiones en instituciones educativas de nivel básico de 30 a 60 minutos y de 60 a 90 minutos en instituciones educativas de nivel medio superior y superior, como se propone en [Bligh, 2000].

Por otro lado, los archivos de presentaciones con diapositivas, deben acotar solo lo significativo para atender sesiones de treinta a cincuenta minutos, no extenderse para no aburrir al estudiante o del público en general y evitar la fatiga cognitiva. Lo que se recomienda en la utilización de este tipo de archivos es evitar incrustar otro tipo de RA como videos, imágenes de gran tamaño, y otros, para evitar el consumo excesivo de recursos, que la aplicación Power Point se vuelva lenta o que falle, además de que el volumen de este archivo de presentación se incremente.

Existe una recomendación por parte de [Harden, 2008], en donde se menciona que una presentación de PowerPoint debe tener diez diapositivas, durar no más de veinte minutos y no contener una fuente de menos de treinta puntos. De acuerdo a estas recomendaciones, los RA de tipo Power Point no deberían de rebasar los 5 MB, siempre y cuando no se integren RA (videos, audios e imágenes) de gran tamaño dentro de dichas presentaciones.

Para determinar el tamaño de los RA de tipo texto es complicado, esto es determinado a nivel sistema operativo en el cual se determina que un documento de texto de 50 páginas puede medir entre 50 a 200KB, dependiendo si presenta algún tipo de estilo, tabla y fuentes incrustadas, eso por mencionar algunas características.

11.1.7 Granularidad de Recursos de Aprendizaje de Tipo Imagen

De acuerdo con [Gardner, 1983], existen 8 diferentes tipos de inteligencias, una de ellas es la inteligencia espacial, la cual indica que es una habilidad para interpretar, comprender y expresar las imágenes visuales y espaciales. Las imágenes pueden ser utilizadas para la enseñanza de conceptos funcionales, además de que facilitan la enseñanza escrita [Cámara, 2000]. En el estudio de [Crozat et al., 1999], los autores mencionan que la representación visual deberá ser simple, ir directamente a lo esencial y no ser ambigua, además de que los cuadros didácticos serán coherentes entre sí y el objeto de la comunicación debe ser claro e identificable.

En ambientes computacionales existen diferentes formatos que pueden ser utilizados para visualizar imágenes, cada formato utiliza un algoritmo de compresión y maneja su propio enfoque, con el objetivo de disminuir el tamaño y de mantener la máxima calidad posible. En la tabla 3 se listan los formatos identificados que pueden ser utilizados en medios educativos, y de acuerdo el formato, el tipo de compresión para reducir el tamaño y las características del formato de acuerdo al tamaño.

Tabla 66. Formato de imagen con características de compresión.

Formato	Tipo de compresión	Características
BMP	Sin compresión	Cada píxel se almacena sin reducción.
TIFF (sin compresión)	Sin compresión	Alta calidad, usado en escáneres o impresión.
TIFF (LZW)	Sin pérdida	Compresión eficiente pero sin perder datos.
PNG	Sin pérdida	Ideal para gráficos o transparencia.
JPEG (alta calidad)	Con pérdida	Compromiso entre calidad y tamaño.
JPEG (calidad media)	Con pérdida	Muy usado en web con pérdida visual mínima.
GIF (256 colores)	Sin pérdida (limitado)	Solo 8 bits de color, útil para dibujos o animaciones.

El número de bytes necesarios para almacenar la información de color de cada píxel depende de la profundidad de color de la imagen. La profundidad de color especifica el número de bits que se utilizan para representar cada canal de color (rojo, verde y azul) en un píxel. Por ejemplo, una profundidad de color de 8 bits permite 256 valores posibles para cada canal de color, mientras que una profundidad de color de 24 bits permite 16,7 millones de valores posibles para cada canal de color.

En la Tabla 67 se listan los tipos de profundidad, el tipo de imagen y el tamaño aproximado en bytes de una imagen de 1000x800 píxeles. Esta tabla muestra el incremento del tamaño o volumen computacional considerando los colores, pasando de 1 bit que ocuparía un tamaño aproximado de 100KB a utilizar 32 bits, este último con un incremento de más de 3,000 por ciento, con respecto del primero.

Tabla 67. Tamaño de imagen de acuerdo a la profundidad.

Profundidad	Tipo de imagen	Tamaño	Tamaño aprox
1 bit	Blanco y negro	1000*800*1 / 8	100 KB
8 bits	Escala de grises	1000*800*8 / 8	800 KB
16 bits	Color (High Color)	1000*800*16 / 8	1.6 MB
24 bits	Color verdadero (RGB)	1000*800*24 / 8	2.4 MB
32 bits	Color + canal alfa (RGBA)	1000*800*32 / 8	3.2 MB

Pasando al enfoque de este estudio, cual es el tamaño adecuado para las imágenes en entornos educativos. Respecto al tamaño en bytes de una imagen, este se calcula multiplicando el número de píxeles de la imagen por el número de bytes necesarios para almacenar la información de color de cada píxel. Para determinar el número de píxeles de una imagen, se puede multiplicar el ancho por su alto (en píxeles). Por ejemplo, una imagen de 1000 píxeles de ancho por 800 píxeles de alto contendría un total de 800,000 píxeles.

Para calcular el tamaño en bytes de una imagen, se utiliza la siguiente fórmula:

$$Tb = px * bpp$$

Donde:

px : Es el número de píxeles de la imagen.

bpp : Es el número de bytes por píxel de la resolución de la imagen.

Por ejemplo, con una imagen con una resolución de 1000×800 píxeles y una profundidad de color de 24 bits (8 bits por canal de color), el tamaño en bytes de esta imagen, se utiliza la siguiente fórmula:

- ◆ Tamaño en bytes = $1000 \times 800 \times (24 / 8)$
- ◆ Tamaño en bytes = $1000 \times 800 \times 3$
- ◆ Tamaño en bytes = 2,400,000 bytes
- ◆ Tamaño en bytes = 2.4 megabytes

11.1.8 Definición de Tamaño de Acuerdo a la Granularidad

Después del análisis en diferentes trabajos de investigación, se ha podido establecer una propuesta que permite identificar la granularidad enfocada en el volumen computacional por tipo de RA. En la Tabla 68 se muestran 4 tipos de RA, de tipo Texto en la que se incluyen los archivos de Word, Excel, Power Point, archivos PDF y de texto plano; los archivos de tipo imagen, de audio y de video. De acuerdo con [Gardner, 1983], existen diferentes tipos de aprendizaje y para el aprendizaje no se debe de excluir a algún tipo de recurso, por lo que se incluyen los tipos de RA para ser entregados por un SWA.

Tabla 68. Granularidad de acuerdo al volumen por tipo de RA.

Tipo de RA	Fina	Media	Gruesa
Texto	1 MB	3 MB	5 MB
Imagen	1 MB	3 MB	5 MB
Audio	1 MB	5 MB	10 MB
Video	5 MB	10 MB	15 MB

Anexo F. Análisis Estadístico de los Resultados de Entregabilidad de SWA

12.1 Análisis Estadístico de los Resultados de Entregabilidad de SWA

En esta sección se presenta el análisis estadístico de los resultados de entregabilidad de SWA. Por la naturaleza de las pruebas y al diseño experimental, se realizaron dos metodologías, la primera metodología se basa en determinar las diferencias significativas de pruebas de dos muestras independientes. La segunda metodología se enfocó en evaluar las diferencias significativas de más de dos muestras o grupos independientes.

12.1.1 Normalidad de los Datos

De acuerdo con [Reynaga, 2012], la distribución normal o distribución de Gauss representa la forma en la que se distribuyen en la naturaleza los diversos valores numéricos de las variables continuas, como pueden ser estatura, peso, etc.

En el estudio de [Gorgas et al., 2009], los autores mencionan que la distribución normal describe con gran aproximación la distribución de las variables asociadas con muchos fenómenos de la naturaleza. En particular, las medidas de magnitudes físicas suelen distribuirse según una distribución normal. Por ejemplo, la distribución de alturas de un grupo de población, las medidas de calidad de procesos industriales, o la distribución de temperaturas de una población, se pueden aproximar por distribuciones normales. Además, los errores en las medidas también se aproximan con mucha exactitud a la distribución normal.

Por definición, se dice que una variable aleatoria continua X sigue una distribución normal de media μ y desviación típica σ si su función de densidad como se presenta en la ecuación 11.

$$f(x) = N(\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (11)$$

Gráficamente (Figura 58), la distribución de probabilidad normal tiene forma de campana (llamada campana de Gauss, o curva normal), simétrica (por depender de x a través del término $(x-\mu)^2$), centrada en μ y con anchura proporcional a σ (como es lógico esperar del significado de la desviación típica). Evidentemente, el máximo de la función de densidad ocurre para $x = \mu$ y, por tanto, media, mediana y moda coinciden en ese punto. Se puede demostrar que los puntos de inflexión de la curva normal están situados en $\mu-\sigma$ y $\mu+\sigma$. La curva tiende asintóticamente a cero al alejarse del valor medio.

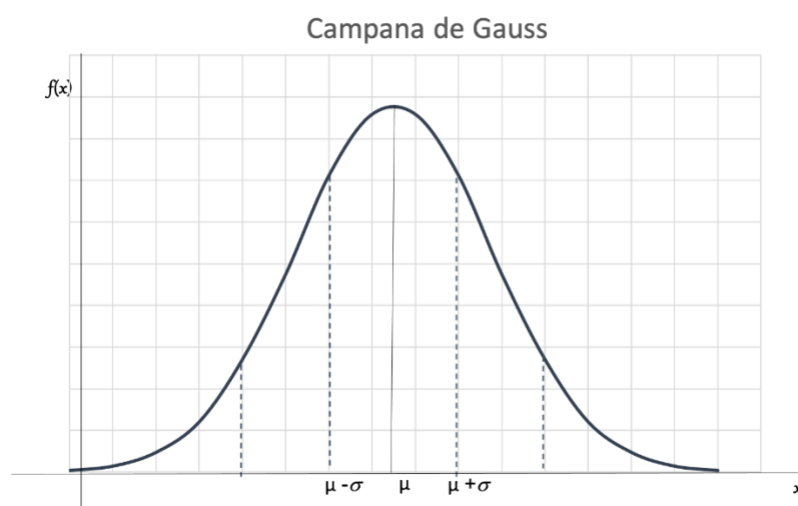


Figura 58. Campana de Gauss.

Tabla 69. Valores de función de densidad.

Valor	Descripción
μ	Media
σ	Desviación típica
σ^2	Varianza
π	3.1416
e	2.7182
x	Abscisa

12.1.2 Homocedasticidad

Es una característica de los datos relacionada con la varianza, la cual es una medida estadística que indica la dispersión de los datos con respecto a la media. En la prueba de la homocedasticidad, las varianzas de cada grupo se comparan y se determina si son iguales o muy similares. Un ejemplo se presenta en la Figura 59, en la que se muestra la diferencia de la homocedasticidad y heterocedasticidad, en el primer ejemplo los puntos están dispersos, pero se mantienen cerca o constante de la tendencia central, lo que no sucede con el ejemplo de la heterocedasticidad, en la que se observan los puntos más alejados de la línea en color rojo que representa en este caso, la tendencia central.

En el caso de entregabilidad de SWA, con dos muestras independientes se utilizó el método de Levene para determinar la existencia de homocedasticidad en los datos a través de la homogeneidad de varianzas.

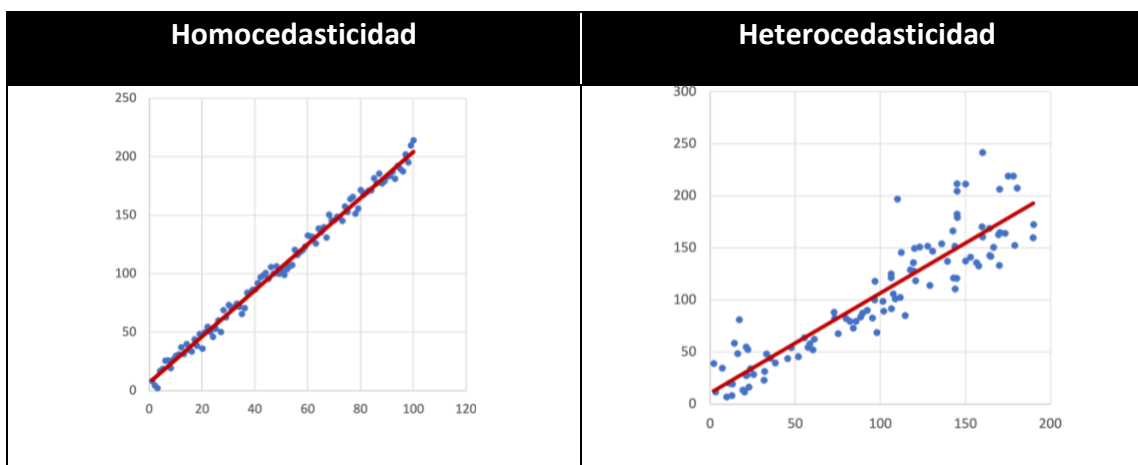


Figura 59. Ejemplo de Homocedasticidad y Heterocedasticidad.

12.1.3 Independencia

La independencia de los datos indica que una observación no depende de otra observación y cada una no influye en la otra. En el caso de que haya dependencia de los datos, las pruebas estadísticas paramétricas pueden fallar.

En el caso de las pruebas de entregabilidad de SWA, no hay independencia de los datos, debido a que se realizaron 100 solicitudes por grupo de solicitudes y en este caso, el que una solicitud resulte no exitosa, depende de que el servidor se encuentre ocupado atendiendo otras solicitudes.

12.1.4 Curtosis

La curtosis es una medida estadística que describe el grado de concentración de los valores de la distribución alrededor de la media y la pesadez de las colas, en la campana de Gauss. El valor de la curtosis 3 es tomado como una curtosis normal, y a partir de este, se determinan los tipos de curtosis, leptocúrtica, mesocúrtica y platicúrtica, como se observa en la Figura 60.

Leptocúrtica. En este tipo de curtosis, los valores están altamente concentrados en la media, con un apuntamiento elevado y las colas gruesas. El valor de la curtosis leptocúrtica es mayor a 3.

Mesocúrtica. En este tipo de curtosis se indica una distribución normal o similar a la campana de Gauss. El valor de la curtosis mesocúrtica es igual a 3.

Platicúrtica. En este tipo de curtosis hay una baja concentración de los datos cerca de la media. El valor de la curtosis platicúrtica es menor a 3.

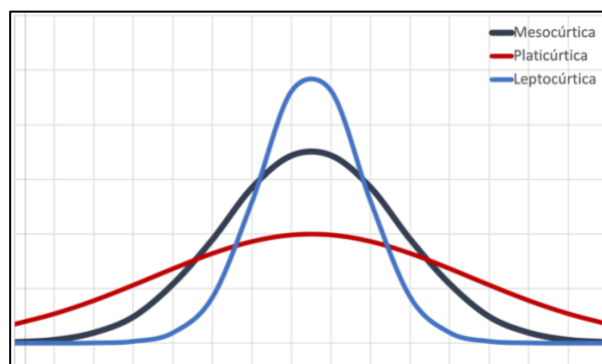


Figura 60. Tipos de curtosis.

12.1.5 Asimetría

La simetría es una medida estadística que describe el grado y la dirección de falta de simetría de la distribución respecto a su media. Existen tres tipos de asimetría, simétrica, positiva y negativa, como se observa en la Figura 61.

En la Tabla 70 se describen los tres tipos de asimetría, y el valor obtenido de asimetría por el tipo de asimetría, y la descripción de su representación en la distribución gaussiana.

Tabla 70. Tipos de asimetría y descripción.

Tipo de Asimetría	Observación	Asimetría
Simétrica	Balanceada	Igual a cero
Asimetría positiva	Cola derecha larga	Mayor que cero
Asimetría negativa	Cola izquierda larga	Menor que cero

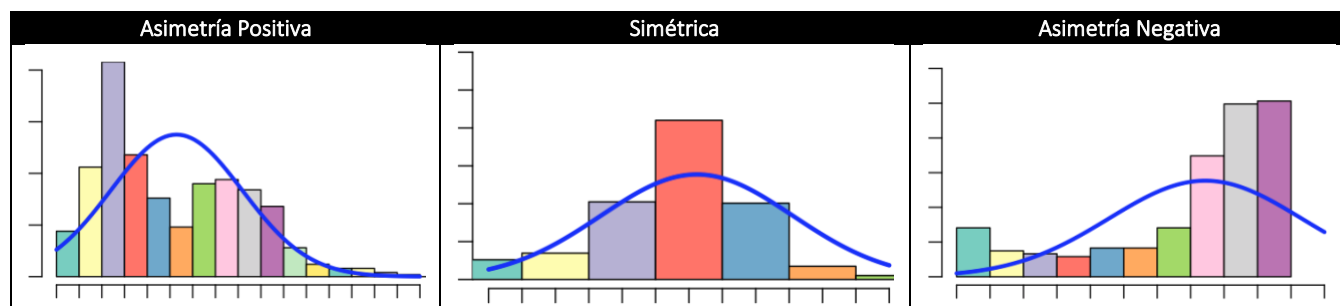


Figura 61. Tipos de asimetría estadística.

13.1.1 Modelo Estadístico de Entregabilidad de SWA

Debido al incumplimiento los principios estadísticos de normalidad, homocedasticidad e independencia de los resultados de entregabilidad de SWA en las métricas de tiempo de respuesta, **throughput** y tasa de éxito, se utilizó el método Yuen.

En la columna Shapiro-W de las tablas 1 a la tabla 14 se encuentran los resultados de la prueba de normalidad, en donde el valor es menor de 0.05, casi en todos los casos.

En la columna Levene de la tabla 86 a la tabla 100, se encuentran los resultados de la prueba de homocedasticidad e igualdad de varianzas, en donde el valor es menor

de 0.05, en la mayoría de los casos, que indica que no hay estas características en la muestra.

12.1.6 Método Yuen

El Método Yuen es un método estadístico para pruebas robustas y es una alternativa a la prueba T de Student. El método Yuen se aplica para confirmar la existencia de diferencias significativas considerando la influencia de valores atípicos y la robustez cuando no existe normalidad ni homocedasticidad. Este se basa en la comparación de medias recortadas al 20 por ciento de los valores extremos de cada cola en lugar de comparar las medias normales. El valor P en Yuen se interpreta igual que en las otras pruebas, el valor P menor a 0.05 indica que existe una diferencias significativa entre las muestras y el tamaño del efecto indica la magnitud de la diferencia entre estas muestras.

12.2 Análisis Estadístico de los Resultados

En esta sección se presenta el análisis estadístico de los resultados de las pruebas de entregabilidad de SWA. De 15 pruebas diferentes realizadas y analizadas, 12 de estas corresponden a pruebas de dos muestras independientes y tres casos de prueba de más de dos muestras independientes.

12.2.1 Análisis Estadístico del Caso de Prueba S-TSW1

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-SM1 se presentan en la Tabla 71, está dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

La sección correspondiente al tiempo de respuesta, las columnas Shapiro-W y Levene indican la falta de normalidad y homocedasticidad. Los valores de asimetría, curtosis indican que existe una asimetría positiva, que conforme el tamaño del SWA, la asimetría también aumenta. Lo mismo sucede con la curtosis, en SWA de 1MB la curtosis es platicúrtica y en SWA de 15MB, esta es muy apuntalada. Aunque los resultados obtenidos con el método Yuen indican diferencia entre las muestras, el valor del efecto indica que esta diferencia es muy baja.

El resultado de Yuen precisa que solo en SWA de 5MB en tiempo de respuesta y **throughput**, es mejor REST que SOAP para entregar SWA, presentados con fondo en azul. En el resto de las pruebas se puede indicar diferencia en el resultado de Yuen, pero el efecto indica que la magnitud es baja o muy baja, por lo que no se puede mencionar que exista una diferencia significativa.

En lo que respecta a los resultados de tasa de éxito, el resultado de Yuen indica que en SWA de 10 y 15MB existe diferencia, aunque esta pueda existir, el resultado del efecto indica que la magnitud de la diferencia es baja. De acuerdo a estos resultados, se puede mencionar que no existe evidencia estadística o diferencia significativa entre SOAP y REST para entregar SWA en lo que respecta a tasa de éxito. En el caso de SWA de 1 y 3MB, el resultado es de 100 por ciento de tasa de éxito en ambos casos.

Los resultados en general, no permiten dar certeza de una diferencia entre utilizar SOAP y REST. Como resultados estadísticos del caso de prueba S-TSW1, podemos indicar que no se rechaza la hipótesis nula, es decir, no existe diferencia significativa entre SOAP y REST.

Tabla 71. Resultados estadísticos del caso de prueba S-TSW1

Tiempo de Respuesta													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media SOAP	Media REST	Yuen	Efecto
PPTX_1MB	23.66	5.17	0.0000	0.144674	23.36	0.000000	2.80970746	12.8105698	15.8105698	23.616	23.704	0.2854	0.0944
PPTX_3MB	65.668	6.05	0.0000	0.007957	65.29	0.000000	2.08638193	8.29371802	11.293718	65.632	65.704	0.0285	0.2106
PPTX_5MB	77.108	64.34	0.0000	0.004328	75.66	0.000008	2.47508952	12.0455506	15.0455506	80.768	73.448	0.0000	0.9008
VIDEO_10MB	68.696	31.34	0.0002	0.165749	68.56	0.016541	0.38780174	-0.5129614	2.48703861	68.904	68.488	0.9120	0.0102
VIDEO_15MB	67.376	59.17	0.0000	0.170234	66.70	0.070674	1.6073624	5.57130635	8.57130635	65.488	69.264	0.0000	0.4630
Throughput													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media SOAP	Media REST	Yuen	Efecto
PPTX_1MB	13.44	5.52	0.000023	0.023928	5.57	0.0000	-1.5192	3.7476	6.7476	5.55	5.52	0.3632	0.0812
PPTX_3MB	14.28	8.26	0.000035	0.008723	5.78	0.0000	-1.7362	4.9273	7.9273	5.76	5.75	0.0250	0.2156
PPTX_5MB	14.96	5.67	0.000000	0.000144	5.58	0.0001	-1.5100	3.0090	6.0090	5.27	5.78	0.0000	0.8933
VIDEO_10MB	12.47	1.42	0.000000	0.049147	5.47	0.0001	-0.3359	-1.0493	1.9507	5.51	5.43	0.0359	0.1991
VIDEO_15MB	11.90	28.17	0.000000	0.000013	5.15	0.8567	0.3412	1.7450	4.7450	5.41	4.91	0.0000	0.6155
Tasa de Éxito													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media SOAP	Media REST	Yuen	Efecto
PPTX_1MB	100	0								100	100		
PPTX_3MB	100	0								100	100		
PPTX_5MB	65.89	1.36	0.000000	0.174006	65.92	0.000000	-0.3042075	0.43859839	17.0915234	65.94	65.85	0.4829	0.0676
VIDEO_10MB	30.43	3.07	0.000000	0.001101	30.36	0.000000	0.39465646	0.34289159	3.34289159	31.26	29.59	0.0000	0.6327
VIDEO_15MB	17.98	1.22	0.000000	0.020809	17.95	0.000000	0.68805415	2.19280686	5.19280686	18.28	17.67	0.0002	0.3134

12.2.2 Análisis Estadístico del Caso de Prueba S-SM1.

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-SM1 se presentan en la Tabla 72, está en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

La sección correspondiente al tiempo de respuesta, las columnas Shapiro-W y Levene indican la falta de normalidad y homocedasticidad. Los valores de asimetría, curtosis y el momento indican que existe una asimetría positiva, una distribución leptocúrtica muy alta y con valores muy extremos, respectivamente. Aunque los resultados obtenidos con el método Yuen indican diferencia entre las muestras, el valor del efecto indica que esta diferencia es muy baja.

La sección correspondiente al *throughput*, Los resultados de asimetría y curtosis muestran que la mayoría de las distribuciones presentan comportamientos cercanos a la simetría, esto no quiere decir que exista normalidad. indicando una distribución leptocúrtica con presencia significativa de valores atípicos y alejamiento de la normalidad estadística. En el resto de las pruebas se puede indicar diferencia en el resultado de Yuen, pero el resultado del efecto indica que la magnitud es muy baja, por lo que no se puede mencionar que exista una diferencia significativa.

En lo que respecta a los resultados de tasa de éxito, los resultados estadísticos indican la falta de normalidad y homocedasticidad. La prueba de asimetría y curtosis indican que existe asimetría negativa y una distribución muy leptocúrtica elevada en SWA de 5MB, en el caso de los SWA de 10 y 15MB indican que existe una distribución platicúrtica moderada, con un poco de asimetría. El resultado de Yuen indica una fuerte diferencia en SWA de 10MB, pero en el resto indica que no existe esa diferencia, por lo que podemos indicar la falta de evidencia estadística o diferencia significativa entre SOAP y SOAP MTOM para entregar SWA.

Los resultados en general, no permiten dar certeza de una diferencia entre utilizar SOAP MTOM y SOAP. Así que en este punto definimos que SOAP MTOM, de la manera en la que están implementados los SWA, no se observa una ventaja significativa en el desempeño de entrega de SWA entre estas alternativas. Como resultados estadísticos del caso de prueba S-SM1, podemos indicar que no se rechaza la hipótesis nula, es decir, no existe diferencia significativa entre SOAP MTOM y SOAP.

Tabla 72. Resultados estadísticos del caso de prueba S-SM1.

Tiempo de Respuesta													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media MTOM	Media SOAP	Yuen	Efecto
AUDIO_1MB	8.98	2.08	0.00000	0.1447	8.88753	0.00000	1.72605	6.81269	9.81269	8.4480	9.5120	0.0000	0.4979
DOCX_3MB	23.56	16.84	0.00000	0.0080	22.92115	0.00000	1.05215	0.12985	3.12985	24.4080	22.7040	0.0004	0.3240
PDF_5MB	26.47	25.01	0.00000	0.0001	25.98680	0.00387	1.34720	2.28977	5.28977	27.8160	25.1280	0.0005	0.3688
AUDIO_10MB	29.70	51.98	0.00000	0.6429	28.97073	0.03807	1.15613	1.50683	4.50683	28.5920	30.8080	0.0035	0.3068
VIDEO_15MB	21.86	8.99	0.00000	0.1702	21.26696	0.00000	2.63891	9.03456	12.03456	22.4560	21.2720	0.7157	0.0385
Throughput													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media MTOM	Media SOAP	Yuen	Efecto
AUDIO_1MB	16.6153	5.81	0.00000	0.0239	16.58662	0.0000	0.15	1.19	4.187085	17.5123	15.7182	0.0000	0.4937
DOCX_3MB	16.9714	6.71	0.00000	0.0087	17.0828817	0.0002	-0.58	-0.73	2.268231	16.3882	17.5547	0.0001	0.3406
PDF_5MB	15.2242	6.28	0.08983	0.0000	15.2541625	0.2473	-0.21	-0.36	2.637464	14.5247	15.9236	0.0003	0.3507
AUDIO_10MB	12.5996	7.59	0.22867	0.5958	12.5919078	0.8265	0.05	-0.56	2.444204	13.4009	11.7984	0.0001	0.4036
VIDEO_15MB	15.3135	2.67	0.00000	0.0000	15.6324276	0.0002	-1.71	2.98	5.975018	14.9474	15.6796	0.2818	0.1597
Tasa de Éxito													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media MTOM	Media SOAP	Yuen	Efecto
AUDIO_1MB	100	0								100	100		
DOCX_3MB	100	0								100	100		
PDF_5MB	54.97	2.65	0.0000	0.0286	54.9652844	0.0001	-0.64	4.74	7.744622	54.952	54.992	0.7109	0.0357
AUDIO_10MB	27.84	2.13	0.0000	0.0928	27.8439327	0.0000	0.00	-0.26	2.735790	28.728	26.952	0.0000	0.9258
VIDEO_15MB	17.28	0.95	0.0000	0.0208	17.2758698	0.0000	-0.08	-0.18	2.820419	17.16	17.392	0.2091	0.1603

12.2.3 Análisis Estadístico del Caso de Prueba S-TPC1

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-TPC1 se presentan en la Tabla 73, está en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito. Los resultados estadísticos indican la falta de normalidad y homocedasticidad de las muestras, en la columna Shapiro-W y Levene, respectivamente.

De acuerdo con los resultados estadísticos de Tiempo de Respuesta, en algunos casos existe diferencia significativa entre HTTP y HTTPS, en SWA de granularidad fina y media combinados con REST y HTTPS se obtiene mejor resultado, como se indica en el resultado de Yuen y el efecto. En contraste, para SWA de granularidad gruesa (10 MB y 15 MB), las diferencias se distribuyen entre ambos protocolos: en un caso el mejor resultado corresponde a HTTP y en otro a HTTPS. Por lo tanto, en el contexto de granularidad gruesa no es posible definir una tendencia clara o concluyente respecto a la utilización de alguno de los protocolos de comunicación para entregar SWA.

El resultado de Yuen precisa que solo en SWA de 10MB en *throughput* existe diferencia significativa entre HTTPS que HTTP con el tipo de implementación REST para entregar SWA con granularidad fina y media, marcados en amarillo. En el caso de la entrega de SWA de granularidad gruesa, en dos casos el mejor resultado corresponde a HTTP y en uno a HTTPS. Por lo tanto, en el contexto de granularidad gruesa no es posible definir una tendencia clara o concluyente respecto a la utilización de alguno de los protocolos de comunicación para entregar SWA.

En el resto de las pruebas se puede indicar diferencia en el resultado de Yuen, pero el resultado del efecto indica que la magnitud es muy baja, por lo que no se puede mencionar que exista una diferencia significativa. En lo que respecta a los resultados de tasa de éxito, los resultados de Yuen con diferencia significativa entre HTTP y HTTPS para se encuentran solo en SWA con granularidad gruesa con tendencia favorable para HTTP.

Los resultados en general, indican que la diferencia al utilizar HTTP y HTTPS para entregar SWA, afecta más a SWA de granularidad gruesa. Por lo que podemos indicar que la hipótesis nula no se rechaza para los casos de SWA de 1,3 y 5MB, y se rechaza para SWA de 10 y 15MB. Así que en este punto definimos que el protocolo de comunicación, de la manera en la que están implementados los SWA, se tiene que definir por parte de los encargados de las plataformas de SWA identificar las mejores alternativas.

Tabla 73. Resultados Estadísticos del caso de prueba S-TPC1.

Tiempo de Respuesta														
SWA	Tipo WS	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media HTTP	Media HTTPS	Yuen	Efecto
AUDIO_1MB	SOAP	8.372	0.95511647	0.000000	1.0000	8.45	0.0000	-0.16	-0.63	-0.632936	8.2960	8.4480	1.0000	0.00
AUDIO_1MB	REST	12.33	7.37126104	0.000000	0.0654	12.25	0.0000	0.66	2.92	2.921337	13.5120	11.1520	0.0000	0.70
DOCX_3MB	SOAP	23.21	15.5653173	0.000000	0.0011	22.67	0.0000	1.19	0.93	0.929777	22.0160	24.4080	0.0000	0.45
DOCX_3MB	REST	26.47	34.7158394	0.000000	0.0000	25.69	0.0026	1.89	5.05	5.049666	28.9920	23.9440	0.0000	0.70
PDF_5MB	SOAP	26.19	14.0006586	0.000000	0.0003	25.68	0.0001	1.27	1.28	1.280298	26.7520	25.6240	0.3318	0.12
PDF_5MB	REST	26.19	14.0006586	0.000000	0.0003	25.68	0.0001	1.27	1.28	1.280298	25.6240	26.7520	0.3318	0.12
VIDEO_10MB	SOAP	24.75	34.813751	0.000000	0.0000	23.56	0.0002	2.65	7.99	7.990928	22.5280	26.9760	0.0000	0.59
VIDEO_10MB	REST	27.82	17.8856867	0.000035	0.2621	27.63	0.0638	0.61	0.20	0.197275	27.0400	28.5920	0.0275	0.22
VIDEO_15MB	SOAP	34.40	191.701992	0.000000	0.0000	34.13	0.0000	0.53	-0.88	-0.882629	46.3360	22.4560	0.0000	0.99
VIDEO_15MB	REST	32.84	31.5914699	0.000000	0.0004	32.24	0.0002	1.17	1.61	1.609387	33.2400	32.4320	0.8739	0.02
Throughput														
SWA	Tipo WS	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media HTTP	Media HTTPS	Yuen	Efecto
AUDIO_1MB	SOAP	17.51	4.53113348	0.000000	1.0000	17.47	0.0000	0.69	0.19	0.19	17.5123	17.5123	1.0000	0.00
AUDIO_1MB	REST	12.47	8.36976785	0.000000	0.0000	12.14	0.0000	1.00	0.72	0.72	11.1815	13.7677	0.0000	0.75
DOCX_3MB	SOAP	23.21	15.5653173	0.000000	0.0011	22.67	0.0000	1.19	0.93	0.93	22.0160	24.4080	0.0000	0.45
DOCX_3MB	REST	15.17	7.43291059	0.000002	0.0014	15.28	0.0113	-0.57	-0.21	-0.21	13.9144	16.4404	0.0000	0.65
PDF_5MB	SOAP	16.53	4.76575874	0.000000	0.0011	16.73	0.0365	-0.70	0.03	0.03	16.3914	16.6851	0.5539	0.07
PDF_5MB	REST	16.53	4.76602011	0.000000	0.0011	16.73	0.0365	-0.70	0.03	0.03	16.6854	16.3914	0.5532	0.07
VIDEO_10MB	SOAP	16.10	9.80671589	0.000000	0.0000	16.38	0.1321	-0.91	0.50	0.50	18.1894	14.0236	0.0000	0.92
VIDEO_10MB	REST	13.41	5.06698401	0.015880	0.0002	13.38	0.0703	0.25	-0.49	-0.49	14.6213	12.2106	0.0000	0.78
VIDEO_15MB	SOAP	11.35	16.1428728	0.000000	0.0974	11.35	0.0002	0.10	-1.59	-1.59	7.7596	14.9474	0.0000	0.98
VIDEO_15MB	REST	10.40	2.5636252	0.129230	0.0000	10.38	0.7184	0.15	-0.07	-0.07	10.7719	10.0383	0.0005	0.36
Tasa de Éxito														
SWA	Tipo WS	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media HTTP	Media HTTPS	Yuen	Efecto
AUDIO_1MB	SOAP	100	0								100	100		0.00
AUDIO_1MB	REST	100	0								100	100		0.00
DOCX_3MB	SOAP	100	0								100	100		0.00
DOCX_3MB	REST	99.24	2.09685141	-2.190554	0.2383	4.59	1.1539	2.88	0.00	0.00	100	99.98	0.0000	0.00
PDF_5MB	SOAP	59.76	3.25733333	0.000000	0.0507	59.97	0.0000	-2.76	11.01	11.01	60.10	59.42	0.0000	0.27
PDF_5MB	REST	59.76	3.25733333	0.000000	0.0507	59.97	0.0000	-2.76	11.01	11.01	60.07	59.35	0.0010	0.27
VIDEO_10MB	SOAP	30.62	5.38314859	0.000000	0.7199	30.63	0.0005	0.01	-1.33	-1.33	32.61	28.65	0.0010	0.95
VIDEO_10MB	REST	29.24	6.05461847	0.000000	0.0000	29.21	0.0001	0.43	-0.83	-0.83	31.00	27.48	0.0000	0.87
VIDEO_15MB	SOAP	17.65	1.32820884	0.000000	0.6107	17.65	0.0000	0.03	0.21	0.21	18.14	17.16	0.0000	0.55
VIDEO_15MB	REST	17.45	1.49367068	0.000000	0.3994	17.45	0.0000	0.05	0.27	0.27	18.08	16.82	0.0000	0.75

12.2.4 Análisis Estadístico del Caso de Prueba C-SC1

Los resultados de los métodos aplicados de las pruebas de entregabilidad de SWA del caso de prueba C-SC1 se presentan en Tabla 74. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba C-SC1 se presentan en la Tabla 73, está en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito. Los resultados estadísticos indican la falta de normalidad y homocedasticidad de las muestras, en la columna Shapiro-W y Levene, respectivamente.

Con respecto a los resultados de tiempo de respuesta, estos indican la falta de normalidad y homocedasticidad, se encontró asimetría y curtosis, lo que indica variabilidad e inestabilidad en el caso de SWA de 15MB. El resultado del efecto y el resultado de Yuen precisa que existe una amplia diferencia entre las muestras y estos están a favor con la implementación de Server Cache para entregar SWA.

En lo que respecta a los resultados de *throughput*, indican que existe inestabilidad y variabilidad, esto se indica en las celdas resaltadas en color amarillo. Dicha inestabilidad se debe a que se entregan cantidades de bytes similares durante el proceso de la prueba.

En lo que respecta a los resultados de tasa de éxito, las pruebas de asimetría y curtosis indican que existe una leve asimetría entre las muestras y una distribución platicúrtica moderada. El resultado de Yuen indica una fuerte diferencia significativa entre las muestras, a favor de la implementación de la arquitectura Server Cache para entregar SWA.

A pesar de que el tiempo de respuesta es más alto en la entrega de SWA de 15MB con la implementación de Server Cache, los resultados evidencian que existe diferencia estadística significativa cuando se utiliza Server Cache para entregar SWA. Por lo que podemos rechazar la hipótesis nula, esto quiere decir que sí existe diferencia significativa entre las muestras.

Tabla 74. Resultados Estadísticos del caso de prueba C-SC1.

Tiempo de Respuesta													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Standalone	Media Cache	Yuen	Efecto
VIDEO_10MB	45.02	1041.41	0.0000	0.0000	45.019811	0.00000	0.099438	-1.792975	-1.792975	76.5280	13.5200	0.0000	0.9540
VIDEO_15MB	92.26	10837.36	0.0000	0.0000	58.363248	0.00000	2.173185	5.087433	5.087433	32.1921	163.4270	0.0000	0.9672
Throughput													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Standalone	Media Cache	Yuen	Efecto
VIDEO_10MB	5.26	0.22	0.0000	0.0089	5.291535	0.00000	0.080400	-1.993460	1.006540	30.78	96.84	0.0000	0.9400
VIDEO_15MB	12.28	24.85	0.0000	0.0000	12.170985	0.00250	0.341993	-0.449493	2.550507	11.9840	12.6217	0.9400	0.0109
Tasa de Éxito													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Standalone	Media Cache	Yuen	Efecto
VIDEO_10MB	63.81	1186.13	0.0000	0.1015	41.503545	0.00000	0.080484	-1.993460	1.006540	30.7760	96.8400	0.0000	0.9149
VIDEO_15MB	48.97	1609.80	0.0000	0.0000	21.858818	0.00000	0.467908	-1.770346	1.229654	17.9400	85.7357	0.0000	0.9239

12.2.5 Análisis Estadístico del Caso de Prueba C-TF2

Los resultados de los métodos aplicados de las pruebas de entregabilidad de SWA del caso de prueba C-SC1 se presentan en la Tabla 75. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito. En la mayoría de los resultados se indican la falta de normalidad y homocedasticidad, de acuerdo a las columnas Shapiro-W y Levene. En las celdas sombreadas en azul aparecen los datos con normalidad y homocedasticidad, que en ninguno de los casos cumple con estos dos principios.

En la sección de tiempo de respuesta, El resultado de Yuen precisa que existe diferencia significativa en los resultados de a favor de la prueba Standalone que con la implementación de Tolerancia a Fallas, esto se puede mencionar a la configuración para cuando una solicitud no es respondida.

En lo que respecta a los resultados de *throughput*, los resultados con diferencias significativas media solo se presentan en SWA de 10MB, con SWA de 15MB de tipo SOAP se presentan resultados altamente significativos. En el caso del SWA de tipo REST, los resultados son similares si se utiliza o no la implementación de Tolerancia a Fallas para entregar SWA, esto se debe a que la prueba Standalone tardo mucho más que en la prueba Standalone con SWA de tipo SOAP.

En lo que respecta a la métrica de tasa de éxito, los resultados estadísticos indican que en todas las pruebas realizadas existe diferencia significativa a favor de la implementación de Tolerancia a Fallas para entregar SWA, en la sección tasa de éxito de la Tabla 75 aparece sombreados los resultados en color azul, con indicadores de Yuen y el efecto con magnitud muy alta.

Finalmente, los encargados de las plataformas de SWA serán los encargados de realizar la configuración de estas, y de definir los mejores valores de la configuración de tolerancia a fallas para obtener los mejores resultados de entrega y definir el tiempo adecuado para la correcta funcionalidad de las plataformas de SWA.

Tabla 75. Resultados Estadísticos del caso de prueba C-TF2.

Tiempo de Respuesta														
SWA	Tipo WS	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Standalone	Media FT	Yuen	Efecto
VIDEO_10MB	SOAP	34.40	113.21	0.000000	0.0808	34.274961	0.000409	0.373678	-0.941315	2.058685	26.8720	41.9200	0.0000	0.8871
VIDEO_10MB	REST	31.59	18.78	0.000000	0.0000	31.657508	0.000024	-0.372489	-0.770389	2.229611	28.5920	34.5920	0.0000	0.9093
VIDEO_15MB	SOAP	32.06	105.75	0.000000	0.0580	31.952156	0.000000	0.052941	-1.301996	1.698004	22.4560	41.6720	0.0000	0.9712
VIDEO_15MB	REST	37.14	31.67	0.000000	0.0000	37.494559	0.000000	-0.385823	-1.306765	1.693235	32.4320	41.8480	0.0000	0.9664
Throughput														
SWA	Tipo WS	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Standalone	Media FT	Yuen	Efecto
VIDEO_10MB	SOAP	13.51	11.69	0.000001	0.1707	13.421292	0.672542	0.373678	2.999044	5.999044	12.9764	14.0529	0.0000	0.4319
VIDEO_10MB	REST	13.07	3.47	0.000263	0.7303	13.045459	0.290301	-0.372489	2.018473	5.018473	13.9202	12.2106	0.0000	0.6498
VIDEO_15MB	SOAP	12.74	7.99	0.000000	0.0412	12.742627	0.003635	0.052941	-1.370446	1.629554	10.5307	14.9474	0.0000	0.9322
VIDEO_15MB	REST	10.00	0.98	0.324023	0.0221	10.008681	0.283786	-0.385823	-0.151059	2.848941	9.9649	10.0383	0.3096	0.1029
Tasa de Éxito														
SWA	Tipo WS	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Standalone	Media FT	Yuen	Efecto
VIDEO_10MB	SOAP	35.16	66.08	0.000000	0.0000	34.416349	0.000000	0.373678	3.075498	6.075498	28.6400	41.6880	0.0000	0.9517
VIDEO_10MB	REST	32.91	36.17	0.000000	0.0000	32.705693	0.000000	-0.372489	-0.989088	2.010912	27.4800	38.3440	0.0000	0.9511
VIDEO_15MB	SOAP	19.98	11.95	0.000000	0.0000	19.734700	0.000000	0.052941	-0.159327	2.840673	17.1600	22.7920	0.0000	0.9866
VIDEO_15MB	REST	19.29	8.41	0.000000	0.0000	19.222001	0.000002	-0.385823	-0.988348	2.011652	16.8240	21.7520	0.0000	0.8977

12.2.6 Análisis Estadístico del Caso de Prueba C-C4

Los resultados de los métodos aplicados de las pruebas de entregabilidad de SWA del caso de prueba C-C4 se presentan en la Tabla 76. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

Los resultados de tiempo de respuesta en la columna Shapiro-W y Levene indican la falta de normalidad, en algunos casos presentan falta de homocedasticidad. Lo que respecta al asimetría, se presenta una asimetría positiva y distribuciones leptocúrticas. El resultado de Yuen precisa que existe diferencia significativa en los resultados de tiempo de respuesta a favor de la prueba Standalone a diferencia de la implementación de Clustering para entregar SWA, esto se debe a que es un reflejo de los resultados de tasa de éxito.

En el caso de la métrica de **throughput**, Los resultados de la columna Shapiro-W y Levene indican la falta de normalidad, en algunos casos presentan falta de homocedasticidad. Lo que respecta al asimetría, se presenta una asimetría positiva y distribuciones leptocúrticas. El resultado de Yuen indica que en algunos casos existe diferencia significativa, como indicativos son los datos que se presentan en color azul.

En lo que respecta a la métrica de tasa de éxito, los resultados de entrega de SWA son mejores cuando se implementa el **Clustering**, marcados en color azul. Prácticamente en los datos mostrados, en la prueba de Yuen y el resultado del efecto con una magnitud significativamente alta.

Los resultados estadísticos correspondientes a la tasa de éxito indican que en todas las pruebas realizadas, existe una diferencia significativa a favor de la implementación de **Clustering** para la entrega de SWA, precisamente en los resultados Tasa de éxito, los resultados de la prueba Yuen y al tamaño del efecto. Por lo que podemos indicar que se rechaza la hipótesis nula, es decir, existe diferencia significativa entre la implementación de Clustering y Standalone, para entregar SWA.

Tabla 76. Resultados estadísticos del caso de prueba C-C4.

Tiempo de Respuesta														
SWA	Tipo WS	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Stand	Media C4	Yuen	Efecto
AUDIO_1MB	SOAP	9.83	3.85	0.000000	0.3974	9.584337	0.000000	1.323411	1.854432	4.854432	10.0800	9.5840	0.0156	0.2101
AUDIO_1MB	REST	12.45	10.69	0.000000	0.1093	12.214425	0.005610	0.540862	-0.512036	2.487964	13.5520	11.3440	0.0000	0.5247
DOCX_3MB	SOAP	26.56	61.38	0.000000	0.0000	25.848258	0.000098	1.109626	0.517107	3.517107	32.3440	20.7840	0.0000	0.8489
DOCX_3MB	REST	37.76	159.74	0.000000	0.0140	37.741108	0.000421	-0.022061	-1.491551	1.508449	49.5360	25.9920	0.0000	0.9861
PDF_5MB	SOAP	37.59	49.97	0.000000	0.0000	36.398217	0.004161	1.506157	2.273033	5.273033	38.9200	36.2560	0.0398	0.2549
PDF_5MB	REST	34.00	42.04	0.000000	0.0149	33.361604	0.031137	1.456968	2.813010	5.813010	31.7520	36.2560	0.0000	0.8008
VIDEO_10MB	SOAP	57.35	125.58	0.000037	0.8317	57.299041	0.035617	0.082527	-0.997139	2.002861	48.7103	65.9936	0.0000	0.9253
VIDEO_10MB	REST	87.84	553.29	0.000000	0.0000	86.556086	0.000053	0.840945	0.176545	3.176545	68.4880	107.1859	0.0000	0.9782
VIDEO_15MB	SOAP	77.43	1815.39	0.000000	0.0000	75.664216	0.000002	0.874313	0.299535	3.299535	41.9360	112.9200	0.0000	0.9347
VIDEO_15MB	REST	71.30	1927.34	0.000000	0.0000	68.501140	0.000000	0.656468	-0.643707	2.356293	32.4320	110.1600	0.0000	0.9860
Throughput														
SWA	Tipo WS	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Stand	Media C4	Yuen	Efecto
AUDIO_1MB	SOAP	13.80	5.70	0.000000	0.9303	13.843775	0.000221	-0.279840	-0.169323	2.830677	13.4701	14.1377	0.0091	0.2211
AUDIO_1MB	REST	11.27	8.31	0.000000	0.3087	11.225175	0.013828	0.305688	-0.869486	2.130514	10.2787	12.2543	0.0000	0.5041
DOCX_3MB	SOAP	15.64	15.14	0.000000	0.0033	15.644851	0.001852	-0.236004	-1.038827	1.961173	12.4471	18.8313	0.0000	0.8652
DOCX_3MB	REST	11.63	18.77	0.000000	0.0000	11.262252	0.000000	0.624506	-0.911086	2.088914	7.8581	15.4043	0.0000	0.9477
PDF_5MB	SOAP	17.16	7.34	0.000000	0.0000	17.383789	0.045119	-0.731554	0.034107	3.034107	16.5231	17.7872	0.0021	0.3619
PDF_5MB	REST	17.66	5.80	0.000000	0.0009	18.057536	0.013536	-1.576056	2.346227	5.346227	17.5374	17.7872	0.1298	0.1718
VIDEO_10MB	SOAP	9.85	6.49	0.019345	0.0000	9.883219	0.063490	-0.041881	-0.463700	2.536300	9.0206	10.6799	0.0004	0.4169
VIDEO_10MB	REST	8.66	12.31	0.000000	0.0000	8.466679	0.000000	0.368999	-1.379855	1.620145	5.4252	11.9002	0.0000	0.9650
VIDEO_15MB	SOAP	9.85	6.49	0.019345	0.0000	9.883219	0.063490	-0.041881	-0.463700	2.536300	9.0206	10.6799	0.0004	0.4169
VIDEO_15MB	REST	10.25	4.47	0.012312	0.0000	10.193865	0.720439	0.208620	0.885905	3.885905	10.0381	10.4543	0.2550	0.1319
Tasa de Éxito														
SWA	Tipo WS	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Stand	Media C4	Yuen	Efecto
AUDIO_1MB	SOAP	100	0.00											
AUDIO_1MB	REST	100	0.00											
DOCX_3MB	SOAP	100	0.00											
DOCX_3MB	REST	100	0.00											
PDF_5MB	SOAP	98.34	4.18	0.000000	0.0000	0.000000	0.000000	-0.796692	-0.711414	2.288586	96.6720	100	0.0000	0.8716
PDF_5MB	REST	92.12	65.29	0.000000	0.0000	92.142579	0.000000	-0.132323	-1.824156	1.175844	84.2400	100	0.0000	0.9482
VIDEO_10MB	SOAP	65.63	1187.49	0.000000	0.0000	65.632000	0.000000	-0.003964	-2.002728	0.997272	31.2640	100	0.0000	0.9137
VIDEO_10MB	REST	64.50	1225.42	0.000000	0.0118	64.504000	0.000000	0.000217	-2.002797	0.997203	29.5920	99.4160	0.0000	0.9158
VIDEO_15MB	SOAP	38.71	407.85	0.000000	0.0000	38.712000	0.000000	0.031294	-1.952852	1.047148	18.7200	58.7040	0.0000	0.9329
VIDEO_15MB	REST	36.51	398.87	0.000000	0.0000	34.566133	0.000000	0.048669	-1.949705	1.050295	16.8240	56.2000	0.0000	0.9345

12.2.7 Análisis Estadístico del Caso de Prueba C-CD4

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba C-CD4 se presentan en la Tabla 77. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

En el caso del tiempo de respuesta, los resultados de las pruebas de Shapiro-W indican la falta de normalidad, y la prueba de Levene indica la falta de homocedasticidad en algunos casos, en color azul se indican los casos que si se presentan similitud de varianzas. El asimetría va de la normalidad a la asimetría de las muestras conforme el SWA es más grande, la curtosis pasa de una muestra mesocúrtica a platicúrtica conforme el SWA es más grande. Los resultados de Yuen y el efecto indican diferencias significativas con magnitudes altas entre las muestras.

En cuanto a la métrica de **throughput**, los resultados de las pruebas de Shapiro-W indican la falta de, y de Levene indican la falta de normalidad y homocedasticidad. El **asimetría** presenta simetría y la curtosis platicúrtica, lo que indica un comportamiento más homogéneo. En todos los casos analizados con la prueba Yuen y el efecto indican que si existe diferencia estadísticamente significativa, lo cual se observa en la sección **throughput** de la Tabla 77, en las celdas resaltadas en color azul.

En lo que respecta a la tasa de éxito, los resultados estadísticos de entrega de SWA indican que existe diferencia significativa a favor de la entrega de SWA con la implementación de contenerización. El resultado de la prueba Yuen y el valor del efecto corroboran esta diferencia, esta información se presenta en la sección tasa de éxito de la Tabla 77, en celdas resaltadas en color azul.

Finalmente, los resultados de este análisis estadístico permiten rechazar la hipótesis nula, e indicar que existe diferencia significativa cuando se utilizan más instancias del servidor de aplicaciones para entregar SWA, específicamente con instancias replicadas como es el caso de la contenerización.

Tabla 77. Resultados estadísticos del caso de prueba C-CD4.

Tiempo de Respuesta													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Standalone	Media Contenerizado	Yuen	Efecto
PPTX_1MB	16.61	53.87	0.000000	0.9442	16.556141	0.000000	0.135134	-1.389341	1.610659	23.6160	9.6000	0.0000	0.9647
PPTX_3MB	44.17	472.98	0.000000	0.0103	44.168000	0.000000	-0.004338	-1.918748	1.081252	65.6320	22.7040	0.0000	0.9365
PPTX_5MB	65.88	278.51	0.000000	0.4113	65.877391	0.000274	0.022371	-1.423917	1.576083	80.7680	51.0000	0.0000	0.9923
VIDEO_10MB	74.63	95.27	0.000000	0.0001	73.717304	0.128005	1.305965	2.447194	5.447194	68.9040	80.3520	0.0000	0.8162
VIDEO_15MB	74.87	664.83	0.000000	0.0000	76.041651	0.000042	-0.598604	1.855422	4.855422	65.4880	84.2560	0.0000	0.8753
Throughput													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Standalone	Media Contenerizado	Yuen	Efecto
PPTX_1MB	9.96	22.63	0.000000	0.0000	9.716884	0.000000	0.352028	-1.445173	1.554827	5.5462	14.3697	0.0000	0.9660
PPTX_3MB	11.42	35.00	0.000000	0.0000	11.229917	0.000000	0.205582	-1.749973	1.250027	5.7555	17.0942	0.0000	0.9410
PPTX_5MB	9.37	18.61	0.000000	0.0000	9.166071	0.000000	0.214432	-1.727314	1.272686	5.2678	13.4626	0.0000	0.9659
VIDEO_10MB	8.34	15.26	0.000000	0.0000	8.149517	0.000000	0.035343	-1.099029	1.900971	5.5083	11.1686	0.0000	0.9157
VIDEO_15MB	6.63	10.33	0.000000	0.0000	6.892259	0.000001	-0.407252	-0.366293	2.633707	5.4098	7.8484	0.0000	0.8235
Tasa de Éxito													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Standalone	Media Contenerizado	Yuen	Efecto
PPTX_1MB	100	0.00	0.000000	0.0000									
PPTX_3MB	100	0.00	0.000000	0.0000									
PPTX_5MB	82.97	291.85	0.000000	0.0000	82.968000	0.000000	-0.006059	-1.999875	1.000125	65.9360	100	0.0000	0.9168
VIDEO_10MB	53.67	623.34	0.000000	0.0000	53.085484	0.000000	0.170292	-1.822482	1.177518	30.2400	77.1040	0.0000	0.9578
VIDEO_15MB	28.87	328.00	0.000000	0.0000	29.388478	0.000000	0.125366	-1.458815	1.541185	18.2800	39.4640	0.0000	0.8875

12.2.8 Primer Análisis Estadístico del Caso de Prueba S-JHS1

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-JHS1 se presentan en la Tabla 78. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

Los resultados estadísticos del tiempo de respuesta indican la falta de normalidad en la columna Shapiro-W, y en algunos casos se presenta similaridad en las varianzas, estas aparecen sombreadas en color azul en la columna Levene. El asimetría indica simetría en solo una prueba y una curtosis platicúrtica. En lo que respecta al resultado de Yuen, en una sola prueba indica diferencia con una alta magnitud, en el resto indica diferencia con magnitud media y en la última no indica diferencia entre las muestras.

En cuanto a la métrica de **Throughput**, no existe normalidad de acuerdo al valor de la columna Shapiro-W, y homocedasticidad solo se presenta en una muestra. El resultado de Asimetría presenta una asimetría a la derecha y la curtosis apuntalada leptocúrtica. También es preciso mencionar que en algunos casos se presentan diferencias significativas favorables a la muestra de las pruebas de la configuración de JHS 1 y en otros casos favorables a la configuración de JHS 2, de acuerdo al resultado de Yuen.

En lo que respecta a la métrica de Tasa de Éxito, los resultados estadísticos de entrega de SWA indican que existen diferencias significativa para ambos casos de la configuración de JHS. En algunos casos es mejor la configuración JHS 1 y en otros la configuración JHS 2. Por lo que en estas pruebas se pueden definir el parámetro XMS o el tamaño inicial de la JHS en cualquier tamaño.

Como conclusión de los resultados de los métodos estadísticos del caso de prueba S-JHS1 en el primer análisis, podemos mencionar que no se rechaza la hipótesis nula. Esto quiere decir que no existe diferencia significativa entre la configuración 1 y la configuración 2 de la JHS en las pruebas de entregabilidad de SWA.

Tabla 78. Resultados estadísticos del caso de prueba S-JHS1.

Tiempo de Respuesta															
SWA	XMS JHS1	XXM JHS1	XMS JHS 2	XXM JHS 2	Media	Varianza	Shapiro-W	Levene	Kolmogorov	Asimetría	Curtosis	Media JHS 1	Media JHS 2	Yuen	Efecto
VIDEO_10MB	512MB	2048MB	1024MB	2048MB	22.71	49.19	0.000000	0.0000	0.000000	2.120876	4.961011	18.3840	27.0400	0.0000	0.9595
VIDEO_10MB	512MB	3096MB	2048MB	3096MB	26.34	1.39	0.000000	0.4886	0.000000	0.786353	0.340553	26.4000	26.2800	0.4420	0.0740
VIDEO_10MB	512MB	4128MB	3096MB	4128MB	43.78	31.96	0.000000	0.0000	0.001405	1.968393	4.779910	45.3280	42.2320	0.0001	0.4085
Throughput															
SWA	XMS JHS 1	XXM JHS 1	XMS JHS 2	XXM JHS 2	Media	Varianza	Shapiro-W	Levene	Kolmogorov	Asimetría	Curtosis	Media JHS 1	Media JHS 2	Yuen	Efecto
VIDEO_10MB	512MB	2048MB	1024MB	2048MB	17.42	16.56	0.000000	0.0000	0.000032	-0.687739	-0.517178	20.8637	13.9837	0.0000	0.9827
VIDEO_10MB	512MB	3096MB	2048MB	3096MB	22.31	0.42	0.000000	0.0210	0.000018	-1.030916	2.482496	22.2533	22.3633	0.0202	0.2327
VIDEO_10MB	512MB	4128MB	3096MB	4128MB	17.13	3.88	0.000000	0.0840	0.314810	-0.916869	1.064353	16.1106	18.1590	0.0000	0.7506
Tasa de Éxito															
SWA	XMS JHS 1	XXM JHS 1	XMS JHS 2	XXM JHS 2	Media	Varianza	Shapiro-W	Levene	Kolmogorov	Asimetría	Curtosis	Media JHS 1	Media JHS 2	Yuen	Efecto
VIDEO_10MB	512MB	2048MB	1024MB	2048MB	29.38	2.21	0.000000	0.0001	0.000000	0.166833	-0.692424	30.1120	28.6480	0.0000	0.6158
VIDEO_10MB	512MB	3096MB	2048MB	3096MB	47.06	3.63	0.000000	0.0000	0.000009	2.054839	0.000001	47.40	47.80	0.0202	0.0000
VIDEO_10MB	512MB	4128MB	3096MB	4128MB	59.27	6.01	0.002695	0.2526	0.012157	-0.166141	-0.198761	57.5840	60.9600	0.0000	0.9476

12.2.9 Segundo Análisis Estadístico del Caso de Prueba S-JHS1

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-JHS1 se presentan en la Tabla 79. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

En la presente los resultados de los métodos estadístico aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-JHS1 para más de dos muestras, en la que se comparan los valores del parámetro XMx, con 2048, 3064 y 4096MB. Las pruebas de normalidad y homocedasticidad realizadas con el test de Anova con un factor, indican la falta de estas características, por lo que se aplicó Kruskal-Wallis y la validación con el método de Dunn, el valor del efecto se indica la existencia de la diferencia significativa entre las muestras.

De acuerdo con la media muestral, la Configuración 3, correspondiente a un tamaño de JHS de 4096 MB, presenta el mejor desempeño en términos de tasa de éxito. Estos resultados sugieren que el tamaño de la JHS influye directamente en la capacidad de entrega del sistema, ya que una mayor asignación de memoria permite la generación de más objetos y el procesamiento exitoso de un mayor número de solicitudes. Por otro lado, la Configuración 2 resultó con los mejores resultados de **throughput**. No obstante, el dimensionamiento de la JHS debe evaluarse de manera equilibrada, considerando las capacidades del hardware, a fin de definir una configuración que permita mejorar la capacidad de respuesta y mantener un alto porcentaje de éxito en la entrega.

Finalmente, como conclusión de este segundo análisis estadístico del caso de prueba S-JHS1, se rechaza la hipótesis nula, es decir, existe diferencia significativa entre las muestras en las métricas de tiempo de respuesta, throughput y tasa de éxito.

Tabla 79. Resultados estadísticos del caso de prueba S-JHS1.

Tiempo de Respuesta						
Configuración	Media	Varianza	Anova	Kruskall-W	Dunn	Efecto
Configuración 1	22.71	49.19	2E-16	2.2E-16	0	0.705
Configuración 2	26.34	1.38				
Configuración 3	43.78	31.95				
Throughput						
Configuración	Media	Varianza	Anova	Kruskall-W	Dunn	Efecto
Configuración 1	17.42	16.56	2.00E-16	2.2E-16	0	0.604
Configuración 2	20.39	0.42				
Configuración 3	17.13	3.88				
Tasa de Éxito						
Configuración	Media	Varianza	Anova	Kruskall-W	Dunn	Efecto
Configuración 1	29.26	5.18	2.00E-16	2.2E-16	0	0.891
Configuración 2	49.04	3.57				
Configuración 3	59.72	6.0				

12.2.10 Análisis Estadístico del Caso de Prueba S-GC5

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-GC5 se presentan en la Tabla 80. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

Los resultados de las pruebas de normalidad y homocedasticidad indican la falta de estas características en las muestras. La prueba de Kruskal–Wallis para evaluar la existencia de diferencias globales entre los grupos, y después se aplicó la prueba post hoc de Dunn para identificar comparaciones específicas entre pares de algoritmos. Por último, el análisis del tamaño del efecto permitió determinar la magnitud de las diferencias observadas, complementando la interpretación de la significancia estadística.

En los resultados estadísticos del tiempo de respuesta indican que existe diferencia estadísticamente significativa entre de las muestras. El resultado de la media estadística indica que la muestra de la prueba del algoritmo G1GC obtiene los mejores resultados del resto de los algoritmos.

Los resultados de las pruebas estadísticas de *throughput*, indican que existe diferencia significativa en una de las muestras, como se presenta en la columna Dunn y el efecto indica una diferencia de magnitud medianamente alta. El resultado de la media estadística precisar que el algoritmo G1GC obtiene mejor resultado de *throughput* en las pruebas de entrega de SWA, respecto a los demás algoritmos de GC.

En lo que respecta a la métrica de tasa de éxito, la prueba de Dunn y el resultado del efecto indican que existen diferencias significativa en las pruebas de entrega de SWA con diferentes algoritmos de GC. En este caso, de acuerdo a la media estadística, el algoritmo CMS presenta mejores resultados.

Los algoritmos de GC influyen en la capacidad de entrega de SWA, esto debe ponerse en una balanza para que los responsables de las plataformas de SWA definan algoritmo de GC adecuado para responder más rápido y con el mayor porcentaje de éxito posible. Como resultado de este análisis estadístico, es importante mencionar que se rechaza la hipótesis nula, esto es porque en las tres métricas de entregabilidad se indica diferencia significativa en los resultados de entrega de SWA del caso de prueba S-GC5.

Tabla 80. Resultados estadísticos del caso de prueba S-GCA5.

Tiempo de Respuesta						
Algoritmo	Media	Varianza	Anova	Kruskall-W	Dunn	Efecto
STD	330.77	1033.63	2.00E-16	2.2E-16	0	0.552
CMS	317.56	711.93				
SGC	304.40	708.00				
PGC	328.23	349.07				
G1GC	138.22	74.60				
Throughput						
Algoritmo	Media	Varianza	Anova	Kruskall-W	Dunn	Efecto
STD	1.92	1.21	2.00E-16	2.2E-16	0	0.853
CMS	3.29	0.05				
SGC	2.99	0.23				
PGC	1.98	0.24				
G1GC	4.75	0.02				
Tasa de Éxito						
Algoritmo	Media	Varianza	Anova	Kruskall-W	Dunn	Efecto
STD	50.49	185.98	2.00E-16	2.2E-16	0	0.758
CMS	82.80	14.98				
SGC	71.96	42.59				
PGC	51.99	51.51				
G1GC	52.99	10.52				

12.2.11 Análisis Estadístico del Caso de Prueba S-G1P3

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-G1P3 se presentan en la Tabla 81. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito. En estos métodos estadísticos se compararon los resultados de la capacidad de entrega de las pausas del GC G1 First de 100, 200 y 300 milisegundos.

Los resultados de los métodos estadísticos del tiempo de respuesta indican que no existe diferencia significativa entre las tres muestras. Aunque el resultado de la prueba de Dunn indica diferencia entre las muestras, el valor del efecto es bajo, lo que indica una diferencia de tiempo de respuesta mínimo.

Lo que respecta al **throughput**, los resultados de la prueba post hoc de Dunn y el tamaño del efecto indican que no existe diferencia significativa entre las muestras. Es decir, las variaciones en la duración de las pausas del GC G1 First no generan cambios relevantes en la capacidad de entrega de SWA.

En lo que respecta a la métrica de tasa de éxito, los resultados estadísticos muestran que la prueba de Dunn detecta diferencias. No obstante, los tamaños del efecto reportados indican que dichas diferencias presentan una magnitud muy baja en las pruebas de entrega de SWA con distintas longitudes de pausa del GC.

Por último, los resultados del análisis realizado permiten determinar que la longitud de las pausas del GC no mejoran la capacidad de entrega de SWA en ninguna de las métricas de entregabilidad de SWA, tiempo de respuesta, **throughput** y tasa de éxito. Aun sabiendo que la prueba de Dunn detecta estas diferencias, la magnitud del efecto es muy bajo. Por lo que finalmente podemos mencionar que no se rechaza la hipótesis nula.

Tabla 81. Resultados estadísticos del caso de prueba S-G1P3.

Tiempo de Respuesta						
Pausa	Media	Varianza	Anova	Kruskall-W	Dunn	Efecto
100 Millis	124.16	154.38	7.637E-08	7.147e-10	0	0.135
200 Millis	110.23	96.74				
300 Millis	115.13	126.94				
Throughput						
Pausa	Media	Varianza	Anova	Kruskall-W	Dunn	Efecto
100 Millis	6.03	0.65	2.00E-16	7.637e-08	0	0.104
200 Millis	6.92	1.00				
300 Millis	6.41	1.07				
Tasa de Éxito						
Pausa	Media	Varianza	Anova	Kruskall-W	Dunn	Efecto
100 Millis	59.52	12.21	2.00E-16	0.008579	0	0.0253
200 Millis	60.43	20.57				
300 Millis	58.21	20.73				

12.2.12 Análisis Estadístico del Caso de Prueba S-G1R1

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-G1R1 se presentan en la Tabla 82. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

Los resultados de las pruebas de Shapiro y Levene indican la falta de normalidad y homocedasticidad en las muestras. El **asimetría** y la curtosis están muy cerca de la normalidad.

Los resultados estadísticos del tiempo de respuesta indican que existe diferencia significativa entre las muestras. El resultado de la prueba de Yuen indica diferencia entre las muestras, el valor de la magnitud del efecto es medio.

En cuanto a la métrica de **throughput**, en la sección correspondiente de la Tabla 82, indican que existe diferencia significativa entre las muestras. La prueba de Yuen indica diferencia entre las muestras, el valor de la magnitud del efecto es medio.

Los resultados estadísticos que se presentan en la Tabla 82 de tasa de éxito, indican que existe diferencia significativa entre las muestras, aunque el valor de la magnitud del efecto es medio.

Por último, los resultados del análisis realizado permiten determinar que la configuración de G1 Region permite mejorar la capacidad de entrega de SWA en todas las métricas de entregabilidad de SWA, tiempo de respuesta, **throughput** y tasa de éxito. Se recomienda hacer más pruebas con otras configuraciones del parámetro G1 First Region y con otros parámetros con el fin de profundizar en el análisis y determinar las alternativas que permitan mejorar aún más la capacidad de entrega de SWA. De acuerdo a los resultados, y a los resultados obtenidos en otras pruebas, concluimos que no se rechaza la hipótesis nula, es decir, no existe diferencia significativa en los resultados.

Tabla 82. Resultados Estadísticos del Caso de Prueba S-G1R1.

Tiempo de Respuesta													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media G1	Media G1 R1	Yuen	Efecto
VIDEO_10MB	141.76	129.45	0.000001	0.0002	141.296268	0.047285	0.679321	-0.228308	2.771692	145.30	138.22	0.0003	0.4197
Throughput													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media G1	Media G1 R1	Yuen	Efecto
VIDEO_10MB	4.53	0.34	0.002445	0.0000	4.549323	0.015157	-0.288058	-0.120401	2.879599	4.30	4.75	0.0000	0.5647
Tasa de Éxito													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media G1	Media G1 R1	Yuen	Efecto
VIDEO_10MB	51.00	17.75	0.000197	0.0005	51.220225	0.000223	-0.228287	-0.028867	2.971133	49.52	52.48	0.0000	0.5543

12.2.13 Análisis Estadístico del Caso de Prueba S-G1R2

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-G1R2 se presentan en la Tabla 83. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito.

Los resultados que se presentan en la columna Shapiro-W indican la falta de normalidad. Lo que corresponde a la homocedasticidad, en la columna Levene, no se presentan resultados de similaridad en las varianzas. El resultado de asimetría indica simetría, y la curtosis platicúrtica baja, casi mesocúrtica.

En los resultados de tiempo de respuesta presentan diferencia estadística significativa. El estadístico de la prueba de Yuen, junto con el tamaño del efecto estimado, confirman la existencia de estas diferencias y evidencian una magnitud alta entre las muestras comparadas.

El estadístico de la prueba de Yuen, junto con el tamaño del efecto estimado, confirman la existencia de estas diferencias entre las muestras y evidencian una magnitud relevante o bastante alta.

En lo que respecta a la métrica de tasa de éxito, los resultados indican que existen diferencias significativa. El estadístico de la prueba de Yuen, junto con el tamaño del efecto estimado, confirman la existencia de estas diferencias y evidencian una magnitud relevante entre los resultados de las pruebas de entregabilidad de SWA.

Estos resultados permiten determinar que la configuración de G1 Ratio mejora la capacidad de entrega de SWA en tiempo de respuesta, **throughput** y tasa de éxito. Los encargados de las aplicaciones de SWA, deben considerar este parámetro como un factor relevante en aplicaciones que entreguen SWA. Por último, podemos rechazar la hipótesis nula e indicar que existen diferencias entre las muestras al utilizar el parámetro G1 First Ratio para entregar SWA.

Tabla 83. Resultados estadísticos del caso de prueba S-G1R2.

Tiempo de Respuesta													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media G1	Media G1R2	Yuen	Efecto
VIDEO_10MB	122.52	249.68	0.000010	0.0000	122.053092	0.124296	0.542171	-0.491898	2.508102	134.90	110.14	0.0000	0.9604
Throughput													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media G1	Media G1R2	Yuen	Efecto
VIDEO_10MB	6.03	1.73	0.000088	0.0003	6.006669	0.070997	0.350028	-0.794745	2.205255	4.97	7.08	0.0000	0.8549
Tasa de Éxito													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media G1	Media G1R2	Yuen	Efecto
VIDEO_10MB	57.63	39.62	0.001704	0.0038	57.521782	0.067157	0.244070	-0.385576	2.614424	53.25	62.01	0.0000	0.9200

12.2.14 Análisis Estadístico del Caso de Prueba S-G1DD

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-G1DD se presentan en la Tabla 84. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito. Los resultados que se presentan en la columna de Shapiro-W indican la falta de normalidad. Con respecto a la característica de homocedasticidad, en la columna Levene, no se presentan resultados de similaridad en las varianzas.

En los resultados estadísticos del tiempo de respuesta, los resultados indican que se presenta diferencia estadística significativa entre las muestras. El estadístico de la prueba de Yuen, junto con el tamaño del efecto estimado, confirman la existencia de estas diferencias y evidencian una magnitud muy alta.

En lo que respecta a la métrica de *throughput*, en la sección con el mismo nombre de la Tabla 84, es preciso mencionar que los resultados indican que se presentan diferencias significativas favorables a la configuración de G1 Ratio. El estadístico de la prueba de Yuen, junto con el tamaño del efecto estimado, confirman la existencia de estas diferencias y evidencian una magnitud relevante bastante considerable.

En lo que respecta a la métrica de tasa de éxito, los resultados indican que existen diferencias significativas entre las muestras. El estadístico de la prueba de Yuen, junto con el tamaño del efecto estimado, confirman la existencia de estas diferencias y evidencian una magnitud muy alta.

Por último, los resultados del análisis realizado permiten determinar que la configuración de G1 First *deduplication* permite mejorar la capacidad de entrega de SWA en todas las métricas de entregabilidad de SWA, tiempo de repuesta, *throughput* y tasa de éxito. Los encargados de las aplicaciones de SWA, deben considerar este parámetro como un factor relevante.

Tabla 84. Resultados estadísticos del caso de prueba S-G1DD1.

Tiempo de Respuesta													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media G1	Media DD	Yuen	Efecto
VIDEO_10MB	109.69	960.57	0.000000	0.0000	109.690000	0.000145	-0.155605	-1.554432	1.445568	138.22	81.16	0.0000	0.9224
Throughput													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media G1	Media DD	Yuen	Efecto
VIDEO_10MB	10.20	33.78	0.000000	0.0000	10.074391	0.000000	0.308716	-1.606027	1.393973	4.75	15.65	0.0000	0.8962
Tasa de Éxito													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media G1	Media DD	Yuen	Efecto
VIDEO_10MB	75.59	555.46	0.000000	0.0000	75.590000	0.000000	0.019911	-1.945262	1.054738	52.48	98.7	0.0000	0.9321

12.2.15 Análisis Estadístico del Caso de Prueba S-JPS1

Los resultados de los métodos estadísticos aplicados de las pruebas de entregabilidad de SWA del caso de prueba S-JSP1 se presentan en la Tabla 85. Esta tabla se presenta dividida en tres secciones, la primera parte indica los resultados del tiempo de respuesta, la segunda parte indica el throughput y la última corresponde a la tasa de éxito. Los resultados que se presentan en la columna de Shapiro-W indican la falta de normalidad en la mayoría de las muestras. Con respecto a la característica de homocedasticidad, en la columna Levene, solo un caso obtuvo resultados de similaridad en las varianzas.

En los resultados de tiempo de respuesta, los casos de los SWA de 5,10 y 15MB presentan diferencia estadística significativa. El estadístico de la prueba de Yuen, junto con el tamaño del efecto estimado, confirman la existencia de estas diferencias y evidencian una magnitud relevante entre las muestras comparadas y en un caso la magnitud se clasifica como media o moderada.

En lo que respecta a la métrica de **throughput**, en la sección correspondiente de la Tabla 85, es preciso mencionar que los resultados indican que se presentan diferencias significativas favorables a la configuración de Singleton. El estadístico de la prueba de Yuen, junto con el tamaño del efecto estimado, confirman la existencia de estas diferencias y evidencian una magnitud relevante entre las configuraciones comparadas en los casos de SWA de 10 y 15MB y en los casos con SWA de 3 y 5MB la magnitud se considera como media o moderada.

En lo que respecta a la métrica de Tasa de Éxito, los resultados estadísticos de entrega de SWA indican que existen diferencias significativa para ambos casos de SWA con diseño Singleton. El estadístico de la prueba de Yuen, junto con el tamaño del efecto estimado, confirman la existencia de estas diferencias y evidencian una magnitud relevante entre los diseños de programación de SWA comparados.

Los resultados del análisis realizado permiten determinar que el diseño de programación Singleton permite mejorar la capacidad de entrega de SWA, en tiempo de respuesta, **throughput** y tasa de éxito. Los encargados de las aplicaciones de SWA, deben considerar diseñar SWA con este patrón y definirlo como un factor relevante. Para concluir este análisis, rechazamos la hipótesis nula para indicar que existe diferencia estadísticamente significativa entre las muestras en las pruebas de entrega de SWA con el patrón de diseño Singleton con Java EJB.

Tabla 85. Resultados estadísticos del caso de prueba S-JPS1.

Tiempo de Respuesta													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Singleton	Media Standalone	Yuen	Efecto
AUDIO_1MB	12.02	5.19	0.000000	0.000000	11.58837	0.000040	1.074566	0.446268	0.607225	11.25	12.80	0.0006	0.4300
IMG_3MB	32.61	28.93	0.000000	0.000908	32.038523	0.000000	0.079518	1.087534	1.254692	31.68	33.54	0.0001	0.3544
IMG_5MB	37.10	215.70	0.000000	0.000017	36.601886	0.000000	0.000030	0.643262	-0.188974	49.79	24.41	0.0000	0.9863
VIDEO_10MB	105.66	4485.44	0.000000	0.974666	105.664000	0.000000	0.000000	0.014185	-1.958176	39.33	172.00	0.0000	0.9400
VIDEO_15MB	125.71	9122.02	0.000000	0.000000	125.712000	0.000000	0.000000	0.013498	-1.988797	30.62	220.80	0.0000	0.9207
Throughput													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Singleton	Media Standalone	Yuen	Efecto
AUDIO_1MB	11.97	7.11	0.000007	0.000000	11.974867	0.001817	-0.076684	-0.882195	-0.882195	13.10	11.92	0.5364	0.0575
IMG_3MB	11.92	3.17	0.024556	0.161839	11.961635	0.258504	-0.219092	-0.237882	-0.237882	12.33	11.52	0.0001	0.3663
IMG_5MB	14.98	7.52	0.055913	0.000000	15.019801	0.437820	-0.081709	-0.431760	-0.431760	13.57	16.39	0.0000	0.6673
VIDEO_10MB	8.14	30.47	0.000000	0.000000	6.863365	0.000000	0.000000	0.103983	-1.895645	13.53	2.76	0.0000	0.9436
VIDEO_15MB	9.29	52.42	0.000000	0.000001	9.288310	0.000000	0.000000	0.015579	-1.986575	16.49	2.08	0.0000	0.9233
Tasa de Éxito													
SWA	Media	Varianza	Shapiro-W	Levene	Huber	Kolmogorov	Asimetría	Curtosis	Moment	Media Singleton	Media Standalone	Yuen	Efecto
AUDIO_1MB	100.00	0								100	100		
IMG_3MB	100.00	0								100	100		
IMG_5MB	98.91	2.61	0.000000	0.003126	0.000000	0.000000	-1.229139	0.301480	0.301480	97.82	100.00	0.0000	0.7533
VIDEO_10MB	68.94	968.45	0.000000	0.000000	39.573687	0.000000	0.000000	0.483689	-1.110841	99.99	37.88	0.0000	0.9841
VIDEO_15MB	33.52	95.56	0.000000	0.001221	25.188952	0.000000	0.000000	0.019694	-1.218528	43.03	24.00	0.0000	0.8473

